Securing Debian HOWTO

Alexander Reelsen <ar@rhwd.net>
Перевод: Ильгиз Кальметьев, <ilgiz@mail.rb.ru>

v1.1 28 diciembre 2003Thu, 7 Dec 2000 19:10:13 +0100

Аннотация

Этот документ описывает настройку защиты и безопасности системы Debian, имеющей установки по умолчанию.

Замечания об авторских правах

Оглавление

1 Введение 1
1.1 Где можно найти этот HOWTO . . . 1
1.2 Организационные примечания/Обратная связь . . . 2
1.3 Предварительные знания . . . 2
1.4 TODO . . . 2
1.5 Благодарности . . . 2

2 Перед и в течение установки 5
2.1 Выбор пароля BIOS . . . 5
2.2 Грамотная разбивка диска на разделы . . . 5
2.3 Установка пароля суперпользователя . . . 6
2.4 Активизация теневых и MD5 паролей . . . 6
2.5 Запуск минимального количества сервисов . . . 6

3 После установки 7
3.1 Установка пароля LILO или GRUB . . . 7
3.2 Запрет загрузки с дискет . . . 8
3.3 Правильное монтирование разделов . . . 8
3.4 PAM - Pluggable Authentication Modules . . . 9
3.5 The limits.conf file . . . 10
3.6 Customize /etc/inetd.conf . . . 10
3.7 Edit /etc/login.defs . . . 11
3.8 Editing /etc/ftpusers . . . 11
3.9 Using tcp wrappers . . . 11

3.10 The importance of logs and alerts . . . 12
3.10.1 Configuring where alerts are sent . . . 12
3.10.2 Using a loghost . . . 13
3.10.3 Logfile permissions . . . 13
3.11 Setting up setuid check . . . 13
3.12 Using su . . . 14
3.13 Using sudo . . . 14
3.14 Using chroot . . . 14
3.15 Configuring some kernel features . . . 15
3.16 Do not use software depending on svgalib . . . 15
3.17 Secure file transfers . . . 15
3.18 Using quotas . . . 15
3.19 chattr/lsattr . . . 16
3.20 Your filesystem integrity . . . 16

4 Securing services running on your system 17
4.1 Securing ssh . . . 17
4.2 Realize the insecurity of X over network . . . 18
4.3 The lpd and lprng issue . . . 18
4.4 Защита почты . . . 18
4.5 Защита BIND . . . 19

5 Before the compromise 21
5.1 Follow Debian security updates . . . 21
5.2 Exchange software . . . 21
5.3 Полезные заплаты для ядра . . . 22
5.4 Genius/Paranoia Ideas, what you could do . . . 23

6 After the compromise 25
6.1 General behavior . . . 25

Глава 1

Введение

Одна из самых сложных вещей при написании документов, посвященных безопасности, состоит в том, что каждый случай уникален. Две вещи, которые требуют вашего особенно пристального внимания к себе, - это враждебное окружение и безопасность индивидуальных сайтов, хостов или сетей. Требования к безопасности домашнего пользователя коренным образом отличаются от требований в банковской сети. В то время, как домашнему пользователю могут угрожать в основном всякие кракеры-шалуны, банки могут подвергаться целенаправленным атакам. Банкам приходится защищать данные своих пользователей от малейших искажений злоумышленниками. В общем, каждому пользователю приходится выбирать между простотой использования и безопасностью/паранойей.

Обратите внимание, что данный HOWTO относится только к программному обеспечению. Никакое ПО не может защитить от физического доступа к машине. Вы можете разместить ее под столом или спрятать в бункер под охраной целой армии. Тем не менее настольный компьютер может быть намного более защищен (с точки зрения программного обеспечения), чем физически защищенная машина, если ПО на этом компьютере правильно настроено и закрыты дыры в безопасности. Obviously, you must consider both issues. # In addition this document just gives a overview of what you can do to increase # the security of your Debian GNU/Linux # installation. Many parts of this HOWTO can be transferred to other # distributions.

Если у вас имеются комментарии, дополнения или предложения, присылайте их по почте автору (mailto:ar@rhwd.net), и они будут учтены при написании этого HOWTO.

