# Linux Intrusion Detection System FAQ

**Sander Klein**

**lids AT roedie DOT nl**

**Linux Intrusion Detection System FAQ**
by Sander Klein

v.20, May 19th, 2003

This is the Linux Intrusion Detection System (LIDS) FAQ. It answers commonly asked questions asked on the LIDS-mailing-list (and more!).

The LIDS version at the time this Document was released was:

- Kernel 2.4: 1.1.1 (stable) 1.1.2-rc6 (developement)
- Kernel 2.2: 0.11.0r2 (stable) 0.11.1pre1 (development)
- Kernel 2.5: 2.0.3rc1 (development)

# Table of Contents

# Chapter 1. Introduction to LIDS

## 1.1. What is LIDS?

LIDS is an enhancement for the Linux kernel written by Xie Huagang (mailto:xie@gnuchina.org) and Philippe Biondi (mailto:philippe.biondi@webmotion.com). It implements several security features that are not in the Linux kernel natively. Some of these include: mandatory access controls (MAC), a port scan detector, file protection (even from root), and process protection.

## 1.2. Why use LIDS?

The current Linux setup has many problems that are inherent in many versions of *nix. Probably the single largest problem is the "all powerful" root account. When a process or user has root privileges, there is little if nothing to prevent that process or user from completely destroying the system. A malicious user/intruder with root access can cause much headache for us hard working sysadmins. LIDS implements access control lists (ACLs) that will help prevent even those with access to the mighty root account from wreaking havoc on a system. These ACLs allow LIDS to protect files as well as processes.

## 1.3. Where can I obtain LIDS?

You can obtain LIDS on http://www.lids.org or one of it’s mirrors. For a list of mirrors visit http://www.lids.org/mirrors.html.

## 1.4. Which versions of the Linux kernel are supported?

Currently, LIDS supports the 2.2 kernel as well as the 2.4 kernel. LIDS development is done using the 2.5 kernel. So, new features will be implemented there first. However, some features may make it back into the 2.2 or 2.4 kernel depending on users’ needs. Any security fixes will be backported to the 2.2 or 2.4 kernels. There’s also a version of LIDS which is integrated in LSM (Linux Security Modules (http://lsm.immunix.org)).

## 1.5. Is there a LIDS mailing list?

Yes. You can post to the list at any time by e-mailing `lids-users@lists.sourceforge.net`. However, if you wish to receive messages posted to the mailing list, you must subscribe to it. To subscribe, go to http://lists.sourceforge.net/lists/listinfo/lids-user ( http://lists.sourceforge.net/lists/listinfo/lids-user) and fill out the form. You will then receive a confirmation request that you must reply to. You can also unsubscribe and change your mailing list options from that page.

## 1.6. What about an archive?

The mailing list archive is located at http://www.geocrawler.com/lists/3/SourceForge/9348/0/ (http://www.geocrawler.com/redir-sf.php3?list=lids-user) The old archive can be found at

http://groups.yahoo.com/group/lids.

# 1.7. Copyright & Disclaimer

This document is `copyright(c) 2000, 2001, 2002 for Steve Bremer - 2002, 2003 for Sander Klein` and it is a FREE document. You may redistribute it under the terms of the GNU General Public License.

The information herein this document is, to the best of Sander's knowledge, correct. However, LIDS and the LIDS-FAQ is written by humans and thus, the chance of mistakes, bugs, etc. might occur from time to time.

No person, group, or other body is responsible for any damage on your computer(s) and any other losses by using the information on this document. i.e.

"*THE AUTHORS AND ALL MAINTAINERS ARE NOT RESPONSIBLE FOR ANY DAMAGES INCURRED DUE TO ACTIONS TAKEN BASED ON THE INFORMATION IN THIS DOCUMENT.*"

# 1.8. Feedback

If you have any questions, comments, suggestions, or corrections for this document, please feel free to contact me at lids AT roedie DOT nl (mailto:lidsATroedieDOTnl). I always welcome feedback whether it's good or bad!

# 1.9. Credit

Special thanks go to:

- **Xie Huagang** - Technical editor and LIDS author.
  - LIDS version question.
  - Subject/object question.
- **Philippe Biondi** - LIDS author.
- **Andy Harrelson** - Grammar/spelling editor.
- **Rob Willis** - Open-SSH, OpenLDAP, and Port Sentry configuration examples.
- **Fred Mobach** - Inspiration and corrections.
- **David Ranch** - I used his excellent Linux IP Masquerade HOW-TO (http://www.linuxdoc.org/HOWTO/IP-Masquerade-HOWTO.html) as an SGML template. His disclaimer also proved useful.
- **Austin Gonyou** -
  - Valuable feedback on FAQ.
  - Alternative fix to the lidsadm compile problem.
  - Warning about updating the inode of the `/etc/passwd` file.
- **Pavel Epifanov** - For a simple fix to the lidsadm compile problem.
- **Justus Pendleton** - Samba configuration example.
- **Nenad Micic**

  - For the hidden process kill script example.
  - His C program to kill hidden processes at shutdown.
  - LD_PRELOAD warning.

- **Bill Phillips** - For pointing out many reference errors in the PDF version.
- **Szymon Juraszczyk**
  - LD_PRELOAD warning.

- **Lorn Kay** -
  - Heartbeat configuration for Linux HA.
  - Sendmail configuration.

- **Bill McKenzie** - Additions to Portsentry configuration.
- **Sander Klein**
  - Question about checking if LIDS is enabled or disabled.
  - Apcupsd configuration.
  - Pump configuration.
  - Snort configuration.
  - getty configuration.
  - login configuration.
  - su configuration.

- **David Spreen** - Time restriction warning and restricting crontab access.
- **Thomas Linden** - For providing his BIND 9.x Configuration
- **Mathias Gygax** - For providing sample configurations for exim, qpopper, and proftp.
- **Dimitri Goldin** - For pointing out that the LIDS password can be obtained by root users if /dev/pts is mounted and left unprotected.
- **BigSam** - Innd config.
- **Ralf Dreibrodt** - Innd config.
- **Steve Bremer** - For starting to write this document which helped me a lot when I started using LIDS.

" **Linux is a trademark of Linus Torvalds** "

## 1.10. Translations

The following is a list of know translations and their locations:

- **Japanese** -- http://www.linux.or.jp/JF/JFdocs/LIDS-FAQ.html
- **Polish** -- http://www.linuxpub.pl/man/lidsfaq/.

## 1.11. Revision History

The latest version of this FAQ can be found at http://www.roedie.nl/lids-faq/. Please check the latest version before reporting any bugs.

- May 19th, 2003. Version .20
  - Added the Postfix example.
  - Updated the Openssh example.
  - Removed the part about compiling lids prior to 1.1.0.
  - Removed the 'I can't see my /etc/lids dir' question.
  - Added question about losing capabilities.
  - Changed the capabilities section.
  - Changed lots of small things to be correct again.

- December 22th, 2002. Version .19
  - This is the first version of the LIDS FAQ that I (Sander Klein) released. Big thanks go to Steve.
  - Replaced lidsadm -P to lidsconf -P.
  - Rewritten small parts to be up-to-date again.
  - Rewritten the whole document in docbook format with the ldp stylesheet.
  - Added the capabilities section.
  - Changed some URLs to point to the right directions again.

- April 27th, 2002. Version .18
  - This will be the last version of the LIDS FAQ that I (Steve Bremer) will produce. Sander Klein will be the new maintainer of the FAQ from this point on.
  - Added aproxy configuration example.
  - Added /dev/pts warning.
  - Added squid configuration example.
  - PowerPC status update.
  - Added Innd configuration example.
  - Other minor corrections.

- January 28th, 2002. Version .17
  - Changed "READ" to "READONLY".

- January 12th, 2002. Version .16
  - Various updates for the changes made in version 1.1.0.
  - Minor corrections.
  - Added sample configurations for exim, qpopper, and proftp.
  - Updated console logging question.
  - Updated LD_PRELOAD warning.
  - Updated file ACL inheritance question.
  - Added file editing question.

- November 12th, 2001. Version .15
  - Added many new configurations (Sendmail,apcupsd,pump,snort,getty,login,su). Thanks to Sander Klein and Lorn Kay.
  - Added Red Hat Kernel patch question.
  - Added User quota question.

- August 26th, 2001. Version .14
  - Added LIDS enabled/disabled question.
  - Improved basic configuration question.
  - Added a notice for Debian users.
  - Added time restriction question.
  - Updated log rotation question to use new time restriction feature.
  - Updated non-Intel hardware question.
  - Added translations section.
  - Added port restriction question.
  - Added BIND 9.x configuration.

- May 20th, 2001. Version .13
  - Added heartbeat configuration for HA Linux.
  - Added read password error question.
  - Added basic configuration question.
  - Minor additions to portsentry configuration.
  - Enhanced (yet again) passwd update question.
  - Other minor corrections.

- April 1st, 2001. Version .12
  - Updated FAQ for new versions of LIDS (1.0.6+ and 0.9.14+).
  - Added warning about LD_PRELOAD environment variable.
  - Updated hardware question.

- March 10th, 2001. Version .11
  - Fixed several reference errors in the PDF version (there are still a few document conversion problems that need looked at).
  - Clarified the Basic System Setup configuration.
  - Updated the mailing list information
  - Updated passwd and log rotation questions.

- March 1st, 2001. Version .10
  - Added Samba configuration example.
  - Added example on how to kill hidden processes at shutdown.
  - Added ssh keygen question.
  - Enhanced passwd update question.

- February 10th, 2001. Version .09
  - Added ssh/scp question.
  - Updated mailing list information.
  - LIDS SMP status update.

- January 27th, 2001. Version .08
  - Modified Apache configuration so the server root is protected as DENY.
  - Modified mysql and courier-imap so their default directories are protected as DENY.
  - Modified ssh config to work with password authentication.
  - Added question regarding ACL reconfiguration.

- January 25th, 2001. Version .07
  - Added a much simpler fix to the lidsadm compile problem. Clarified the sealing the kernel question (hopefully). Minor corrections.

- January 24th, 2001. Version .06
  - Removed ACL example from /etc/mtab mount question because /etc/mtab is recreated at system boot and each time a file system is unmounted.
  - Added alternative fix to the lidsadm compile problem.
  - Minor corrections.

