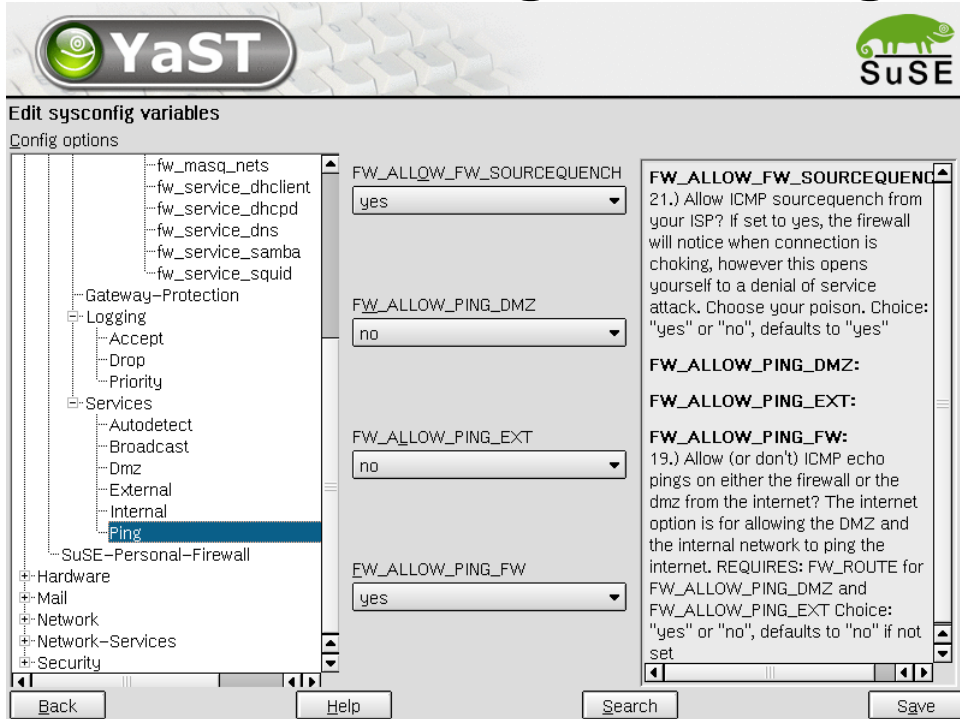


# SuSE Firewall2

## Understanding and Using



## Unofficial SuSEFAQ

by Togan Muftuoglu

---

## SuSE Firewall2: Understanding and Using

Unofficial SuSEFAQ

by Togan Muftuoglu

Author of SuSEfirewall2: Marc Heuse

Cookbook solutions: SuSE Security Mailinglist participants

Configuring VPN with SSH Sentinel: Nadeem Hasan

Copyright © 2002 Togan Muftuoglu

Copyright (c) 2002 by Togan Muftuoglu. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v0.4 or later (the latest version is presently available at <http://www.open-content.org/openpub/>).

Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.

Distribution of the work or derivative of the work in any standard (paper) book form is prohibited unless prior permission is obtained from the copyright holder.

A copy of the license is included in Appendix C .

The only exception to the License is Section 5.2 by Nadeem Hasan which has a CopyLeft license

*Linux* is a registered trademark of *Linus Torwalds*. *Windows*, *Windows95™*, *Windows98™* *WindowsXP™* *Windows2000™* *Windows NT™* are registered trademarks of *Microsoft Corporation*. *SuSE YaST™* and *YaST2™* are registered trademarks of *SuSE Gmbh*. Other products mentioned in this manual may be trademarks of the respective people, organizations, manufacturers, companies.

---

# Dedication

In memory of Ali Mandalinci who passed away at the age of 38 whom I have shared invaluable moments of true friendship.

Life is a tale told by an idiot,  
Full of sound and furry,  
Signifying nothing.

DRAFT

DRAFT

# Table of Contents

Preface .....	xi
1. Audience .....	xi
2. Layout of the document .....	xi
3. Examples .....	xii
4. Conventions used in this document .....	xii
5. Comments and Questions .....	xiii
6. Acknowledgments .....	xiii
I. Introduction .....	15
1. Introduction to SuSEfirewall2 .....	17
1.1. Where to get from .....	17
1.2. How to install .....	17
1.3. Technical Background .....	17
1.4. How SuSEfirewall2 works .....	20
2. Configuration .....	23
2.1. Variables used in SuSEfirewall2 .....	23
2.2. SuSEfirewall2 Command parameters .....	33
2.2.1. Configuring SuSEfirewall2 for different rules via cron .....	33
II. Configuration .....	35
3. Configuration .....	37
3.1. Using the Easy Configuration option .....	37
3.2. Configuring SuSEfirewall2 with YaST2 .....	38
4. Editing The Configuration file .....	47
4.1. Basic Configuration of SuSEFirewall2 .....	47
4.1.1. Single user .....	47
4.1.2. Examples for basic configuration .....	48
4.2. Proxy masquerading .....	48
4.2.1. Examples for Proxy Configuration .....	51
4.3. Firewall Masquerading .....	52
4.3.1. Examples for Masquerading firewall configuration .....	53
4.4. Configuring the SuSEfirewall2 for using DMZ .....	53
4.4.1. Setting up Services for DMZ .....	54
4.4.2. Services Autoprotect .....	54
4.4.3. FW_FORWARD .....	56
4.4.4. Kernel security options .....	56
4.4.5. Routing state .....	56
4.4.6. Ping .....	56
4.4.7. Example for DMZ configuration .....	57
5. Expert Level Configuration .....	61
5.1. Editing the Expert Options .....	61
5.1.1. How FW_AUTOPROTECT_SERVICES works .....	61
5.1.2. Configuring HIGHPORTS .....	62
5.1.3. Configuring FW_REDIRECTT .....	64
5.1.4. Expert log Format .....	65
5.1.5. Allowing Ping .....	66
5.1.6. Configuring the SuSEfirewall2 for Traceroute .....	67
5.1.7. Understanding FW_ALLOW_FW_SOURCEQUENCH .....	67
5.1.8. Understanding BROADCAST options .....	68
5.1.9. Routing interfaces of same class .....	69
5.2. Building a VPN with FreeS/WAN, SuSEfirewall2 and SSHSentinel .....	69
5.2.1. Introduction .....	70
5.2.2. Prerequisites .....	70
5.2.3. Firewall Tweaks .....	70
5.2.4. OpenSSL and Certification Authority (CA) .....	71
5.2.5. FreeS/WAN Gateway Certificate .....	72
5.2.6. Installing the Private Key .....	72

5.2.7. Configuring FreeS/WAN .....	73
5.2.8. Poking Holes in the Wall .....	74
5.2.9. Configuring SSH Sentinel .....	74
5.2.10. Putting It All Together .....	75
5.2.11. Conclusion .....	76
5.3. Configuring SuSEfirewall2 for VPN .....	76
6. Customizing SuSEfirewall2 .....	79
6.1. Using custom rules with SuSEfirewall2 .....	79
6.1.1. Enabling FW_CUSTOM .....	79
6.1.2. Understanding how FW_CUSTOM works .....	79
6.2. Identd Dilemma .....	81
6.2.1. Definition of REJECT target .....	81
6.2.2. Deciding the error code to send back .....	82
III. Logs and Interperating them .....	83
7. Log messages .....	85
7.1. Log messages and understanding them .....	85
7.1.1. Understanding Iptables' log parameters .....	85
7.1.2. Variables used in the SuSEfirewall2 generated logs .....	86
7.1.3. Analyzing SuSEfirewall2 generated logs .....	88
7.2. Firewall Log Analysis Software .....	88
IV. Troubleshooting .....	91
8. Troubleshooting .....	93
8.1. Cookbook Recipes .....	93
8.1.1. Configuring SuSEfirewall2 for pcAnywhere .....	93
8.1.2. Configuring SuSEfirewall2 for edonkey .....	93
8.1.3. Configuring SuSEfirewall2 for Kazaa .....	93
8.1.4. Configuring Masquerading for specific port numbers .....	94
8.1.5. Configuring SuSEfirewall2 for use with time servers .....	94
8.1.6. Configuring SuSEfirewall2 for Internal Access to External IP .....	94
8.1.7. Accessing the webserver in DMZ .....	95
8.1.8. Protection for DOS .....	95
8.1.9. Security Aspects & Troubleshooting for FTP .....	96
V. Appendixes .....	99
A. ICMP types .....	101
B. Resources .....	103
B.1. Resources on the Web .....	103
C. OPEN PUBLICATION LICENSE .....	105
C.1. REQUIREMENTS ON BOTH UNMODIFIED AND MODIFIED VERSIONS .....	105
C.2. COPYRIGHT .....	105
C.3. SCOPE OF LICENSE .....	105
C.4. REQUIREMENTS ON MODIFIED WORKS .....	106
C.5. GOOD-PRACTICE RECOMMENDATIONS .....	106
C.6. LICENSE OPTIONS .....	106
C.7. OPEN PUBLICATION POLICY APPENDIX: .....	106
Glossary of terminology .....	111

---

## List of Figures

4.1. Dual-homed Host .....	48
4.2. Using Proxy Services .....	49
4.3. Proxy Masquearading .....	52
4.4. Sample DMZ .....	58
5.1. Transparent proxy .....	64

DRAFT

DRAFT

---

## List of Tables

2.1. SuSEfirewall2 command options .....	33
6.1. REJECT target .....	81
7.1. LOG target options .....	85
7.2. SuSEfirewall2 LOG Prefixes .....	87
7.3. SuSEfirewall2 log explanations .....	88
A.1. ICMP types .....	101

DRAFT

DRAFT

---

## List of Examples

4.1. Single User ADSL .....	48
4.2. Single User ISDN .....	48
4.3. Proxy Masquerading with ISDN .....	51
4.4. Masquerading Only .....	53
4.5. DMZ configuration .....	57
5.1. Transparent Proxy .....	64
5.2. Transparent Squid Proxy .....	65
5.3. Sample VPN configuration .....	76
5.4. VPN with no masquerading .....	76
7.1. Drop Default .....	88

DRAFT

DRAFT

---

# Preface

This document tries to explain how to configure the excellent SuSEfirewall2 packet filtering script in a layman's level. Most of the information you will find on these pages are basically collected by reading the documentation, the SuSEfirewall2 script itself and **searching the fine web** for SuSEfirewall2 related questions. The SuSE-Security mailing list archives were extremely helpful in finding answers especially on troubleshooting.

"Although naming the program as a firewall is theoretically wrong as it is a highly configurable and effective packet filter"

says Marc Heuse yet I will refer it as a *firewall* just to make things easier for readability.

A basic definition for firewall can be *a component or set of components that restricts access between a protected network and the Internet, or between other sets of networks.*

## 1. Audience

Questions on configuring SuSEfirewall2 is a common topic both on SuSE Linux and SuSE-Security mailinglists. It is sometimes difficult to understand how to configure the application so the intended protection takes place. Therefore the intended audience is both the the new user and the savy system admin. The document tries to provide information covering a wide interest area.

In the hope of making SuSEfirewall2 usage easier and to avoid fighting simple issues encountered by SuSE Linux users this document tries to cover the following

- Understanding how SuSEfirewall2 works
- Understanding the parameters of SuSEfirewall2 and assigning the needed variables.
- How to do troubleshooting

## 2. Layout of the document

### Chapter 1 Technical Background

In order to effectively configure SuSEfirewall2, one must understand the process happening in the background. SuSEfirewall2 initiates in three steps and in each step network protection is available. It is recommended to read this chapter before moving further into the configuration itself

### Chapter 2 Variables and Parameters

Since there is no preset firewall configuration for protecting your network, SuSEfirewall2 needs some parameters to be set before it can operate.

This chapter is a **must** to read before configuring the application. You can configure complicated firewalls if you really understand what the variables do mean and how information can be entered into them.

### Chapter 3 YaST2 GUI configuration

Using YaST2 it is possible to configure SuSEfirewall2. The firewall module enables the user to set a basic firewall configuration.

### Chapter 4 Editing the Configuration file

If you would like to have more control in the configuration of the variables this is the chapter you would like to read.

understanding the examples in this chapter should aid you in setting your firewall.

### Chapter 5 Expert Configuration

If your needs are beyond the basic configuration, or you would like to fine tune SuSEfirewall2, this chapter should help you in achieving these.

Autoprotecting services, forwarding ports and redirecting are discussed in this chapter along with examples.

**Chapter 6 Customizing**

It is possible to extend SuSEfirewall2 so it's capabilities are expanded. Sometimes you may need to have some special rules inserted to the firewall to allow or deny specific settings.

Be careful as this area is highly expert level and you may lose the protection if misconfigurations are done.

**Chapter 7 Logs and Understanding them**

In this section understanding SuSEfirewall2 generated logs is the key topic. Interpreting the generated logs is very useful especially when you are getting suspicious about what's happening behind.

It is also useful in the debugging procedure when SuSEfirewall2 is not behaving as you are expecting it to.

**Chapter 8 Troubleshooting**

Sometimes you need quickfixes, which will satisfy the needs with immediate gratification. This chapter tries to come out with such recipes.

## 3. Examples

In order to make the document more understandable examples are used which are from the SuSEfirewall2 documentation. When it was necessary additional examples have been supplied. Before applying the examples please make sure you understand the intent of the situation and then adapt it to your network needs.

## 4. Conventions used in this document

In the hope of making ease of readability the following are used:

**Choices**

In order to clarify the choices available for the selected variable these are blue colored if there is a default setting this is highlighted with bold black color.

Choice "yes" or "no" defaults to "no"

**Requires**

Sometimes in order for the parameter to function another variable needs to be configured also. This is identified as follows.

**REQUIRES:FW\_ROUTE**

with the required variable linked

**Note**

In order to identify **Notes** the following notation is used.

**Note**

This is informative note

**Tip**

In order to identify **Tip** the following notation is used.

**Tip**

This is a helpful hint

**Warning**

In order to point out a **Warning**, the following notation is used.

## Warning

---

This is a warning message. Be sure to read it carefully.

---

### Caution

In order to identify a **Caution** information, the following notation is used.

## Caution

---

Be careful this is a **Caution** area

---

## 5. Comments and Questions

The program coding screens directly come from the SuSEfirewall2 script. Should there be any error in those areas please check the script itself that is installed in your computer. If there are conflicts between the document and the script itself, it is highly recommended that you follow the script installed in your computer. As there are bug fixes and feature improvements involved with the application, it is difficult to maintain consistency.

The recipes that are available Chapter 8 are from SuSE-Security mailinglist archives. Although they should work without a major problem, it is recommended before you imply them you have a through understanding how things work.

Nevertheless it is very likely that there will be typographical errors or a solution may not work. In these cases please report any bugs, patches, recommendations to the project website located at <http://sourceforge.net/projects/susefaq> .There is a special tracker for SuSEfirewall2 [http://sourceforge.net/tracker/?at\\_id=509170&group\\_id=42064&func=browse](http://sourceforge.net/tracker/?at_id=509170&group_id=42064&func=browse).

## 6. Acknowledgments

- Actually the whole credit should go to Marc Heuse since I just reformatted what was in the script and the configuration file
- SuSE-Security mailing list has valuable members. Many of the cookbook recipes came from that mailing list
- In Chapter 5, Section 5.2 is work of Nadeem Hasan. The original article's location listed in Appendix B. It is included in this document considering the user may not have a Internet connection while setting up SuSEfirewall2.

DRAFT

---

# Part I. Introduction

This is the part where we will start understanding how SuSEfirewall2 works. Once we have a better understanding what's in the background we will then have a look to the variables used in the configuration file `/etc/susconfig/SuSEfirewall2`

It is extremely important to understand the background process and the variables used in the SuSEfirewall2 to have a working configuration that best meets the needs of your network environment

DRAFT

DRAFT

# 1

## Introduction to SuSEfirewall2

### 1.1. Where to get from

The most current version of the SuSE Firewall2 can be found at <http://www.suse.de/~marc/suse.html>. The version that comes with SuSE 7.3 CDs' is old and has some bugs which were fixed later and hence the reason to get the latest from Marc's page.

If you are using SuSE 8.0 due to the fact of `/etc/sysconfig` directory structure, the correct version is the one that comes with the *SuSE 8.0 CD/DVD*

#### Note

You can get version 3.1 for 8.0 from <ftp://ftp.gwdg.de/pub/linux/suse/people/garloff/linux/SuSE/RPMS/8.0/SuSEfirewall2-3.1-19.noarch.rpm> there is a new option which is called *easy*. This basically makes SuSEfirewall2 to act like the personal-firewall with rejecting all incoming connections.

Please note that when using packages from the *people* directory you are not using the SuSE supported version

This article will cover the version compatible with SuSE 8.1

### 1.2. How to install

Depending on where you get the SuSEfirewall2 the installation method could be via YaST or via `rpm -Uhv package-name.rpm` or via `tar zxvf package.tar.gz` and changing in to the directory and issuing the `./INSTALL` command

Regardless of the method of installation you should read the documentation that comes with the firewall package which will walk you through how it should be configured for your specific case.

Here is the list of **documentation files** located at `/usr/share/doc/packages/SuSEfirewall2`. You may want read these before configuring the firewall

- SuSEfirewall2.conf.EXAMPLE
- FAQ
- PROBLEMS

The configuration file is placed under the `/etc/sysconfig/SuSEfirewall2` and it does include explanations of the various parameters.

### 1.3. Technical Background

How does SuSEfirewall2 starts at boot time

This section will try to explain what happens when you have configured `/etc/sysconfig/SuSEfirewall2` and the program SuSEfirewall2 starts.

1. Important is the initialization of the firewall during the boot up. First `/etc/init.d/SuSEfirewall2_init` is called, making sure no incoming network traffic allowed except `dhcpcd` + `ping` (used for boot security).

```
case "$1" in
start)
echo -n "Starting Firewall Initialization: "
echo -n '(phase 1 of 3) '
( $SUSEFWALL close ) > /dev/null 2>&1 || return=$rc_failed
echo -e "$return" ;;
```

## Note

`/etc/init.d/SuSEfirewall2_init` *start* calls `/sbin/SuSEfirewall2` with the *close* parameter.

```
test "$1" = close && {
set_basic_rules ❶
{echo 0 > /proc/sys/net/ipv4/ip_forward 2> /dev/null ;} >/dev/null 2>&1❷
$IPTABLES -A INPUT -j ACCEPT -p udp --sport 67 -d 255.255.255.255/32 --dport 68❸
$IPTABLES -A INPUT -j ACCEPT -p icmp --icmp-type echo-request ❹
$IPTABLES -A INPUT -j ACCEPT -i lo ❺
$IPTABLES -A INPUT -j "$REJECT" ❻
test -x "$LOGGER" && \
$LOGGER -p kern.info -t SuSEfirewall2 "Firewall rules set to \
CLOSE all network traffic."❼
exit 0
}
```

## Close parameter

- ❶ `set_basic_rules` function is called
- ❷ Ip forwarding is disabled
- ❸ DHCP client's IP retrieve is allowed
- ❹ ICMP with echo-request is allowed
- ❺ Packets on loopback device is accepted
- ❻ Default INPUT is set as `$REJECT`
- ❼ Status is logged

```
function set_basic_rules() {
{ rmmod ipfwadm; rmmod ipchains
modprobe ip_tables; modprobe ip_conntrack; modprobe ip_conntrack_ftp
modprobe ip_nat_ftp
} > /dev/null 2>&1 ❶
$IPTABLES -F INPUT
$IPTABLES -F OUTPUT
$IPTABLES -F FORWARD 2> /dev/null ❷
test "$DROP" = ACCEPT && DROP_JUMP="ACCEPT"
test "$DROP" = ACCEPT || DROP_JUMP="DROP"
$IPTABLES -P INPUT "$DROP_JUMP"
$IPTABLES -P OUTPUT "$ACCEPT"
$IPTABLES -P FORWARD "$DROP_JUMP" 2> /dev/null❸
$IPTABLES -F
$IPTABLES -X
# Special REJECT function #
$IPTABLES -t nat -F
$IPTABLES -t nat -X
$IPTABLES -t mangle -F
```

```

$IPTABLES -t mangle -X ④
$IPTABLES -N reject_func
$IPTABLES -A reject_func -p tcp -j REJECT --reject-with tcp-reset ⑤
$IPTABLES -A reject_func -p udp -j REJECT --reject-with icmp-port-unreachable
$IPTABLES -A reject_func -j REJECT --reject-with icmp-proto-unreachable ⑥
}

```

## Basic rules setup

- ① `ipfwadm`, `ipchains` modules are unloaded and `iptables` related modules are loaded.
  - ② Rules for default chains are flushed
  - ③ Default policies are set
  - ④ Tables are flushed
  - ⑤ For tcp protocol packets are rejected with `tcp-reset`
  - ⑥ For UDP and other protocols packets are rejected with `icmp-proto-unreachable`
2. Next when all other related init scripts have started (`SuSEfirewall2_init` `network` `route` `dhclient`) `/etc/init.d/SuSEfirewall2_setup` is called which will read and parse the `/etc/sysconfig/SuSEfirewall2` configuration file generating firewall rules for all interfaces available at this time
  3. However, because more dynamic connections and services might be run later (ie. `rpc`, `named`, `sshd`, `inetd`, `dhcp`, `nscd`, `nessusd`, `wpmc`, `squid`, `ipsec`, after all these have started then `/etc/init.d/SuSEfirewall2_final` is called. All possible firewall rules are generated and also if there are error messages are given back.

It is important that from dynamic dialup scripts, the firewall should always be called from `/etc/ppp/ip-up` so that firewall rules are reloaded. This is already done by SuSE so no worries here.

Just by configuring these settings and using the `SuSEfirewall2` you are not secure per se! There is not such a thing you install and hence you are saved from all (security) hazards.

### Enhanced Security

The below list is placed at the very beginning of `/etc/sysconfig/SuSEfirewall2` and it says ensure your security, you need also:

- Secure all services you are offering to untrusted networks (internet) You can do this by using software which has been designed with security in mind (like `postfix`, `apop3d`, `ssh`), setting these up without misconfiguration and praying, that they have got really no holes. `SuSEcompartment` can help in most circumstances to reduce the risk.
- Do not run untrusted software. (philosophical question, can you trust SuSE or any other software distributor?)
- Harden your server(s) with the `hardened_suse` package/script
- Recompile your kernel with the `openwall-linux` kernel patch <http://www.openwall.com> (former `secure-linux` patch, from Solar Designer)
- Check the security of your server(s) regularly
- If you are using this server as a firewall/bastion host to the internet for an internal network, try to run proxy services for everything and disable routing on this machine.
- If you run DNS on the firewall: disable untrusted zone transfers and either don't allow access to it from the internet or run it split-brained.

## 1.4. How SuSEfirewall2 works

This section will try to explain what happens when you have configured `/etc/sysconfig/SuSEfirewall2` and the program SuSEfirewall2 starts.

In this section is often talked about filter rules on internal interfaces and networks. These will only be set up if you defined those networks and interfaces and they are up and running when you are starting the firewall. (If not, no traffic on/to these is allowed)

This is how the SuSEfirewall2 script, which resides in `/sbin`, works:

- a. First it reads the configuration file `/etc/sysconfig/SuSEfirewall2`. You must first configure this configuration file before anything can happen.
- b. All tools like `sed`, `awk`, `grep`, `ifconfig`, `netstat` and of course `iptables` are searched for and if not available it terminates with an error message. If the kernel version can not be determined, it prints a warning; if it is not a 2.4.x kernel it terminates with an error message.
- c. Now all the configuration options are processed, some logical settings done and some data is read from the system, e.g. IP addresses and netmasks of interfaces. If an interface is not up, there will only some small protection rules generated. After any new connection with a dynamic IP address, this script should be run again.