1.1 Где можно найти этот HOWTO

Вы можете скачать или посмотреть новые версии Securing Debian HOWTO в следующих форматах:

- Textonly (http://joker.rhwd.de/doc/Securing-Debian-HOWTO/Securing-Debian-HOWTO.txt)

- HTML (http://joker.rhwd.de/doc/Securing-Debian-HOWTO/Securing-Debian-HOWTO.html)
- HTML, tarred gzipped (http://joker.rhwd.de/doc/Securing-Debian-HOWTO/Securing-Debian-HOWTO.tar.gz)
- SGML (http://joker.rhwd.de/doc/Securing-Debian-HOWTO/Securing-Debian-HOWTO.sgml)

1.2 Организационные примечания/Обратная связь

Теперь перейдем к официальной части. На данный момент мною написано большинство параграфов этого HOWTO, но на мой взгляд, это не окончательный вариант. but in my opinion this should not stay the case. I grew up and live with free software, it is part of my everyday use and I guess yours, too. I encourage everybody to send me feedback, hints additions or any other suggestions, you might have. If you think, you can maintain a certain section or paragraph better than me, then write this to me and you are welcome to do it. Especially if you find a section marked as FIXME, what means I did not have the time yet or the needed knowledge about the topic, drop me a mail immediately. The topic of this HOWTO makes is quite clear, that it is important to keep uptodate, and you can help to keep the quality of this HOWTO up, so do it.

1.3 Предварительные знания

Установка Debian GNU/Linux не очень сложное дело и вы должны суметь установить его. Если у вас уже есть некоторые понятия о Linux и других юниксах, а также о основах безопасности, то вам будет легче читать этот документ, так как я не собираюсь углубляться в излишние подробности (в противном случае это уже будет книга, а не HOWTO).

1.4 TODO

- suidmanager/dpkg-statoverrides
- lpr and lprng
- Switching off the gnome IP things
- LKM, linux kernel modules, bad and good ones

1.5 Благодарности

- Alexander Reelsen, за написание первоначального документа

- Robert van der Meulen, за раздел о quota и множество хороших идей
- Ethan Benson, за коррекцию раздела о PAM и также несколько полезных идей
- Всем тем, кто поддерживал меня при написании этого HOWTO
- Всему проекту Debian

Глава 2

Перед и в течение установки

2.1 Выбор пароля BIOS

Перед установкой любой из операционных систем на ваш компьютер, установите пароль в BIOS и запретите загрузку с дискеты. Иначе говоря, кракеру достаточно будет только загрузиться с дискеты, чтобы получить доступ к вашей системе.

Запрет загрузки без пароля - это еще лучше. Это может быть очень эффективно, если вы запускаете сервер, потому что он не перегружается слишком часто. Прибегать к этому не стоит, если машина физически труднодоступна, поскольку при перезагрузке машине потребуется вмешательство человека.

2.2 Грамотная разбивка диска на разделы

Грамотная разбивка диска на разделы зависит от области применения машины. Хорошее правило для этого - быть беспристрастным и обратить внимание на следующие вещи:

- Любой раздел, на который пользователь должен иметь возможность писать, например, /home и /tmp, должен быть отдельным. Это сокращает риск отказа в обслуживании пользователя (DoS) в результате переполнения корневого каталога "/" и вследствие этого отказа системы. (Вообще-то это не совсем так, поскольку для суперпользователя резервируется место, которым обычный пользователь не может воспользоваться).
- Любой раздел, который может сильно распухнуть, напр., /var (особенно /var/log). В контексте Debian вы должны создать /var несколько больше, чем обычно, потому что скачанные пакеты (кеш программы apt) сохраняются в каталоге /var/apt/cache/archives.
- Любой раздел, куда вы планируете устанавливать недистрибутивное ПО. В соответствии с File Hierarchy Standard - это каталог /opt или /usr/local. Будучи отдельными разделами они не пропадут при переустановке системы.

2.3 Установка пароля суперпользователя

Установка хорошего пароля суперпользователя - это наиболее основное требование обеспечения безопасности системы.

2.4 Активизация теневых и MD5 паролей

В конце установки вас спросят, не хотите ли вы установить теневые пароли. Ответьте да, и пароли будут храниться в файле /etc/shadow. Доступ к этому файлу будут иметь только пользователь root и пользователи группы root, так что пользователи не смогут скопировать себе этот файл, чтобы прогнать его через программу-взломщик паролей. Вы можете в любой момент переключиться между теневыми и обычными паролями командой 'shadowconfig'. Кроме того, при установке системы вас спросят, не желаете ли вы использовать хешированные MD5 пароли. Обычно это стОящая вещь, поскольку она позволяет использовать более длинные пароли и улучшенные алгоритмы зашифровывания.

2.5 Запуск минимального количества сервисов

Не устанавливайте лишние сервисы на машине без необходимости. Каждый сервис - это потенциальная дыра в вашей защите. Если вы хотите оставить какой-либо сервис, но пользуетесь им нечасто, то используйте команду update-commands, т.е. 'update-inetd' для предотвращения загрузки сервиса при запуске системы. # This section needs a list of services,and what they do and the risk level # involved, as newbies don't have a clue.