- January 22nd, 2001. Version .05
  - Minor additions to Basic System Setup sample configuration. Added section on configuring e-mail alerts.

- January 19th, 2001. Version .04
  - Minor correction to lidsadm compile problem question.

- January 17th, 2001. Version .03
  - Added information about the new file ACL inheritance "-i" option in LIDS-0.9.12. Also updated the configuration examples to use the "-i" option when required. Other minor updates including information about lidsadm compile problems, enabling/disabling capabilities, and how to setup ACLs for a new program.

- January 15th, 2001. Version .02
  - Minor corrections.

- January 15th, 2001. Version .01
  - Initial release.

## 1.12. To-do

Things that still need to be done in this document:

- Add directions for LIDS-LSM.
- Probably a lot more...

## 1.13. Where can I get this faq?

The html version of the LIDS-FAQ can be found at the LIDS (http://www.lids.org/lids-faq/lids-faq.html) website. Other formats are also available at http://www.roedie.nl/lids-faq/.

# Chapter 2. Installing LIDS

## 2.1. How do I apply the LIDS kernel patch?

Xie has included instructions (http://www.lids.org/install.html) on how to patch the kernel in the LIDS download. However, I will briefly cover the necessary steps. This example assumes your kernel sources are installed in `/usr/src/linux`.

`Always make sure you are using the latest LIDS version with the corresponding kernel version. LIDS development can go very rapid from time to time and documentation changes just as fast.`

- First you need to download the LIDS patch from www.lids.org/download.html (http://www.lids.org/download.html). Make sure you get the version that matches your kernel. (People who use kernels provided by their Linux distribution please read the following.
- Then, expand the tarball:

```
bash$ tar -zxvf lids-lids_version-kernel_version.tar.gz
```

- Apply the lids patch to the existing kernel sources:

```
bash$ cd /usr/src/linux
bash$ patch -p1 < /path/to/lids/patch/lids-lids_version-kernel_version.patch
```

- Then configure your kernel. For an excellent source of information on recompiling your Linux kernel, see the Linux Kernel HOW-TO (http://www.tldp.org/HOWTO/Kernel-HOWTO.html).

There are several kernel configuration options for LIDS. In order for LIDS to work, you must make sure the following options are enabled:

```
[*]   Prompt for development and/or incomplete code/drivers
[*]   Sysctl Support
```

## 2.2. How do I install the LIDS administration utilities (lidsadm & lidsconf)?

- **LIDS 1.1.2+**

**NOTE:** If you are upgrading LIDS, you should backup everything in the /etc/lids directory first!)

From your LIDS source directory, type:

```
bash$ tar -zvxf lidstools-version.tar.gz
bash$ cd lidstools-version
bash$ ./configure
bash$ make
bash$ su -
bash# make install
```

This will install lidsadm and lidsconf in the /sbin directory. It will also create an /etc/lids directory and place a few default configuration files in it for you. The configuration files will be updated with the proper inode and device information for your system. You will also be prompted to enter the LIDS password at this time. The inode update process can give some errors if it is looking for files that are non-existant on your system. These errors are not harmful.

If you do not wish to enable the view option (-V) in lidsadm, specify `--disable-view` when running configure.

## 2.3. What next?

Before you reboot into your LIDS enhanced kernel, you should configure your LIDS ACLs first. Otherwise your system may be unusable when you reboot. Configuring LIDS ACLs is covered later.

## 2.4. When I try to compile lidsadm, gcc reports that lidstext.h doesn't exist. How do I fix this problem?

This happens on systems where `/usr/include/linux` is not a symbolic link to `/usr/src/linux/include/linux`. The complete error message is:

```
lidsadm.c:30: linux/lidsext.h: No such file or directory make: *** [lidsadm.o] Error 1
```

To fix this problem, edit the Makefile in the lidsadm source directory and add `-I/usr/src/linux/include` to the CFLAGS option. At this point, you should be able to compile lidsadm normally.

## 2.5. A note for Debian users...

David Spreen is maintaining the Debian package for LIDS. He would like you to email package specific LIDS configurations to netzwurm AT debian DOT org (netzwurm@debian.org). He also recommends that Debian users use the Debian package for LIDS because it includes Debian specific modifications.

## 2.6. I tried to apply the LIDS patch to kernel version 2.x.x-x that is shipped with my distro and I received errors. What's wrong?

LIDS is developed using the "vanilla" kernel as developed by Linus. Many distributions,including Red Hat, Debian, and Suse, customize their kernels. There is nothing wrong with this, but you should be aware that their kernels are not the same as Linus' kernel. If you want to apply it to a non-Linus kernel, you're on your own. (Debian users, see above notice)

# Chapter 3. lidsadm and lidsconf

## 3.1. What is lidsadm?

lidsadm is the LIDS administration utility that you will use to administer LIDS on your system. This includes enabling/disabling LIDS, sealing the kernel, and viewing the status of LIDS.

## 3.2. What is lidsconf?

lidsconf is used to configure the access control lists (ACLs) for LIDS. It is also used to set the LIDS password. *NOTE:* In versions prior to LIDS 1.1.0, lidsadm also performed all of the tasks of that lidsconf now performs.

## 3.3. What options are available for lidsadm?

To get a list of the available options, enter the following:

```
bash# lidsadm -h
```

This will return the following output:

```
lidsadm version 0.4.1 for LIDS project
       Huagang Xie <xie@gnuchina.org>
       Philippe Biondi <pbi@cartel-info.fr>

Usage: lidsadm -[S|I] -- [+|-][LIDS_FLAG] [...]
       lidsadm -V
       lidsadm -h

Commands:
       -S  To submit a password to switch some protections
       -I  To switch some protections without submitting password (sealing time)
       -V  To view current LIDS state (caps/flags)
       -v  To show the version
       -h  To list this help

Available capabilities:
           CAP_CHOWN chown(2)/chgrp(2)
    CAP_DAC_OVERRIDE DAC access
 CAP_DAC_READ_SEARCH DAC read
          CAP_FOWNER owner ID not equal user ID
          CAP_FSETID effective user ID not equal owner ID
            CAP_KILL real/effective ID not equal process ID
          CAP_SETGID set*gid(2)
          CAP_SETUID set*uid(2)
         CAP_SETPCAP transfer capability
 CAP_LINUX_IMMUTABLE immutable and append file attributes
CAP_NET_BIND_SERVICE binding to ports below 1024
   CAP_NET_BROADCAST broadcasting/listening to multicast
       CAP_NET_ADMIN interface/firewall/routing changes
         CAP_NET_RAW raw sockets
```

```
        CAP_IPC_LOCK locking of shared memory segments
       CAP_IPC_OWNER IPC ownership checks
      CAP_SYS_MODULE insertion and removal of kernel modules
       CAP_SYS_RAWIO ioperm(2)/iopl(2) access
      CAP_SYS_CHROOT chroot(2)
      CAP_SYS_PTRACE ptrace(2)
       CAP_SYS_PACCT configuration of process accounting
       CAP_SYS_ADMIN tons of admin stuff
        CAP_SYS_BOOT reboot(2)
        CAP_SYS_NICE nice(2)
    CAP_SYS_RESOURCE setting resource limits
        CAP_SYS_TIME setting system time
  CAP_SYS_TTY_CONFIG tty configuration
          CAP_MKNOD mknod operation
          CAP_LEASE taking leases on files
         CAP_HIDDEN hidden process
  CAP_KILL_PROTECTED kill protected programs
       CAP_PROTECTED Protect the process from signals

Available flags:
               LIDS de-/activate LIDS locally (the shell & childs)
         LIDS_GLOBAL de-/activate LIDS entirely
         RELOAD_CONF reload config. file and inode/dev of protected programs
```

# 3.4. What options are available for lidsconf?

To get a list of the available options, enter the following:

```
bash# lidsconf -h
```

This will return the following output:

```
lidsconf version 0.4.1 for the LIDS project
       Huagang Xie <xie@gnuchina.org>
       Philippe Biondi <philippe.biondi@webmotion.net>

Usage: lidsconf -A [-s subject] -o object [-d] [-t from-to] [-i level] -j ACTION
       lidsconf -D [-s file] [-o file]
       lidsconf -Z
       lidsconf -U
       lidsconf -L [-e]
       lidsconf -P
       lidsconf -v
       lidsconf -[h|H]

Commands:
       -A,--add To add an entry
       -D,--delete      To delete an entry
       -Z,--zero        To delete all entries
       -U,--update      To update dev/inode numbers
       -L,--list        To list all entries
       -P,--passwd      To encrypt a password with RipeMD-160
       -v,--version     To show the version
```

```
       -h,--help        To list this help
       -H,--morehelp    To list this help with CAP/SOCKET name

subject: -s,--subject subj
       can be any program, must be a file
object: -o,--object [obj]
       can be a file, directory or Capability, Socket Name
ACTION: -j,--jump
       DENY     deny access
       READONLY read only
       APPEND   append only
       WRITE    writable
       GRANT    grant capability to subject
       IGNORE   ignore any permissions set on this object
       DISABLE  disable some extension feature
OPTION:
      -d,--domain       The object is an EXEC Domain
      -i,--inheritance Inheritance level
      -t,--time Time dependency
      -e,--extended     Extended list
```

## 3.5. Nice, what do all those capabilities mean?

Capabilities allow you to set certain system wide permissions on what actions are allowed or disallowed. If you disable `CAP_SETUID` then it's impossible for any program to transfer the UID. With LIDS you can enable/disable certain capabilities for certain programs. The function of each capability is described in `/etc/lids/lids.cap` or in `/path/to/lidstools/example/lids.cap` if you didn't install LIDS yet.

# Chapter 4. LIDS Administration

## 4.1. How do I set my LIDS password?

If the install command somehow didn't ask you to set you password then you must set it before you reboot into your LIDS enhanced kernel by entering the following at the command prompt:

```
bash# lidsconf -P
```

You will then be prompted for a LIDS password:

```
MAKE PASSWD
enter new password:
reenter new password:
wrote password to /etc/lids/lids.pw
```

This will write your RipeMD-160 encrypted password to the `/etc/lids/lids.pw` file. You need this password if you want to change some ACL's, capabilities or when you want to start a LIDS free session.