### Note

If you are connected via `ppp` or `ippp` the script `/etc/ip-up` takes care of this restart automatically.

- d. The script starts! All former rules are flushed and defaults for incoming packets and those which has to be routed are set to `DROP`. This means that if there's no rule found for a special packet, it will be thrown away. All packets which leave the firewall are *allowed*.
- e. If the firewall is configured to route, the routing is activated. (You need a kernel configured for this, however, the default kernel from SuSE is.)
- f. The `/proc` system allows easy online kernel configuration. The script uses this possibility to add some security settings. However, to set most of these, you have to set the option `FW_KERNEL_SECURITY` in the configuration file to `yes`, because the results can be very complex ...
- g. Very important: all traffic is allowed via the `localhost` interface.
- h. Now the IP spoofing and circumvention rules are set, which prevents attacks in which an intruder tries to disguise his data as coming from the internal network, and direct access to the internal network.
- i. The redirection rules, which come next, can be used to direct access to the internal network or local ports to special port defined on the firewall.
- j. Is an internal (trusted?) network connected to the firewall? If this is the case, and an internal interface is defined, now the configuration option `FW_PROTECT_FROM_INTERNAL` is processed, which allows all traffic from the internal network to the firewall, if set to `yes`. Otherwise, all other filter rules notes below are also done against the internal network, unless noted otherwise.
- k. Now all ICMP, TCP and UDP rules are generated.

The ICMP rules are generated in two waves, first special ICMP packets are allowed or denied, depending on the configuration, e.g. ping to the firewall, sourcequench messages from the connected routers and replies to traceroute like programs (TTL exceeded). Next the general config which denied dangerous ICMP packets and allows important ones are set. The internal network is always allowed to ping the firewall.

- l. For the TCP rules, first all configured services which are allowed to the outside, to trusted networks and the internal one are allowed. Then port 113 is set to send `REJECT` with TCP connection resets to all incoming connections (this is important to prevent long mail sending delays). For more information please see Section 6.2

Then a fine automation process is done. If the option `FW_AUTOPROTECT_GLOBAL_SERVICES` was set to `yes` in the configuration file, all services which are listening on all interfaces (e.g. public services, using `0.0.0.0` or `INADDR_ANY`) will be protected. This is useful if you have to open your high ports to incoming connection, but you want to ensure that e.g. your database running on port 4545 is protected.

Finally, it is decided from the configuration if/how the high/unprivileged ports (between 1024 and 65535) may be accessed: not at all, only DNS packets from name servers which are defined in `/etc/resolv.conf`, only packets which come from a special source port (not recommended, this protection is easy to subvert), or everyone. Internal networks are always allowed to access unprivileged ports (except to those protected by the AUTOPROTECT routine).

- m. The same is done for UDP, but the connection reset for port 113 is not needed here.
- n. Then come the routing rules, those, which defined to which internal systems external machines are allowed to have access to. This should of course not be used!!
- o. Now are the masquerading rules set.
- p. Very important for the configuration are additional logging mechanisms for special packets. E.g. logging all incoming TCP packets which want to initiate a connection. If a `LOG_*_ALL` option is set, all packets are logged! This is for debugging filter problems.

Last, some optimization rules are installed, to make the transfer with ssh, ftp, www, syslog and snmp faster or more reliable.

This is the end of the script. If an error occurred, it returns 1, if everything went fine, it returns 0.

DRAFT

# 2

## Configuration

### 2.1. Variables used in SuSEfirewall2

#### 1. FW\_QUICKMODE

Should the Firewall run in quickmode?

*Quickmode* means that only the interfaces pointing to external networks are secured, and no other. all interfaces not in the list of FW\_DEV\_EXT are allowed full network access! Additionally, masquerading is automatically activated for FW\_MASQ\_DEV devices. and last but not least: all incoming connection via external interfaces are **REJECTED**.

You will only need to configure

- FW\_DEV\_EXT
- FW\_MASQ\_DEV.

Optionally, you may add entries to section FW\_SERVICES\_QUICK\_\*

**Choice:** "yes" or "no", if not set defaults to "no"

```
FW_QUICKMODE="no"
```

### Interfaces

#### 2. FW\_DEV\_EXT

Which is the interface that points to the Internet/untrusted networks?

Enter all the network devices here which are untrusted.

**Choice:** any number of devices, separated by a space e.g. "eth0", "ipp0 ipp1 eth0:1"

### Tip

If you have more then one device connected to the Internet you can have multiple definitions ie. ppp+, ipp0, eth0 entered spearted by a space

```
FW_DEV_EXT="ppp+ ipp0 eth0"
```

#### 3. FW\_DEV\_INT

Which is the interface that points to the internal network?

Enter all the network devices here which are trusted. If you are not connected to a trusted network (e.g. you have just a dialup) leave this empty.

**Choice:** leave empty or any number of devices, separated by a space e.g. "tr0", "eth0 eth1 eth1:1" or " "

#### 4. **FW\_DEV\_DMZ**

Which is the interface that points to the dmz or dialup network?

Enter all the network devices here which point to the dmz/dialups. A *dmz* is a special, separated network, which is only connected to the firewall, and should be reachable from the Internet to provide services, e.g. WWW, Mail, etc. and hence are at risk from attacks. See `/usr/share/doc/packages/SuSEfirewall12/EXAMPLES` for an example.

### Note

---

You have to configure `FW_FORWARD` to define the services which should be available to the Internet and set `FW_ROUTE` to "yes".

---

**Choice:** leave empty or any number of devices, separated by a space e.g. "tr0", "eth0 eth1 eth1:1" or " "

#### 5. **FW\_ROUTE**

Should routing between the Internet, dmz and internal network be activated?

**REQUIRES:** `FW_DEV_INT` or `FW_DEV_DMZ`

You need only set this to "yes", if you either want to masquerade internal machines or allow access to the dmz (or internal machines, but this is not a good idea). This option supersedes `IP_FORWARD` from `/etc/sysconfig/network/options`

Setting this option one alone doesn't do anything. Either activate masquerading with `FW_MASQUERADE` below if you want to masquerade your internal network to the Internet, or configure `FW_FORWARD` to define what is allowed to be forwarded!

**Choice:** "yes" or "no", defaults to "no"

### Masquerading

#### 6. **FW\_MASQUERADE**

Do you want to masquerade internal networks to the outside?

**REQUIRES:** `FW_DEV_INT` or `FW_DEV_DMZ`, `FW_ROUTE`

"Masquerading" means that all your internal machines which use services on the Internet seem to come from your firewall.

### Note

---

Please note that it is more secure to communicate via proxies to the Internet than masquerading. This option is required for `FW_MASQ_NETS` and `FW_FORWARD_MASQ`.

---

**Choice:** "yes" or "no", defaults to "no"

#### **FW\_MASQ\_DEV**

You must also define on which interface(s) to masquerade on. This is normally your external device(s) to the Internet.

Most users can leave the default below.

e.g. "ipp0" or \$FW\_DEV\_EXT

### FW\_MASQ\_NETS

Which internal computers/networks are allowed to access the Internet directly (not via proxys on the firewall)?

Only these networks will be allowed access and will be masqueraded!

**Choice:** leave empty or any number of hosts/networks separated by a space.

Every host/network may get a list of allowed services, otherwise everything is allowed. A target network, protocol and service is appended by a comma to the host/network. e.g. 10.0.0.0/8 allows the whole 10.0.0.0 network with unrestricted access. 10.0.1.0/24,0/0,tcp,80 10.0.1.0/24,0/0tcp,21 allows the 10.0.1.0 network to use www/ftp to the Internet. 10.0.1.0/24,tcp,1024:65535 10.0.2.0/24 is OK too.

Set this variable to "0/0" to allow unrestricted access to the Internet.

## General protection

### 7. FW\_PROTECT\_FROM\_INTERNAL

Do you want to protect the firewall from the internal network?

**REQUIRES:** FW\_DEV\_INT

If you set this to "yes", internal machines may only access services on the machine you explicitly allow. They will be also affected from the FW\_AUTOPROTECT\_SERVICES option.

If you set this to "no", any user can connect (and attack) any service on the firewall.

**Choice:** "yes" or "no", defaults to "yes"

### 8. FW\_AUTOPROTECT\_SERVICES

Do you want to autoprotect all running network services on the firewall?

If set to "yes", all network access to services TCP and UDP on this machine will be prevented (except to those which you explicitly allow, see below: FW\_SERVICES\_{EXT,DMZ,INT}\_{TCP,UDP})

**Choice:** "yes" or "no", defaults to "yes"

### 9. FW\_SERVICES\_\*\*

Which services **ON THE FIREWALL** should be accessible from either the Internet (or other untrusted networks), the dmz or internal (trusted networks)?

## Note

(see FW\_FORWARD& FW\_FORWARD\_MASQ if you want to route traffic through the firewall)

Enter all ports or known portnames below, separated by a space. TCP services (e.g. SMTP, WWW) must be set in FW\_SERVICES\_\*\_TCP, and UDP services (e.g. syslog) must be set in FW\_SERVICES\_\*\_UDP. e.g. if a webserver on the firewall should be accessible from the Internet:

```
FW_SERVICES_EXT_TCP="www"
```

e.g. if the firewall should receive syslog messages from the dmz:

```
FW_SERVICES_DMZ_UDP="syslog"
```

For IP protocols (like GRE for PPTP, or OSPF for routing) you need to set `FW_SERVICES_*_IP` with the protocol name or number (see `/etc/protocols`)

**Choice:** leave empty or any number of ports, known portnames (from `/etc/services`) and port ranges separated by a space. Port ranges are written like this: allow port 1 to 10 -> "1:10" e.g. " ", "smtp", "123 514", "3200:3299", "ftp 22 telnet 512:514" For `FW_SERVICES_*_IP` enter the protocol name (like "igmp") or number ("2")

#### **FW\_SERVICES\_EXT\_TCP**

Common: smtp domain

```
FW_SERVICES_EXT_TCP=" "
```

#### **FW\_SERVICES\_EXT\_UDP**

Common: domain

```
FW_SERVICES_EXT_UDP
```

#### **FW\_SERVICES\_EXT\_IP**

For VPN/Routing which END at the firewall!!

```
FW_SERVICES_EXT_IP=" "
```

#### **FW\_SERVICES\_DMZ\_TCP**

Common: smtp domain

```
FW_SERVICES_DMZ_TCP=" "
```

#### **FW\_SERVICES\_DMZ\_UDP=""**

Common: domain

```
FW_SERVICES_DMZ_UDP=" "
```

#### **FW\_SERVICES\_DMZ\_IP**

For VPN/Routing which END at the firewall!!

```
FW_SERVICES_DMZ_IP=" "
```

#### **FW\_SERVICES\_INT\_TCP**

Common: ssh smtp domain

```
FW_SERVICES_INT_TCP=" "
```

**FW\_SERVICES\_INT\_UDP**

Common: domain syslog

```
FW_SERVICES_INT_UDP= " "
```

**FW\_SERVICES\_INT\_IP**

For VPN/Routing which END at the firewall!!

```
FW_SERVICES_INT_IP= " "
```

**10. FW\_TRUSTED\_NETS**

Which services should be accessible from trusted hosts/nets?

Define trusted hosts/networks (doesn't matter if they are internal or external) and the TCP and/or UDP services they are allowed to use. Please note that a trusted host/net is **not** allowed to ping the firewall until you set it to allow also icmp!

**Choice:** leave `FW_TRUSTED_NETS` empty or any number of computers and/or networks, separated by a space. e.g. `"172.20.1.1 172.20.0.0/16"`

Optional, enter a protocol after a comma, e.g. `"1.1.1.1,icmp"`

Optional, enter a port after a protocol, e.g. `"2.2.2.2,tcp,22"`

**11. FW\_ALLOW\_INCOMING\_HIGHPORTS\_\***

How is access allowed to high (unprivileged [above 1023]) ports?

```
FW_ALLOW_INCOMING_HIGHPORTS_TCP
FW_ALLOW_INCOMING_HIGHPORTS_UDP
```

You may either allow everyone from anyport access to your highports ("yes"), disallow anyone ("no"), anyone who comes from a defined port (portnumber or known portname) [note that this is easy to circumvent!], or just your defined nameservers ("DNS").

**Note**

Note that you can't use rpc requests (e.g. `rpcinfo`, `showmount`) as root from a firewall using this script (well, you can if you include range `600:1023` in `FW_SERVICES_EXT_UDP` ...).

Please note that with v2.1 "yes" is **not mandatory** for active FTP from the firewall anymore.

**Choice:** "yes", "no", "DNS", portnumber or known portname, defaults to "no" if not set

**FW\_ALLOW\_INCOMING\_HIGHPORTS\_TCP**

Common: "ftp-data", better is "yes" to be sure that everything else works :-)

**FW\_ALLOW\_INCOMING\_HIGHPORTS\_UDP**

Common: "DNS" or "domain" "ntp", better is "yes" to be sure ...

**12. FW\_SERVICE\_AUTODETECT**

Are you running some of the services below?

They need special attention - otherwise they won't work!

Set services you are running to "yes", all others to "no", defaults to "no" if not set.

#### **FW\_SERVICE\_DNS**

If you are running bind/named set to yes. Remember that you have to open port 53 (or "domain") as udp/tcp to allow incoming queries.

Also FW\_ALLOW\_INCOMING\_HIGHPORTS\_UDP needs to be "yes"

#### **FW\_SERVICE\_DHCLIENT**

if you use dhclient to get an ip address you have to set this to "yes" !

#### **FW\_SERVICE\_DHCPD**

set to "yes" if this server is a DHCP server

#### **FW\_SERVICE\_SQUID**

set to "yes" if this server is running squid. You still have to open the tcp port 3128 to allow remote access to the squid proxy service.

#### **FW\_SERVICE\_SAMBA**

set to "yes" if this server is running a samba server. You still have to open the tcp port 139 to allow remote access to SAMBA.

### 13. **FW\_FORWARD**

Which services accessed from the Internet should be allowed to the dmz (or internal network - if it is not masqueraded)?

**REQUIRES: FW\_ROUTE**

With this option you may allow access to e.g. your mailserver. The machines must have valid, non-private, IP addresses which were assigned to you by your ISP. This opens a direct link to your network, so only use this option for access to your dmz!!!!

**Choice:** leave empty (good choice!) or use the following explained syntax of forwarding rules, separated each by a space.

A forwarding rule consists of:

- source IP/net
- destination IP separated by a comma. e.g. 1.1.1.1,2.2.2.2 3.3.3.3/16,4.4.4.4/24

Optional is a protocol, separated by a comma, e.g. 5.5.5.5,6.6.6.6,igmp Optional is a port after the protocol with a comma, e.g. 0/0,0/0,udp,514

### 14. **FW\_FORWARD\_MASQ**

Which services accessed from the Internet should be allowed to masqueraded servers (on the internal network or dmz)?

**REQUIRES: FW\_ROUTE**

With this option you may allow access to e.g. your mailserver. The machines must be in a masqueraded segment and may not have public IP addresses!

## Tip

If `FW_DEV_MASQ` is set to the external interface you have to set `FW_FORWARD` from internal to DMZ for the service as well to allow access from internal!

Please note that this should **not** be used for security reasons! You are opening a hole to your precious internal network. If e.g. the webserver there is compromised - your full internal network is compromised!!

**Choice:** leave empty (good choice!) or use the following explained syntax of forward masquerade rules, separated each by a space.

A forward masquerade rule consists of:

- source IP/net
- destination IP (dmz/intern)
- a protocol (tcp/udp only!)
- destination port, separated by a comma (",")

, e.g. `4.0.0.0/8,1.1.1.1,tcp,80`

Optional is a port after the destination port, to redirect the request to a different destination port on the destination IP, e.g. `4.0.0.0/8,1.1.1.1,tcp,80,81`

### 15. `FW_REDIRECT`

Which accesses to services should be redirected to a localport on the firewall machine?

This can be used to force all internal users to surf via your squid proxy, or transparently redirect incoming web traffic to a secure webserver.

**Choice:** leave empty or use the following explained syntax of redirecting rules, separated by a space.

A redirecting rule consists of:

- source IP/net
- destination IP/net,
- protocol (tcp or udp)
- original destination port
- local port to redirect the traffic to, separated by a colon.

e.g.: `10.0.0.0/8,0/0,tcp,80,3128 0/0,172.20.1.1,tcp,80,8080`

### 16. `FW_LOG_*`

Which logging level should be enforced?

You can define to log packets which were accepted or denied. You can also the set log level, the critical stuff or everything.

## Note

Note that logging \*\_ALL is only for debugging purpose ...

**Choice:** "yes" or "no", FW\_LOG\_\*\_CRIT defaults to "yes", FW\_LOG\_\*\_ALL defaults to ""no""

- FW\_LOG\_DROP\_CRIT="yes"
- FW\_LOG\_DROP\_ALL="no"
- FW\_LOG\_ACCEPT\_CRIT="yes"
- FW\_LOG\_ACCEPT\_ALL="no"

### FW\_LOG=

```
FW_LOG="--log-level warning --log-tcp-options --log-ip-option --log-prefix SuSE-FW"
```

Only change/activate this if you know what you are doing!. To have a better understanding of SuSEfirewall2 log process see Section 7.1

## 17. FW\_KERNEL\_SECURITY

Do you want to enable additional kernel TCP/IP security features? If set to yes, some obscure kernel options are set.

- icmp\_ignore\_bogus\_error\_responses
- icmp\_echoreply\_rate
- icmp\_destunreach\_rate
- icmp\_paramprob\_rate
- icmp\_timeexceed\_rate
- ip\_local\_port\_range
- log\_martians
- mc\_forwarding
- mc\_forwarding
- rp\_filter
- routing flush

### Kernel Security

This is actually what is happening when you set this option and it could be very problematic in some cases. For example if you have this option turned on at first, but later you change your mind and turn it off since changes have been implemented in the /proc area the results you will be getting from SuSE-firewall2 may not be what you are after

```
test "$FW_KERNEL_SECURITY" = no || {
    echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses 2> /dev/null
    echo 5 > /proc/sys/net/ipv4/icmp_echoreply_rate 2> /dev/null
    echo 5 > /proc/sys/net/ipv4/icmp_destunreach_rate 2> /dev/null
    echo 5 > /proc/sys/net/ipv4/icmp_paramprob_rate 2> /dev/null
    echo 6 > /proc/sys/net/ipv4/icmp_timeexceed_rate 2> /dev/null
    echo 20 > /proc/sys/net/ipv4/ipfrag_time 2> /dev/null
    echo 1 > /proc/sys/net/ipv4/igmp_max_memberships 2> /dev/null
    echo "1024 29999" > /proc/sys/net/ipv4/ip_local_port_range 2> /dev/null
    for i in /proc/sys/net/ipv4/conf/*; do
        echo 1 > $i/log_martians 2> /dev/null
        echo 0 > $i/bootp_relay 2> /dev/null
        test "$FW_ROUTE" = yes || ( echo 0 > $i/forwarding ) > /dev/null 2>>&1
        echo 0 > $i/proxy_arp 2> /dev/null
        echo 1 > $i/secure_redirects 2> /dev/null
    done
}
```

```
done
echo 1 > /proc/sys/net/ipv4/route/flush
}
```

## Tip

Set this to "no" until you have verified that you have got a configuration which works for you. Then set this to "yes" and keep it if everything still works. (It should!) ;-)

**Choice:** "yes" or "no", defaults to "yes"

### 18. FW\_STOP\_KEEP\_ROUTING\_STATE

Keep the routing set on, if the firewall rules are unloaded?

**REQUIRES:** FW\_ROUTE

If you are using diald, or automatic dialing via ISDN, if packets need to be sent to the Internet, you need to turn this on. The script will then not turn off routing and masquerading when stopped.

You *might* also need this if you have got a DMZ.

## Note

Please note that this is **insecure** ! If you unload the rules, but are still connected, you might your internal network open to attacks!

The better solution is to remove `"/sbin/SuSEfirewall2 [stop]"` or `"/sbin/SuSEfirewall2 [stop]"` from the `ip-down` script!

**Choices:** "yes" or "no", defaults to "no"

### 19. FW\_ALLOW\_PING\_\*

Allow (or don't) ICMP echo pings on either the firewall or the dmz from the Internet? The Internet option is for allowing the DMZ and the internal network to ping the Internet.

**REQUIRES:** FW\_ROUTE for FW\_ALLOW\_PING\_DMZ and FW\_ALLOW\_PING\_EXT

**Choice:** "yes" or "no", defaults to "no" if not set

**FW\_ALLOW\_PING\_FW="yes"**

**FW\_ALLOW\_PING\_DMZ="no"**

**FW\_ALLOW\_PING\_EXT="no"**

### 20. FW\_ALLOW\_FW\_TRACEROUTE

Allow (or don't) ICMP `time-to-live-exceeded` to be send from your firewall. This is used for traceroutes to your firewall (or traceroute like tools).

Please note that the unix traceroute only works if you say "yes" to FW\_ALLOW\_INCOMING\_HIGH-PORTS\_UDP, and Windows™ traceroutes only if you say *additionally* "yes" to FW\_ALLOW\_PING\_FW

**Choice:** "yes" or "no", defaults to "no" if not set.

21. **FW\_ALLOW\_FW\_SOURCEQUENCH**

Allow ICMP sourcequench from your ISP?

If set to yes, the firewall will notice when connection is choking, however this opens yourself to a denial of service attack. Choose your poison.

**Choice:** "yes" or "no", defaults to "yes"

22. **FW\_\*\_FW\_BROADCAST**

Allow/Ignore IP Broadcasts?

If set to yes, the firewall will not filter broadcasts by default. This is needed e.g. for Netbios/Samba, RIP, OSPF where the broadcast option is used.

If you do not want to allow them however ignore the annoying log entries, set FW\_IGNORE\_FW\_BROADCAST to "yes".

**Choice:** "yes" or "no", defaults to "no" if not set.

**FW\_ALLOW\_FW\_BROADCAST="no"**

**FW\_IGNORE\_FW\_BROADCAST="yes"**

23. **FW\_ALLOW\_CLASS\_ROUTING**

Allow same class routing per default?

**REQUIRES:** FW\_ROUTE

Do you want to allow routing between interfaces of the same class (e.g. between all Internet interfaces, or all internal network interfaces) be default (so without the need setting up FW\_FORWARD definitions)?

**Choice:** "yes" or "no", defaults to "no"

24. **FW\_CUSTOMRULES**

Do you want to load customary rules from a file?

## Warning

---

This is really an expert option. **NO HELP WILL BE GIVEN FOR THIS! READ THE EXAMPLE CUSTOMARY FILE AT** `/etc/sysconfig/scripts/SuSEfirewall12-custom`

---

**FW\_CUSTOMRULES="/etc/sysconfig/scripts/SuSEfirewall12-custom"**

You will find more information on this topic in Section 6.1.

25. **FW\_REJECT**

Do you want to **REJECT** packets instead of **DROP**ing?

*DROP*ing (which is the default) will make portscans and attacks much slower, as no replies to the packets will be sent. *REJECT*ing means, that for every illegal packet, a connection reject packet is sent to the sender.

**Choice:** "yes" or "no", if not set defaults to "no"

## 2.2. SuSEfirewall2 Command parameters

Although if you have configured SuSEfirewall2 properly, it will start during the system initialization time and you would not need the possible command parameters. In this section you will find the various options `/sbin/SuSEfirewall2` can take and how they can help you.