Глава 3

После установки

3.1 Установка пароля LILO или GRUB

Любой человек может легко войти в оболочку суперпользователя и изменить ваши пароли, если введет строку “<имя-файла-образа-ядра> init=/bin/sh” в строке приглашения загрузчика. После изменения пароля и перезагрузки системы негодяй получит неограниченные права суперпользователя и сможет сделать с вашей системой все, что захочет. Вы же не сможете войти в систему, поскольку пароль суперпользователя изменен.

Чтобы этого не произошло, установите пароль на системный загрузчик. Пароль можно установить как вообще на загрузку системы, так и на загрузку определенного ядра.

Для LILO вам нужно отредактировать настроечный файл /etc/lilo.conf и добавить строки “password” и “restricted”, как указано в примере ниже.

```
image=/boot/2.2.14-vmlinuz
  label=Linux
  read-only
  password=hackme
  restricted
```

Затем перезпустите lilo. Указание строки “restricted” заставляет lilo твсегда запрашивать пароль , regardless of whether LILO was passed parameters. Не забудьте ограничить доступ к файлу настроек lilo, т.е., сделать chmod 600 /etc/lilo.conf. # is this Debian’s default? if so say so or remove this , jfs

Если вы используете GRUB вместо LILO, отредактируйте файл /boot/grub/menu.lst и добавьте следующие строки в начало. Они устанавливают загрузочный пароль и азгружают систему по умолчанию по истечении трех секунд:

```
timeout 3
password hackme
```

3.2 Запрет загрузки с дискет

По умолчанию MBR в Debian до версии 2.2 не действует как обычный MBR и оставляет легкий путь проникновения в систему:

- при загрузке нажмите shift и дождитесь появления приглашения MBR
- затем нажмите F, и ваша система начнет загружаться с дискеты, что может быть использовано для доступа к системе с правами суперпользователя.

Это поведение можно изменить командой: lilo -b /dev/hda Теперь LILO располагается в MBR. Этот же эффект достигается добавлением “boot=/dev/hda” в lilo.conf. Есть другое решение, которое позволит полностью запретить строку приглашения mbr: install-mbr -i n # is this install-mbr /dev/hda ?? jfs # check whether this really is true as of 2.2 or was it 2.1? INFO: The bootdisks as of Debian 2.2 do NOT install the mbr, but only LILO

3.3 Правильное монтирование разделов

При монтировании разделов ext2 вы можете добавить несколько дополнительных опций монтирования в командной строке или в файл /etc/fstab. Вот пример записи в моем fstab для раздела /tmp:

```
/dev/hda7    /tmp    ext2    defaults,nosuid,noexec,nodev    0    2
```

Опция nosuid полностью игнорирует биты setuid и setgid, опция noexec запрещает выполнение программ в данной точке монтирования, а опция nodev игнорирует устройства. Это все хорошо, но

- применимо только к системе ext2
- легко обходится

Опция noexec предотвращает прямой запуск бинарных файлов, но легко обходится:

```
alex@joker:/tmp# mount | grep tmp
/dev/hda7 on /tmp type ext2 (rw,noexec,nosuid,nodev)
alex@joker:/tmp# ./date
bash: ./date: Permission denied
alex@joker:/tmp# /lib/ld-linux.so.2 ./date
Sun Dec  3 17:49:23 CET 2000
```

Однако многие глуповатые сценарии для взлома пытаются создать и выполнить файлы в /tmp. Если это у них не получается, то они терпят неудачу в этой точке.

3.4 PAM - Pluggable Authentication Modules

PAM (загружаемые модули аутентификации) помогают системным администраторам выбирать, как приложения будут аутентифицировать пользователей. Заметьте, что приложение должно иметь вкомпилированную поддержку PAM, чтобы этот метод работал. Большинство приложений в составе Debian 2.2 имеют в себе эту поддержку. Обратите внимание, что Debian до версии 2.2 не имел поддержки PAM. Настройка модулей PAM для каждого из приложений осуществляется в настроечных файлах в каталоге /etc/pam.d/. PAM позволяет вам настроить шаги процесса аутентификации пользователя, ничего при этом не зная о самом пользователе. Вы можете аутентифицировать пользователя через базу данных Berkeley и файл passwd, и пользователь будет допущен в систему только если пройдет обе проверки. Вы можете как закрыть множество вещей, так и широко открыть двери в вашу систему. Будьте внимательны. Обычно в строках настроек управляющую информацию содержит третье поле. Обычно он должен быть установлен в “requisite”, что приведет к отказу регистрации пользователя, если хотя бы один модуль вернул ошибку. Первое, что я делаю, - это разрешаю поддержку MD5 в приложениях PAM, так как это помогает защититься от взлома паролей с помощью словаря. Следующие две строки должны быть добавлены ко всем файлам в каталоге /etc/pam.d/ для предоставления доступа к машине, как ’login’ и ’ssh’.

```
password   required     pam_cracklib.so retry=3 minlen=12 difok=3
password   required     pam_unix.so use_authtok nullok md5
```