## 4.2. How do I change my LIDS password once it is set?

You must first create a LIDS free session. Then set your password using the "-P" option just like you did the first time (you will not be prompted for your current password). After resetting your LIDS password, you must tell LIDS to reload its configuration files.

*WARNING:* It is possible for someone with root access to obtain the LIDS password if /dev/pts is mounted. In order to prevent this, you can either unmount /dev/pts or protect it.

## 4.3. What is a LIDS free session and how do I create one?

A LIDS free session (LFS) is a terminal session that is not restricted by LIDS. This option is available so you can administer your system without having to reboot into a non-LIDS kernel. In order for this to work, you must have selected this option when you compiled your LIDS enhanced kernel:

```
  [*] Allow switching LIDS protections
```

To create an LFS, enter the following at the prompt:

```
bash# lidsadm -S -- -LIDS
```

You will then be prompted for your LIDS password. This terminal is now LIDS free. It will remain LIDS free until you:

- Enable LIDS again (`lidsadm -S -- +LIDS`).
- Log out of the terminal.

You can only have one LFS active at any one time. Even though `lidsadm -S -- -LIDS` will not fail if entered on another terminal, you can have only one LFS.

## 4.4. I created a LIDS free session, but LIDS still appears to be active! What's wrong?

This can happen if you create an LFS on a virtual console and then switch to another virtual console and try to administer your machine. To clear it up, try enabling LIDS and then disabling it again (entering passwords when prompted):

```
# lidsadm -S -- +LIDS
# lidsadm -S -- -LIDS
```

**NOTE:** Be sure that there is no other administrator in an LFS because you will close his LFS!

## 4.5. How do I tell LIDS to reload its configuration files?

In order for LIDS to be able to reload its configuration files, you must enable this option when you configure your LIDS enhanced kernel:

```
  [*]  Allow switching LIDS protections
  (3)    Number of attempts to submit password
  (30)     Time to wait after a fail (seconds)
  [ ]    Allow remote users to switch LIDS protections
  [ ]    Allow any program to switch LIDS protections
  [*]    Allow reloading config. file   <-----------------------------
```

NOTE: You must allow switching LIDS protections in order to enable reloading of configuration files. From an LFS (or with LIDS_GLOBAL disabled), execute the following command to instruct LIDS to reload its configuration files:

```
# lidsadm -S -- +RELOAD_CONF
```

This will reload the following configuration files:

- `/etc/lids/lids.conf` - LIDS ACL configuration file.
- `/etc/lids/lids.cap` - LIDS capabilities file.
- `/etc/lids/lids.pw` - LIDS password file.
- `/etc/lids/lids.net` - LIDS mail alert configuration file.

## 4.6. Help!!! My system is totally unusable! What do I do?

You can reboot into a non-LIDS enhanced kernel, or boot into your LIDS enhanced kernel with LIDS disabled to try and patch things up. To boot with LIDS disabled, specify `lids=0` at the lilo prompt. For example, if your LIDS enhanced kernel is called `lids-kernel` you would enter the following at the lilo prompt:

```
lilo: lids-kernel lids=0
```

That's the easy part. The difficult part is getting your LIDS enabled system to shutdown. You may not be able to shutdown successfully depending on your LIDS configuration.

**WARNING:** Rebooting your LIDS enabled system when it is not properly configured can cause file system corruption and/or loss of data!!

## 4.7. I've updated/moved a system binary. How do I tell LIDS that the file changed/moved?

Whenever the device that a file resides on, or a file's inode number changes, you must update your `/etc/lids/lids.conf` file with the proper information. Fortunately, Xie has provided us with an option just for this occasion:

```
bash# lidsadm -U
```

You must then reload the configuration files.

## 4.8. OK, without rebooting, how do I completely disable LIDS?

Besides using an LFS, LIDS can be turned off globally. This will only work if you compiled the option into your kernel.

```
bash# lidsadm -S -- -LIDS_GLOBAL
```

When `LIDS_GLOBAL` is disabled, your system will operate like a "normal" Linux system. To re-enable LIDS globally, perform the opposite:

```
bash# lidsadm -S -- +LIDS_GLOBAL
```

**NOTE:** This will not affect your LFS if you currently have one enabled.

## 4.9. What does it mean to "seal the kernel"?

At the end of the bootup process, you should seal the kernel. This sets the global capabilities on your system according to your `/etc/lids/lids.cap` file. File ACLs are enforced even before the kernel is sealed, however. To seal the kernel, put the following at the end of your `rc.local` (assuming SysV style init):

```
/sbin/lidsadm -I
```

The "-I" option is only used to seal the kernel. After it's sealed, you must use the "-S" option to make changes to your system. **WARNING:** If you do not seal your kernel at boot time, you will not receive the full benefits of a LIDS enhanced system.

## 4.10. How do I view the status of my LIDS system?

In order to use the "-V" option, you must have compiled lidsadm with the view option enabled. (which is standard behaviour, see above). At the command line, enter:

```
bash# lidsadm -V
```

This will produce output similar to the following on a 2.4.x kernel:

```
VIEW
                     CAP_CHOWN 0
                CAP_DAC_OVERRIDE 0
             CAP_DAC_READ_SEARCH 0
                      CAP_FOWNER 0
                      CAP_FSETID 0
                        CAP_KILL 0
                      CAP_SETGID 0
                      CAP_SETUID 0
                     CAP_SETPCAP 0
             CAP_LINUX_IMMUTABLE 0
           CAP_NET_BIND_SERVICE 0
              CAP_NET_BROADCAST 0
                   CAP_NET_ADMIN 0
                     CAP_NET_RAW 0
                    CAP_IPC_LOCK 0
                   CAP_IPC_OWNER 0
                  CAP_SYS_MODULE 0
                   CAP_SYS_RAWIO 0
                  CAP_SYS_CHROOT 0
                  CAP_SYS_PTRACE 0
                   CAP_SYS_PACCT 0
                   CAP_SYS_ADMIN 0
                    CAP_SYS_BOOT 1
                    CAP_SYS_NICE 0
                CAP_SYS_RESOURCE 1
                    CAP_SYS_TIME 0
              CAP_SYS_TTY_CONFIG 0
                      CAP_MKNOD 0
                      CAP_LEASE 0
                     CAP_HIDDEN 1
              CAP_KILL_PROTECTED 0
                   CAP_PROTECTED 0
                            LIDS 0
                     LIDS_GLOBAL 1
                     RELOAD_CONF 0
```

As you can see from the output above, this system has an LFS active. However, LIDS is enabled globally. The items with a "1" next to them are enabled, and those items with a "0" next to them are disabled. Except for the last two capabilities, root normally has all of the above capabilities. Thanks to LIDS, root only has capabilities CAP_SYS_BOOT, CAP_SYS_RESOURCE, and CAP_HIDDEN in this particular case (NOTE: CAP_HIDDEN isn’t a capability provided by the standard Linux kernel).

## 4.11. How do I configure the port scan detector in LIDS?

You don’t. As long as you selected the option when you configured your LIDS enhanced kernel, the port scan detector is enabled.

```
[*]  Port Scanner Detector in kernel
```

## 4.12. What are the subject and object in a LIDS ACL?

The subject is a program that can run on a Linux system, such as a binary or shell script. The object is what the subject wants to access. This includes files, directories, capabilities, etc.

## 4.13. Can I enable/disable a system capability without modifying /etc/lids/lids.cap and reloading the configuration files?

Yes. However, this method will not save the changes past system shutdown. To enable a capability:

```
bash# lidsadm -S -- +CAP_SYS_ADMIN
```

To disable a capability:

```
bash# lidsadm -S -- -CAP_SYS_ADMIN
```

## 4.14. I’ve reconfigured my LIDS ACLs, but my changes don’t seem to take effect. What’s wrong?

There are two things you should do when re-configuring LIDS:

1. Reload the configuration files.
2. Restart the service or services that your changes affected.

## 4.15. Why won’t lidsconf -L list my ACLs?

lidsconf -L must be used from an LFS or when LIDS_GLOBAL is disabled. If neither of those conditions are true, you will see the following error message:

```
lidsconf: can not open conf file
reason:: Permission denied
LIST
```

## 4.16. Is there anyway to reduce the number of LIDS violations that get reported on the console?

Yes. The syslog init script can be modified to start klogd with the "-c" option. This options sets the default level of system messages that get logged to the console. Any message with a value less than the value specified will appear on the console (see `include/linux/kernel.h`). For example:

```
klogd -c 4
```

Tells klogd to log all messages below level 4 will be logged to the console.

Another way to change the console log level is to modify the values in `/proc/sys/kernel/printk`. View the documentation provided in `/usr/src/linux/Documentation/sysctl/kernel.txt` for more information.

## 4.17. Should I be concerned about the LD_PRELOAD environment variable with LIDS?

Yes, if you are running an version of LIDS older than 1.1.1preX please read on.

For setuid programs, the LD_PRELOAD env var is "cleansed" so that it can't affect the libraries loaded by a program (with the exception of recent glibc vulnerabilities).

Problems arise when you grant special capabilities or file access permissions to non-setuid binaries. Since the LD_PRELOAD env var isn't "cleansed" before loading libraries, someone with malicious intent could load a trojaned library and it would have the same special capabilities/file access permissions that were given to the original program.

*Possible options to reduce your risk:*

- Any program with special capabilities or file access permissions should be restricted with the standard unix file permissions so that not everyone is allowed to execute it (e.g. `chmod o-rwx /path/to/program` )
- Another option may be to make the file setuid and change the ownerhip to a non-root user. That way the LD_PRELOAD env var is "cleansed" before the program is executed.

**SECURITY UPDATE:** Starting with LIDS 1.1.1preX, the LD_PRELOAD environment variable is disabled automatically for any program that has been given special privileges via LIDS. This has also been back ported to LIDS 0.10.3.

## 4.18. When I boot up, the message "read password file error" appears. How do I fix the problem?

This happens when you somehow forget or did not set the LIDS password before booting into LIDS the first time. To fix the problem, reboot your machine (see booting an unusable system) and set your LIDS password.

# 4.19. How do I check if LIDS is enabled/disabled??

If you have compiled the lidsadm with ’make VIEW=1’ then you can use ’lidsadm -V’ to see if LIDS is enabled. If it says ’LIDS_GLOBAL 0’ then LIDS is disabled. If it says ’LIDS 0’ then someone is in a Lids Free Session. If you haven’t compiled lidsadm with the VIEW option there are several ways to determine if LIDS is running.