**Table 2.1. SuSEfirewall2 command options**

Option	Explanation
start	generate and load the firewall filter rules from <code>/etc/sysconfig/SuSEfirewall2</code>
stop	unload all filter rules
easy	set easy filter rules which rejects all incoming access <sup>1</sup>
close	no incoming network traffic except bootp+ping (used for boot security)
file <i>FILENAME</i>	same as <code>start</code> but load alternate config file <i>FILENAME</i>
test	generate and load the filter rules but do not drop any packet but log to syslog anything which <b>would</b> be denied
status	print the output of <code>iptables -L -nv</code> , <code>iptables -nat -L -nv</code> and <code>iptables -t mangle -L -nv</code>
debug	print the iptables command to stdout instead of executing them
help	the output of above options

<sup>a</sup>This is available with version 3.1

### Note

Calling `/sbin/SuSEfirewall2` without any option is the same as the `start` option. The `file FILENAME` option may be used with the `start`, `test` and `debug` options.

### 2.2.1. Configuring SuSEfirewall2 for different rules via cron

One possibly is to make use of cron to load a different configuration file.

For example during night time you may disable access to your proxy so your employees can not use your highbandwidth to download software or MP3's

```
# run at 8 pm to disable proxy and ftp access
0 20 * * * * /sbin/SuSEfirewall2 start file /home/admin/conf/firewall-nightrules
# run at 7.30 am so normal firewall rules apply
30 7 * * * * /sbin/SuSEfirewall2 start
```

<sup>1</sup> <sup>a</sup>This is available with version 3.1

DRAFT

---

# Part II. Configuration

This is the part where we will start configuring SuSEfirewall2. You will see that YaST2 has a new module now for firewall configuration. This is new for SuSE 8.1. It can be used to quickly configure a very basic firewall.

If your needs are above the basic configuration then it is suggested you follow starting with Chapter 4, `/etc/susconfig/SuSEfirewall2`

It is extremely important to understand the background process and the variables used in the SuSEfirewall2 to have a working configuration that best meets the needs of your network environment

DRAFT

DRAFT

# 3

## Configuration

### 3.1. Using the Easy Configuration option

Starting with version 3.1 that comes with SuSE 8.1 (it is also available for 8.0 from [ftp://ftp.gwdg.de/pub/linux/suse/people/garloff/linux/SuSE/RPMS/8.0/SuSEfirewall2-3.1-19.noarch.rpm](http://ftp.gwdg.de/pub/linux/suse/people/garloff/linux/SuSE/RPMS/8.0/SuSEfirewall2-3.1-19.noarch.rpm) ) there is a new option which is called *easy*. This basically makes SuSEfirewall2 to act like the personal-firewall with rejecting all incoming connections.

#### FW\_QUICKMODE

Should the Firewall run in quickmode?

*Quickmode* means that only the interfaces pointing to external networks are secured, and no other. all interfaces not in the list of FW\_DEV\_EXT are allowed full network access! Additionally, masquerading is automatically activated for FW\_MASQ\_DEV devices. and last but not least: all incoming connection via external interfaces are **REJECTED**.

You will only need to configure

- FW\_DEV\_EXT
- FW\_MASQ\_DEV.

Optionally, you may add entries to section FW\_SERVICES\_QUICK\_\*

**Choice:** "yes" or "no", if not set defaults to "no"

```
FW_QUICKMODE="no"
```

#### FW\_SERVICES\_QUICK\_\*

External services in *QUICKMODE*. This is only used for QUICKMODE ! (The settings here are similar to section FW\_SERVICES\_\*. ) Which services **ON THE FIREWALL** should be accessible from either the Internet (or other untrusted networks), i.e. the external interface(s) FW\_DEV\_EXT

Enter all ports or known port names below, separated by a space. TCP services (e.g. *SMTP*, *WWW*) must be set in FW\_SERVICES\_QUICK\_TCP, and UDP services (e.g. *syslog*) must be set in FW\_SERVICES\_QUICK\_UDP. e.g. if a secure shell daemon on the firewall should be accessible from the Internet:

```
FW_SERVICES_QUICK_TCP="ssh"
```

e.g. if the firewall should receive isakmp (IPsec) Internet:

```
FW_SERVICES_QUICK_UDP="isakmp"
```

For IP protocols (like IPsec) you need to set

```
FW_SERVICES_QUICK_IP=" 50 "
```

**Choice:** leave empty or any number of ports, known port names (from `/etc/services`) and port ranges separated by a space. Port ranges are written like this: allow port 1 to 10 -> "1:10" e.g. "", "smtp", "123 514", "3200:3299", "ftp 22 telnet 512:514" For `FW_SERVICES_QUICK_IP` enter the protocol name (like "igmp") or number ("2")

```
QUICKMODE: TCP services open to external networks (Internet)

# (Common: ssh smtp)
FW_SERVICES_QUICK_TCP=""

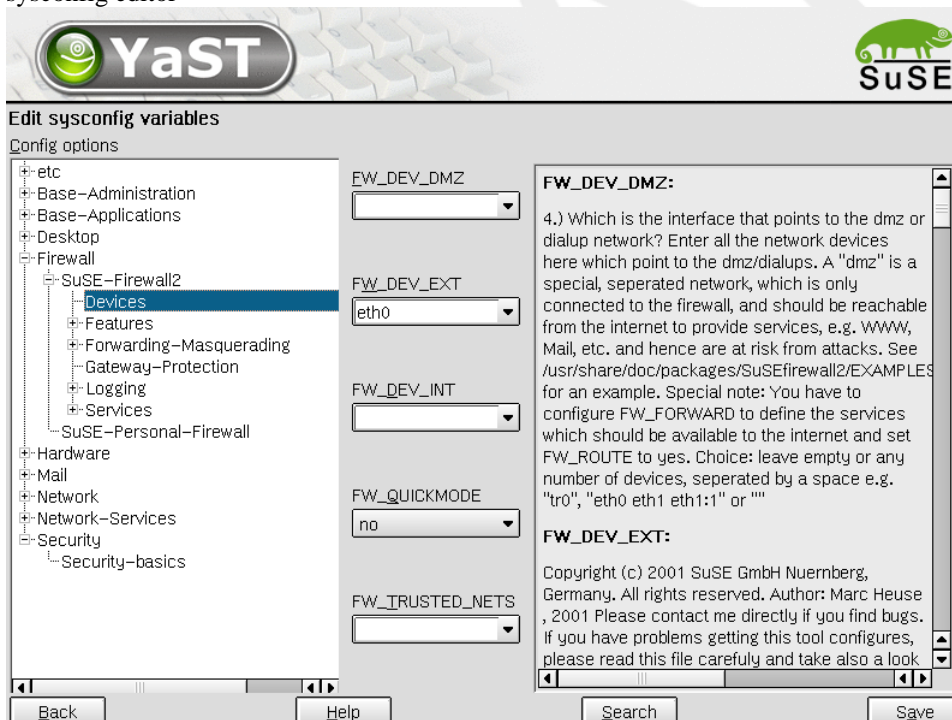
# QUICKMODE: UDP services open to external networks (Internet)
# (Common: isakmp)
FW_SERVICES_QUICK_UDP=""

# QUICKMODE: IP protocols unconditionally open to external networks (Internet)
# (For VPN firewall that is VPN gateway: 50)
FW_SERVICES_QUICK_IP=""
```

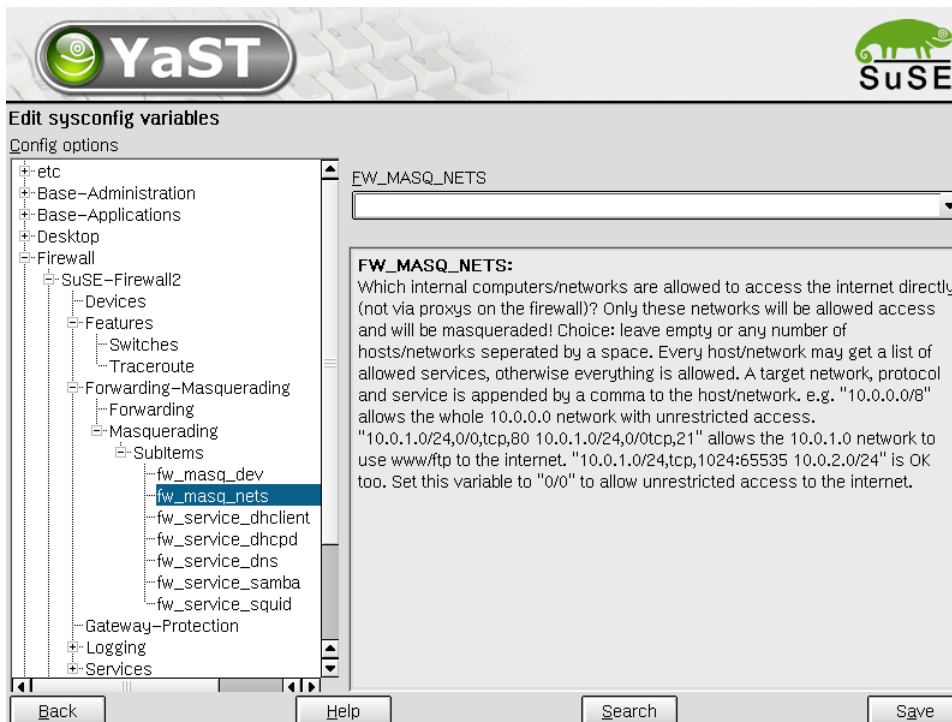
## 3.2. Configuring SuSEfirewall2 with YaST2

For those who would like to configure the firewall parameters via GUI interface there exists two choices

- sysconfig editor



By using sysconfig editor you will be able to reach the `/etc/sysconfig/SuSEfirewall2` and make your configuration there.



It should be noted that this option does not list the items in the order they are listed in the file. Although it can be easier for many people to use sysconfig editor in configuring the firewall

- YaST2 firewall module is basically enables configuration of the `QUICK_MODE` parameters via graphical user interface. It will provide a quick and convenient configuration for a very basic setup.

In this section we will have a look to the second option using YaST2 firewall module

### Procedure 3.1. Step by step procedures for using the YaST2 firewall Module

1. **Starting YaST2**
  - a. Start YaST2



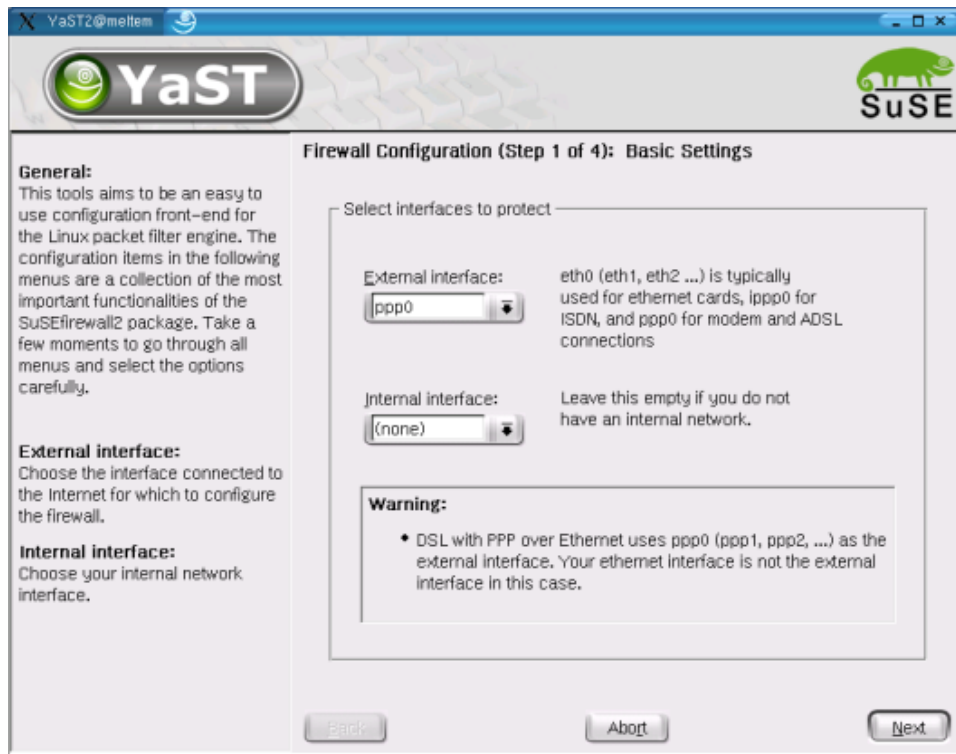
- b. From the left menu choose Security and Users where you will notice on the right hand menu there is a Firewall. Select Firewall



## 2. Choosing the interfaces

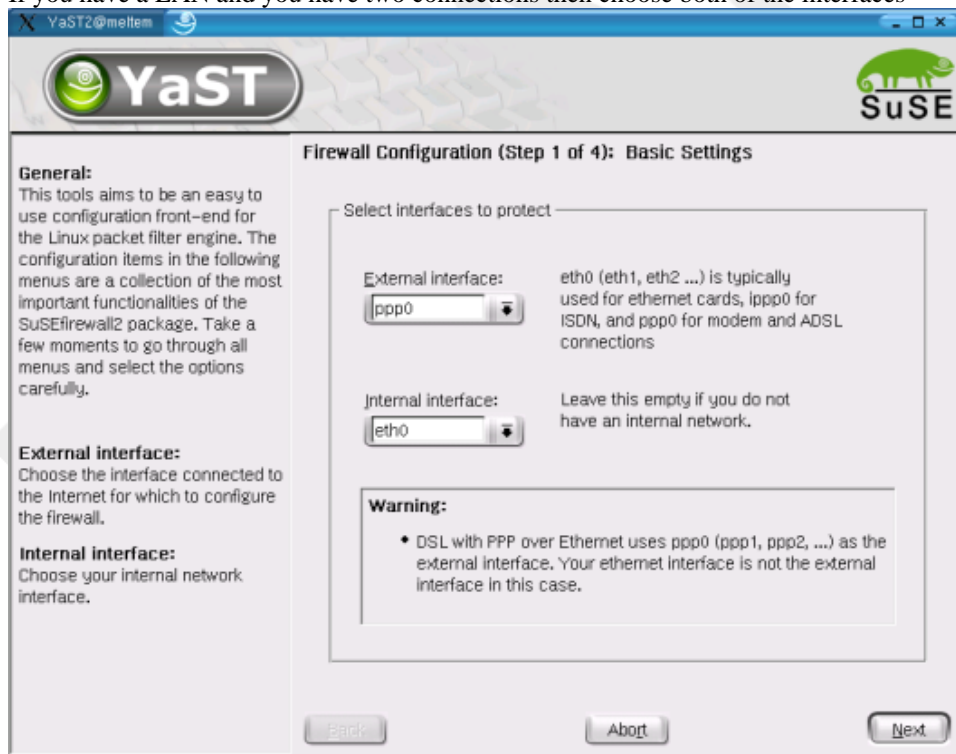
### Single interface

If you are connected via dialup or ADSL (PPPOE) choose ppp0

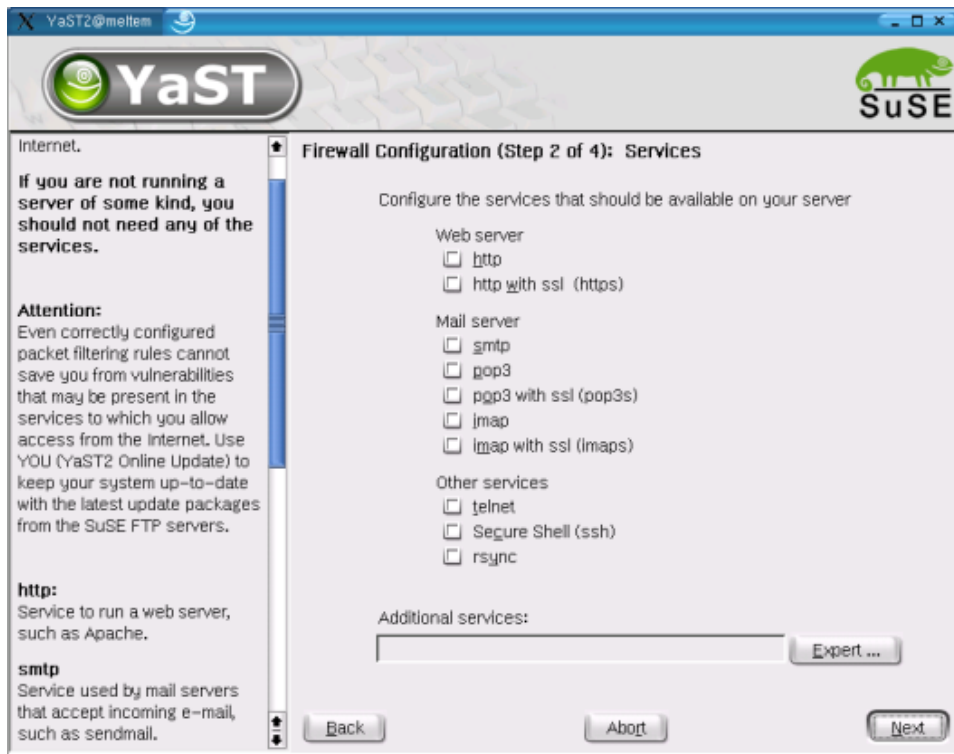


### Two interface

If you have a LAN and you have two connections then choose both of the interfaces



### 3. Choosing the services



- a. If you need to add additional services that you would like to offer then

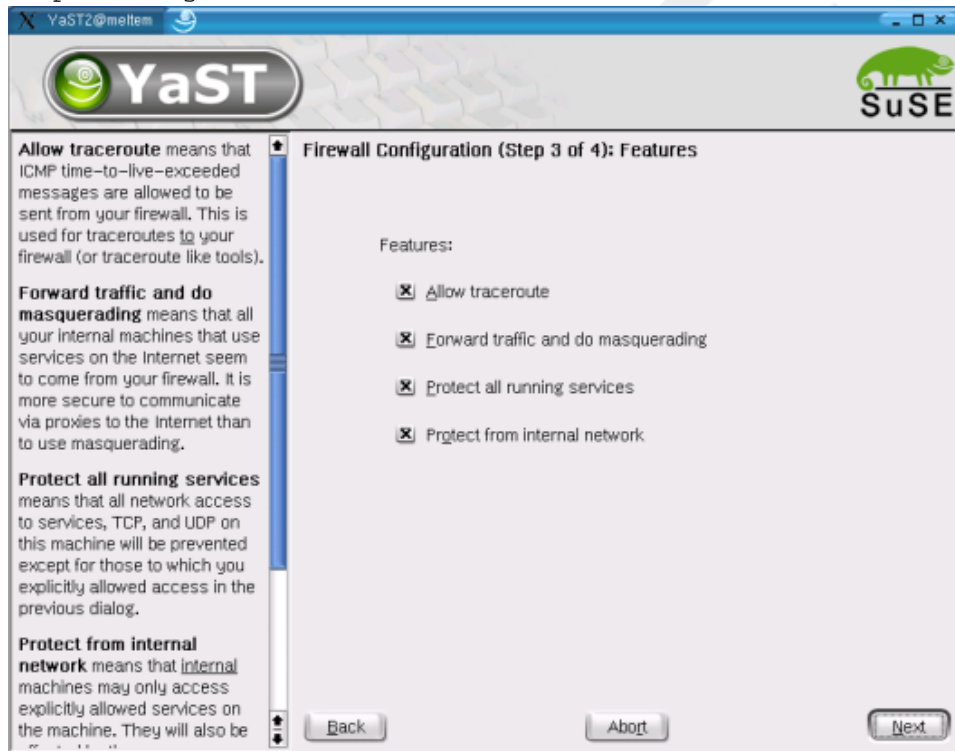


- b. For example if your ISP is blocking access to port 80 and you really want to serve your own web pages you can define a different port like 81 and have your friends know it as `http://your.server.com:81`



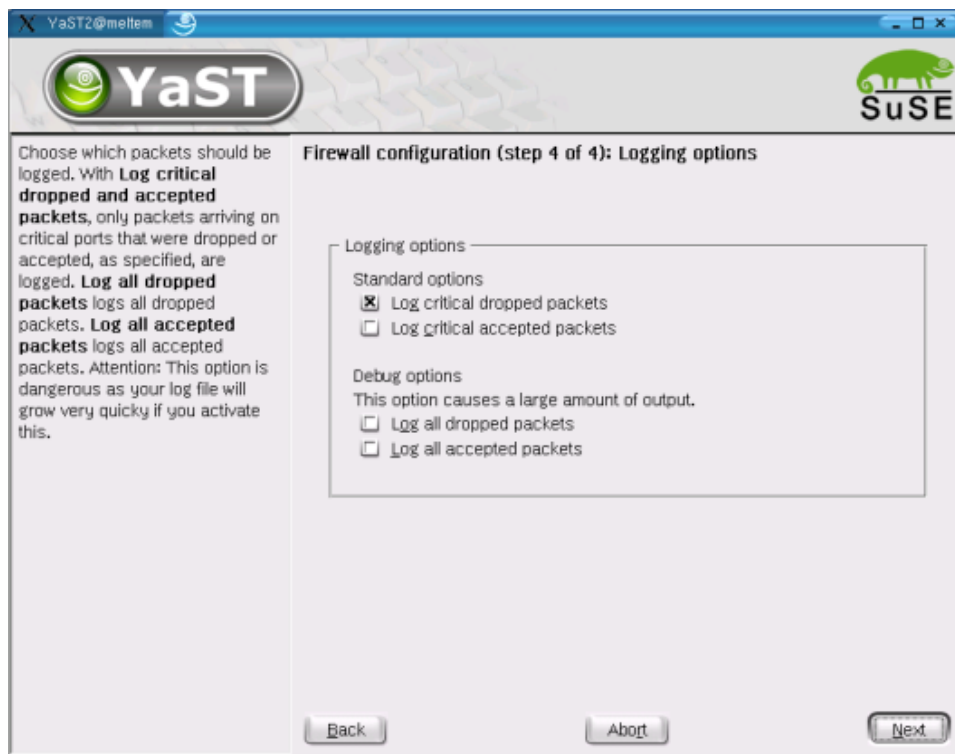
#### 4. Additional features

In this step you will define if features like traceroute masquerading Auto protecting services and protecting from internal network



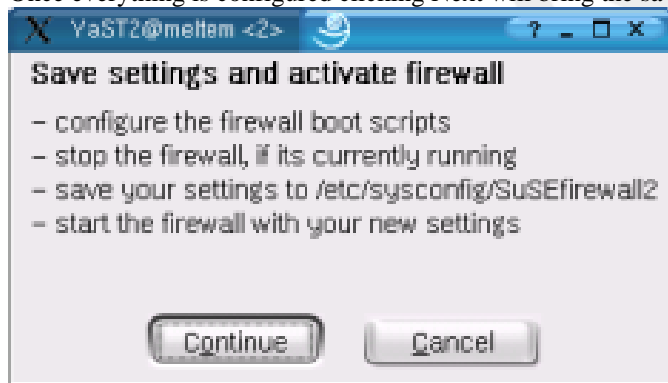
#### 5. Logging

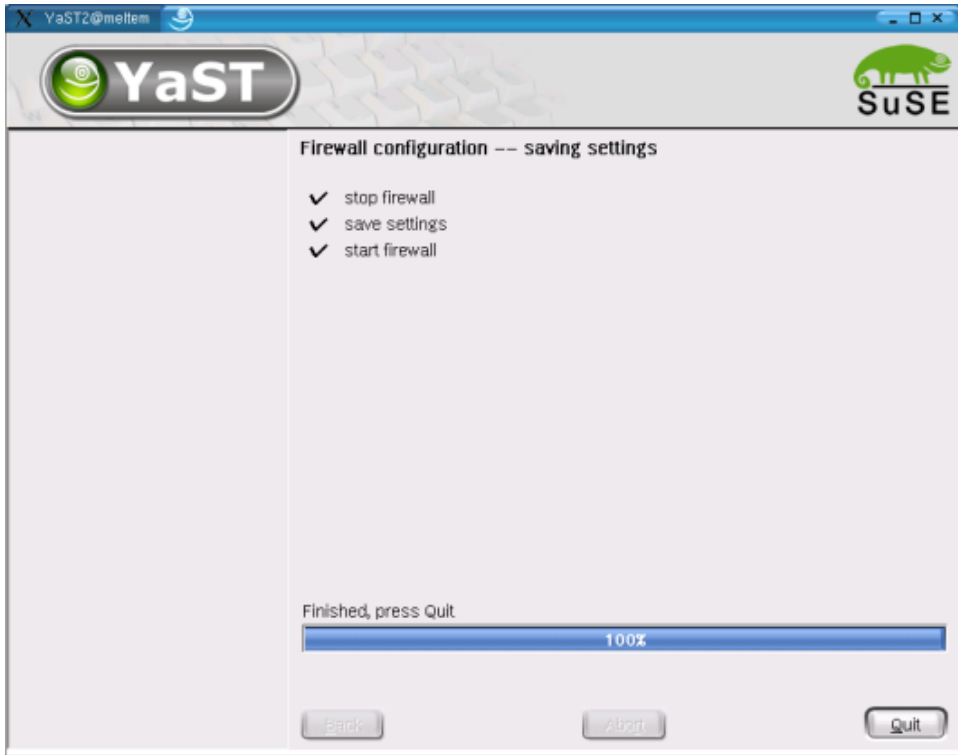
Now choose the logging options



## 6. Finishing up

Once everything is configured clicking Next will bring the save settings window





DRAFT

# 4

## Editing The Configuration file

### 4.1. Basic Configuration of SuSEFirewall2

This section will try to explain the basics of configuring `/etc/sysconfig/SuSEfirewall2`. Please consult the file as this page covers very basic details.