Итак, что это за загадочные письмена? первая строка загружает модуль PAM, который осуществляет первичную проверку пароля, запрашивает ввод нового пароля длиной как минимум 12 символов, отличающегося не менее чем на 3 символа от старого пароля и разрешает предпринимать 3 попытки. Вторая строка запускает стандартный модуль аутентификации с паролями md5 и позволяет иметь пароли нулевой длины. Директива use_authtok нужна для принятия пароля из предыдущего модуля. Чтобы убедиться, что суперпользователь может регистрироваться в системе только с локальных терминалов, в файле /etc/pam.d/login должна присутствовать следующая строка: auth requisite pam_securetty.so Далее вы должны добавить терминалы, с которых суперпользователь может входить в систему в файл /etc/security/access.conf. Last but not least the following line should be enabled if you want to set up user limits. session required pam_limits.so This restricts the system resources that users are allowed. For example, you could restrict the number of concurrent logins users may have. Now edit the file /etc/pam.d/passwd and change the first line. You should add the option “md5” to use md5 passwords, change the minimum length of password from 4 to 6 (or more) and set a maximum length, if you desire. The resulting line will look something like: password required pam_unix.so nullok obscure min=6 max=11 md5 If we want to protect su, so that only some people can use it to become суперпользователь on your system, we need to add a new group “wheel” to your system (that is the cleanest way, since no file has such a group permission yet). Add суперпользователь and the other users that should be able to “su” to the суперпользователь user to this group. Then add the following line to /etc/pam.d/su: auth requisite pam_wheel.so group=wheel debug This makes sure that only people from the group wheel can use to su to become суперпользователь. If others try, they

will get a message telling them access is denied. Last, but not least, create /etc/pam.d/other and enter the following lines:

```
auth     required       pam_securetty.so
auth     required       pam_unix_auth.so
auth     required       pam_warn.so
auth     required       pam_deny.so
account  required       pam_unix_acct.so
account  required       pam_warn.so
account  required       pam_deny.so
password required       pam_unix_passwd.so
password required       pam_warn.so
password required       pam_deny.so
session  required       pam_unix_session.so
session  required       pam_warn.so
session  required       pam_deny.so
```

These lines will provide a good default configuration for all applications that support PAM (access is denied by default).

3.5 The limits.conf file

You should really take a serious look into this file. Here you can define user resource limits. If you use PAM, this file is not valid and you should use /etc/security/limits.conf instead. FIXME: Get a good limits.conf up here

3.6 Customize /etc/inetd.conf

You should stop all unneeded services on your system, like echo. charges, disca rd, daytime, time, talk, ntalk and the HIGHLY insecure considered r-services (rsh, rlogin and rcp. Use ssh instead). After disabling those, you should check if you really need the inetd daemon. Many people prefer to use daemons instead of calling services via inetd. Denial of Service possibilities exist against inetd, which can increase the machine's load tremendously. If you still want to run some kind of inetd service, switch to a more configurable inet daemon like xinetd or rlinetd.

You can do this editing the inetd.conf directly, but Debian provides an alternative to this: update-inetd. You could remove the telnet daemon by do:

```
# /usr/sbin/update-inetd --disable telnet
# /etc/init.d/inetd restart
```

3.7 Edit /etc/login.defs

The next step is to edit the basic configuration and action upon user login. FAIL_DELAY 10 This variable should be set to a higher value to make it harder using the terminal to log in using brute force style. If a wrong password is typed in, he has to wait for 10 seconds to get a new login prompt, what is quite time consuming when you test passwords. Pay attention to the fact, that this setting is useless if using program other than getty, such as mingetty. FAILLOG_ENAB yes If you enable this variable, failed logins will be logged. It is important to keep track of them to catch someone who tries a brute force attack. LOG_UNKFAIL_ENAB yes If you set the variable "FAILLOG_ENAB" to yes, then you should also set yes to this variable. This will record unknown usernames if the login failed. If you do this, make sure the logs have to the proper permissions (640 for example, with an appropriate group setting such as adm), because users often accidentally enter their password as the username and you do not want others to see it. SYSLOG_SU_ENAB yes This one enables logging of su attempts to syslog. Quite important on serious machines but note that this can create privacy issues as well. SYSLOG_SG_ENAB yes The same as SYSLOG_SU_ENAB but applies to the sg call. MD5_CRYPT_ENAB yes As stated above, md5 sum passwords greatly reduce the problem of dictionary attacks, because it is very difficult to perform a crack against MD5 hashed passwords. At least it is hard to do successfully. If you are using slink, read the docs about MD5 before enabling this option. Otherwise this is set in PAM. PASS_MAX_LEN 50 If MD5 passwords are activated in your PAM configuration, then this variable should be set to the same value as used there.