1. You can check for the line ’Linux Intrusion Detection System `<lids-version>` for `<kernel-version>` doesn’t start’ is in your dmesg. If it says ’Linux Intrusion Detection System `<lids-version>` for `<kernel-version>` starts’ then LIDS is started of course.
2. You can try to do something that you are sure of you can’t do to see if LIDS takes action. If there’s no action the LIDS is not active.

# Chapter 5. Configuring LIDS

## 5.1. How do I protect a file as read only?

```
bash# lidsconf -A -o /some/file -j READONLY
```

This will prevent anyone (including root) from modifying or deleting `/some/file` as long as LIDS is enabled. If you are in an LFS, you are free to modify `/some/file` assuming you have appropriate file system permissions and the partition isn't mounted read-only.

## 5.2. OK, so how do I protect a directory as read only?

Same as above, only specify `/some/directory`

```
bash# lidsconf -A -o /some/directory  -j READONLY
```

When the object is a directory, LIDS protects the directory itself, and it recursively protects everything underneath it *within the same file system*. (e.g. **LIDS ACLs do not cross file system boundaries!**) This is very important to remember so you don't accidentally leave part of your system unprotected.

A directory that you may want to protect as read only is the `/etc` directory.

```
bash# lidsconf -A -o /etc -j READONLY
```

## 5.3. How can I hide a file/directory from everyone?

```
bash# lidsconf -A -o /some/file_or_directory   -j DENY
```

Again, this will prevent even root from accessing it. And, if it is a directory, all files and directories underneath it are also hidden (within the same file system, of course).

## 5.4. How can I protect log files so they can only be appended to?

```
bash# lidsconf -A -o /some/log/file  -j APPEND
```

This will allow someone to write to the end of the file while at the same time preventing him/her from erasing or modifying its existing contents. An easy way to protect your system logs as append only would be:

```
bash# lidsconf -A -o /var/log  -j APPEND
```

## 5.5. If nothing is allowed to read my /etc/shadow file, how can I authenticate myself to the system?

In order to allow users to authenticate themselves to the system, it is necessary to give certain programs read only access to the `/etc/shadow`. Some of the programs you may want to consider giving read access to are: `login`, `sshd`, `su`, and `vlock`. To allow the login program to read `/etc/shadow`, use the following ACL:

```
bash# lidsconf -A -s /bin/login -o /etc/shadow -j READONLY
```

The "-s" option specifies a subject, which is `/bin/login` in this case. We are giving the subject read only access to the object (`/etc/shadow` in this case). This will protect all files under `/var/log` as append only. As with READ and DENY, this target is also recursive.

## 5.6. If I protect /etc as read only, how will mount be able to write to /etc/mtab?

It won't. To fix this problem, you can remove the `/etc/mtab` file and replace it with a symbolic link to `/proc/mounts`. In order for this to work, you must modify your startup scripts to use the "-n" option with every `mount` and `umount` command. This tells `mount` and `umount` not to update the `/etc/mtab` file.

For example, if you find:

```
mount -av -t nonfs,noproc
```

in your init scripts, you will need to change it to:

```
mount -av -n -t nonfs,noproc
```

These `mount` commands may be scattered throughout your init scripts. Use grep to make sure you catch them all. You will also want to modify all of the `umount` commands in the same manner.

## 5.7. LIDS complains that it can't write to my modules.dep file during startup. What's wrong?

This happens when you protect `/lib` as read only (a good thing to do). The error received is something similar to:

```
LIDS: depmod (3 12 inode 16119) pid 13203 user (0/0) on tty2: Try to open /lib/modules/2.2.18/mod
```

This occurs during startup because the `/etc/rc.d/rc.sysinit` init script tries to recreate all of your module dependencies. Normally this is not needed because the module dependencies don't change unless you add, change, or delete modules. The error is harmless, but if you don't like seeing it, you can simply comment out the line in your `/etc/rc.d/rc.sysinit` script that recreates the module dependencies (Look for `depmod -a` or something similar).

## 5.8. If I protect my logs as append only, how will logrotated rotate my logs?

It won't. Log rotation is something that will have to be done manually by executing your log rotation utility when LIDS_GLOBAL is disabled. You should disable the cron job that initiates log rotation. (See below for a possible alternative).

## 5.9. Why can't I just give my log rotation utility write access to the directory containing my log files so it can rotate them?

You can, but it's not recommended. If someone were to break into your system, even though they couldn't modify your logs, they could rotate them enough times (by executing the log rotation utility manually) that the log containing the information gathered during the intrusion is dropped off the face of the earth. This is part of the price you pay for high security.

An alternative solution to giving your log rotation utility write access to `/var/log`, is to give the cron daemon write access to /var/log and make it inheritable:

```
lidsconf -A -s /usr/sbin/crond -i -o /var/log -j WRITE
```

This prevents someone from manually executing your log rotation utility, but will allow it to work when it is executed by the cron daemon. **WARNING:** If a vulnerability is found in your cron daemon, someone could exploit it and wipe your logs since cron would have write access to `/var/log`. This defeats the purpose of using MAC in the first place since your access controls can now be circumvented if a vulnerability is found. Use this option at your own discretion!

**UPDATE:** Because of the new time restriction feature, it is recommended that if crond has write access to `/var/log`, it should be limited to a specific time period. For example, if logrotated is executed every day at 6:00 AM by crond, limit crond's write access to a 1 minute window:

```
/sbin/lidsconf -A -s /usr/sbin/crond -i 2 -o /var/log -t 0600-0601 -j WRITE
```

If 1 minute isn't long enough, extend the time by 1 minute increments until logrotated is executed successfully.

## 5.10. When LIDS is active, my file systems won't unmount during shutdown. What do I do?

This happens when you have disabled the CAP_SYS_ADMIN capability globally and have not given the proper authority to unmount your file systems to your shutdown script(s). For example, on Red Hat 6.2, the `/etc/rc.d/init.d/halt` script unmounts your file systems. You must give it the CAP_SYS_ADMIN capability so it can unmount your file systems:

```
bash# lidsconf -A -s /etc/rc.d/init.d/halt -o CAP_SYS_ADMIN -i 1 -j GRANT
```

The target "GRANT" tells LIDS to grant the subject (`/etc/rc.d/init.d/halt` in this case) the CAP_SYS_ADMIN capability. The "-i 1" option sets the "inheritance level" of the ACL to 1.

Beware that this also allows anyone who can execute your `/etc/rc.d/init.d/halt` script to unmount your file systems. If you have physical access to your box, you may just want to turn off LIDS_GLOBAL before shutting

down your system rather than grant capabilities to your shutdown scripts. However, if you have a UPS that can shutdown your system in case of power failure, you may not be around to disable LIDS_GLOBAL.

## 5.11. Why can’t I start a service that runs on a privileged port as root?

Services that run a privileged port (those below 1024) require the CAP_NET_BIND_SERVICE capability in order to bind to the port. If you have disabled this capability globally in the `/etc/lids/lids.cap` file, you must either grant the program that capability

```
bash# lidsconf -A -s /usr/local/bin/apache -o CAP_NET_BIND_SERVICE 80 -j GRANT
```

or, start the service when LIDS_GLOBAL is disabled.

## 5.12. Why can’t I start a service that runs on a privileged port from an LFS?

An LFS applies to a single terminal session. A daemon forks itself in order to separate itself from the controlling terminal. Once this happens, it is no longer connected to the LFS on your terminal and is now protected by LIDS.

## 5.13. How do I disable/enable capabilities?

The `/etc/lids/lids.cap` file contains a list of all the capabilities available under a LIDS enhanced Linux kernel. Those that have a "+" in front of them are enabled, and those with a "-" in front of them are disabled. To change the status of a capability, simply edit the text file and change the "+" to a "-" to disable a capability and vice-versa to enable it. After you’re done editing the file, you must tell LIDS to reload the configuration files.

## 5.14. Why won’t the X Window System work with LIDS enabled?

The X server that you are using requires the CAP_SYS_RAWIO capability. Try

```
bash# lidsconf -A -s /path/to/your/X_server -o CAP_SYS_RAWIO -j GRANT
```

## 5.15. With all of these ACLs, how can I possibly keep track of my configuration?

It is recommended that you create a shell script of all the ACLs that you wish to add to your system. That way you don’t accidentally leave something unprotected when you make changes to your system. You can start the script

out by flushing your old ACLs so you don't create duplicates.

```
bash# lidsconf -Z
```

To protect this shell script, you can either create an ACL to DENY access to it, or put it in the `/etc/lids` directory since it is automatically protected as DENY.

## 5.16. How can I give init write access to /etc/initrunlvl so LIDS doesn't complain about it during startup and shutdown?

Unfortunately, there isn't much you can do about this. Because `init` recreates this file each time you boot, it will have a different inode number every time. This makes it difficult for LIDS to handle. It is a harmless error, and your system will still function properly without `/etc/initrunlvl`.

## 5.17. Can a process inherit file ACLs from its parent?

Yes. Up until version 0.9.12-2.2.18, this was the default behavior. Now the default is for children *not* to inherit the file ACLs from their parents. To allow a file ACL to be passed from a parent process to a child process, you must use the "-i <inheritance level>" option.

Where "inheritance level" (a.k.a. TTL) determines how far the ACL is inherited. If the TTL specified is 1, then the subject specified in the ACL and all of its children will inherit the ACL. However, the children's children (a.k.a. a grandchild of the subject in the ACL) will not inherit the ACL (a TTL of 2 would be needed for this to occur).

**Note:** These same inheritance rules apply to ACLs that grant capabilities.

**SECURITY UPDATE:** Starting with LIDS 1.1.1preX and 0.10.1, only protected programs are allowed inherit ACLs from their parent. Allowing non-protected processes to inherit ACLs led to an exploit.

## 5.18. Help! I can't seem to get program xyz to work under LIDS. How do I determine what files/capabilities it needs access to?

The first thing to do is simply try running the program and see what violations get reported by LIDS. However, many times this doesn't give you enough information. When this happens, you can try using strace to follow the program through and see which system call fails. This will usually give you a good indication as to which capability is being violated.