#### 4.1.1. Single user

If you are a end-user who is NOT connected to two networks (read: you have got a single user system and are using a dialup or cable modem to the internet and you are not offering any services to the world) you just have to configure the following (all other settings are OK): `FW_DEV_EXT`.

**`FW_DEV_EXT=""`**

Which is the interface that points to the internet/untrusted networks?

If you have a dialup connection or ADSL with PPPOE then your internet connection is Point to Point Protocol and you have `ppp+` device. For ISDN users the device is `ipp`

```
FW_DEV_EXT="ppp+"
```

If you have cable modem or direct Ethernet connection then you need to have your device as `ethX` where X is the ethernet number

```
FW_DEV_EXT="ethX"
```

#### Note

When you have PPPOE although you are connecting the ethernet device to the ADSL modem you are actually conneted to the Internet via virtual device `ppp` so ADSL users who have PPPOE connections must use `ppp+` as their connecting device

**`FW_STOP_KEEP_ROUTING_STATE="no"`**

If you are using diald, or automatic dialing via ISDN, if packets need to be sent to the internet, you need to turn this on. The script will then not turn off routing and masquerading when stopped.

Choices "yes" or "no", defaults to "no"

```
FW_STOP_KEEP_ROUTING_STATE="yes"
```

## 4.1.2. Examples for basic configuration

### Example 4.1. Single User ADSL

A User with his nice SuSE Linux PC wants to be protected when connected to the internet via the ADSL dialup of his ISP. He wants to offer NO services to the internet. He is NOT connected to any other network, nor are any other network cards active.

```
FW_DEV_EXT="ppp0" # this is the adsl interface, which is using pppoe
```

### Example 4.2. Single User ISDN

A User with his nice SuSE Linux PC wants to be protected when connected to the internet via the ISDN dialup of his ISP. He has *dial on demand* configuration He wants to offer NO services to the internet. He is NOT connected to any other network, nor are any other network cards active.

```
FW_DEV_EXT="ipp0" # this is the isdn interface
FW_STOP_KEEP_ROUTING_STATE="yes"
```

## 4.2. Proxy masquerading

Proxy services are specialized application or server programs that run on a firewall host: either a dual-homed host with an interface on the internal network and one on the external network, or some other bastion host that has access to the Internet and is accessible from the internal machines. These programs take users' requests for Internet services (such as FTP and HTTP) and forward them, as appropriate according to the site's security policy, to the actual services. The proxies provide replacement connections and act as gateways to the services. For this reason, proxies are sometimes known as application-level gateways.

Dual-homed host is a general-purpose computer system that has at least two network interfaces (or homes)

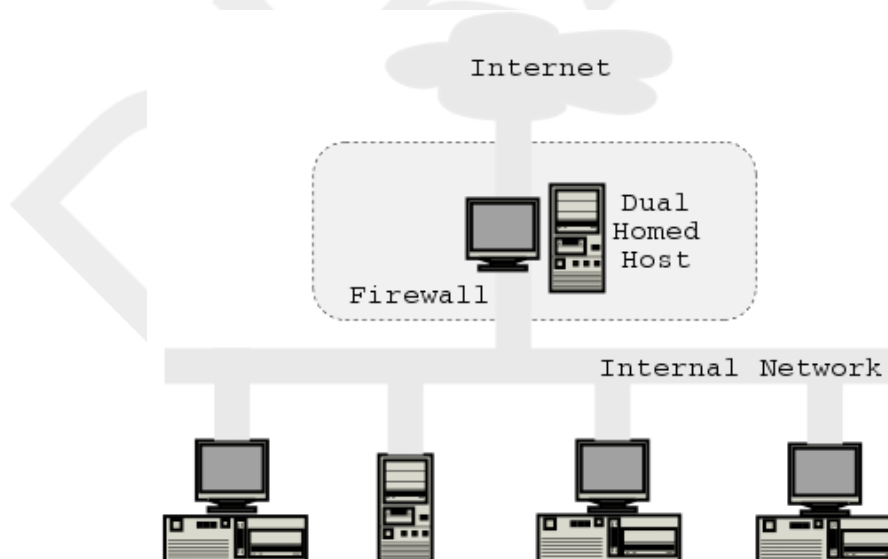
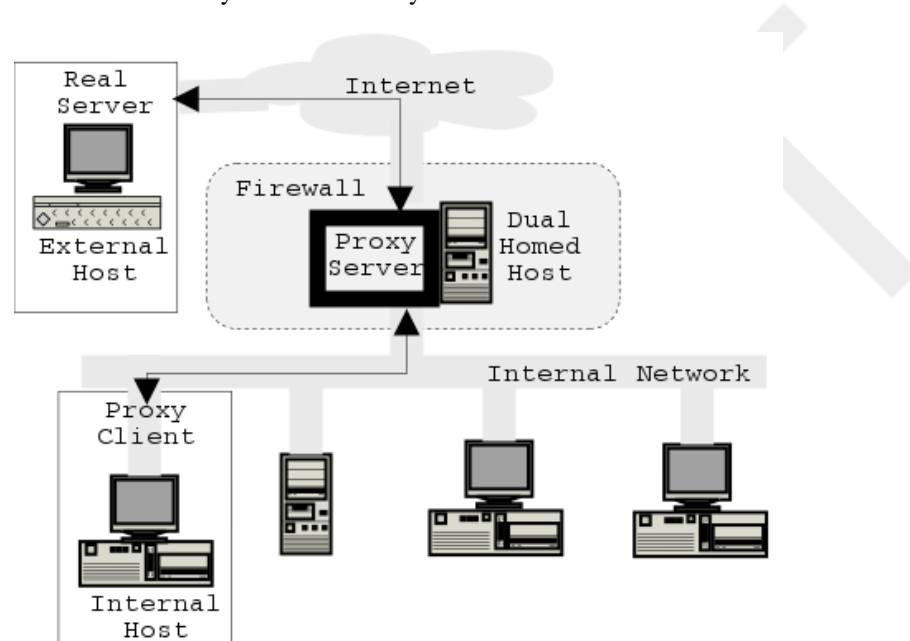


Figure 4.1. Dual-homed Host

Proxy services sit, more or less transparently, between a user on the inside (on the internal network) and a service on the outside (on the Internet). Instead of talking to each other directly, each talks to a proxy. Proxies handle all the communication between users and Internet services behind the scenes.

Transparency is the major benefit of proxy services. It's essentially smoke and mirrors. To the user, a proxy server presents the illusion that the user is dealing directly with the real server. To the real server, the proxy server presents the illusion that the real server is dealing directly with a user on the proxy host (as opposed to the user's real host).

As Figure 4.2 shows, a proxy service requires two components: a proxy server and a proxy client. In this situation, the proxy server runs on the dual-homed host. A proxy client is a special version of a normal client program (i.e., a Telnet or FTP client) that talks to the proxy server rather than to the "real" server out on the Internet; in addition, if users are taught special procedures to follow, normal client programs can often be used as proxy clients. The proxy server evaluates requests from the proxy client, and decides which to approve and which to deny. If a request is approved, the proxy server contacts the real server on behalf of the client (thus the term "proxy"), and proceeds to relay requests from the proxy client to the real server, and responses from the real server to the proxy client. Almost all web browsers have the ability to work as Proxy client



**Figure 4.2. Using Proxy Services**

The proxy server doesn't always just forward users' requests on to the real Internet services. The proxy server can control what users do, because it can make decisions about the requests it processes. Depending on your site's security policy, requests might be allowed or refused. For example, the HTTP proxy might allow users to surf only certain sites. More sophisticated proxy services might allow different capabilities to different hosts, rather than enforcing the same restrictions on all hosts.

If this server is a firewall, which should act like a proxy (no direct routing between both networks), or you are an end-user connected to the Internet and to an internal network, you have to setup your proxies and reconfigure (all other settings are OK): `FW_DEV_EXT`, `FW_DEV_INT`, `FW_EXT_SERVICES_*_*` and maybe `FW_PROTECT_INTERNAL`, `FW_ALLOW_INCOMING_HIGH`, `FW_FORWARD_MASQ`

**FW\_DEV\_INT=""**

Which is the interface that points to the internal network?

Enter all the network devices here which are trusted. If you are not connected to a trusted network (e.g. you have just a dialup) leave this empty.

```
FW_DEV_INT= " "
```

If you have a LAN and other PC's are connected to the *firewall PC* then enter the device for the LAN connection. Again you can enter multiple variables separated with space if that is your case.

```
FW_DEV_INT="eth1"
```

## Note

You can also enter virtual interfaces ie. `eth0:1`

**FW\_PROTECT\_FROM\_INTERNAL=""**

Do you want to protect the firewall from the internal network?

## Warning

**REQUIRES:** FW\_DEV\_INT

If you set this to *yes*, internal machines may only access services on the machine you explicitly allow. They will be also affected from the FW\_AUTOPROTECT\_SERVICES option.

## Tip

The way you set FW\_AUTOPROTECT\_SERVICES may require additional configurations.

For instance if you have chosen **"yes"** for both of the parameters, the you have to state the services that are running on the firewall which local users may access. For that you will need to enter the necessary ports to FW\_SERVICES\_INT\_TCP and FW\_SERVICES\_INT\_UDP and maybe FW\_SERVICES\_INT\_IP

One thing to make clear is the services mentioned here are like SSH server or SMTP server running on the *firewall machine*

If you set this to *no*, any user can connect (and attack) any service on the firewall.

Choice: "yes" or "no", defaults to "yes"

```
FW_PROTECT_FROM_INTERNAL="yes" # "yes" is a good choice
```

**FW\_ALLOW\_INCOMING\_HIGHPORTS**

How is access allowed to high (unprivileged [above 1023]) ports?

You may either allow everyone from any port access to your high ports ("yes"), disallow anyone ("no"), anyone who comes from a defined port (port number or known port name) [note that this is easy to circumvent!], or just your defined nameservers ("DNS").

## Note

Note that if you want to use normal (active) ftp, you have to set the TCP option to **ftp-data**. If you use passive ftp, you don't need that. Note that you can't use rpc requests (e.g. rpcinfo, showmount) as root from a firewall using this script (well, you can if you include range 600:1023 in FW\_SERVICES\_EXT\_UDP ...).

Choice: "yes", "no", "DNS", port number or known port name, defaults to "no" if not set

```
FW_ALLOW_INCOMING_HIGHPORTS_TCP="no" # Common: "ftp-data"
FW_ALLOW_INCOMING_HIGHPORTS_UDP="DNS" # Common: "DNS" or "domain ntp"
```

**FW\_FORWARD\_MASQ=**

Which services accessed from the Internet should be allowed to masqueraded servers (on the internal network or dmz)?

**Warning**

**REQUIRES: FW\_ROUTE**

With this option you may allow access to e.g. your mail server. The machines must be in a masqueraded segment and may not have public IP addresses!

**Tip**

If `dev_masq` is set to the external interface you have to set `FW_FORWARD` from internal to DMZ for the service as well to allow access from internal!

**Caution**

Please note that this should **not** be used for security reasons! You are opening a hole to your precious internal network. If e.g. the web server there is compromised - your full internal network is compromised!!

Choice: leave empty (good choice!) or use the following explained syntax of forward masquerade rules, separated each by a space. A forward masquerade rule consists of

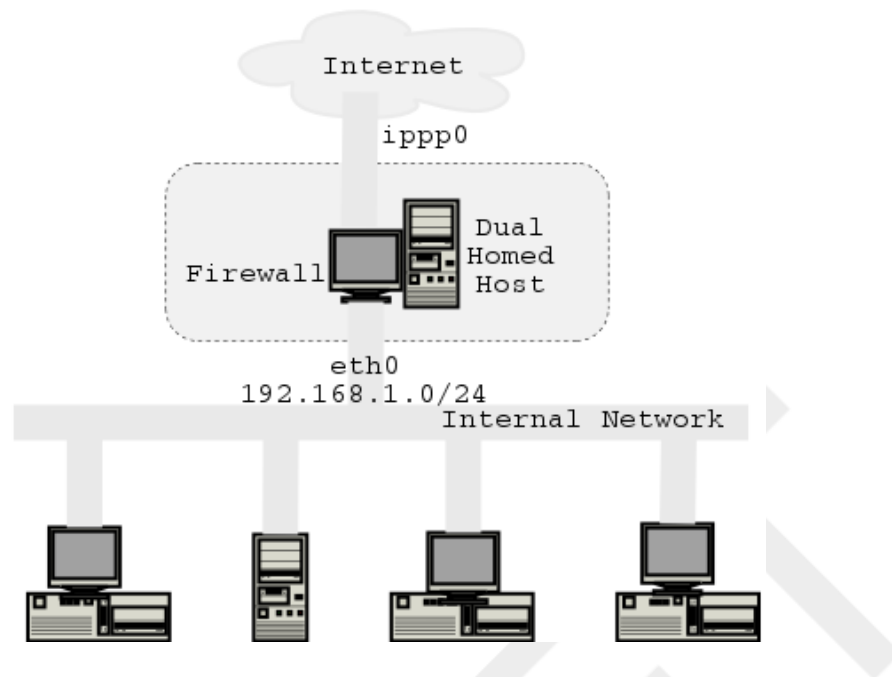
1. source IP/net
2. destination IP(dmz/intern)
3. a protocol (tcp/udp only!)
4. destination port, separated by a comma (","), e.g. **4.0.0.0/8,1.1.1.1,tcp,80** Optional is a port after the destination port, to redirect the request to a different destination port on the destination IP, e.g. **"4.0.0.0/8,1.1.1.1,tcp,80,81"**

```
FW_FORWARD_MASQ="0/0,10.0.1.2,tcp,80" # Beware to use this!
```

## 4.2.1. Examples for Proxy Configuration

### Example 4.3. Proxy Masquerading with ISDN

A company uses it's SuSE Linux PC to access the Internet via an ISDN dialup of it's ISP.



**Figure 4.3. Proxy Masquearading**

It has got a web server running on the PC plus it's the mail-/pop3-server for the company. Squid is running to cache WWW traffic. No internal PC should have direct access to the Internet.

The network address of the internal LAN is 192.168.1.0 netmask 255.255.255.0

**TODO:** users have to configure their mail server, pop3server and DNS to the IP of the firewall, and their web client software to use the firewall on port 3128.

**TODO:** The services mail, squid and pop3 have to be set up (securely).

```
FW_DEV_EXT="ipp0"
FW_DEV_INT="eth0"
FW_ROUTE="yes"
FW_SERVICES_EXT_TCP="25 80"
FW_SERVICES_INT_TCP="25 53 80 110 3128"
FW_SERVICES_INT_UDP="53"
FW_SERVICE_DNS="yes"
FW_STOP_KEEP_ROUTING_STATE="yes"
```

## 4.3. Firewall Masquerading

If this server is a firewall, and should do routing/masquerading between the untrusted and the trusted network, you have to reconfigure (all other settings are OK): FW\_DEV\_EXT, FW\_DEV\_INT, FW\_ROUTE, FW\_MASQUERADE, FW\_MASQ\_NETS, FW\_SERVICES

If you have applications running on your system that are providing services to the Internet (for example webserver) then you have to configure also the following

**FW\_SERVICES\_EXT\_TCP=""**

Enter all ports or known port names, separated by a space. Please note that if you use service names, that they exist in `/etc/services`. There is no service dns, it's called domain; email is called smtp etc. TCP services (e.g. SMTP, WWW) must be set in FW\_SERVICES\_\*\_TCP e.g. if a webserver on the firewall should be accessible from the Internet:

```
FW_SERVICES_EXT_TCP="www"
```

or if a webserver on the firewall should be reachable along with SSH you have to edit like

```
FW_SERVICES_EXT_TCP="www ssh"
```

### Note

This means the services you will be listing here are the actual services running on the firewall. If you want to use services that are available on other sites ie ftp.suse.com **do not** enter 21 here thinking you are providing access permission to your local LAN to access the ftp servers which are located on the Internet.

Any service type and port should be actually running on the firewall machine

#### FW\_SERVICES\_EXT\_UDP

Enter all ports or known port names, separated by a space. UDP services (e.g. domain) must be set in FW\_SERVICES\_EXT\_UDP

```
FW_SERVICES_EXT_UDP="53 123 " # Common: domain
```

## 4.3.1. Examples for Masquerading firewall configuration

### Example 4.4. Masquerading Only

A small university unit wants to use masquerading to access the Internet directly but wants the client PCs not directly reachable from the outside (and hence provide limited protection through this). The Firewall provides no services whatsoever.

- external fw interface=eth1
- internal fw interface=eth0
- internal LAN: 192.168.10.0/24

```
FW_DEV_EXT="eth1"  
FW_DEV_INT="eth0"  
FW_ROUTE="yes"  
FW_MASQUERADE="yes"  
FW_MASQ_NETS="192.168.10.0/24"
```

## 4.4. Configuring the SuSEfirewall2 for using DMZ

If you want to run a DMZ in either of the above three standard setups, you just have to configure **additionally**

- FW\_DEV\_DMZ
- FW\_SERVICES\_DMZ
- FW\_SERVICES\_AUTODETECT
- FW\_FORWARD
- FW\_KERNEL\_SECURITY

- `FW_ALLOW_PING_*`

The very first variable you need to set is `FW_DEV_DMZ`. Once that is set you need to define the services that DMZ will be using that are on the **firewall** machine.

## Tip

When you are providing the service on the DMZ you do not enter the same service on the `FW_SERVICES_EXT_*` and more important you do not define them here as `FW_SERVICES_DMZ_*`. The services you provide on the DMZ machine are either defined in `FW_FORWARD` or `FW_FORWARD_MASQ` depending on your use of public or private IP

### 4.4.1. Setting up Services for DMZ

This is where you define the services that DMZ network is allowed to access on the firewall machine. It could be mailserver or a name server.

- **FW\_SERVICES\_DMZ\_TCP**

Enter the services the DMZ server is allowed to use on the **firewall**. For example to provide access **from DMZ** to to your mailserver and SSH server **on the firewall** you may enter

```
FW_SERVICES_DMZ_TCP="22 25"
```

- **FW\_SERVICES\_DMZ\_UDP**

Enter the services the DMZ server is allowed to use on the **firewall** machine. For example to provide access to DNS and syslog server you may enter

```
FW_SERVICES_DMZ_UDP="domain syslog"
```

- **FW\_SERVICES\_DMZ\_IP**

For IP protocols (like GRE for PPTP, or OSPF for routing)

```
FW_SERVICES_DMZ_IP=" "
```

### 4.4.2. Services Autoprotect

Actually Autoprotect services feature is a complicated part of the SuSEfirewall2 even says so the author.

"... these long lines of code try to identify which services were allowed via the config file (including the DNS port for UDP) and which others are open which have to be protected. This could even be more optimized by resolving name->number and just protecting ports > 1023. I know it looks ... weird ... but it works! ;-)"

```
test "$FW_AUTOPROTECT_SERVICES" = no || {
  PROTECT_GLOBAL=`$NETSTAT -an | \
  $GREP -E '^tcp .* 0.0.0.0:[1-9].*LISTEN|^tcp .* :::[1-9].*LISTEN' | \
  $AWK '{print $4}' | $SED 's/.*/\`
  for IP in $DEV_EXT; do
    PROTECT_EXT="$PROTECT_EXT ` $NETSTAT -an | \
    $AWK '/^tcp .* "$IP":[1-9].*LISTEN/ {print $4}' | $SED 's/.*/\`"
  done
  for IP in $DEV_DMZ; do
    PROTECT_DMZ="$PROTECT_DMZ ` $NETSTAT -an | \
```

```

    $AWK '/^tcp .* "$IP":[1-9].*LISTEN/ {print $4}' | $SED 's/.*///'\`"
done
test "$FW_PROTECT_FROM_INTERNAL" = no || {
  for IP in $DEV_INT; do
    PROTECT_INT="$PROTECT_INT `NETSTAT -an | \
    $AWK '/^tcp .* "$IP":[1-9].*LISTEN/ {print $4}' | $SED 's/.*///'\`"
done
PROTECT=`for S in $PROTECT_INT $PROTECT_GLOBAL; do echo $S; done | $SORT -n`
OPENED_INT=`echo "$FW_SERVICES_INT_TCP" | $SED 's/ //g'\`
PROTECT_INT=`for S in $PROTECT; do echo $S; done | grep -Evw "$OPENED_INT"`
for PORT in $PROTECT_INT; do
  test -z "$LDC" -o -z "$LDA" && $IPTABLES -A input_int -j LOG ${LOG}"-DROP
" -p tcp --dport $PORT --syn
  $IPTABLES -A input_int -j "$DROP" -p tcp --dport $PORT --syn
done
}
PROTECT=`for S in $PROTECT_DMZ $PROTECT_GLOBAL; do echo $S; done | $SORT -n`
OPENED_DMZ=`echo "$FW_SERVICES_DMZ_TCP" | $SED 's/ //g'\`
PROTECT_DMZ=`for S in $PROTECT; do echo $S; done | grep -Evw "$OPENED_DMZ"`
for PORT in $PROTECT_DMZ; do
  test -z "$LDC" -o -z "$LDA" && $IPTABLES -A input_dmz -j LOG ${LOG}"-DROP
" -p tcp --dport $PORT --syn
  $IPTABLES -A input_dmz -j "$DROP" -p tcp --dport $PORT --syn
done
}
PROTECT=`for S in $PROTECT_EXT $PROTECT_GLOBAL; do echo $S; done | $SORT -n`
OPENED_EXT=`echo "$FW_SERVICES_EXT_TCP" | $SED 's/ //g'\`
PROTECT_EXT=`for S in $PROTECT; do echo $S; done | grep -Evw "$OPENED_EXT"`
for PORT in $PROTECT_EXT; do
  test -z "$LDC" -o -z "$LDA" && $IPTABLES -A input_ext -j LOG ${LOG}"-DROP
" -p tcp --dport $PORT --syn
  $IPTABLES -A input_ext -j "$DROP" -p tcp --dport $PORT --syn
done
}
}

```

This basically searches for the listening ports on the server using netstat and then using grep and awk and sed identifies the ports to protect

```

test "$FW_SERVICE_SQUID" = yes -o "$START_SQUID" = yes && SQUID_PORT=`LSOF -i -n
-P | \
  $GREP '^squid .* UDP \*:.' | $AWK -F: '{print $2}' | \
  $AWK '{print $1}' | $SORT -un`

```

Here we also see the use of lsof in identification of the port used

```

test "$FW_SERVICE_SQUID" = yes && {
  test -z "$SQUID_PORT" && \
  echo 'Warning: FW_SERVICE_SQUID defined, but no Squid server found running!'
  test -z "$SQUID_PORT" || {
    for PORT in $SQUID_PORT; do
      for CHAIN in input_int input_dmz input_ext; do
        $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p udp --dport $PORT
        $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW,ESTABLISHED,RELATED
      -p udp --dport $PORT
      done
    done
  }
}

```

### 4.4.3. FW\_FORWARD

Since with this option you may allow access to e.g. your mailserver. The machines must have valid, non-private, IP addresses which were assigned to you by your ISP. This opens a direct link to your network, so only use this option for access to your dmz!!!!