3.8 Editing /etc/ftpusers

This file contains a list of users who are not allowed to log into the host using ftp. Only use this file if you really want to allow ftp (which is not recommended in general, because it uses cleartext passwords). If your daemon supports PAM, you can also use this allow and deny users for certain services.

3.9 Using tcp wrappers

TCP wrappers were developed when there were no real packet filters available and access control was needed. The TCP wrappers allow you to allow or deny a service for a host or a domain and define a default allow or deny rule. If you want more informations look into the manpage hosts_access(5). Now, here comes a small trick and probably the smallest intrusion detection system available. In general you should have a decent firewall policy as a first line and tcp wrappers as the second line of defense. One little trick is to set up a spawn command in /etc/hosts.deny that sends mail to суперпользователь whenever a denied service triggers wrappers:

```
ALL: ALL: spawn ( \
  echo -e "\n\
  TCP Wrappers\: Connection refused\n\
  By\: $(uname -n)\n\
```

```
  Process\: %d (pid %p)\n\
  User\: %u\n\
  Host\: %c\n\
  Date\: $(date)\n\
" | /bin/mail -s "Connection to %d blocked" root)
```

Beware: The above printed example can easily be DoSed by doing lots of connections in a short period of time. Many emails mean a lot of file I/O by sending only a few packets. # Could this example be more interesting? # It also relates to the next sectionjfs #

```
#ALL: ALL: spawn ( \
#  /usr/local/sbin/send_syslog %u %c %d )
#
```

Whit send_syslog as: ##!/usr/bin/perl -w # #use Sys::Syslog qw(:DEFAULT setlogsock); # #$user=shift(@ARGV) || 'unkown'; #$host=shift(@ARGV) || 'unkown'; #$service=shift(@ARGV) || 'unkown'; #setlogsock('unix'); #openlog("alert",", 'user'); #syslog('warning', 'Connection from %s at %s to %s blocked.', ($user, $host, $service)); #closelog(); # #exit 0;

3.10 The importance of logs and alerts

How log and alerts are treated is an important issue in a secure system. It is easily to see that, even if the system is perfectly configured and, supposedly, 99% secure. If the 1% comes to happen, if there are no security measures in place to, first, detect this and, second, raise alarms. The system is not secured at all.

3.10.1 Configuring where alerts are sent

Debian comes with a standard syslog configuration (in /etc/syslog.conf) that logs messages to the appropiate files depending on the system facility. If you intend to maintain a secure system you should be wary of where this messages are sent so they do not go unnoticed.

For example, sending messages to the console also is an interesting setup useful for many production-level systems. But for many such systems it is important to also add a new machine that will serve as loghost (receives logs from all other systems).

суперпользователь's mail should be considered also, many security controls (like snort) send alerts to суперпользователь's mailbox. This mailbox usually points to the first user created in the system (check /etc/aliases). Take care to send суперпользователь's mail to some place where it will be read (either locally or remotely). # Note: it would be interesting to tell how a Debian system can send SNMP traps # related to security problems, jfs # check: snmptraglogd, snmp and snmpd

3.10.2 Using a loghost

A loghost is a host which collects syslog data remotely over the network. If one of your machines is cracked, the intruder is not able to cover his tracks, unless he hacks the loghost as well. So, the loghost should be especially secur e. Making a machine a loghost is simple. Just start the syslogd with 'syslogd -r' and a new loghost is born. Next configure the other machines to send data to the loghost. Add an entry like the following to /etc/syslog.conf:

```
facility.level          @your_loghost
```

facility should be one of authpriv, cron, daemon, kern, lpr, mail, news, syslog, user, uucp and local1 up to local7. level should be alert, crit, err, warning, notice, info debug. If you want to log everything remote, just write:

```
*.*                     @your_loghost
```

into your syslog.conf. Logging remotely as well as locally is the best solution (the attacker might presume to have covered their tracks after deletin g the local log files). See the syslog(3), syslogd(8) and syslog.conf(5) manpage for additional information.

3.10.3 Logfile permissions

It is not only important to decide how alerts are used but also who has access to them, i.e. can read or modify the logfiles (if not using a remote loghost). Since, in the event of an intrusion, even with security alerts in place, if an attacker is able to change them security goes back to nil

It should be explained why after installation this is not # already done, jfs Some logfile permissions are not perfect after the installation. First /var/log/lastlog and /var/log/faillog do not need to be readable by normal users. In the lastlog file you can see who logged in the lasttime and in the faillog you see a summary of failed logins. The author recommends chmod'ing both to 660. Take a brief look over your log files and decide very carefully which logfiles you make read/writeable for a user with another UID than 0 and a group other than 'adm' or 'суперпользователь'.