NOTE: If you have disabled CAP_SYS_PTRACE globally, you will need to **temporarily** give strace the CAP_SET_PTRACE capability so it can trace your program while LIDS is enabled.

# 5.19. How do I give passwd the proper permissions to update the /etc/shadow file?

Unfortunately there isn't an easy solution. This is because the `passwd` utility recreates the `/etc/shadow` file every time you change your password. Because of this, it will start on a different inode each time you use the `passwd` utility successfully.

For the system administrator, there is an easy work around. Start an LFS and use the `passwd` utility from within the LFS. If there are users that need to change their passwords, LDAP can provide an alternate means of client authentication that will allow users to change their passwords.

There is an option available that will allow users to change their system passwords when using the standard unix system filesUNIX authentication, but it's *not* recommended. `/usr/bin/passwd` can be given write access to /etc so it can always modify the shadow file regardless of it's inode number.

**WARNING:** If someone were to find a vulnerability in `/usr/bin/passwd`, or any of the libraries/PAM modules that it uses, he/she could potentially gain write access to your `/etc` directory. This defeats the purpose of using MAC in the first place since your access controls can now be circumvented if a vulnerability is found. Use this option at your own discretion!

If you are going to give `/usr/bin/passwd` write access to `/etc`, it is recommended that you create ACLs to protect every file and directory under `/etc` that you don't want `/usr/bin/passwd` to be able to modify. This will significantly reduce the risk (and possibly completely remove it if done correctly) mentioned above. For example:

```
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc                    -j WRITE
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/hosts.allow        -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/hosts.deny         -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc0.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc1.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc2.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc3.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc4.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc5.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/rc6.d              -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/init.d             -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/cron.d             -j READONLY
/sbin/lidsconf -A -s /usr/bin/passwd -o /etc/pam.d              -j READONLY
...
```

The above is not a complete list by any stretch of the imagination, but it's a start. Also keep in mind that anytime you add a file or directory to `/etc` that you don't want `passwd` to have access to, you must create a new ACL to protect it.

*A note about updating inodes:* If you have defined ACLs to grant access to `/etc/shadow` or `/etc/passwd` you must make sure to tell lids to update the inodes and then reload it's config files. Otherwise you may encounter problems.

For example: Assume `/etc/passwd` is protected as DENY, and `/bin/login` can read `/etc/passwd`. If the inodes aren't updated after changing a password, problems may arise the next time someone tries to log in. `/bin/login` won't be able to read `/etc/passwd` and you won't be able to log in, or worse yet, you will be able to login by just pressing the <ENTER> key.

## 5.20. Why doesn't ssh or scp work when LIDS is enabled?

By default, ssh/scp try to use a privileged port for the source port when creating an outgoing connection. This requires the CAP_NET_BIND_SERVICE capability. However, you can specify the following option in the `ssh_config` file to force it to use a port above 1023 for the source port:

```
UsePrivilegedPort no
```

Or, you can grant CAP_NET_BIND_SERVICE to ssh (since scp uses ssh, it will work also):

```
lidsconf -A -s /usr/bin/ssh -o CAP_NET_BIND_SERVICE 22 -j GRANT
```

## 5.21. Open-SSH won't start at boot time. LIDS reports that bash tried to access a hidden file. How can I fix this?

This can happen when you protect your private keys with a default policy of DENY. The init script provided in the openssh-server rpm checks to see if the private key files exist under `/etc/ssh`. If the script can't find them, it executes ssh-kegen to generate them. The keys are actually there, ssh-keygen fails and the startup script exits.

To fix this, remove the check for the key files in the startup script:

```
start)
      # Create keys if necessary
      #do_rsa_keygen;  <------------ Comment out these lines
      #do_dsa_keygen;

      echo -n "Starting sshd: "
      if [ ! -f $PID_FILE ] ; then
              sshd
              RETVAL=$?
              if [ "$RETVAL" = "0" ] ; then
                      success "sshd startup"
                      touch /var/lock/subsys/sshd
              else
                      failure "sshd startup"
              fi
      fi
      echo
      ;;
```

**NOTE:** This means the private keys will have to be generated manually prior to starting sshd. Otherwise, it will fail to start.

## 5.22. Some of my file systems won't unmount at shutdown because I have hidden processes running. How can I kill

# them?

A hidden process can still be killed if you know it's process id (pid). Many systems store the pid of all the processes started at boot time somewhere under `/var` (`/var/run` is commonly used). Your shutdown scripts can be modified to read the pids from these files and send them the appropriate signals.

For example, if your system stores the pids in `/var/run/<process_name>.pid`, then you can add the following lines to your shutdown scripts:

```
for p in ‘ls /var/run/*.pid‘
do
   kill -15 ‘cat $p‘
done
sleep 5
sync;sync;sync

for p in ‘ls /var/run/*.pid‘
do
   kill -9 ‘cat $p‘
done
sleep 5
sync;sync;sync
```

The CAP_KILL and CAP_INIT_KILL capabilities must be granted to the shutdown script containing these lines. It is probably a good idea to hide the `/var/run` directory from everything but the init scripts too.

Another option would be to just send every process the TERM and KILL signals.

```
MAX_PROC=65535
trap : 1 2 15
I=1;while (( $I < $MAX_PROC ));do
        I=$(($I+1));
        if (( $$ != $I ));then
                kill -15 $I;
        fi;
done
sleep 5
sync;sync;sync;
I=1;
while (( $I < $MAX_PROC ));do
        I=$(($I+1));
        if (( $$ != $I ));then
                kill -9 $I;
        fi;
done
sync;sync;sync
```

Nenad Micic wrote his own C program (http://www.bg.ac.yu/~mclaffin/lids/brc.c) to kill hidden processes at shutdown.

# 5.23. I just want to start with a basic configuration. Can you recommend a setup that will provide additional protection and still leave most of my system functioning as normal?

Make sure to select the following kernel options:

```
...
[*]    Security alert when execing unprotected programs before sealing
[*]      Do not execute unprotected programs before sealing lids
...
[*]    Allow switching LIDS protections
...
[*]      Allow reloading config. file
```

A good starting point would be to protect your init scripts, system binaries, and libraries (Note that these may vary depending upon your distro):

```
/sbin/lidsconf -A -o /etc/rc0.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc1.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc2.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc3.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc4.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc5.d                 -j READONLY
/sbin/lidsconf -A -o /etc/rc6.d                 -j READONLY
/sbin/lidsconf -A -o /etc/init.d                -j READONLY
/sbin/lidsconf -A -o /etc/rc                    -j READONLY
/sbin/lidsconf -A -o /etc/rc.local              -j READONLY
/sbin/lidsconf -A -o /etc/rc.sysconfig          -j READONLY

/sbin/lidsconf -A -o /bin                       -j READONLY
/sbin/lidsconf -A -o /sbin                      -j READONLY
/sbin/lidsconf -A -o /lib                       -j READONLY

/sbin/lidsconf -A -o /usr/bin                   -j READONLY
/sbin/lidsconf -A -o /usr/sbin                  -j READONLY
/sbin/lidsconf -A -o /usr/lib                   -j READONLY
```

If `/usr/local` is on a separate partition, add the following ACLs also:

```
/sbin/lidsconf -A -o /usr/local/bin             -j READONLY
/sbin/lidsconf -A -o /usr/local/sbin            -j READONLY
/sbin/lidsconf -A -o /usr/local/lib             -j READONLY
```

You should also disable CAP_SYS_RAWIO and CAP_SYS_PTRACE in the `/etc/lids/lids.cap` file. If you don’t disable CAP_SYS_RAWIO, then someone can override the above file protections by writing directly to your devices.

If you are running the X Window System, please see above about getting X to work under LIDS.

# 5.24. Is it possible to limit access based on time of day?

Yes. There is a new feature in LIDS version 0.10.1 for 2.2.19 and version 1.0.10 for 2.4.5 that allows a time restriction to be placed on ACLs. For example, to only allow logins between the hours of 9:00 AM and 6:00 PM (18:00):

```
/sbin/lidsconf -A -s /bin/login -o /etc/shadow -t 0900-1800 -j READONLY
```

Now, `/bin/login` can only read the `/etc/shadow` file during the specified time period and any login attempts outside of that time period will fail. You can also use the "!" operator for negation (e.g. The ACL permits access all the time except for the time period listed).

If you grant privileges to crond based on time restrictions, it is *highly recommended* that you hide your crontabs from everyone (including root), and only allow crond to read them. Otherwise, someone could figure out what time of day they should try and exploit something by looking at your crontabs. Remember to protect the system crontabs as well as the user crontabs.

For example, the following should be hidden:

```
/var/spool/cron/
/etc/crontab
/etc/cron.hourly/
/etc/cron.daily/
/etc/cron.weekly/
/etc/cron.monthly/
/etc/cron.d/
```

**WARNING:** Because this new feature relies on the system time, you should *not* grant CAP_SYS_RAWIO to any program that can change the system time (e.g. `/sbin/hwclock`. This would allow someone to bypass the time restriction by changing the system time.

# 5.25. How do I limit the ports that a program can bind to?

As of version 0.10.1 for 2.2.19 and version 1.0.11 for 2.4.6, you can limit the privileged ports that a program can bind to. When granting CAP_NET_BIND_SERVICE to a program, specify the port or ports that the program is allowed to bind to after the capability, like this:

```
/sbin/lidsconf -A -s /bin/httpd -o CAP_NET_BIND_SERVICE 80-80 -j GRANT
```

Or, if you also need to bind to port 443 for SSL:

```
/sbin/lidsconf -A -s /bin/httpd -o CAP_NET_BIND_SERVICE 80-80,443-443 -j GRANT
```

If you have a program that requires a range of ports, try this:

```
/sbin/lidsconf -A -s /path/to/program -o CAP_NET_BIND_SERVICE 423-867 -j GRANT
```

## 5.26. If I make /etc/mtab a symbolic link to /proc/mounts, will user quotas still work?

Yes, as long as you are starting quotaon with the "-a" option.

## 5.27. When I edit a file protected by LIDS, it appears to lose it’s LIDS protections. Why?

Many editors (e.g. vi) copy the file you are editing to a temporary file. All your changes are made to that temporary file. When you exit the editor, the temporary file is moved over the original file. This changes the inode of the original file and any previous LIDS ACLs that affected the file will no longer work. Type:

```
/sbin/lidsconf -U
```

To update the inodes in the lids.conf file.