Lets say your web server has got an official IP address of 1.1.1.1 which you received from your ISP. You would just configure FW\_FORWARD\_TCP like this:

```
FW_FORWARD="0/0,1.1.1.1,tcp,80"
```

In the case you have only one official IP address thats your external Firewall IP address and you have got a web-server with a private ip placed in the dmz, you can use backward masquerading.

For this you need to set FW\_ROUTE and FW\_MASQUERADE to **"yes"**, and additionally FW\_FORWARD\_MASQ for the web servers private IP (lets say it is 10.0.0.1):

```
FW_FORWARD_MASQ="0/0,10.0.0.1,tcp,80"
```

### 4.4.4. Kernel security options

Due to the fact that when you enable this option after you disable it the results may be not exactly what you want. It is a good idea to have this option to be disabled until you are sure that everything works.

```
FW_KERNEL_SECURITY="no"
```

#### Note

In order to refresh what happens when enabled please refer to Kernel Security

### 4.4.5. Routing state

You *might* also need to enable FW\_KEEP\_ROUTING\_STATE got a DMZ.

This is defined in the functions part of the SuSEfirewall2

```
function reset_rules() {
    echo -n "SuSEfirewall2: clearing rules now ..."
    test "$FW_STOP_KEEP_ROUTING_STATE" = "yes" || (
        echo 0 > /proc/sys/net/ipv4/ip_forward
    ) > /dev/null 2>&1
    ....
}
```

```
FW_STOP_KEEP_ROUTING_STATE="yes"
```

### 4.4.6. Ping

Allow (or don't) ICMP echo pings on the dmz from the Internet? In order to have this parameter to work FW\_ROUTE should have a value of **"yes"**

```
# FORWARD ICMP rules
test "$FW_ALLOW_PING_DMZ" = yes -a "$FW_ROUTE" = yes && {
  for DEV in $FW_DEV_DMZ; do
    for CHAIN in forward_ext forward_int; do
      $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-
type echo-request -o $DEV
      $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW -p icmp --icmp-
type echo-request -o $DEV
    done
  done
  $LAA $IPTABLES -A forward_dmz -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-type
echo-reply
  $IPTABLES -A forward_dmz -j "$ACCEPT" -m state --state ESTABLISHED -p icmp -
-icmp-type echo-reply
}
```

```
FW_ALLOW_PING_DMZ="no"
```

### Tip

During the configuration of the firewall and the network it is helpful to have ping working so you may set this to "yes" during this phase and once sure everything works can change it to "no"

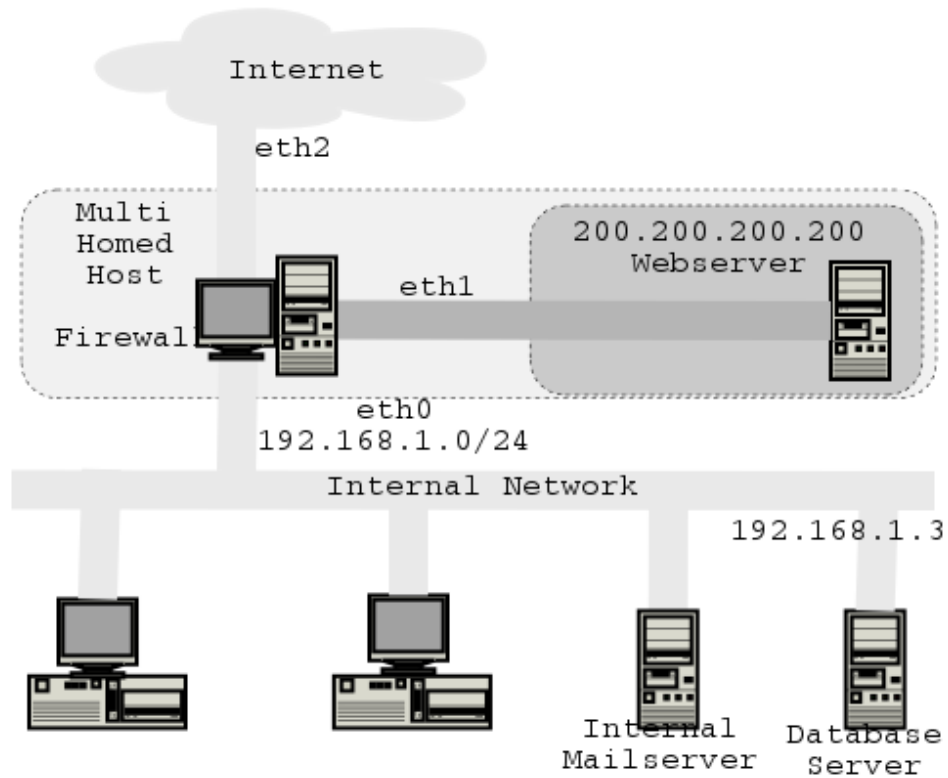
## 4.4.7. Example for DMZ configuration

### Example 4.5. DMZ configuration

As shown in Figure 4.4 There's a DMZ where a Web server resides (port 80 and port 443) which needs to connect to the Firewall to deliver mail to internal, send syslog messages and do domain lookups. It needs also direct access to the internal database (bad idea!).

All mail which is delivered to the firewall, is sent to the internal mail server. The mail server sends all mail to the Internet to the firewall. Internal PCs which access the Internet should be masqueraded.

All Mail is delivered to the firewall. It also provides DNS service to internal and external.



**Figure 4.4. Sample DMZ**

```
external fw interface: eth2
dmz fw interface: eth1
internal fw interface: eth0
ip of database: 192.168.1.3, tcp port for database is 4545
ip of web server: 200.200.200.200 (this is an official, assigned address!)
internal LAN: 192.168.1.0 netmask 255.255.255.0
```

**TODO:** the name server on the firewall needs to be setup "*split-brained*". See the DNS How-to. The mail server on the firewall needs to be setup as a forwarder/relay. The mail server on the internal network gets the firewall as forwarder/relay configured.

```
FW_DEV_EXT="eth2"
FW_DEV_INT="eth0"
FW_DEV_DMZ="eth1"
FW_ROUTE="yes"
FW_MASQUERADE="yes"
FW_MASQ_NETS="192.168.1.0/24"
FW_SERVICES_EXT_TCP="25 53"
FW_SERVICES_EXT_UDP="53"
FW_SERVICES_DMZ_TCP="25 53"
FW_SERVICES_DMZ_UDP="53 514"
FW_SERVICES_INT_TCP="25 53"
FW_SERVICES_INT_UDP="53"
FW_SERVICE_DNS="yes"
FW_FORWARD="0/0,200.200.200.200,tcp,80 0/0,200.200.200.200,tcp,443 \
  200.200.200.200,192.168.1.3,tcp,4545" # access to the web server and allow
  # access from the web server to the database
FW_REDIRECT="192.168.1.0/24,0/0,tcp,53,53 192.168.1.0/24,0/0,tcp,25,25" # all
  # DNS and mail is done by the firewall
FW_REDIRECT="192.168.1.0/24,0/0,udp,53,53" # all DNS is done by the firewall
```

```
FW_ALLOW_PING_DMZ="yes"
```

## **Tip**

the redirect statements here are gimmicks to show how to use it. in this example they send \*any\* traffic from the internal network, which go via the firewall and a are destined to a target port of 53 (DNS) or 25 (Mail) to the local servers on the firewall.

DRAFT

DRAFT

# 5

## Expert Level Configuration

### 5.1. Editing the Expert Options

If you know what you are doing, you may also change

- FW\_AUTOPROTECT\_SERVICES
- FW\_ALLOW\_HIGHPORTS\_\*
- FW\_REDIRECT
- FW\_LOG\_\*

and the expert options at the far end, but you should **NOT**.

- FW\_ALLOW\_PING\_\*
- FW\_TRACEROUTE
- FW\_ALLOW\_FW\_SOURCEQUENCH
- FW\_\*\_FW\_BROADCAST
- FW\_ALLOW\_CLASS\_ROUTING

#### 5.1.1. How FW\_AUTOPROTECT\_SERVICES works

Basically the easiest way for configuring SuSEfirewall2 is to have this option set to "yes". If there are services that the internal LAN needs to reach than you have to specify them on FW\_SERVICES\_\*\_\* parameter

You also need to adjust the services FW\_SERVICE\_AUTODETECT accordingly. SuSEfirewall2 will check these adjust them in the memory and give you a warning message so you can correct your configuration

```
Warning: detected activated named, enabling FW_SERVICE_DNS!  
You still have to allow tcp/udp port 53 on internal, dmz and/or external.
```

```
test "$FW_SERVICE_AUTODETECT" = yes && {  
    test "$FW_SERVICE_DNS" = no && check_srv named && {  
        echo -e 'Warning: detected activated named, enabling FW_SERVICE_DNS!  
You still have to allow tcp/udp port 53 on internal, dmz and/or external.'  
        FW_SERVICE_DNS=yes  
        FW_ALLOW_INCOMING_HIGHPORTS_UDP=yes  
    }  
    test "$FW_SERVICE_SQUID" = no && check_srv squid && {  
        echo -e 'Warning: detected activated squid, enabling FW_SERVICE_SQUID!
```

```

You still have to allow tcp port 3128 on internal, dmz and/or external.'
    FW_SERVICE_SQUID=yes
}
test "$FW_SERVICE_DHCPD" = no && check_srv dhcpd && {
    echo 'Warning: detected activated dhcpd, enabling FW_SERVICE_DHCPD!'
    FW_SERVICE_DHCPD=yes
}
test "$FW_SERVICE_SAMBA" = no && check_srv smb && {
    echo -e 'Warning: detected activated samba, enabling FW_SERVICE_SMB!'
You still have to allow tcp port 139 on internal, dmz and/or external.'
    FW_SERVICE_SAMBA=yes
}
test "$FW_SERVICE_DHCLIENT" = no && {
    test "$IFCONFIG_0" = dhcpclient -o "$IFCONFIG_1" = dhcpclient \
        -o "$IFCONFIG_0" = bootp -o "$IFCONFIG_1" = bootp && {
        echo 'Warning: detected BOOTP/DHCLIENT usage for interfaces in \
/etc/sysconfig/network/config,\
            enabling FW_SERVICE_DHCLIENT!'
        FW_SERVICE_DHCLIENT=yes
    }
}
}
}
}

```

#### Advanced options for FW\_SERVICE\_\*

If you want to offer the below services to your DMZ as well, (and not just internally), set the switches below to **"dmz"**, if you even want to offer to the world as well, set to **"ext"** instead of **"yes"** (NOT RECOMMENDED FOR SECURITY REASONS!)

```

test "$FW_SERVICE_AUTODETECT" = no -a "$FW_ALLOW_INCOMING_HIGHPORTS_UDP" != yes
-a "$FW_SERVICE_DNS" = yes && \
    echo 'FW_ALLOW_INCOMING_HIGHPORTS_UDP should be set to yes, if you are running
a DNS server!'

test "$FW_SERVICE_AUTODETECT" = yes -o "$FW_SERVICE_AUTODETECT" = dmz -o "$FW_SER-
VICE_AUTODETECT" = ext && {
    test "$FW_SERVICE_DNS" = no -a '!' "$START_NAMED" = no && check_srv named &&
{
    echo -e 'Warning: detected activated named, enabling FW_SERVICE_DNS!'
You still have to allow tcp/udp port 53 on internal, dmz and/or external.'
    FW_SERVICE_DNS=$FW_SERVICE_AUTODETECT
    FW_ALLOW_INCOMING_HIGHPORTS_UDP=yes
}
    test "$FW_SERVICE_SQUID" = no -a '!' "$START_SQUID" = no && check_srv squid &&
{
    echo -e 'Warning: detected activated squid, enabling FW_SERVICE_SQUID!'
You still have to allow tcp port 3128 on internal, dmz and/or external.'
    FW_SERVICE_SQUID=$FW_SERVICE_AUTODETECT
}
}
}
}

```

This way services you define, will be available to either the DMZ or the World. It could be helpful if you are servicing a public DNS server.

## 5.1.2. Configuring HIGHPORTS

By configuring the two variables

```

FW_ALLOW_INCOMING_HIGHPORTS_TCP
FW_ALLOW_INCOMING_HIGHPORTS_UDP

```

in this area you decide how access is allowed to high (unprivileged [above 1023]) ports

You may either allow everyone from any port access to your high ports ("yes"), disallow anyone ("no"), anyone who comes from a defined port (port number or known port name) [note that this is easy to circumvent!], or just your defined nameservers ("DNS").

```
test -e /etc/resolv.conf && NAMESERVERS=`$AWK '/^nameserver/ {print $2}'
/etc/resolv.conf | \
  $GREP -iv yast 2> /dev/null`
```

```
DONE_ALL=no
test "$FW_ALLOW_INCOMING_HIGHPORTS_TCP" = yes || {
  $LAC $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --dport 1024:65535 --
syn
  $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --dport 1024:65535
  $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state ESTABLISHED,RELATED -p tcp
--dport 1024:65535
}
for j in $FW_ALLOW_INCOMING_HIGHPORTS_TCP; do
  case "$j" in
    no) ;;
    yes)
      for CHAIN in input_int input_dmz input_ext; do
        $LAC $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --dport 1024:65535
--syn
        $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --dport 1024:65535
        $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW,ESTABLISHED,RELATED
-p tcp --dport 1024:65535
        done
        DONE_ALL=yes
        ;;
      [Dd][Nn][Ss])
        test -z "$NAMESERVERS" && \
        echo 'Warning: No nameservers in /etc/resolv.conf!'
        test "$DONE_ALL" = yes || for k in $NAMESERVERS; do
          test "$k" = 127.0.0.1 || for CHAIN in input_int input_dmz input_ext; do
            $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp -s $k --sport 53
--dport 1024:65535
            $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state ESTABLISHED,RELATED -
p tcp -s $k --sport 53 --dport 1024:65535
            done
            done
            ;;
          *)
            test "$DONE_ALL" = yes || for CHAIN in input_int input_dmz input_ext; do
              $LAC $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --sport $j --
dport 1024:65535 --syn
              $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT " -p tcp --sport $j --
dport 1024:65535
              $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW,ESTABLISHED,RELATED
-p tcp --sport $j --dport 1024:65535
              done
            done
            ;;
          esac
        done
      done
    done
  done
done
```

So if you enter DNS as the allowed port you are basically allowing communication with your defined nameservers in `/etc/resolv.conf`. If you are only querying other nameservers for name resolution you don't need to have DNS in `FW_ALLOW_HIGHPORTS_TCP` to have DNS.

## Note

Note that you can't use rpc requests (e.g. `rpcinfo`, `showmount`) as root from a firewall using this script (well, you can if you include range `600:1023` in `FW_SERVICES_EXT_UDP` ...).

## Tip

Please note that with v2.1 "yes" is **not mandatory** for active FTP from the firewall anymore. If you want to have active FTP then you should enter "**ftp-data**"

### 5.1.3. Configuring FW\_REDIRECTT

This can be used to force all internal users to surf via your squid proxy, or transparently redirect incoming web traffic to a secure web server.

#### 5.1.3.1. Transparent Proxy Configuration for FTP

Lets' assume the company is using SuSE-FTP proxy application and wants to implement it as a transparent proxy so the IT supervisor does not have configure the employees' ftp clients and force the users to the SuSE-FTP proxy

##### How does Transparent Proxy works?

Its very simple: the ip filter of you kernel redirects all packages to the ftp port (21) in "external networks" to the proxy and the proxy connects the server you wanted to go using informations from the ip package of your request.

##### Example 5.1. Transparent Proxy

Here is a example network configuration:

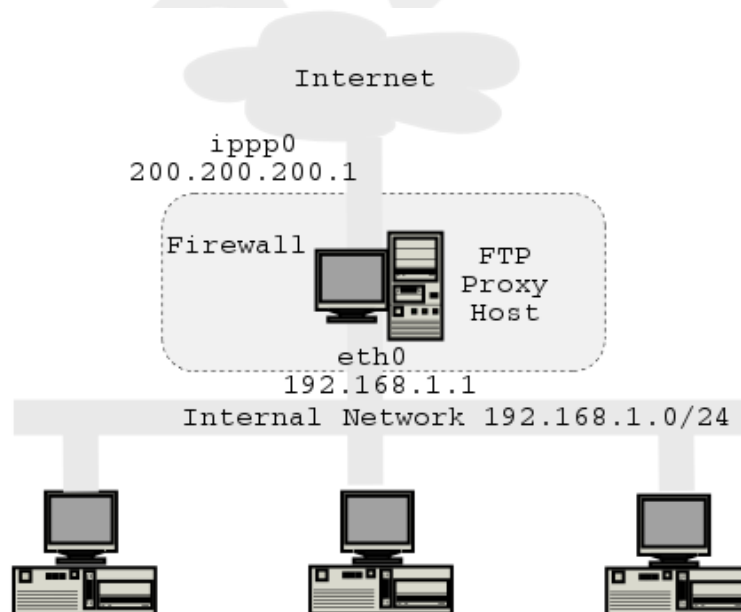


Figure 5.1. Transparent proxy

First, you want to enable the `AllowTransProxy` flag in the `ftp-proxy.conf(5)` file and start the proxy.

```
# grep -v ^# /etc/proxy-suite/ftp-proxy.conf | grep -v ^$
```

```
[ -Global- ]
ServerType                standalone
LogDestination            daemon
DestinationTransferMode  passive
PortResetsPasv           yes
Listen                   192.168.1.1
# may be needed in NAT'ed/Masqueraded environments
#TranslatedAddress       200.200.200.1
UseMagicChar              %
AllowMagicUser            yes
AllowTransProxy           yes
```

Second, you want to enable the transparent proxy feature in your kernel and set up the redirect rules

### Note

Default SuSE kernel has the transparent proxy feature enabled so kernel reconfiguration is not necessary

```
FW_REDIRECT="192.168.1.0/24,!192.168.1.1,tcp,21,21"
```

## 5.1.3.2. Transparent Proxy Configuration for Squid

### Example 5.2. Transparent Squid Proxy

As you have seen in How does Transparent Proxy works? first prepare `/etc/squid.conf` for transparent proxy support and the as seen in Figure 5.1 configure `FW_REDIRECT` parameter

```
# this is needed for transparent proxy:
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

and configure the firewall for using Squid

```
FW_SERVICE_SQUID="yes"

FW_REDIRECT="192.168.1.0/24,0/0,tcp,80,3128"
```

## 5.1.4. Expert log Format

The configuration of `FW_LOG_*` options results to the noise of your logging process:

- `FW_LOG_DROP_CRIT="yes"`
- `FW_LOG_ACCEPT_CRIT="yes"`

If you turn these to **"no"** the amount of logging will be less. It would be wiser to only change `FW_LOG_DROP_CRIT` to **"no"**.

If for some reason you are having problems with the configuration, ie `SuSEfirewall2` is preventing an access whereas you would like to permit you may change the following to **"yes"** for debugging reasons

- `FW_LOG_DROP_ALL="no"`
- `FW_LOG_ACCEPT_ALL="no"`

## Note

This will log everything so your logs will grow really fast

Another option is turn on kernel logging of matching packets. When this option is set for a rule, the Linux kernel will print some information on all matching packets (like most IP header fields) via the kernel log (where it can be read with dmesg or syslogd).

You can read more about the options in Section 7.1.1.

```
FW_LOG="--log-level warning --log-tcp-options --log-ip-option --log-prefix SuSE-FW"
```

### 5.1.5. Allowing Ping

ping uses the ICMP protocol's mandatory ECHO\_REQUEST datagram to elicit an ICMP ECHO\_RESPONSE from a host or gateway. ECHO\_REQUEST datagrams ("pings") have an IP and ICMP header, followed by a struct timeval and then an arbitrary number of "pad" bytes used to fill out the packet.<sup>2</sup>

```
test "$FW_ROUTE" = no -a "$FW_ALLOW_PING_DMZ" = yes -o "FW_ROUTE" = no -a "$FW_ALLOW_PING_EXT" = yes && \
    echo 'Warning FW_ROUTE needs to be set to yes, so that the FW_ALLOW_PING_EXT
and/or FW_ALLOW_PING_DMZ works!'
```

```
test "$FW_ALLOW_PING_FW" = yes && for CHAIN in input_ext input_dmz input_int; do
    $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-type
echo-request
    $IPTABLES -A $CHAIN -j "$ACCEPT" -p icmp --icmp-type echo-request
done
for TYPE in echo-reply destination-unreachable time-exceeded \
    parameter-problem timestamp-reply address-mask-reply; do
    for CHAIN in input_ext input_dmz input_int; do
        $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT-ICMP " -p icmp --icmp-type
$TYPE
        $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state ESTABLISHED,RELATED -p
icmp --icmp-type $TYPE
    done
done
# DROP rules for input ICMP are after trusted handling (see below)
....
....