I want to emphasize that the apache logfile permissions are really screwed due to the fact that the apache user owns the apache log files. If a user gets a shell with a back door in apache, they can easily remove the logfiles. # This is quite personal, IMHO, since this is due to the fact that # суперпользователь's priviledges are dropped on startup. I prefer an attacker to erase # a service's logfiles than to erase all of my system's logs. Anyhow, this # can be improved by changing user permissions after rotation

3.11 Setting up setuid check

Debian provides a cron job that runs daily in /etc/cron.daily/standard this cron job will run the /usr/sbin/checksecurity script that will store information of this changes. # What is the defaul for this in cron package? jfs

In order for this check to be made you must set in /etc/checksecurity.conf:

CHECKSECURITY_DISABLE="FALSE"

Is this sent to root? jfs

3.12 Using su

If you really need to become the super user on your system, eg for installing packages or adding users, you can use the command su to change your identity. You should try to avoid any login as user суперпользователь and instead use su. Actually, the best solution is to remove su and switch to sudo, as it has more features than su. However, su is more common as is used on many other Unix es.

3.13 Using sudo

sudo allows the user to execute defined commands under another users identity, even as суперпользователь. If the user is added to /etc/sudoers and authenticates himself correctly, he is able to run commands which have been defined in /etc/sudoers. Violations, such as incorrect passwords or trying to run a program you don't have permission for, are logged and mailed to суперпользователь.

3.14 Using chroot

chсуперпользователь is one of the most powerful possibilities to restrict a daemon or a user or another service. Just imagine a jail around your target, where the target cannot escape from (normally, but there are still a lot of conditions that allow one to escape out of such a jail). If you do not trust a user, you c an create a change суперпользователь environment for him. This can use quite a bit of disk spac e as you need to copy all needed executables, as well as libraries, into the jail . Even if the user does something malicious, the scope of the damage is limited t o the jail. A good example for this case is, if you do not authenticate against /etc/passwd but LDAP or MySQL instead. So your ftp-daemon needs a binary and perhaps a few libraries. A chrooted environment would be an excellent security improvement, if a new exploit is known for this ftp-daemon. It is then only possible to exploit the UID of the ftp-daemon-user and nothing else. Of course, many other daemons could benefit from this as well.

As an additional Обратите внимание, the Debian default BIND (the name-service) is not shipped chrooted per default, in fact no daemons come chrooted. I hope this will change in the woody release.

3.15 Configuring some kernel features

FIXME - Content missing Many features of the kernel can be modified while running by echoing something into the /proc file system or by using sysctl. By entering sysctl -A y ou can see what you can configure and what the options are. Only in rare cases do you need to edit something here, but you can increase security that way as well. net/ipv4/icmp_echo_ignore_broadcasts = 0 This is a 'windows emulator' because it acts like windows on broadcast ping if this is set to 1. Otherwise, it does nothing. net/ipv4/icmp_echo_ignore_all = 0 If you don't want to block ICMP on your firewall, enable this.

3.16 Do not use software depending on svgalib

SVGAlib is very nice for console lovers like me, but in the past it has been proven several times, that it is very insecure. Exploits against zgv were released, and it was simple to become суперпользователь. Try to prevent using SVGAlib programs wherever possible.

3.17 Secure file transfers

Copying files in a secure manner from a host to another can be achieved by using 'scp' which is included in the ssh package. It works like rcp but is encrypted completely, so the bad guys cannot even find out WHAT you copy.

3.18 Using quotas

Having a good quota policy is important, as it keeps users from filling up the hard disk(s).

You can use two different quota systems - user quota and group quota. As you probably figured out, user quota limits the amount of space a user can take up, group quota does the equivalent for groups. Keep this in mind when you're working out quota sizes.

There are a few important points to think about in setting up a quota system:

- Keep the quotas small enough, so users do not eat up your disk space
- Keep the quotas big enough, so users do not complain or their mail quota keeps them from accepting mail over a longer period
- Use quotas on all user-writable areas, on /home as well as on /tmp.

Every partition/directory users have full write access should be quota enabled. So find out those partitions and directories and calculate a valuable quota size, which concatenates usability and security. So, now you want to use quotas. First of all you need to check whether you enabled quota support in your kernel. If not, you will need to recompile it. After this, control whether

the package 'quota' is installed. If not you will need this one as well. Enabling quota for the respective filesystems is as easy as modifying the 'defaults' setting to 'defaults,usrquota' in your /etc/fstab file. If you need group quota, substitute 'usrquota' to 'grpquota'. You can also use them both. Then create empty quota.user and quota.group files in the суперпользователь of the filesystems you want to use quotas on (e.g. touch /home/quota.user, touch /home/quota.group for a /home filesystem). Restart quota by doing /etc/init.d/quota stop;/etc/init.d/quota start. Now quota should be running, and quota sizes can be set. Editing quotas for a specific user (say 'ref') can be done by edquota -u ref. Group quotas can be modified withedquota -g <group>. Then set the soft and hard quota and/or inode quotas as needed. For more information about quotas, read the quota man page, and the quota mini-howto.