## 5.28. When I update my LIDS configuration some processes seem to lose their capabilities

This can happen when a process got it’s capabilities through inheritance. Think of the following:

The parent process gives it’s capabilities to a child proces, the parant process exits but the child remains running. If you start an LFS, change some ACL’s and reload your config, LIDS will re-attach the capabilities based on the parents process capabilities and it’s own capabilities. If the parent process is not running anymore the process will not get those capabilities again and may give errors.

# Chapter 6. Configuring Security Alerts

## 6.1. Which kernel configuration options do I need to select in order to send security alerts through the network?

```
[*]   Send security alerts through network
[ ]      Hide klids kernel thread
(3)      Number of connection tries before giving up
(30)     Sleep time after a failed connection
(16)     Message queue size
[*]      Use generic mailer pseudo-script
```

The first option enables the use of security alerts. The second option allows you to hide the process that sends the alerts. Until you have your mail notification working, it is recommended that you leave this option disabled because it will also prevents error messages from being logged. The last option tells LIDS to use the generic mailer script provided with LIDS to send any alert messages to your mail server. This is currently the only option.

## 6.2. Where do I specify the mail server information and e-mail address to send the LIDS alerts to?

All information required for sending security alerts must be configured in the `/etc/lids/lids.net` file. A description of each option is provided in the configuration file itself. When specifying an e-mail address, be sure not to leave *any* leading or trailing spaces around the e-mail address. This may cause problems with delivery. For example, the following two MAIL_TO examples won't work:

```
"MAIL_TO= steve@somedomain.org"
"MAIL_TO=steve@somedomain.org "
```

NOTE: The double quotes are used only to show you the trailing space. They should not be included in your configuration.

After making changes to the `/etc/lids/lids.net` file, you must tell LIDS to reload it's configuration files.

## 6.3. LIDS can't seem to deliver alerts to my qmail SMTP server. Is there a fix for this?

Yes. For LIDS versions 0.9.12 and older, a patch is required in order to make LIDS e-mail alerts work with a qmail SMTP mail server. The patch can be found here: http://www.egroups.com/message/lids/1896.

# Chapter 7. Sample Configurations

**Note:** Because LIDS is progressing very quickly, and the fact that software packages change, some of these configurations may not work correctly "out of the box". However, they should provide a very good starting point for anyone interested in configuring one of the services listed here.

## 7.1. Basic System Setup

The following is a sample configuration for basic system setup.

```
# Protect System Binaries
#
/sbin/lidsconf -A -o /sbin                                  -j READONLY
/sbin/lidsconf -A -o /bin                                   -j READONLY

# Protect all of /usr and /usr/local
# (This assumes /usr/local is on a separate file system).
#
/sbin/lidsconf -A -o /usr                                   -j READONLY
/sbin/lidsconf -A -o /usr/local                             -j READONLY

# Protect the System Libraries
#(/usr/lib is protected above since /usr/lib generally isn't
# on a separate file system than /usr)
#
/sbin/lidsconf -A -o /lib                                   -j READONLY

# Protect /opt
#
/sbin/lidsconf -A -o /opt -j READONLY

# Protect System Configuration files
#
/sbin/lidsconf -A -o /etc                                   -j READONLY
/sbin/lidsconf -A -o /usr/local/etc                         -j READONLY
/sbin/lidsconf -A -o /etc/shadow                            -j DENY
/sbin/lidsconf -A -o /etc/lilo.conf                         -j DENY

# Enable system authentication
#
/sbin/lidsconf -A -s /bin/login -o /etc/shadow              -j READONLY
/sbin/lidsconf -A -s /usr/bin/vlock -o /etc/shadow          -j READONLY
/sbin/lidsconf -A -s /bin/su -o /etc/shadow                 -j READONLY
/sbin/lidsconf -A -s /bin/su \
                  -o CAP_SETUID                             -j GRANT
/sbin/lidsconf -A -s /bin/su \
                  -o CAP_SETGID                             -j GRANT

# Protect the boot partition
#
/sbin/lidsconf -A -o /boot                                  -j READONLY

# Protect root's home dir, but allow bash history
#
/sbin/lidsconf -A -o /root                                  -j READONLY
```

```
/sbin/lidsconf -A -s /bin/bash -o /root/.bash_history    -j WRITE

# Protect system logs
#
/sbin/lidsconf -A -o /var/log                            -j APPEND
/sbin/lidsconf -A -s /bin/login -o /var/log/wtmp         -j WRITE
/sbin/lidsconf -A -s /bin/login -o /var/log/lastlog      -j WRITE
/sbin/lidsconf -A -s /sbin/init -o /var/log/wtmp         -j WRITE
/sbin/lidsconf -A -s /sbin/init -o /var/log/lastlog      -j WRITE
/sbin/lidsconf -A -s /sbin/halt -o /var/log/wtmp         -j WRITE
/sbin/lidsconf -A -s /sbin/halt -o /var/log/lastlog      -j WRITE
/sbin/lidsconf -A -s /etc/rc.d/rc.sysinit \
                  -o /var/log/wtmp -i 1                  -j WRITE
/sbin/lidsconf -A -s /etc/rc.d/rc.sysinit \
                  -o /var/log/lastlog -i 1               -j WRITE

# Startup
#
/sbin/lidsconf -A -s /sbin/hwclock -o /etc/adjtime       -j WRITE


# Shutdown
#
/sbin/lidsconf -A -s /sbin/init -o CAP_INIT_KILL         -j GRANT
/sbin/lidsconf -A -s /sbin/init -o CAP_KILL              -j GRANT

# Give the following init script the proper privileges to kill processes and
# unmount the file systems.  However, anyone who can execute these scripts
# by themselves can effectively kill your processes.  It's better than
# the alternative, however.
#
# Any ideas on how to get around this are welcome!
#
/sbin/lidsconf -A -s /etc/rc.d/init.d/halt \
                  -o CAP_INIT_KILL -i 1                  -j GRANT
/sbin/lidsconf -A -s /etc/rc.d/init.d/halt \
                  -o CAP_KILL -i 1                       -j GRANT
/sbin/lidsconf -A -s /etc/rc.d/init.d/halt \
                  -o CAP_NET_ADMIN -i 1                  -j GRANT
/sbin/lidsconf -A -s /etc/rc.d/init.d/halt \
                  -o CAP_SYS_ADMIN -i 1                  -j GRANT

# Other
#
/sbin/lidsconf -A -s /sbin/update -o CAP_SYS_ADMIN       -j GRANT
```

## 7.2. Apache

This sample configuration assumes Apache was installed in `/usr/local/apache` with a log directory of `/var/log/httpd` and a configuration directory of `/etc/httpd`. You can adjust the paths in the ACLs to match

your own configuration. With this configuration, Apache must be started prior to sealing the kernel, or when LIDS_GLOBAL is disabled so it can bind to port 80 (and possibly 443).

```
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o CAP_SETUID                          -j GRANT
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o CAP_SETGID                          -j GRANT

# Config files
/sbin/lidsconf -A -o /etc/httpd                          -j DENY
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o /etc/httpd                          -j READONLY

# Server Root
/sbin/lidsconf -A -o /usr/local/apache                   -j DENY
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o /usr/local/apache                   -j READONLY

# Log Files
/sbin/lidsconf -A -o /var/log/httpd                      -j DENY
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o /var/log/httpd                      -j APPEND
/sbin/lidsconf -A -s /usr/local/apache/bin/httpd \
                  -o /usr/local/apache/logs              -j WRITE
```

## 7.3. Qmail

These ACLs were written for a qmail setup that was installed according to Dave Sill's *Life with qmail (http://Web.InfoAve.Net/~dsill/lwq.html).* With this configuration, qmail must be started prior to sealing the kernel, or when LIDS_GLOBAL is disabled so tcpserver can bind to port 25.

```
# setup
/sbin/lidsconf -A -o /var/qmail                          -j READONLY
/sbin/lidsconf -A -s /usr/local/bin/multilog \
                  -o /var/log/qmail                      -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/svc \
                  -o /var/qmail/supervise                -j WRITE

# queue access
#
/sbin/lidsconf -A -s /var/qmail/bin/qmail-inject \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-rspawn \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-lspawn \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-queue \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-clean \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-send \
                  -o /var/qmail/queue                    -j WRITE
/sbin/lidsconf -A -s /var/qmail/bin/qmail-remote \
```

```
                  -o /var/qmail/queue                    -j WRITE

# Access to local mail boxes
/sbin/lidsconf -A -s /var/qmail/bin/qmail-lspawn \
                  -o CAP_SETUID                          -j GRANT
/sbin/lidsconf -A -s /var/qmail/bin/qmail-lspawn \
                  -o CAP_SETGID                          -j GRANT
/sbin/lidsconf -A -s /var/qmail/bin/qmail-lspawn \
                  -o CAP_DAC_OVERRIDE                    -j GRANT
/sbin/lidsconf -A -s /var/qmail/bin/qmail-lspawn \
                  -o CAP_DAC_READ_SEARCH                 -j GRANT


# Remote delivery
/sbin/lidsconf -A -s /var/qmail/bin/qmail-rspawn \
                  -o CAP_NET_BIND_SERVICE -i -1          -j GRANT

# supervise

/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/qmail/supervise/qmail-smtpd/supervise     -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/qmail/supervise/qmail-smtpd/log/supervise -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/qmail/supervise/qmail-send/supervise      -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/qmail/supervise/qmail-send/log/supervise  -j WRITE
```

## 7.4. Dnscache & Tinydns (djbdns)

The following ACLs were written for a djbdns setup based on Jeremy Rauch's *Installing djbdns (DNScache) for Name Service* parts 1 (http://www.securityfocus.com/focus/sun/articles/dnscache.html) & 2 (http://www.securityfocus.com/focus/sun/articles/dnscache2.html). With this configuration, dnscache and tinydns must be started prior to sealing the kernel, or when LIDS_GLOBAL is disabled so they can bind to port 53.

```
# dnscache
#
/sbin/lidsconf -A -o /var/dnscache                        -j READONLY
/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/dnscache/dnscache/supervise     -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/dnscache/dnscache/log/supervise -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/multilog \
                  -o /var/dnscache/dnscache/log/main      -j WRITE

# tinydns
#
/bin/echo "tinydns"

/sbin/lidsconf -A -s /usr/local/bin/supervise \
                  -o /var/dnscache/tinydns/supervise      -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/supervise \
```

```
                  -o /var/dnscache/tinydns/log/supervise  -j WRITE
/sbin/lidsconf -A -s /usr/local/bin/multilog \
                  -o /var/dnscache/tinydns/log/main       -j WRITE
```