# FORWARD ICMP rules
test "$FW_ALLOW_PING_DMZ" = yes -a "$FW_ROUTE" = yes && {
    for DEV in $FW_DEV_DMZ; do
        for CHAIN in forward_ext forward_int; do
            $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-
type echo-request -o $DEV
            $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW -p icmp --icmp-
type echo-request -o $DEV
        done
    done
    $LAA $IPTABLES -A forward_dmz -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-type
echo-reply
    $IPTABLES -A forward_dmz -j "$ACCEPT" -m state --state ESTABLISHED -p icmp -
-icmp-type echo-reply
}
```

<sup>2</sup> <sup>1</sup> refer to man ping(8) for more information

```

test "$FW_ALLOW_PING_EXT" = yes -a "$FW_ROUTE" = yes && {
  for DEV in $FW_DEV_EXT; do
    for CHAIN in forward_int forward_dmz; do
      $LAA $IPTABLES -A $CHAIN -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-
type echo-request -o $DEV
      $IPTABLES -A $CHAIN -j "$ACCEPT" -m state --state NEW -p icmp --icmp-
type echo-request -o $DEV
    done
  done
  $LAA $IPTABLES -A forward_ext -j LOG ${LOG}"-ACCEPT-PING " -p icmp --icmp-type
echo-reply
  $IPTABLES -A forward_ext -j "$ACCEPT" -m state --state ESTABLISHED -p icmp -
icmp-type echo-reply
}
# drop rule for forwarding chains are at the end of the forwarding rules

```

## 5.1.6. Configuring the SuSEfirewall2 for Traceroute

**What is traceroute ?** The Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route one's packets follow (or finding the miscreant gateway that's discarding your packets) can be difficult. Traceroute utilizes the IP protocol `time to live` field and attempts to elicit an ICMP `TIME_EXCEEDED` response from each gateway along the path to some host.<sup>3</sup>

```

# OUTPUT ICMP rules
test -z "$LDC" -o -z "$LDA" -o -z "$LAC" -o -z "$LAA" && $IPTABLES -A OUTPUT -j
LOG ${LOG}"-TRACEROUTE-ATTEMPT " -p icmp --icmp-type time-exceeded
test "$FW_ALLOW_FW_TRACEROUTE" = yes && {
  $IPTABLES -A OUTPUT -j "$ACCEPT" -p icmp --icmp-type time-exceeded
  $IPTABLES -A OUTPUT -j "$ACCEPT" -p icmp --icmp-type port-unreachable
}
test "$FW_ALLOW_FW_TRACEROUTE" = yes || \
  $IPTABLES -A OUTPUT -j "$DROP" -p icmp --icmp-type time-exceeded
for TYPE in fragmentation-needed network-prohibited host-prohibited communication-
prohibited; do
  $IPTABLES -A OUTPUT -j "$ACCEPT" -p icmp --icmp-type $TYPE
done
$IPTABLES -A OUTPUT -j "$DROP" -p icmp --icmp-type destination-unreachable # we
deny all other icmp type 3 codes

```

To get programs like traceroute to your firewall to work is a bit tricky, you have to set the following options in items

- `FW_ALLOW_INCOMING_HIGHPORTS_*`
- `FW_ALLOW_PING_*`
- `FW_ALLOW_FW_TRACEROUTE`

```

FW_ALLOW_INCOMING_HIGHPORTS_UDP=yes
FW_ALLOW_FW_PING=yes          #( this is for Windows PC's)
FW_ALLOW_FW_TRACEROUTE=yes

```

## 5.1.7. Understanding FW\_ALLOW\_FW\_SOURCEQUENCH

ICMP message types are not listed in `/etc/services`. However you may find a helpful file in `/usr/include/netinet/ip_icmp.h` that defines the names of the ICMP message numbers. ICMP, the Internet Control Message Protocol,

<sup>3</sup> <sup>2</sup>for more information look at traceroute man or info pages by issuing the **man traceroute** command

is not exploited to break in to your site's computer systems. However, it is being used, for numerous denial of service attacks. ICMP was designed as a network health indicator

## Note

Time exceeded and port unreachable messages are also a potential result of running the traceroute program from one of your site's host computers. Traceroute sends out packets to probe for the identities of all the routers along a network path and gathers the data it needs from the ICMP messages.

SuSEfirewall2 will let only your ISP to send you this when you enable `FW_ALLOW_FW_SOURCEQUENCH`

```
test "$FW_ALLOW_FW_SOURCEQUENCH" = no || for NET in $DEV_EXT_NET; do
    test -z "$LAC" -o -z "$LAA" && $IPTABLES -A input_ext -j LOG ${LOG}"-ACCEPT-
SOURCEQUENCH " -p icmp -s $NET --icmp-type source-quench
    $IPTABLES -A input_ext -j "$ACCEPT" -p icmp -s $NET --icmp-type source-quench
done
```

## 5.1.8. Understanding BROADCAST options

If set to yes, the firewall will not filter broadcasts by default. This is needed e.g. for Netbios/Samba, RIP, OSPF where the broadcast option is used.

```
test "$FW_ALLOW_FW_BROADCAST" = yes && {
    for NET in $DEV_EXT_BCAST; do
        for DEV in $FW_DEV_EXT; do
            $IPTABLES -A INPUT -j input_ext -i $DEV -d $NET
            $IPTABLES -A INPUT -j input_ext -i $DEV -d 255.255.255.255
        done
    done
    for NET in $DEV_DMZ_BCAST; do
        for DEV in $FW_DEV_DMZ; do
            $IPTABLES -A INPUT -j input_dmz -i $DEV -d $NET
            $IPTABLES -A INPUT -j input_dmz -i $DEV -d 255.255.255.255
        done
    done
    for NET in $DEV_INT_BCAST; do
        for DEV in $FW_DEV_INT; do
            $IPTABLES -A INPUT -j input_int -i $DEV -d $NET
            $IPTABLES -A INPUT -j input_int -i $DEV -d 255.255.255.255
        done
    done
done
}
```

If you do not want to allow them however ignore the annoying log entries, set `FW_IGNORE_FW_BROADCAST` to **"yes"**

```
broadcast stuff
test "$FW_IGNORE_FW_BROADCAST" = yes && {
    for NET in $DEV_EXT_BCAST; do
        for DEV in $FW_DEV_EXT; do
            $IPTABLES -A INPUT -j "$DROP" -i $DEV -d $NET
            $IPTABLES -A INPUT -j "$DROP" -i $DEV -d 255.255.255.255
        done
    done
    for NET in $DEV_DMZ_BCAST; do
        for DEV in $FW_DEV_DMZ; do
            $IPTABLES -A INPUT -j "$DROP" -i $DEV -d $NET
            $IPTABLES -A INPUT -j "$DROP" -i $DEV -d 255.255.255.255
        done
    done
}
```

```

done
done
for NET in $DEV_INT_BCAST; do
for DEV in $FW_DEV_INT; do
    $IPTABLES -A INPUT -j "$DROP" -i $DEV -d $NET
    $IPTABLES -A INPUT -j "$DROP" -i $DEV -d 255.255.255.255
done
done
}

```

## 5.1.9. Routing interfaces of same class

If you have two internal network cards, communication between the two networks connected to the firewall is not possible. This works as designed. For security reasons, no network may communicate to another until configured otherwise. Even if both are *trusted* internal networks. You can allow full traffic with `FW_ALLOW_CLASS_ROUTING` or specifying all allowed traffic with `FW_FORWARD`

If you have more than one device connected as Local net you can have multiple definitions ie. `eth0, eth3, ppp+` entered separted by a space

```
FW_DEV_INT="ppp+ eth0 eth3"
```

```

test "$FW_ALLOW_CLASS_ROUTING" = yes && {
for DEV1 in $FW_DEV_INT; do
for DEV2 in $FW_DEV_INT; do
test "$DEV1" = "$DEV2" || {
    $LAA $IPTABLES -A forward_int -j LOG ${LOG}"-ACCEPT-CLASS " -i
$DEV1 -o $DEV2
    $IPTABLES -A forward_int -j "$ACCEPT" -i $DEV1 -o $DEV2
}
done
done
for DEV1 in $FW_DEV_DMZ; do
for DEV2 in $FW_DEV_DMZ; do
test "$DEV1" = "$DEV2" || {
    $LAA $IPTABLES -A forward_dmz -j LOG ${LOG}"-ACCEPT-CLASS " -i
$DEV1 -o $DEV2
    $IPTABLES -A forward_dmz -j "$ACCEPT" -i $DEV1 -o $DEV2
}
done
done
for DEV1 in $FW_DEV_EXT; do
for DEV2 in $FW_DEV_EXT; do
test "$DEV1" = "$DEV2" || {
    $LAA $IPTABLES -A forward_ext -j LOG ${LOG}"-ACCEPT-CLASS " -i
$DEV1 -o $DEV2
    $IPTABLES -A forward_ext -j "$ACCEPT" -i $DEV1 -o $DEV2
}
done
done
}
}

```

## 5.2. Building a VPN with FreeS/WAN, SuSEfirewall2 and SSHSentinel

*Nadeem Hasan*

This article explains building a VPN connection between a mobile user (so called roadwarrior) and a local network using a gateway running SuSE 7.3 with SuSEFirewall2 and FreeS/WAN.

## 5.2.1. Introduction

A VPN (Virtual Private Network) lets you create a secure link between two private networks over an unsecure public network such as the internet. In this article we will build such a network using IPSec. IPSec (Internet Security Protocol) is a generic framework to provide security enhancements to IP (Internet Protocol). Three protocols are used to handle encryption and authentication, namely AH (Authentication Header) provides a packet-level authentication service, ESP (Encapsulating Security Payload) provides encryption plus authentication and IKE (Internet Key Exchange) negotiates connection parameters, including keys, for the other two. IPSec supports two encryption modes, transport and tunnel. The transport mode encrypts only the payload part, while the tunnel mode encrypts both the header and the payload. The secure channel negotiated between any two IPSec peers is called an SA (Security Association). SAs are uni-directional, i.e. you need to established a pair of them for bi-directional communication. IPSec is an open standard developed by IETF and is supported by all major operating systems and security product vendors.

FreeS/WAN is a portable, open source implementation of IPSec specification for Linux available under GPL. It has three main parts, namely KLIPS (Kernel IPSec) implements AH, ESP, and packet handling within the kernel, pluto (an IKE daemon) implements IKE, negotiating connections with other systems and various scripts, that provide an administrator's interface to the machinery. FreeS/WAN currently only supports 3DES (triple Data Encryption Standard) for encryption. Authentication is carried out by using MD5 hash of a shared key. This shared key could be a character string (shared secret), RSA crypto keypair or X.509 certificate (requires a patch). For more information about IPSec, see the introduction and also the FAQ on the FreeS/WAN website.

## 5.2.2. Prerequisites

I am assuming that you already have a box configured as a firewall using SuSE 7.3 and SuSEFirewall2 (iptables). You are free to use any other distribution and firewall system (such as ipchains) as long as you adapt these instructions to your system. Also, make sure that you have FreeS/WAN compiled with the X.509 patch. This patch adds support for digital (X.509) certificates, required for the purpose of this article. SuSE 7.3 ships with FreeS/WAN 1.91 with X.509 patch version 0.9.2, although other versions might work too. We would be using X.509 certificates for authentication, using OpenSSL as the CA (Certification Authority) to generate the certificates. You need to have root privileges on the system to do the tasks below.

Make sure you have freeswan and openssl packages installed. You can check this by issuing the following command:

```
# rpm -q package_name
```

## 5.2.3. Firewall Tweaks

First off, you need to allow the IKE (UDP port 500) and ESP (protocol 50) packets on your firewall. Optionally you can also allow AH (protocol 51), but we won't need it for this setup. If you are using SuSEfirewall2, add 500 to the list of ports for variable FW\_SERVICES\_EXT\_UDP, also add 50 to the protocols listed under FW\_SERVICES\_INT\_IP in the firewall config file `/etc/sysconfig/SuSEfirewall2`, e.g.:

```
FW_SERVICES_EXT_UDP="500"  
FW_SERVICES_EXT_IP="50"
```

In case you are cooking your own firewall rules using iptables:

```
# IKE negotiations on UDP port 500  
iptables -A INPUT -p udp --sport 500 --dport 500 -j ACCEPT  
iptables -A OUTPUT -p udp --sport 500 --dport 500 -j ACCEPT  
# ESP (protocol 50, not port) encryption and authentication  
iptables -A INPUT -p 50 -j ACCEPT iptables -A OUTPUT -p 50 -j ACCEPT
```

Also, you need to add `ipsec0` to the `FW_DEV_EXT` variable. Assuming your external interface is `eth0`, it should now read:

```
FW_DEV_EXT="eth0 ipsec0"
```

It is important to add `ipsec0` here so that SuSEfirewall2 does not turn on the reverse path filter (aka `rp_filter`), used for IP spoofing protection, on all the interfaces. Although, I am not sure what the exact interaction is, but having `rp_filter` turned on for the external interface has been reported to break things with `ipsec`. I would be happy to include a good explanation here if anyone comes up with one. If you omit this step, you will receive error messages like the following when you start FreeS/WAN:

```
ipsec_setup: WARNING: ipsec0 has route filtering turned on, KLIPS may not work
ipsec_setup: (/proc/sys/net/ipv4/conf/ipsec0/rp_filter = `1`, should be 0)
ipsec_setup: WARNING: eth0 has route filtering turned on, KLIPS may not work
ipsec_setup: (/proc/sys/net/ipv4/conf/eth0/rp_filter = `1`, should be 0)
```

For more information on this topic, see [FreeS/WAN and Firewalls](#) document.

## 5.2.4. OpenSSL and Certification Authority (CA)

Each X.509 certificate is issued (signed) by a CA (Certification Authority). This CA can be intermediate, i.e. its certificate is in turn signed by another CA, or root, i.e. it uses a self signed certificate. A certificate is authenticated by going up the trust chain until a trusted CA is found. If no trusted CA is found until the root CA is reached, the certificate fails the authentication. In our example, we will use OpenSSL to generate a self-signed root CA certificate. Since this OpenSSL CA and the FreeS/WAN gateway run on the same machine, care must be taken to use a different DN (Distinguished Name) for the CA and gateway certificates.

To generate a self-signed CA certificate, issue the following commands:

```
# cd
# mkdir certs
# cd certs
# ln -s /usr/share/ssl/misc/CA.sh CA.sh
# ./CA.sh -newca
```

Press enter when asked for CA certificate filename. You will then be asked for a passphrase twice. Enter the passphrase and remember it, otherwise you won't be able to sign any certificates later using this CA. Answer the other questions appropriately. When prompted for Common Name (CN), enter an alias for your hostname, such as `ca.mysite.dom`. This name need not be a valid DNS name. When finished, you will have a directory structure under `demoCA` for all the CA related files. The CA private key (1024 bit RSA) will be created in `private/akey.pem` and a self-signed CA certificate in `cacert.pem` under the `demoCA` directory. The CA certificate will have a validity of 1 year. If you want a different validity period, edit the `CA.sh` script and change the variable `DAYS` appropriately. Use the following command to verify the certificate content:

```
# openssl x509 -in ./demoCA/cacert.pem -noout -text
```

In order for this CA certificate to be used with SSH Sentinel and FreeS/WAN, it must be converted to DER (binary) format from the PEM (base64) format. Later versions of X.509 patch (0.9.3+) also accept PEM encoded CA certificates.

```
# openssl base64 -d -in ./demoCA/cacert.pem -out cacert.bin
```

In order for FreeS/WAN to trust this CA certificate, it should be copied to the `/etc/ipsec.d/cacerts` directory:

```
# mv cacert.bin /etc/ipsec.d/cacerts/
```

## 5.2.5. FreeS/WAN Gateway Certificate

Next, we will generate a host certificate for the FreeS/WAN gateway. In order to make this certificate easier to use, we need to add a `subjectAltName` field to it and use the FQDN (Fully Qualified Domain Name) to identify the host. To achieve this, put the following line under [ `usr_cert` ] section of OpenSSL configuration file `/usr/share/ssl/openssl.cnf`:

```
subjectAltName=DNS:host.mysite.dom
```

Replace `host.mysite.dom` with the FQDN of your FreeS/WAN gateway machine. This will allow us to use this FQDN as the `id` in FreeS/WAN instead of the certificate DN. More on this later. Now issue the following command to generate a certificate request:

```
# ./CA.sh -newreq
```

You will again be prompted for a passphrase twice. Remember this passphrase as it will provide the proof that you have authority over this request. Please provide the appropriate requested information, making sure to use the FQDN of the gateway as the CN (Common Name). At the end of this, you will have a PEM encoded certificate request and the private key in `newreq.pem`. Next, this certificate request should be signed by the CA:

```
# ./CA.sh -signreq
```

When prompted for the passphrase, type the CA's passphrase. You will then be asked twice to confirm, first to sign the request and then to commit it. Answer "y" to both questions. The certificate request will be signed and a text copy printed on the screen. The signed certificate is stored in `newcert.pem`. At this point, remember to remove the `subjectAltName` line from `openssl.cnf` file.

FreeS/WAN uses the X.509 certificate stored in DER format in `/etc/x509cert.der` to identify itself to its peers. Before we can use our PEM host certificate, we need to convert it into DER format:

```
# openssl x509 -in newcert.pem -outform DER -out newcert.der
# mv newcert.der /etc/x509cert.der
```

## 5.2.6. Installing the Private Key

Now that we have the host certificate stored, we need to have its private key available in `/etc/ipsec.secrets`. SuSE installs a private key by default upon installation in this file, so we need to move the existing file out of the way and create a new one with the private key of our gateway.

```
# mv /etc/ipsec.secrets /etc/ipsec.secrets.orig
# /usr/lib/ipsec/fswcert --key newreq.pem > /etc/ipsec.secrets
```

When prompted for a passphrase, use the passphrase you entered during the host certificate request process before. Verify that the private key has been stored in `/etc/ipsec.secrets`. It should look something like:

```
RSA {
  Modulus:      0xE0....
  PublicExponent: 0x010001
  PrivateExponent: 0xD7....
  Prime1:       0xF8....
  Prime2:       0xE6....
  Exponent1:    0xB0....
```

```
    Exponent2:      0x55....
    Coefficient:    0x8D....
}
```

Private keys should be protected zealously, make sure ours is:

```
# chmod 600 /etc/ipsec.secrets
```

## 5.2.7. Configuring FreeS/WAN

FreeS/WAN stores its configuration in `/etc/ipsec.conf`. Following is a complete configuration for our setup:

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file
# More elaborate and more varied sample configurations can be found
# in FreeS/WAN's doc/examples file, and in the HTML documentation.

config setup
    # THIS SETTING MUST BE CORRECT or almost nothing will work;
    # %defaultroute is okay for most simple cases.
    interfaces=%defaultroute
    # Debug-logging controls: "none" for (almost) none, "all" for lots.
    klipsdebug=none
    plutodebug=none
    # Use auto= parameters in conn descriptions to control startup actions.
    plutoload=%search
    plutostart=%search
    # Close down old connection when new one using same ID shows up.
    uniqueids=yes

# defaults for subsequent connection descriptions
conn %default
    type=tunnel
    #
    left=%defaultroute
    # leftnexthop would default to %defaultroute
    leftsubnet=192.168.200.0/24
    leftupdown=/usr/lib/ipsec/_updown_custom
    #
    authby=rsasig
    #
    keyexchange=ike
    keyingtries=1
    ikelifetime=240m
    keylife=20m
    leftid=@host.mysite.dom
    leftrsasigkey=%cert
    rightrsasigkey=%cert
    #
    auth=esp
    pfs=yes
    compress=yes

conn vpn
    right=%any
    auto=add
```

For detailed description of each parameter in this file, see the manual page for `/etc/ipsec.conf`. Make sure to change the `host.mysite.dom`, listed as `leftid`, to the FQDN of your gateway. Also, change the `leftsubnet` parameter to reflect your setup. Note that I have turned on compression (`compress=yes`). Though not completely reliable, it can be a

real winner for roadwarriors connected via slow dialup access. It has worked for me so far. If you face problems, try setting it to no. You can also increase the keylife from 20m (default 60m), but using a smaller value helps in cleaning up the dead connections faster. This is because the pluto daemon cannot detect a dead peer until it tries to renegotiate the keys. You should leave the rest of the parameters intact, unless you have a very good reason to change them.

## 5.2.8. Poking Holes in the Wall

An important parameter to consider here is leftupdown. Everytime an IPSec SA is established, a hole needs to be created in the firewall to allow the traffic flow back and forth for that SA. Pluto can call a script specified by the leftupdown parameter each time a connection (SA) comes up or goes down. FreeS/WAN ships such a script called `_updown` in `/usr/lib/ipsec`, but it is only suitable for firewalls using the ancient ipfwadm firewall system. The FreeS/WAN developers recommend using this default script as the starting point to create a custom updown script for other firewall systems. Only caveat is that we should use a different name for this custom script so that it does not get overwritten during the next upgrade. We are going to patch this default script and call it `_updown_custom`.

The patch is listed below. You can also download it from [here](#), you can also download a patched `_updown_custom` file.

Save it under `/usr/lib/ipsec` as `_updown.diff` and do the following to create our custom updown script.

```
# cd /usr/lib/ipsec
# cp _updown _updown_custom
# patch < _updown.diff
```

At this point make sure that `START_IPSEC` variable in `/etc/rc.config` is set to yes and reboot. You could also start ipsec manually using `"rcipsec start"` at this point, but you would get the `rp_filter` related error described elsewhere. This is because during the last startup, SuSEfirewall didn't see ipsec0 listed and it turned on the `rp_filter` for all the interfaces. Just restarting the firewall would have no effect as it would then leave the `rp_filter` at its previous state. At this point you should have FreeS/WAN running on your gateway machine. If you get errors, verify that you have followed the steps outlined earlier properly. You can also refer to the troubleshooting guide on FreeS/WAN website.

## 5.2.9. Configuring SSH Sentinel

Next, we install SSH Sentinel 1.2.3 on a Windows 2000 laptop. SSH Sentinel is an IPSec implementation that supports almost all flavors of Windows(TM) operating system. You can download it with a 30 day evaluation license. Go through the installation and create a self-signed certificate for Sentinel. Although we won't be using this certificate, Sentinel somehow requires that it be present at all times. At the end of the installation, You will be asked to reboot in true Windows tradition. After reboot, you will see an icon for SSH Sentinel in the system tray. Now, we will create a certificate request, sign it using our OpenSSL CA and import it back into Sentinel along with the CA certificate.

Right click on the Sentinel tray icon and select "Run Policy Editor..." to open the policy editor window. Click on the "Key Management" tab. Under "Authentication Keys", right click on "primary host key". Select "Add New Auth. Key..." to open the "New Authentication Key" wizard. Select "Enroll for a new certificate" and click "Next". By doing this, we will be using the existing private key to generate a certificate request. You can also choose to generate a new key by selecting "Create a new authentication keypair and certificate". On the "Identity Information" screen, select "Administrator Email" as the primary identifier and enter your email address in the next field and click "Next". On the "Enrollment Information" screen, select "File Based Enrollment (PKCS #10)" as the "Enrollment protocol" and enter `"a:\request.req"` in the "File Location" field. Place a formatted disk in the floppy drive and Click "Finish". On the policy editor window, click "Apply".

Insert the floppy disk into linux gateway. To sign the certificate request contained in it, issue the following commands:

```
# mount /floppy
# cd ~/certs
# openssl ca -policy policy_anything -in /floppy/request.req -out /floppy/request.cer
```

You will be prompted for the CA passphrase. Enter "y" for both the prompts asking you to confirm the signing and committing of the request. For Sentinel to trust this certificate, it should also have and trust the CA certificate. To accomplish this, we copy the CA certificate to the floppy disk:

```
# cp /etc/ipsec.d/cacerts/cacerts/cacert.bin /floppy/
# umount /floppy
```

Put this floppy back in the laptop drive. Open "Policy Editor" and go to the "Key Management" tab. Right click on "Trusted Root CA" and select "Add New Certificate...", select the CA certificate (cacert.bin) from the floppy and click "OK" on the confirmation dialog. The OpenSSL CA certificate should now appear under "Trusted Root CA" as ca.mysite.dom or whatever CN you provided for the CA. Click "Apply" to apply the change. Next, we will import our user certificate for which we have a pending request in Sentinel. Right click on the "primary host key" and select "Import Auth. Key...". Now select the certificate file (request.cer) from the floppy and click "OK" on the confirmation dialog. Now you should see our certificate under "primary host key". You may want to rename it to something more helpful such as your name or email address. Make sure to click "Apply" to apply the changes.