3.19 chattr/lsattr

These two commands are very useful, but they only work for the ext2 filesystem. With 'lsattr' you can list the attributes of a file and with 'chattr' you can change them. Note that attributes are not the same thing as permissions. There are many attributes, but only the most important for increasing security are mentioned here. There are two flags which can only be set by the superuser. First there is the 'a' flag. If set on a file, this file can only be opened for appending. This attribute is useful for some of the files in /var/log/, though you should consider they get moved sometimes due to the log rotation scripts. The second flag is the 'i' flag, short for immutable. If set on a file, it can't be modified or deleted or renamed and no link be created to it. If you do not want users to look into your config files you could set this flag and remove readability. Furthermore it can give you a little bit more security against intruders, because the cracker might be confus ed by not being able to remove a file. Nevertheless, you should never assume that the cracker is blind. After all, he got into your system.

3.20 Your filesystem integrity

Are you sure /bin/login on your hard drive is still the binary you installed there some months ago? What if it is a hacked version, which stores the entered password in a hidden file or mails it in cleartext version all over the internet? The only method to have some kind of protection is to check your files every day/hour/month (I prefer daily) by comparing the actual and the old md5sum of this file. Two files cannot have the same md5sum, so you're on the secure site here, except someone hacked the algorithm to create md5sums on that machine, what is, well, sticky. You really should consider this auditing of your binaries as very important, since it is an easy way to recognize changes at your binaries. Common tools used for this are sXid, AIDE (Advanced Intrusion Detection Environment) and TripWire (non-free, the new version will be GPL).

Furthermore you can exchange your 'locate' package with 'slocate'. slocate is a security enhanced version of GNU locate. When using slocate, the user only sees the files he really has access to and you can exclude any files or directories on the system.

Глава 4

Securing services running on your system

4.1 Securing ssh

If you are still running telnet instead of ssh, you should take a break from th is manual and change this. Ssh should be used for all remote logins instead of telnet. In an age where it is easy to sniff internet traffic and get cleartext passwords, you should use only protocols which use cryptography. So, perform an apt-get install ssh on your system no w. Encourage all the users on your system to use ssh instead of telnet, or even better, uninstall telnet. In addition you should avoid logging into the system using ssh as суперпользователь and use alternative methods to become суперпользователь instead, like su or sudo. Finally, the sshd_config file, /etc/ssh, should be modified to increase security as well: PermitRootLogin No Try not to permit Root Login wherever possible. If anyone wants to become суперпользователь via ssh, now two logins are needed and the суперпользователь password cannot be brute forced via SSH. Listen 666 Change the listen port, so the intruder cannot be completely sure whether a sshd daemon runs. PermitEmptyPasswords no Empty passwords make a mockery of system security. AllowUsers alex ref Allow only certain users to have access via ssh to this machine. AllowGroups wheel admin Allow only certain group members to have access via ssh to this machine. AllowGroups and AllowUsers have equivalent directives for denying access to a machine. Not surprisingly they are called “DenyUsers” and “DenyGroups”. PasswordAuthentication yes It is completely your choice what you want to do. It is more secure only to allow access to machine from users with ssh-keys placed in the ~/.ssh/authorized_keys file. If you want so, set this one to “no”. As a final note be aware, that these directives are from a OpenSSH configuration file. Right now, there are three commonly used SSH daemons, ssh1, ssh2 and OpenSSH by the OpenBSD people. Ssh1 was the first ssh daemon available and it is still the most commonly used (there are rumors that there is even a windows port). Ssh2 has many advantages over ssh1 except it is released under an non-opensource license. OpenSSH is completely free ssh daemon, which supports both ssh1 and ssh2. OpenSSH is the version installed on Debian when the package ’ssh’ is chosen.

4.2 Realize the insecurity of X over network

Today X-Terminals are being used by more and more companies where one server is needed for a lot of workstations. This can be dangerous, because you need to allow the file server to connect to the the clients (X server from the X point of view. X switches the definition of client and server). If you follow the (very bad) suggestion of many docs, you type xhost + on your machine. This allows any X client to connect to your system. For slightly better security, you can use the command xhost +hostname instead to only allow access from specific hosts. A much more secure solution, though, is to use ssh to tunnel X and encrypt the whole session. This is done automatically when you ssh to another machine. Of course, even this can be disabled from /etc/ssh/ssh_config. For best security, if you do not need X access from other machines, is to switc h off the binding on tcp port 6000 simply by typing: startx — -nolisten tcp