## 7.5. Courier-imap

The following ACLs assume courier-imap was installed into `/usr/local/courier-imap`. With this configuration, courier-imap must be started prior to sealing the kernel, or when LIDS_GLOBAL is disabled so it can bind to port 143.

```
/sbin/lidsconf -A -o /usr/local/courier-imap                     -j DENY

/sbin/lidsconf -A -s /usr/local/courier-imap/sbin/imaplogin \
                  -o /etc/shadow                                 -j READONLY
/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/authlib/authpam \
                  -o /etc/shadow                                 -j READONLY
/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/couriertcpd \
                  -o /usr/local/courier-imap                     -j READONLY

/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/couriertcpd \
                  -o CAP_SETUID -i 3                             -j GRANT
/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/couriertcpd \
                  -o CAP_SETGID -i 3                             -j GRANT
/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/couriertcpd \
                  -o CAP_DAC_OVERRIDE -i 3                       -j GRANT
/sbin/lidsconf -A -s /usr/local/courier-imap/libexec/couriertcpd \
                  -o CAP_DAC_READ_SEARCH -i 3                    -j GRANT
```

## 7.6. MySQL

The following ACLs assume MySQL was installed into `/usr/local/mysql`.

```
/sbin/lidsconf -A -o /usr/local/mysql/var               -j APPEND

/sbin/lidsconf -A -o /usr/local/mysql                   -j DENY
/sbin/lidsconf -A -s /usr/local/mysql/libexec/mysqld \
                  -o /usr/local/mysql                   -j READONLY
/sbin/lidsconf -A -s /usr/local/mysql/libexec/mysqld \
                  -o /usr/local/mysql/var               -j WRITE
```

## 7.7. OpenSSH (3.4p1)

The following configuration will work after boot and while LIDS_GLOBAL is on because it gives sshd the CAP_NET_BIND_SERVICE capability.

```
/sbin/lidsconf -A -s /usr/sbin/sshd -o /etc/shadow      -j READONLY

/sbin/lidsconf -A -o /etc/ssh/sshd_config               -j DENY
/sbin/lidsconf -A -o /etc/ssh/ssh_host_key              -j DENY
/sbin/lidsconf -A -o /etc/ssh/ssh_host_dsa_key          -j DENY

/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o /etc/ssh/sshd_config               -j READONLY
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o /etc/ssh/ssh_host_key              -j READONLY
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o /etc/ssh/ssh_host_dsa_key          -j READONLY

/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o /var/log/wtmp                      -j WRITE
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o /var/log/lastlog                   -j WRITE

/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_SETUID                         -j GRANT
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_SETGID                         -j GRANT
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_FOWNER                         -j GRANT
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_CHOWN                          -j GRANT
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_DAC_OVERRIDE                   -j GRANT
/sbin/lidsconf -A -s /usr/sbin/sshd \
                  -o CAP_NET_BIND_SERVICE 22-22         -j GRANT
/sbin/lidscond -A -s /usr/sbin/sshd \
  -o CAP_SYS_CHROOT -j GRANT
/sbin/lidscond -A -s /usr/sbin/sshd \
  -o CAP_SYS_RESOURCE -j GRANT
/sbin/lidscond -A -s /usr/sbin/sshd \
  -o CAP_SYS_TTY_CONFIG -j GRANT
```

## 7.8. OpenLDAP (slapd)

The following configuration will work after boot and while LIDS_GLOBAL is on because it gives slapd the CAP_NET_BIND_SERVICE capability.

```
/sbin/lidsconf -A -s /usr/local/libexec/slapd \
                  -o /usr/local/ldapdb  -j WRITE
/sbin/lidsconf -A -s /usr/local/libexec/slapd \
                  -o CAP_NET_BIND_SERVICE               -j GRANT
/sbin/lidsconf -A -s /usr/local/libexec/slapd \
                  -o CAP_INIT_KILL                      -j GRANT
```

```
/sbin/lidsconf -A -s /usr/local/libexec/slapd \
                  -o CAP_SYS_MODULE                       -j GRANT
```

## 7.9. Port Sentry

The following configuration will work after boot and while LIDS_GLOBAL is on because it gives portsentry the CAP_NET_BIND_SERVICE capability. Depending on what you want portsentry to do, you may or may not need all of the following ACLs.

```
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o /usr/local/psionic/portsentry  -j WRITE
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o /var/log  -j WRITE
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o CAP_NET_BIND_SERVICE                       -j GRANT

# For portsentry to be able to update the firewall:
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o CAP_NET_RAW -i 1                           -j GRANT

# For portsentry to be able to update /etc/hosts.allow and/or /etc/hosts.deny:
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o /etc/hosts.allow              -j WRITE
/sbin/lidsconf -A -s /usr/local/psionic/portsentry/portsentry \
                  -o /etc/hosts.deny              -j WRITE
```

## 7.10. Samba

With this configuration, Samba must be started prior to sealing the kernel, or when LIDS_GLOBAL is disabled so it can bind to ports 137 & 139.

```
/sbin/lidsconf -A -o /etc/samba -j READONLY
/sbin/lidsconf -A -o /var/samba -j READONLY
/sbin/lidsconf -A -s /usr/sbin/smbd -o /var/samba -j WRITE
/sbin/lidsconf -A -s /usr/sbin/nmbd -o /var/samba -j WRITE

# smbd needs write access to smbpasswd to chmod it.  i think it
# also needs access to MACHINE.SID
/sbin/lidsconf -A -s /usr/sbin/smbd -o /etc/samba -j WRITE
/sbin/lidsconf -A -s /usr/sbin/smbd -o /etc/shadow -j READONLY

/sbin/lidsconf -A -s /usr/sbin/smbd -o CAP_SETUID -j GRANT
/sbin/lidsconf -A -s /usr/sbin/smbd -o CAP_SETGID -j GRANT
/sbin/lidsconf -A -s /usr/sbin/smbd -o CAP_HIDDEN -j GRANT

# LIDS complains about smbd trying to chroot to /
# everything still seems to work without it, though
# (and isn't chrooting to / kinda pointless anyway?)
#/sbin/lidsconf -A -s /usr/sbin/smbd -o CAP_SYS_CHROOT -j GRANT
```

```
/sbin/lidsconf -A -s /usr/sbin/nmbd -o CAP_HIDDEN -j GRANT
```

## 7.11. Linux HA heartbeat

```
/sbin/lidsconf -A -o /usr/lib/heartbeat/heartbeat               -j READONLY
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_NET_BIND_SERVICE -i -1                 -j GRANT
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_SYS_RAWIO -i -1                        -j GRANT
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_NET_BROADCAST -i -1                    -j GRANT
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_NET_ADMIN -i -1                        -j GRANT
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_NET_RAW -i -1                          -j GRANT
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o CAP_SYS_ADMIN -i -1                        -j GRANT

# For sending Gratuitous Arps

/sbin/lidsconf -A -o /usr/lib/heartbeat/send_arp                -j READONLY
/sbin/lidsconf -A -s /usr/lib/heartbeat/send_arp \
                  -o CAP_NET_RAW -i -1                          -j GRANT

# For modifying the routing table when the IP address changes

/sbin/lidsconf -A -o /sbin/route                                -j READONLY
/sbin/lidsconf -A -s /sbin/route -o CAP_NET_ADMIN -i 0          -j GRANT

#
# Protect the heartbeat configuration and authentication key.
#
/sbin/lidsconf -A -o /etc/ha.d/ha.cf                            -j READONLY
/sbin/lidsconf -A -o /etc/ha.d/haresources                      -j READONLY
/sbin/lidsconf -A -o /etc/ha.d/authkeys                         -j DENY

#
# Only heartbeat can see the authkey
#
/sbin/lidsconf -A -s /usr/lib/heartbeat/heartbeat \
                  -o /etc/ha.d/authkeys                         -j READONLY
```

## 7.12. Bind 9.x

```
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_NET_BIND_SERVICE 53 -j GRANT
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_SETPCAP             -j GRANT
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_SYS_CHROOT          -j GRANT
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_SYS_RESOURCE        -j GRANT
```

```
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_SETUID              -j GRANT
/sbin/lidsconf -A -s /usr/sbin/named  -o CAP_SETGID              -j GRANT
```

## 7.13. Sendmail

```
# Sendmail LIDS rules (using infinite inheritance for the sendmail
# children and delivery agents to work properly, but a lower inheritance
# like 2 or 3 would probably work as well.)

# Lock down /etc/mail if it's not already done elseware
/sbin/lidsconf -A -o /etc/mail -j READONLY

/sbin/lidsconf -A -o /usr/sbin/sendmail -j READONLY
/sbin/lidsconf -A -s /usr/sbin/sendmail -o /etc/shadow -j READONLY -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o /etc/passwd -j READONLY -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o /etc/mail   -j READONLY -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o /etc/mail/aliases   -j WRITE -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o /etc/mail/aliases.db   -j WRITE -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o CAP_SETUID -j GRANT -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o CAP_SETGID -j GRANT -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o CAP_SYS_ADMIN -j GRANT -i -1
/sbin/lidsconf -A -s /usr/sbin/sendmail -o CAP_NET_BIND_SERVICE 25-25 -j GRANT -i -1

# Depending on how you have the log files secured
# (The maillog will normally get rotated out and this
# rule will stop working when that happens unless you
# stop the log rotation.)

/sbin/lidsconf -A -s /usr/sbin/sendmail -o /var/log/maillog -j APPEND -i -1
```

## 7.14. Apcupsd

```
/sbin/lidsconf -A -o /etc/apcupsd                                    -j DENY
/sbin/lidsconf -A -s /sbin/apcupsd -o /etc/apcupsd                   -j READONLY
/sbin/lidsconf -A -s /sbin/apcupsd -o CAP_HIDDEN -i -1               -j GRANT
```