Next, we will create a security policy for our setup. Click on the "Security Policy" tab in the policy editor window. Right click on "VPN Connections" and select "Add New Rule...". In the "Add VPN Connection" screen, enter the hostname of our FreeS/WAN gateway. If you want, you can also enter the IP address by clicking the "IP" button next to the field. Select our certificate as the authentication key from the list. Click to select "Use legacy proposal" option. Enter the intranet ip address and subnet mask, which in our case are 192.168.200.0 and 255.255.255.0. You may need to adjust these to match your network. Click "OK" to create the policy. Sentinel will now try to probe this connection. If the probe is successful, Sentinel will show you the connection properties. If the probe fails, choose to add the connection anyway and then check the parameters to make sure they are correct. It will also fail if you are not connected to the internet. Click "Apply" to apply the changes. Open the properties dialog of our VPN connection again. Under "General" tab, make sure "Encryption" is set to 3DES, "IKE Mode" is set to Main Mode and "IKE Group" is set to MODP 1024. Click on the "SA Lifetimes" tab and change "IPSec SA Lifetime" to 20. Now click on the "Advanced tab" and select the "Apply IP compression" option. Also select the PMTU and PFS options. Click "OK" and "Apply" to apply the changes.

## 5.2.10. Putting It All Together

Finally, its time to eat the fruits of our hard work. If you are not already connected, dial your ISP and establish an internet connection. Open the Sentinel statistics window by right clicking on the Sentinel icon in the system tray and selecting "View Statistics...". Now try to ping any machine inside the internal network protected by FreeS/WAN gateway. Please note that you will not be able to ping the FreeS/WAN gateway itself as explained here. If the ping does not work the first time, try again or try a different machine, in case routing on the first is broken. You should see the ping echo reply from that host. At this point, the statistics window should show information about the established IPsec SA. Check the FreeS/WAN messages on the gateway by issuing the following command:

```
# tail -f /var/log/messages
```

You should see something like:

```
Pluto[10532]: "vpn" #1: ignoring informational payload, type IPSEC_INITIAL_CONTACT
Pluto[10532]: "vpn" #1: Peer ID is ID_DER_ASN1_DN: 'CN=nhasan@nadmm.com'
Pluto[10532]: "vpn" #1: STATE_MAIN_R3: sent MR3, ISAKMP SA established
Pluto[10532]: "vpn" #1: responding to Quick Mode
Pluto[10532]: "vpn" #1: STATE_QUICK_R2: IPsec SA established
```

The last line tells us that the IPsec SA was established successfully with the peer, in this case our roadwarrior laptop. If you are connecting to a Windows network and using NT domain logins, make sure to specify the WINS server address on the laptop. This will make sure that you find the domain controller that you authenticate with.

## 5.2.11. Conclusion

I hope that you found this article useful and were successful in getting the desired results. To the best of my knowledge, the information presented within is accurate. I would appreciate any input in case you find any inaccuracies or mistakes. This document is still a work in progress and will be periodically updated. I would also like to thank everyone who helped me with advice and suggestions when I first tried to use FreeS/WAN.

## 5.3. Configuring SuSEfirewall2 for VPN

Ipsec interface must be on the same FW\_DEV\_\* as the real interface accepting the ipsec traffic. in this examples it's eth0, so ipsec must be in FW\_DEV\_EXT.

### Example 5.3. Sample VPN configuration

A small company wants access to the Internet for it's client PCs and additionally IPSEC with another office on another continent.

```
external fw interface=eth0
internal fw interface=eth1
frees/wan ipsec device=ipsec0
internal LAN: 192.168.0.0/16
remote LAN: 10.0.0.0/16
the incoming frees/wan traffic is accepted on eth0, the external interface
```

```
FW_DEV_EXT="eth0 ipsec0"
FW_DEV_INT="eth1"
FW_ROUTE="yes"
FW_MASQUERADE="yes"
FW_MASQ_NETS="192.168.0.0/16"
FW_SERVICES_EXT_UDP="500"
FW_SERVICES_EXT_IP="50 51"
FW_FORWARD="192.168.0.0/16,10.0.0.0/16 10.0.0.0/16,192.168.0.0/16"
```

### Example 5.4. VPN with no masquerading

This example is from the SuSE-Security Mailing list

```
FW_DEV_EXT="eth0 ipsec0"
FW_ROUTE="yes"
FW_SERVICES_EXT_UDP="500"
FW_SERVICES_EXT_IP="50"
FW_AUTOPROTECT_SERVICES="no"
```

In order to have the above to work you need to either modify `/usr/lib/ipsec/_updown` or better have a custom script `/usr/lib/ipsec/_updown_custom`. Copy the original to `_updown_custom` and add the following lines

```
up-client:)
# connection to my client subnet coming up
# If you are doing a custom version, firewall commands go here.
iptables -I FORWARD 1 -s $PLUTO_MY_CLIENT_NET/$PLUTO_MY_CLIENT_MASK \
-d $PLUTO_PEER_CLIENT_NET/$PLUTO_PEER_CLIENT_MASK -j ACCEPT
iptables -I FORWARD 1 -d $PLUTO_MY_CLIENT_NET/$PLUTO_MY_CLIENT_MASK \
```

```
        -s $PLUTO_PEER_CLIENT_NET/$PLUTO_PEER_CLIENT_MASK -j ACCEPT
    ;;
down-client:)
    # connection to my client subnet going down
    # If you are doing a custom version, firewall commands go here.
    iptables -D FORWARD -s $PLUTO_MY_CLIENT_NET/$PLUTO_MY_CLIENT_MASK \
        -d $PLUTO_PEER_CLIENT_NET/$PLUTO_PEER_CLIENT_MASK -j ACCEPT
    iptables -D FORWARD -d $PLUTO_MY_CLIENT_NET/$PLUTO_MY_CLIENT_MASK \
        -s $PLUTO_PEER_CLIENT_NET/$PLUTO_PEER_CLIENT_MASK -j ACCEPT
    ;;
```

It is necessary also to modify the config file for ipsec (located in `/etc/ipsec.conf`) add the line with leftupdown:

```
conn %default
    leftupdown=/usr/lib/ipsec/_updown_custom
```

Of course, if you are using SuSE on both sides, you should set the other computer too (also only leftupdown script).

DRAFT

# 6

## Customizing SuSEfirewall2

### 6.1. Using custom rules with SuSEfirewall2

If you are in need of some special rules for your firewall you can have SuSEfirewall2 load it automatically. This section will try explain the `/etc/sysconfig/scripts/SuSEfirewall2-custom`

#### **Warning**

This is file is for SuSEfirewall2 and is an example for using the hooks which are supplied to load customized iptables rules.

**THERE IS NO HELP FOR USING HOOKS EXCEPT THIS FILE ! SO READ CAREFULLY !  
IT IS USEFUL TO CROSS-READ `/sbin/SuSEfirewall2` TO SEE HOW HOOKS WORK !**

#### 6.1.1. Enabling FW\_CUSTOM

In order to use custom rules together with SuSEfirewall2 you should first enable the FW\_CUSTOM by removing the comment character #

```
FW_CUSTOMRULES="/etc/sysconfig/scripts/SuSEfirewall2-custom"
```

```
test -z "$FW_CUSTOMRULES" || {
    test -r "$FW_CUSTOMRULES" || {
        echo "Error: Can not read custom rules file: $FW_CUSTOMRULES"
        test -x "$LOGGER" && \
            $LOGGER -p kern.error -t SuSEfirewall2 "Firewall customary rules file
can not be read: $FW_CUSTOMRULES"
        exit -1
    }
    . "$FW_CUSTOMRULES"
    test -x "$LOGGER" && \
        $LOGGER -p kern.info -t SuSEfirewall2 "Firewall customary rules loaded from
$FW_CUSTOMRULES"
}
```

#### 6.1.2. Understanding how FW\_CUSTOM works

##### `fw_custom_before_antispoofing()`

these rules will be loaded before any anti spoofing rules will be loaded. Effectively the only filter lists already effective are

1. allow any traffic via the loopback interface

2. allow DHCP stuff,
3. allow SAMBA stuff [2 and 3 only if FW\_SERVICE\_... are set to "yes"]

You can use this hook to prevent logging of uninteresting broadcast packets or to allow certain packet through the anti-spoofing mechanism.

```
fw_custom_before_antispoofing() {
#example: allow incoming multicast packets for any routing protocol
#iptables -A INPUT -j ACCEPT -d 224.0.0.0/24

true
}
```

#### **fw\_custom\_before\_port\_handling()**

could also be named "after\_antispoofing()" these rules will be loaded after the anti-spoofing and icmp handling but before any IP protocol or TCP/UDP port allow/protection rules will be set.

You can use this hook to allow/deny certain IP protocols or TCP/UDP ports before the SuSEfirewall2 generated rules are hit.

```
fw_custom_before_port_handling() { # could also be named "after_antispoofing()"
#example: always filter backorifice/netbus Trojan connect requests and log them.
#for target in LOG DROP; do
#   for chain in input_ext input_dmz input_int forward_int forward_ext forward_dmz;
do
#       iptables -A $chain -j $target -p tcp --dport 31337
#       iptables -A $chain -j $target -p udp --dport 31337
#       iptables -A $chain -j $target -p tcp --dport 12345:12346
#       iptables -A $chain -j $target -p udp --dport 12345:12346
#   done
#done

true
}
```

#### **fw\_custom\_before\_masq()**

could also be named "after\_port\_handling()" these rules will be loaded after the IP protocol and TCP/UDP port handling, but before any IP forwarding (routing), masquerading will be done.

### **Note**

NOTE: reverse masquerading is before directly after fw\_custom\_before\_port\_handling !!!!

You can use this hook to ... hmmm ... I'm sure you'll find a use for this ..

```
fw_custom_before_masq() { # could also be named "after_port_handling()"
true
}
```

#### **fw\_custom\_before\_denyall()**

could also be named "after\_forwardmasq()" these are the rules to be loaded after IP forwarding and masquerading but before the logging and deny all section is set by SuSEfirewall2. You can use this hook to prevent the logging of annoying packets.

```
fw_custom_before_denyall() { # could also be named "after_forwardmasq()"
#example: prevent logging of talk requests from anywhere
#for chain in input_ext input_dmz input_int forward_int forward_ext forward_dmz;
do
# iptables -A $chain -j DENY -p udp --dport 517:518
#done

true
}
```

## 6.2. Identd Dilemma

Analysis of the Hard coded rule for port 113

It is quite common to have a portscan check once you have your firewall up and running. It is always better to find the open ports before someone else has the opportunity to sneak in. One of the sites that provide such a scan is <http://www.grc.com> which is designed to find open ports in a firewall typically running on a Windows™ machine.

When you test your firewall at <http://www.grc.com> the analysis will say identd, is visible. This will be the same even if you have identd disabled.

This is because when someone tries to connect port 113 they will receive TCP RST. This means the identd port has a netfilter rule that says REJECT all incoming connections; every other port is configured as DROP by default.

### Note

You may change what DROP means with SuSEfirewall2 version 3.1

This is intentional. Otherwise you will get long long delays when trying to connect to a service that will first do an IDENTD lookup before processing your connection. Most IRC servers do that, and some FTP servers as well. You also need it for mail sending. Here is the code from SuSEfirewall2-3.1

```
# If port 113 (auth/identd) will not allowed below, outgoing mail would
# be delayed most of the time. Hence we put a hard coded reject line
# in.
$IPTABLES -I input_ext 1 -j "$REJECT" -p tcp --dport 113 --syn 2> /dev/null
```

### 6.2.1. Definition of REJECT target

Before we go into details of the reasoning, it would be worthy to recall the definition of REJECT from **man iptables**

**Table 6.1. REJECT target**

Option	<b>--reject-with</b>
Example	<b>iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset</b>

Explanation	<p>This option tells the <b>REJECT</b> target what response to send to the host that sent the packet that we are rejecting. Once we get a packet that matches a rule in which we have specified this target, our host will first of all send the associated reply, and the packet will then be dropped dead, just as the <b>DROP</b> target would drop it. The following reject types are currently valid: <code>icmp-net-unreachable</code>, <code>icmp-host-unreachable</code>, <code>icmp-port-unreachable</code>, <code>icmp-protocol-unreachable</code>, <code>icmp-net-prohibited</code> and <code>icmp-host-prohibited</code>. The default error message is to send an <b>port-unreachable</b> to the host. All of the above are ICMP error messages and may be set as you wish. You can find further information on their various purposes in the appendix Table A.1. There is also the option <b>echo-reply</b>, but this option may only be used in conjunction with rules which would match ICMP ping packets. Finally, there is one more option called <b>tcp-reset</b>, which may only be used together with the TCP protocol. The <b>tcp-reset</b> option will tell <b>REJECT</b> to send an TCP RST packet in reply to the sending host. TCP RST packets are used to close open TCP connections gracefully. For more information about the TCP RST read <a href="#">RFC 793 - Transmission Control Protocol</a>. As stated in the <b>iptables</b> man page, this is mainly useful for blocking identd probes which frequently occur when sending mail to broken mail hosts, that won't otherwise accept your mail.</p>
-------------	---

### 6.2.2. Deciding the error code to send back

When scanned firewalls give back some responses either that reply as the service available or they produce error messages. For the person analyzing the error messages given back the firewall, making an assumption on which ports are really filtered is not that difficult

When you use `tcp-reset` you basically shut off TCP connection without using ICMP. This is the response that a machine would give if it received a TCP packet bound for a port where nothing was listening. Although TCP resets give away less information than ICMP error codes, they still speed up attack problems.

When you just `REJECT` than the default behavior is to send `port-unreachable` Sending back an ICMP error code has the effect of warning the sending machine not to retry sending the packet, thereby saving some network traffic.

---

## Part III. Logs and Interperating them

Once you have the firewall configured and working obviously you would want to have information on what's happening with the firewall. In this part a we will try to understand how SuSEfirewall2 creates the logs, and what do they mean.

DRAFT

DRAFT

# 7

## Log messages

### 7.1. Log messages and understanding them

Starting with SuSE 8.1 you need to specify in `/etc/syslog.conf` where you want the logs to be written. The following makes sure the firewall logs will be written to `/var/log/firewall`

```
*.*;!kern.* - /var/log/messages
kern.* - /var/log/firewall
```

SuSEfirewall2 will create log files which are typically written to `/var/log/firewall`. The level of logging is based on your choice in `FW_LOG_*` parameter.

```
# Logging setup
LOG="--log-level warning --log-tcp-options --log-ip-options --log-prefix SuSE-FW"
test -z "$FW_LOG" || LOG="$FW_LOG"
test "$FW_LOG_DROP_CRIT" = no -o "$FW_LOG_DROP_ALL" = yes && LDC=":"
test "$FW_LOG_ACCEPT_CRIT" = no -o "$FW_LOG_ACCEPT_ALL" = yes && LAC=":"
test "$FW_LOG_DROP_ALL" = yes || LDA=":" # it might look weird - a ":"
test "$FW_LOG_ACCEPT_ALL" = yes || LAA=":" # disables logging of this type
```

#### 7.1.1. Understanding Iptables' log parameters

Before we can go in analyzing the logs of SuSEfirewall2 a basic understanding of Iptables log target options is necessary.

**Table 7.1. LOG target options**

Option	<code>--log-level</code>
Example	<code>iptables -A FORWARD -p tcp -j LOG --log-level debug</code>

Explanation	This is the option to tell <b>iptables</b> and <b>syslog</b> which log level to use. For a complete list of log levels read the <code>syslog.conf</code> manual. Normally there are the following log levels, or priorities as they are normally referred to: <code>debug</code> , <code>info</code> , <code>notice</code> , <code>warning</code> , <code>warn</code> , <code>err</code> , <code>error</code> , <code>crit</code> , <code>alert</code> , <code>emerg</code> and <code>panic</code> . The keyword <code>error</code> is the same as <code>err</code> , <code>warn</code> is the same as <code>warning</code> and <code>panic</code> is the same as <code>emerg</code> . Note that all three of these are deprecated, in other words do not use <code>error</code> , <code>warn</code> and <code>panic</code> . The priority defines the severity of the message being logged. All messages are logged through the kernel facility. In other words, setting <b>kern.=info /var/log/iptables</b> in your <code>syslog.conf</code> file and then letting all your <b>LOG</b> messages in <b>iptables</b> use log level <code>info</code> , would make all messages appear in the <code>/var/log/iptables</code> file. Note that there may be other messages here as well from other parts of the kernel that uses the <code>info</code> priority. For more information on logging I recommend you to read the <b>syslog</b> and <code>syslog.conf</code> man pages as well as other HOWTOs etc.
Option	<b>--log-prefix</b>
Example	<b>iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"</b>
Explanation	This option tells <b>iptables</b> to prefix all log messages with a specific prefix, which can be easily combined with <b>grep</b> or other tools to track specific problems and output from different rules. The prefix may be up to 29 letters long, including whitespace and other special symbols.
Option	<b>--log-tcp-sequence</b>
Example	<b>iptables -A INPUT -p tcp -j LOG --log-tcp-sequence</b>
Explanation	This option will log the TCP Sequence numbers, together with the log message. The TCP Sequence numbers are special numbers that identify each packet and where it fits into a TCP sequence, as well as how the stream should be reassembled. Note that this option constitutes a security risk if the logs are readable by unauthorized users, or by the world for that matter. As does any log that contains output from <b>iptables</b> .
Option	<b>--log-tcp-options</b>
Example	<b>iptables -A FORWARD -p tcp -j LOG --log-tcp-options</b>
Explanation	The <b>--log-tcp-options</b> option logs the different options from the TCP packet headers and can be valuable when trying to debug what could go wrong, or what has actually gone wrong. This option does not take any variable fields or anything like that, just as most of the <b>LOG</b> options don't.
Option	<b>--log-ip-options</b>
Example	<b>iptables -A FORWARD -p tcp -j LOG --log-ip-options</b>
Explanation	The <b>--log-ip-options</b> option will log most of the IP packet header options. This works exactly the same as the <b>--log-tcp-options</b> option, but instead works on the IP options. These logging messages may be valuable when trying to debug or track specific culprits, as well as for debugging - in just the same way as the previous option.

### 7.1.2. Variables used in the SuSEfirewall2 generated logs

SuSEfirewall2 has a complex and helpful logging options. Before we start analyzing the logs it would be better to understand the remarks used in the logs. All logs have the default **SuSE-FW** in addition to the prefixes shown below. You can change the `SuSE-FW` by editing the `--log-prefix` in the `FW_LOG` parameter

Obviously you will not be seeing all those unless you are debugging your configuration

**Table 7.2. SuSEfirewall2 LOG Prefixes**

Remark	Explanation
ACCEPT	Packet is accepted
DROP	Packet is dropped
REJECT	Packet is rejected. This is used when there is an <code>identd</code> request
ACCEPT-CLASS	This is used when <code>FW_ALLOW_CLASS_ROUTING</code> is enabled and you have enabled <code>LOG_ACCEPT_ALL</code> for debugging purposes
ACCEPT-ICMP	This is used when <code>FW_ALLOW_PING_FW</code> is enabled and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes
ACCEPT-ALL-INTERNAL	This is used when <code>FW_PROTECT_FROM_INTERNAL</code> is set to <b>"no"</b> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes
ACCEPT-MASQ	This is used when you have defined <code>FW_MASQ</code> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes
ACCEPT-REVERSE-MASQ	This is used with <code>FW_FORWARD_MASQ</code> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes or <code>LOG_ACCEPT_CRITICAL</code> is enabled
ACCEPT-PING	This is used when <code>FW_ALLOW_PING_DMZ</code> and/or <code>FW_ALLOW_PING_EXT</code> is enabled and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes
ACCEPT-SOURCEQUENCH	This is used when you have enabled <code>FW_ALLOW_SOURCEQUENCH</code> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes or <code>LOG_ACCEPT_CRITICAL</code> is enabled
ACCEPT-REDIRECT	This is used when you have defined <code>FW_REDIRECT</code> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes or <code>LOG_ACCEPT_CRITICAL</code> is enabled
ACCEPT-TRUST	This is used when you have defined <code>FW_TRUSTED_NETS</code> and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes or <code>LOG_ACCEPT_CRITICAL</code> is enabled
DROP-ANTI-SPOOF	This is used when <code>LOG_DROP_CRITICAL</code> or <code>LOG_DROP_ALL</code> is enabled and if IP spoofing is encountered on the interface
DROP-CIRCUMVENTION	This used when <code>FW_ROUTE</code> is enabled and <code>LOG_DROP_CRITICAL</code> or <code>LOG_DROP_ALL</code> is enabled
DROP-ICMP-CRIT	If <code>LOG_DROP_CRITICAL</code> is enabled then critical ICMP requests are logged as seen in ICMP Drop Rules.
DROP-DEFAULT	These are the packet dropped by default when <code>LOG_DROP_CRITICAL</code> is enabled
DROP-DEFAULT-INVALID	These are the packet dropped by default if <code>state</code> is <code>INVALID</code> when <code>LOG_DROP_CRITICAL</code> is enabled
FORWARD-RELATED	This is used if you have <code>FW_ROUTE</code> enabled and <code>LOG_ACCEPT_ALL</code> is set to yes for debugging purposes
NO-ACCESS-INT->FWEXT	This is used to identify Anti Spoofing/Circumvention protection - interface dependent
TRACEROUTE-ATTEMPT	This is used with all logging options if you have enabled <code>FW_ALLOW_FW_TRACEROUTE</code>
UNAUTHORIZED-ROUTING	This is used to identify Anti Spoofing/Circumvention protection - interface dependent seen if <code>FW_ROUTE</code> is enabled and either <code>LOG_DROP_CRITICAL</code> or <code>LOG_DROP_ALL</code> is enabled
UNAUTHORIZED-TARGET	This is used to identify Anti Spoofing/Circumvention protection - interface dependent and either <code>LOG_DROP_CRITICAL</code> or <code>LOG_DROP_ALL</code> is enabled

**ICMP Drop Rules**

```
# ICMP drop rules must be here to allow trusted rules for ICMP
for CHAIN in input_ext input_dmz input_int; do
    $LDC $IPTABLES -A $CHAIN -j LOG ${LOG}"-DROP-ICMP-CRIT " -p icmp --icmp-type
    redirect
    $LDC $IPTABLES -A $CHAIN -j LOG ${LOG}"-DROP-ICMP-CRIT " -p icmp --icmp-type
    source-quench
    $LDC $IPTABLES -A $CHAIN -j LOG ${LOG}"-DROP-ICMP-CRIT " -p icmp --icmp-type
    timestamp-request
    $LDC $IPTABLES -A $CHAIN -j LOG ${LOG}"-DROP-ICMP-CRIT " -p icmp --icmp-type
    address-mask-request
    $LDC $IPTABLES -A $CHAIN -j LOG ${LOG}"-DROP-ICMP-CRIT " -p icmp --icmp-type 2
```

**7.1.3. Analyzing SuSEfirewall2 generated logs****Example 7.1. Drop Default**

```
Oct  1 14:21:32 zeus kernel: SuSE-FW-DROP-DEFAULT IN=eth0 OUT=
MAC=00:10:4b:10:69:c1:00:20:6f:13:82:d2:08:00 SRC=111.222.333.444
DST=555.666.777.888 LEN=60 TOS=0x00 PREC=0x00 TTL=51 ID=10094 DF PROTO=TCP
SPT=1332 DPT=443 WINDOW=5840 RES=0x00 SYN URGP=0 OPT
(020405B40402080A03E4463C0000000001030300).
```

**Table 7.3. SuSEfirewall2 log explanations**

Option	Explanation
SUSE-FW-DROP-DE-FAULT	Log title produced by the SuSEfirewall2 script describing the action taken
IN	interface the packet came in on. In this case this was eth0
OUT	interface packet went out on. In this case, not applicable
MAC	Combined MAC address of sender and recipient
SRC	Source IP. "this is the ip of the attacker"
DST	Destination IP
LEN, TOS, PREC, TTL, ID	various stuff in the TCP/IP headers
PROTO	protocol of the packet
SPT	Source port "this is the port they're coming from"
DPT	Destination Port "this is the port they're trying to gain access to"
WINDOW, RES	more packet header stuff
SYN	The packet was a SYN packet, i.e. the first packet in a TCP negotiation.

**Note**

The details of the header fields can be found in the RFC documents on TCP and IP [rfc793](#), [rfc791](#)).

**7.2. Firewall Log Analysis Software**

Many people on the SuSE-Security mailinglist recommend a tool named fwlogwatch available <http://cert.uni-stuttgart.de/projects/fwlogwatch/> and from <http://www.kyb.uni-stuttgart.de/boris/software.shtml>.

It is a highly configurable tool with the ability to parse nice HTML output of not only netfilter messages but also snort rules. Here is a list of it's feature taken from the README file:

## FEATURES

### General Features

- Can detect and process log entries in the following formats:
  - Linux ipchains
  - Linux netfilter/iptables
  - Solaris/BSD/Irix/HP-UX ipfilter
  - Cisco IOS
  - Cisco PIX,
  - NetScreen
  - Windows XP firewall
  - Elsa Lancom router
  - Snort IDS.
- Entries can be parsed in combined log files, the parsers to be used can be selected.
- Gzip-compressed logs are supported.
- Can separate recent from old entries and detects timewarps in log files.
- Can recognize 'last message repeated' entries concerning the firewall.
- Integrated resolver for protocols, services and host names.
- Can do lookups in the whois database.
- Own DNS and whois information cache for faster lookups.
- Hosts, ports, chains and branches (targets) can be selected or excluded as needed.
- Support for internationalization (available in English, German, Portuguese, simplified and traditional Chinese and Swedish).

### Log summary mode:

- A lot of options to find and display relevant patterns in connection attempts.
- Intelligent selection of certain fields (e.g. the host name column is omitted and the host mentioned in the header of the summary if the log is from a single host, the same happens with the chains, targets and interfaces).
- Plain text and HTML (with CSS) output with many sort options.
- Can send reports by email.

**Interactive report mode:**

- The integrated report generator fills and presents a report that can be sent to abuse contacts of attacking sites or computer emergency response teams (CERTs).
- Supports templates and incident number generation.

All fields can be adjusted as needed interactively.

**Realtime response mode:**

- The program detaches and stays in the background as a daemon.
- Detection of the necessary ipchains rules with logging turned on can be configured.
- Response can be a notification (in form of a log file entry, an email, a remote winpopup message or whatever you can put into a shell script), or a customizable firewall modification.
- The included response script adds a new chain for fwlogwatch to ipchains or netfilter setups and attackers are blocked with new firewall rules.
- Supports trusted hosts (anti-spoofing).
- The current status of the program can be followed through a web interface (supports IPv6).

The commented configuration file supports and explains all options and will get you started quickly. Please read the man page for details on the command line options.

---

# Part IV. Troubleshooting

This part will try to help you to quickly identify the troubles you may be encountering and also includes some cookbook receipes which were posted to <suse-security@suse.com> Mailinglist. If you are not already subscribed to this very valuable list it is time to start considering to subscribe since you are caring for security of your system.

DRAFT

DRAFT

# 8

## Troubleshooting

### 8.1. Cookbook Recipes

Although SuSEfirewall2 is a straightforward application there are times when you may have difficulty in setting up things. In this section you will find quick fixes for your problems.

#### 8.1.1. Configuring SuSEfirewall2 for pcAnywhere

Here is an example 192.168.0.23 is an internal IP of the system behind firewall. this will accept connections from anywhere

```
FW_FORWARD_MASQ="0/0,192.168.0.23,tcp,5631 0/0,192.168.0.23,udp,5632"
```

Another example is to specify the source IP

```
FW_FORWARD_MASQ="12.45.26.3,192.168.0.23,tcp,5631 \  
12.45.26.3,192.168.0.23,udp,5632"
```

In the case you want to use pcAnywhere in reaching more than one Windows™ PC. Obviously, we can't use ports 5631 and 5632 on the firewall, those are now port-forwarded to the 192.168.0.15 machine. So... We'll pick a different pair (5633, and 5634), and forward them to 5631 and 5632 on 192.168.0.30.

Now our forward statement will look like this:

```
FW_FORWARD_MASQ="1.2.3.0/24,192.168.0.15,tcp,5631 \  
1.2.3.0/24,192.168.0.15,udp,5631 \  
1.2.3.0/24,192.168.0.15,tcp,5632 \  
1.2.3.0/24,192.168.0.15,udp,5632 \  
5.6.7.8/32,192.168.0.30,tcp,5633,5631 \  
5.6.7.8/32,192.168.0.30,udp,5633,5631 \  
5.6.7.8/32,192.168.0.30,tcp,5634,5632 \  
5.6.7.8/32,192.168.0.30,udp,5634,5632"
```

#### 8.1.2. Configuring SuSEfirewall2 for edonkey

Here is an example for port forwarding edonkey to 192.168.0.2

```
FW_FORWARD_MASQ="0.0.0.0/0,192.168.0.2,tcp,4662 0.0.0.0/0,192.168.0.2,udp,4665"
```

#### 8.1.3. Configuring SuSEfirewall2 for Kazaa

To enable Kazaa clients to share with other Internet users; Internal network is 192.168.0.0/24

Use following Rule with SuSEfirewall2 and **No Masquerading**:

```
FW_FORWARD="192.168.0.0/24,0.0.0.0/0,tcp,1024: \
192.168.0.0/24,0.0.0.0/0,udp,1024: 0.0.0.0/0,192.168.0.0/24,tcp.1024: \
0.0.0.0/0,192.168.0.0/24,udp,1024:"
```

Use following Rule with SuSEfirewall2 and **Masquerading**:

```
FW_FORWARD_MASQ="192.168.0.0/24,0.0.0.0/0,tcp,1024: \
192.168.0.0/24,0.0.0.0/0,udp,1024: 0.0.0.0/0,192.168.0.0/24,tcp.1024: \
0.0.0.0/0,192.168.0.0/24,udp,1024:"
```

## Note

Interfaces have to be set correct in FW\_DEV\_EXT, FW\_DEV\_INT

### 8.1.4. Configuring Masquerading for specific port numbers

8.1.4.1. Which options I have to change in `/etc/sysconfig/SuSEfirewall2`, that a Client behind the wall can access the Ports 2401 (tcp/udp) and 21 (tcp) on the Internet directly?

try, for your example,

```
FW_MASQ_NETS=" 192.168.0.0/24,0/0,tcp,21 \
192.168.0.0/24,0/0,tcp,2401 \
192.169.0.0/24,0/0,udp,2401"
```

## Note

(change 192.168.0.0/24 to suite your needs)

### 8.1.5. Configuring SuSEfirewall2 for use with time servers

To configure Network time server access and response which is using UDP protocol and port number 123 you have couple of options. NTP uses source and destination port 123.

```
FW_SERVICES_EXT_UDP="123"
```

or

```
FW_TRUSTED_NETS="IP_address_of_timeserver,udp,123"
```

### 8.1.6. Configuring SuSEfirewall2 for Internal Access to External IP

The problem is that you can't seem to access the server from the internal machine ie laptop using the external address. You can't even ping it. Syslog on the server says:

```
SuSE-FW-NO_ACCESS_INT->FWEXT IN=eth0 OUT=
```

This gives you the clue: the firewall is set up to prevent internal machines from accessing the external port (it's an anti-spoofing measure). You will need to add a rule (or rules) to `/etc/sysconfig/scripts/SuSEfirewall2-custom`. In `fw_custom_before_antispoofing()`, add:

```
iptables -A INPUT -i eth0 -s internal net/mask -d external i/f -j ACCEPT
```

internal net/mask is like `192.168.0.0/24` (as appropriate to your LAN) external i/f is the IP address for the external interface.

Note that you want to specify both the source interface (eth0) and the source address, so that the rule is NOT applied to packets coming from the Internet side (spoofing an internal address). You could improve this further by adding controls for ports and protocols to narrow its behavior even further, for example add a rule (as above but with port specified) for each of http, pop and smtp. That way only those ports will be handled, all others will generate a log message.

## 8.1.7. Accessing the webserver in DMZ

If you want to access your web server from a machine connected to the internal network (`192.168.1.0/24`), you need to type the private IP address of the web server: `http://192.168.5.2` to your browser's address bar. However SuSEfirewall2 will normally drop this request

There are two reasons for this: Access to the external/public IP of the firewall is prohibited from the internal network for security reasons, you will see a `SuSE-FW-NO_ACCESS_INT->FW_EXT` if you log dropped packets. Furthermore, the firewall does only DNAT (redirect) connections coming in from the masquerading interface (ie `ppp0`) to the web server in the DMZ.

To be able to access your web server in the DMZ from the `192.168.1.0/24` net, you need to forward connections between the DMZ and the internal net. SuSEfirewall2 does not route anything between different subnets by default. Masquerading is not involved here, therefore you need at least to set

```
FW_FORWARD="192.168.1.0/24,192.168.5.2,tcp,80"
```

This will allow access to port 80 on your web server from any machine in the `192.168.1.0/24` net. And, please set

```
FW_LOG_DROP_ALL="yes"  
FW_LOG_ACCEPT_CRIT="yes"
```

to be able to see what exactly is going on. Once you have SuSEfirewall2 acting as you want it you should change the above logging options so that your logs are not filling with unnecessary information.<sup>4</sup>

## 8.1.8. Protection for DOS

By using the `SuSEfirewall2-custom` you can add some rules for protecting the webserver from DOS

```
iptables -A INPUT -p tcp --dport 80 -m limit --limit 50/s -j ACCEPT  
iptables -A INPUT -p tcp --dport 80 -m limit --limit 2/m -j LOG \  
    --log-prefix "DoS attempt: "  
iptables -A INPUT -p tcp --dport 80 -j DROP
```

Of course you may need to adjust the `--limit 50/s` to reflect your needs.<sup>5</sup>

<sup>4</sup> <sup>3</sup>By Andreas J. Mueller

<sup>5</sup> <sup>4</sup>By Michal Ludvig

## 8.1.9. Security Aspects & Troubleshooting for FTP

FTP is a problematic protocol.<sup>6</sup>

There are two different connection types that are used in FTP.

The first is the "control" connection. This connection is initiated by the FTP client software. This is a connection from client:X to server:ftp (port 21)

```
client:X -> server:ftp
```

where "X" is some high (>1023) port number. From a firewall standpoint, the control connection is very straightforward and easy to accommodate.

The second is the "data" connection. There are two types of data connection: "active" and "passive".

"active" connections are often considered the normal means of getting data from an FTP server. Here are the steps involved in active data transfer:

- the FTP client binds to a high port on the client machine (call it "Y")
- the FTP client tells the FTP server what port it has chosen
- the FTP server initiates a connection from its ftp-data port (port 20) to the port chosen by the FTP client

So the active connection looks like

```
server:ftp-data -> client:Y
```

"passive" connections work basically like this:

- the FTP client asks the FTP server to use a passive connection for data
- the FTP server binds to a high port (call it "N")
- the FTP server tells the client which port it has chosen
- the FTP client connects from some high port to the port chosen by the server So the passive connection looks like

```
client:Z -> server:N
```

There are firewall problems with both data transfer methods, at least as long as we're using more primitive tools like 'ipchains' that cannot keep track of stateful information; e.g. ipchains cannot tell if a connection from some remote machine's ftp-data port is servicing an FTP session, or simply probing high TCP ports.

The problem with active FTP was just described. FTP clients can bind to virtually any high port on the client machine, and we really don't want to allow any remote machine to initiate a connection to any high port on this machine (which is why we have those awful TCP\_BLOCKED\_SERVICES settings).

Briefly, "active" FTP can be a risk for the client.

The problem with passive FTP is that the FTP server must be prepared to accept TCP connections on various high ports. We don't want to allow any remote machine to connect to high ports on the FTP server, either.

Briefly, "passive" FTP requires loosening firewall restrictions on the server.

<sup>6</sup> <sup>5</sup>Taken from Bastille project Peter Watkins

### 8.1.9.1. Configuring your server to allow *passive* FTP clients

The recommended way to allow your FTP server to handle passive clients is to configure it to use a certain range of ports, and only loosen the firewall restrictions for those ports.

If you install proxy-suite application from your *SuSE CD /DVD* you will also use proxy capabilities along with port restriction increasing the security

DRAFT

DRAFT

---

# Part V. Appendixes

DRAFT

DRAFT

# A

## ICMP types

This is a complete listing of all ICMP types:

**Table A.1. ICMP types**

TYPE	CODE	Description	Query	Error
0	0	Echo Reply	x	
3	0	Network Unreachable		x
3	1	Host Unreachable		x
3	2	Protocol Unreachable		x
3	3	Port Unreachable		x
3	4	Fragmentation needed but no frag. bit set		x
3	5	Source routing failed		x
3	6	Destination network unknown		x
3	7	Destination host unknown		x
3	8	Source host isolated (obsolete)		x
3	9	Destination network administratively prohibited		x
3	10	Destination host administratively prohibited		x
3	11	Network unreachable for TOS		x
3	12	Host unreachable for TOS		x
3	13	Communication administratively prohibited by filtering		x
3	14	Host precedence violation		x
3	15	Precedence cutoff in effect		x
4	0	Source quench		
5	0	Redirect for network		
5	1	Redirect for host		
5	2	Redirect for TOS and network		
5	3	Redirect for TOS and host		
8	0	Echo request	x	
9	0	Router advertisement		
10	0	Route solicitation		
11	0	TTL equals 0 during transit		x
11	1	TTL equals 0 during reassembly		x
12	0	IP header bad (catchall error)		x

TYPE	CODE	Description	Query	Error
12	1	Required options missing		x
13	0	Timestamp request (obsolete)	x	
14		Timestamp reply (obsolete)	x	
15	0	Information request (obsolete)	x	
16	0	Information reply (obsolete)	x	
17	0	Address mask request	x	
18	0	Address mask reply	x	

DRAFT

# B

## Resources

There were many valuable sources of information that assisted in the creation of this book. The following sections list additional resources to help you learn even more SuSEfirewall2 and Iptables.

### B.1. Resources on the Web

#### Unix SysAdm Resources: Firewalls & Unix Security

This is a nice collection of resources related to firewalls and security. <http://www.stokely.com/unix.sysadm.resources/security.html>

#### Iptables Tutorial

This is a in depth tutorial on Iptables. Usefull when you want to understand how things work. <http://iptables-tutorial.frozentux.net/>

#### Ipsysctl-tutorial

Currently at the beat stage according to the author the website, still has information revealed about the secrets of `/proc/sys/net/ipv4/` . <http://ipsysctl-tutorial.frozentux.net/>

#### Linux Networking-concepts HOWTO

A gentle introduction to networking concepts by Rusty Russell

#### Linux 2.4 NAT HOWTO

Another howto by Rusty Russell explaining the Network Address Translation in the Linux 2.4 kernels.

#### Packet Filtering

Another howto by Rusty Russel on Packet filtering basics. <http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html>

#### Netfilter Extensions

There are many extensions to the netfilter code. Some are included with the SuSE kernel like ULOG module which can do very detailed logging to different targets (plaintext files, MySQL databases, ...). ulogd supports plugins for different output formats, as well as for new protocols. <http://www.netfilter.org/documentation/HOWTO/netfilter-extensions-HOWTO.html>

#### Connection Tracking

[http://www.sns.ias.edu/~jns/security/iptables/iptables\\_contrack.html](http://www.sns.ias.edu/~jns/security/iptables/iptables_contrack.html)

#### Netfilter FAQ

FAQ docuemntation on Netfilter mailinglist. <http://www.netfilter.org/documentation/FAQ/netfilter-faq.html>

#### SFIRESCAN

SFIRESCAN is a software, which analyzes the file `/var/log/firewall`, created if the software SuSEfirewall or SuSEfirewall2 is installed and activated. <http://seismo.ethz.ch/linux/sfirescan.html>

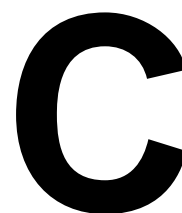
**Firewall Forensics (What am I seeing ?)**

This document explains what you see in firewall logs, especially what port numbers means. You can use this information to help figure out what hackers are up to. [http://www.linuxsecurity.com/resource\\_files/firewalls/firewall-seen.html](http://www.linuxsecurity.com/resource_files/firewalls/firewall-seen.html)

**Netfilter Log Format**

Here is a quick reference for the format used by the netfilter log messages. <http://logi.cc/linux/netfilter-log-format.php3>

DRAFT



# OPEN PUBLICATION LICENSE

Draft v0.4, 8 June 1999

## C.1. REQUIREMENTS ON BOTH UNMODIFIED AND MODIFIED VERSIONS

The Open Publication works may be reproduced and distributed in whole or in part, in any medium physical or electronic, provided that the terms of this license are adhered to, and that this license or an incorporation of it by reference (with any options elected by the author(s) and/or publisher) is displayed in the reproduction.

Proper form for an incorporation by reference is as follows:

Copyright (c) <year> by <author's name or designee>. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, vX.Y or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The reference must be immediately followed with any options elected by the author(s) and/or publisher of the document (see section VI).

Commercial redistribution of Open Publication-licensed material is permitted.

Any publication in standard (paper) book form shall require the citation of the original publisher and author. The publisher and author's names shall appear on all outer surfaces of the book. On all outer surfaces of the book the original publisher's name shall be as large as the title of the work and cited as possessive with respect to the title.

## C.2. COPYRIGHT

The copyright to each Open Publication is owned by its author(s) or designee.

## C.3. SCOPE OF LICENSE

The following license terms apply to all Open Publication works, unless otherwise explicitly stated in the document.

Mere aggregation of Open Publication works or a portion of an Open Publication work with other works or programs on the same media shall not cause this license to apply to those other works. The aggregate work shall contain a notice specifying the inclusion of the Open Publication material and appropriate copyright notice.

### **SEVERABILITY**

If any part of this license is found to be unenforceable in any jurisdiction, the remaining portions of the license remain in force.

### **NO WARRANTY.**

Open Publication works are licensed and provided "as is" without warranty of any kind, express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose or a warranty of non-infringement.

## C.4. REQUIREMENTS ON MODIFIED WORKS

All modified versions of documents covered by this license, including translations, anthologies, compilations and partial documents, must meet the following requirements:

1. The modified version must be labeled as such.
2. The person making the modifications must be identified and the modifications dated.
3. Acknowledgement of the original author and publisher if applicable must be retained according to normal academic citation practices.
4. The location of the original unmodified document must be identified.
5. The original author's (or authors') name(s) may not be used to assert or imply endorsement of the resulting document without the original author's (or authors') permission.

## C.5. GOOD-PRACTICE RECOMMENDATIONS

In addition to the requirements of this license, it is requested from and strongly recommended of redistributors that:

1. If you are distributing Open Publication works on hardcopy or CD-ROM, you provide email notification to the authors of your intent to redistribute at least thirty days before your manuscript or media freeze, to give the authors time to provide updated documents. This notification should describe modifications, if any, made to the document.
2. All substantive modifications (including deletions) be either clearly marked up in the document or else described in an attachment to the document.

Finally, while it is not mandatory under this license, it is considered good form to offer a free copy of any hardcopy and CD-ROM expression of an Open Publication-licensed work to its author(s).

## C.6. LICENSE OPTIONS

The author(s) and/or publisher of an Open Publication-licensed document may elect certain options by appending language to the reference to or copy of the license. These options are considered part of the license instance and must be included with the license (or its incorporation by reference) in derived works.

- A. To prohibit distribution of substantively modified versions without the explicit permission of the author(s). "Substantive modification" is defined as a change to the semantic content of the document, and excludes mere changes in format or typographical corrections.

To accomplish this, add the phrase 'Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.' to the license reference or copy.

- B. To prohibit any publication of this work or derivative works in whole or in part in standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.

To accomplish this, add the phrase 'Distribution of the work or derivative of the work in any standard (paper) book form is prohibited unless prior permission is obtained from the copyright holder.' to the license reference or copy.

## C.7. OPEN PUBLICATION POLICY APPENDIX:

(This is not considered part of the license.)

Open Publication works are available in source format via the Open Publication home page at <http://works.open-content.org/>.

Open Publication authors who want to include their own license on Open Publication works may do so, as long as their terms are not more restrictive than the Open Publication license.

If you have questions about the Open Publication License, please contact TBD, and/or the Open Publication Authors' List at [opal@opencontent.org](mailto:opal@opencontent.org), via email.

DRAFT

DRAFT

---

# Colophon

## **Colophon.**

This document was written entirely in XML, using Docbook 4.2 XML DTD. The PDF file was generated using Docbook XSL Stylesheets 1.58.1. Transforming the XML file to Fo was done using Saxon 6.5.2 and the resulting Fo file was converted to PDF by XEP 2.77

Emacs was used as the editing processor along with psgml extensions. SuSE Linux 8.1 was the operating platform for all of the tasks

The Unofficial SuSE FAQ project is aiming to produce documents which are supplementary to those that come with the distribution itself with an overall objective to provide answers to frequently asked questions

You are more than welcomed to contribute to the project. Details may be find in the project web site <http://sf.net/projects/susefaq> hosted at Sourceforge

DRAFT

DRAFT

---

# Glossary of terminology

## Stateful Firewall

A stateful firewall not only checks for source and destination IP addresses and port numbers, but it also **listens** to all TCP/IP communications to make sure that all of the *communications* are following all procedures. Think of it as a realtime grammar and spell checker for *languages* like TELNET, WWW, etc. Hackers try to re-write the *language* to try to break into it, crash it, etc. A stateful firewall will see a given TCP/IP connection running a "language" like TELNET doing weird stuff that it shouldn't be doing and then it simply drops that weird packet.

## Split Zone

The **Split Zone** feature means that there will be (2) named processes running on your machine. One daemon will run and answer DNS queries for the external interface while the other daemon will answer on the internal interface for the private network. This setup helps protect your internal network IP addresses and names from being exposed to people out on the Internet.

## Chroot

The CHROOTed feature means that the named daemon which runs as `root` will run in its own isolated area. This behavior is very similar to the access that an anonymous FTP user logs in and can only see a subset of the entire remote file system. The reason to implement this is that if some new named security exploit comes out and a hostile user (cracker) finds your machine, they will be extremely limited to what they **can** and **can not** do. This is a good thing in the name of security. CHROOTing daemons like named isn't perfect but it does help security.

## egress filtering

use google to look it up basically: only allow source IP's to match the IP addresses supplied in the connection drop all outgoing traffic that doesn't have the expected source IP addresses

## acknowledgment

Abbreviated ACK. In communications, ACK is a control code, ASCII 06, sent by the receiving computer to indicate that the data has been received without error and that the next part of the transmission may be sent.

## negative acknowledgment

Abbreviated NAK. In communications, a control code, ASCII 21, sent by the receiving computer to indicate that the data was not properly received and should be sent again.