## 7.15. Pump

```
/sbin/lidsconf -A -s /sbin/pump -o CAP_NET_BIND_SERVICE 68-68        -j GRANT
/sbin/lidsconf -A -s /sbin/pump -o CAP_NET_RAW                       -j GRANT
/sbin/lidsconf -A -s /sbin/pump -o CAP_HIDDEN                        -j GRANT
```

## 7.16. Snort

```
/sbin/lidsconf -A -s /usr/sbin/snort -o CAP_DAC_OVERRIDE                -j GRANT
/sbin/lidsconf -A -s /usr/sbin/snort -o CAP_NET_RAW                     -j GRANT
/sbin/lidsconf -A -s /usr/sbin/snort -o CAP_HIDDEN                      -j GRANT
/sbin/lidsconf -A -s /usr/sbin/snort -o CAP_SETUID                      -j GRANT
/sbin/lidsconf -A -s /usr/sbin/snort -o CAP_SETGID                      -j GRANT
```

## 7.17. Getty

```
/sbin/lidsconf -A -s /sbin/getty -o CAP_DAC_OVERRIDE                    -j GRANT
/sbin/lidsconf -A -s /sbin/getty -o CAP_HIDDEN                          -j GRANT
```

## 7.18. Login

```
/sbin/lidsconf -A -s /bin/login -o /etc/shadow                          -j READONLY
/sbin/lidsconf -A -s /bin/login -o CAP_SETUID                           -j GRANT
/sbin/lidsconf -A -s /bin/login -o CAP_SETGID                           -j GRANT
/sbin/lidsconf -A -s /bin/login -o CAP_CHOWN                            -j GRANT
/sbin/lidsconf -A -s /bin/login -o CAP_FSETID                           -j GRANT
```

## 7.19. Su

```
/sbin/lidsconf -A -s /bin/su -o /etc/shadow                             -j READONLY
/sbin/lidsconf -A -s /bin/su -o CAP_SETUID                              -j GRANT
/sbin/lidsconf -A -s /bin/su -o CAP_SETGID                              -j GRANT
```

## 7.20. Exim

```
/sbin/lidsconf -A -s /usr/sbin/exim -o CAP_SETGID -j GRANT
/sbin/lidsconf -A -s /usr/sbin/exim -o CAP_SETUID -j GRANT
```

## 7.21. Qpopper

```
/sbin/lidsconf -A -s /usr/sbin/in.qpopper -o /etc/shadow -j READONLY
```

## 7.22. Proftp

```
/sbin/lidsconf -A -s /usr/sbin/proftpd  -o CAP_SETGID -j GRANT
/sbin/lidsconf -A -s /usr/sbin/proftpd  -o CAP_SETUID -j GRANT
/sbin/lidsconf -A -s /usr/sbin/proftpd  -o CAP_SYS_CHROOT -j GRANT
/sbin/lidsconf -A -s /usr/sbin/proftpd  -o /etc/shadow -j READONLY
```

## 7.23. Aproxy

```
/sbin/lidsconf -A -s /path/to/aproxy -i 2 -o CAP_NET_BIND_SERVICE 25,110,119 -j GRANT
```

## 7.24. Squid

```
/sbin/lidsconf -A -o /var/spool/squid -j DENY
/sbin/lidsconf -A -s /usr/sbin/squid   -i 2 -o /var/spool/squid               -j WRITE
/sbin/lidsconf -A -s /usr/sbin/squid   -i 2 -o /var/log/squid                 -j WRITE
/sbin/lidsconf -A -s /etc/init.d/squid -i 2 -o /var/spool/squid               -j WRITE
/sbin/lidsconf -A -s /usr/sbin/squid        -o CAP_NET_BIND_SERVICE 3128,3130 -j GRANT
```

## 7.25. Innd

```
/sbin/lidsconf  -A -o /usr/local/news -j DENY

/sbin/lidsconf  -A -s /usr/local/news/bin/ctlinnd    -o /usr/local/news      -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/innd       -o /usr/local/news      -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/nnrpd      -o /usr/local/news      -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/nnrpd \
                                     -o /usr/local/news/spool/overview  -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/rc.news    -o /usr/local/news      -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/shlock     -o /usr/local/news/run/ -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/innwatch   -o /usr/local/news/run/ -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/innconfval -o /usr/local/news/     -j WRITE
/sbin/lidsconf  -A -s /usr/local/news/bin/innmail    -o /usr/local/news/     -j WRITE
```

```
/sbin/lidsconf  -A -s /usr/local/news/bin/inndstart  -o /usr/local/news/     -j WRITE

/sbin/lidsconf  -A -s /usr/local/news/bin/inndstart \
                                      -o CAP_NET_BIND_SERVICE 119 -j GRANT
/sbin/lidsconf  -A -s /usr/local/news/bin/inndstart  -o CAP_SETGID -j GRANT
/sbin/lidsconf  -A -s /usr/local/news/bin/inndstart  -o CAP_SETUID -j GRANT
/sbin/lidsconf  -A -s /usr/local/news/bin/nnrpd      -o CAP_SETUID -j GRANT
/sbin/lidsconf  -A -s /usr/local/news/bin/nnrpd      -o CAP_SETGID -j GRANT
```

## 7.26. Postfix

The following example is for postfix on a Debian GNU/Linux Woody (3.0) system with all capabilities disabled. The CAP_HIDDEN parts are ofcourse optional.

```
/sbin/lidsconf -A -o /etc/postfix               -j DENY
/sbin/lidsconf -A -o /var/spool/postfix         -j DENY

/sbin/lidsconf -A -s /etc/init.d/postfix \
  -o /etc/postfix -j READONLY -i 1
/sbin/lidsconf -A -s /etc/init.d/postfix \
  -o /var/spool/postfix -j WRITE    -i 1
/sbin/lidsconf -A -s /usr/sbin/postfix   \
  -o /etc/postfix -j READONLY -i 4
/sbin/lidsconf -A -s /usr/sbin/postfix  \
  -o /var/spool/postfix -j WRITE    -i 4

/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o CAP_SETGID -j GRANT    -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o CAP_SETUID -j GRANT    -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o CAP_HIDDEN                 -j GRANT    -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o CAP_DAC_OVERRIDE           -j GRANT    -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o CAP_SYS_CHROOT             -j GRANT    -i 1

/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o /etc/aliases.db -j READONLY -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o /var/spool/postfix -j WRITE    -i 1
/sbin/lidsconf -A -s /usr/lib/postfix/master \
  -o /etc/postfix         -j READONLY -i 1

/sbin/lidsconf -A -s /usr/sbin/postdrop \
  -o /etc/postfix -j READONLY
/sbin/lidsconf -A -s /usr/sbin/postdrop \
  -o /var/spool/postfix -j WRITE

/sbin/lidsconf -A -s /usr/sbin/sendmail \
  -o /etc/postfix -j READONLY
/sbin/lidsconf -A -s /usr/sbin/sendmail \
  -o /var/spool/postfix -j WRITE
```

# Chapter 8. LIDS Technical

## 8.1. Will LIDS work with a file system other than ext2?

Yes. To quote LIDS co-author Philippe Biondi: “"LIDS works on top of the VFS layer, so that it can handle every fs linux supports."”

## 8.2. Will LIDS run on an SMP system?

There have been problems reported with SMP systems running LIDS. Many of the problems have been fixed, so it is recommended that you try out the latest version and see for yourself. Xie and Philippe are very dedicated to fixing any such problems, so please make sure to report any to the LIDS mailing list.

UPDATE (2/10/01): Many users have reported success using LIDS-1.0.5 for the 2.4.x kernel on SMP systems.

## 8.3. Will LIDS coexist with Solar Designer’s Openwall patch?

Yes. If you apply both the LIDS and Openwall patches yourself, one of the hunks will fail (as of release 0.9.11 for kernel 2.2.18). It is a minor error that won’t affect your system security.

## 8.4. Will LIDS run on non-Intel hardware?

I’m not aware of any confirmed success stories on other hardware platforms. If you get LIDS to work on another architecture, be sure to let everyone know of your efforts.

*Update:* Johannes Helje has successfully installed LIDS on a pair of SUN IPXs. He is using Debian with a 2.2.18 kernel.

*Update:* Joseph P. Garcia (jpgarcia AT execpc DOT com (mailto:jpgarcia@execpc.com)) has attempted to install LIDS on a PowerPC based Macintosh PowerBook G3 without much success. Here is an excerpt from his e-mail with the details of his problems:

```
I am currently pursuing trying out LIDS on my 30-month old powerpc-based
Macintosh PowerBook G3. (’oldworld’ powermac for those who know what that
is)  I use the BootX boot loader to boot Linux, loosely based on LinuxPPC
2000 Q4, using kernel 2.4.7pre3, glibc 2.2.3, and gcc 2.95.4.

My attempts to use LIDS on my system have yielded little results.  With the
patch applied and LIDS disabled via config, the kernel works fine.  With
LIDS enabled in any degree, even just CONFIG_LIDS and security=0, my kernel
does not boot.  The normal routine is BootX cleans out MacOS, sets up
hardware (like harddrive spin down), the kernel clears the screen, shows
simple settings via ’BootX text’, and begins booting with output on a
framebuffer console.  With LIDS enabled, the kernel doesn’t even clear the
screen.  I looked at the code that does this, and to my best understanding,
it just writes memory.  I can’t tell just how far it gets.

As far as i know, lids should not be active until much later.  So this
would either be caused by a fundamental code modification LIDS performs
```

```
that I do not understand, or a possible feature that prevents the kernel to
boot normally on my system.

I am unaware of any other efforts to run LIDS on powerpc at this time.
I am willing to lend my time when available to test theories and
modifications people may have to add support to LIDS for the
increasingly popular PowerPC architecure.

As it stands, after writing this, I think I will try disabling the BootX
text option and see what happens.  Attached is my kernel config (bz2)
before this modification.

Thank you all for your time and consideration.
```

*Update:* Joseph P. Garcia was able to get LIDS working on the PowerPC. He did this by reducing the maximum number of objects that LIDS will manage from 1024 to 512 in his kernel configuration.

## 8.5. What is the difference between the 0.x, 1.x and 2.x versions of LIDS?

LIDS `0.x` is for the `2.2.x` Linux kernel, LIDS `1.x` is for the `2.4.x` Linux kernel and LIDS `2.x` is for the `2.5.x` Linux kernel.