4.3 The lpd and lprng issue

Imagine, you arrive at work, and the printer is spitting out endless amounts of paper because someone is DoS'ing your line printer daemon. Nasty, isn't it? So keep your printer servers specially secure. FIXME. Content missing. (No lpr experience)

4.4 Защита почты

Чтение/прием почты - это наиболее распространенный вид соединений с открытыми паролями. Если для приема почты вы используете POP3 или IMAP, то ваши пароли идут по сети в открытом виде, и почти любой может их перехватить. Лучше использовать для приема почты SSL (Secure Socket Layer). Другая альтернатива - это ssh, если у вас есть доступ к оболочке на вашей машине. Вот базовый fetchmailrc:

```
poll my-imap-mailserver.org via "localhost"
  with proto IMAP port 1236
      user "ref" there with password "hackme" is alex here warnings 3600
    folders
      .Mail/debian
    preconnect 'ssh -f -P -C -L 1236:my-imap-mailserver.org:143 -l ref
     my-imap-mailserver.org sleep 15 </dev/null > /dev/null'
```

Строка preconnect - важная строка. Она запускает ssh-сессию и создает необходимый туннель, который автоматически перенаправляет уже зашифрованные соединения в localhost порт 1236 на почтовый сервер imap. Также можно использовать fetchmail с ssl.

Если вы хотите работать с шифрованными почтовыми сервисами наподобие POP и IMAP, то запустите команду apt-get install stunnel и запускайте ваши демоны вот так: stunnel -p /etc/ssl/certs/stunnel.pem -d pop3s -l /usr/sbin/popd Эта команда запускает из себя указанный демон (-l) на порту (-d) и использует сертификацию ssl (-p).

4.5 Защита BIND

В стандартной установке Debian демон доменных имен, BIND, запускается от имени суперпользователя и группы суперпользователя. Можно легко запустить BIND от ID (UID) другого пользователя. Однако, если BIND будет запущен от другого пользователя, то BIND не сможет определить новые интерфейсы автоматически. Например, если вы вставляете PCMCIA-карточку в свой ноутбук. Подробнее этот вопрос описан в файле README.Debian в каталоге документации named. Есть и другие проблемы безопасности, связанные с BIND, потому переключиться при возможности на другого пользователя крайне полезно.

Чтобы запустить BIND от другого пользователя, сначала создайте отдельную группу и пользователя для него (запускать от nobody и nogroup сервисы, который нежелательно запускать от суперпользователя, - не очень хорошая идея). В данном примере были использованы пользователь и группа 'named'. Создаем их:

```
# addgroup named
# adduser --system --ingroup named named
```

Теперь редактируем /etc/init.d/bind вашим любимым редактором и заменяем строку, начинающуюся с

start-stop-daemon —start на start-stop-daemon —start —quiet —exec /usr/sbin/named — -g named -u named

Остается только перезапустить bind командой '/etc/init.d/bind restart', и затем убедиться, что в вашем файле syslog есть записи типа:

```
Sep  4 15:11:08 nexus named[13439]: group = named
Sep  4 15:11:08 nexus named[13439]: user = named
```

Вуаля! Ваш named теперь работает не от суперпользователя. Чтобы еще больше защитить BIND, сделайте для этого демона chroot (См. 3.13). # I'm not sure about this, shouldn't bind files be chown'ed to # the groups created. This should be stated. jfs

Глава 5

Before the compromise

5.1 Follow Debian security updates

As soon as new security bugs are revealed in packages, debian maintainers and upstream authors generally patch them within days or even hours. After the bug is fixed, a new package is provided on http://security.debian.org. Put the following line in your sources.list and you will get security updates automatically, whenever you update your syste m.

deb http://security.debian.org/debian-security potato/updates main contrib non-free Most people, who don't live in a country which prohibits importing or using strong cryptography, should add this line as well: deb http://security.debian.org/debian-non-US stable/non-US main contrib non -free If you want, you can add the deb-src lines to apt as well. See the apt manpage for further details.

5.2 Exchange software

Вы должны попытаться избавиться от сетевых сервисов, которые отправляют и принимают пароли по сети открытым текстом, таких как FTP/Telnet/NIS/RPC. Автор всем советует использовать ssh вместо telnet и ftp. Также, по возможности не используйте NIS, Network Information Service, поскольку он разрешает совместное использование паролей. При неаккуратной настройке это очень небезопасно. Last, but not least, disable RPC wherever possible. Many security holes for this service are known and can be easily exploited. On the other hand NFS services are quite important in some networks, so find a balance of security and usability in a network. Most of the DDoS (distributed denial of service) attacks use rpc exploits to get into the system and act as a so called agent/handler. Запретить portmap проще простого. Есть два способа. Простейший способ для системы Debian состоит в том, чтобы дать команду update-rc.d portmap remove. This in fact removes every symlink relating to portmap in /etc/rc${runlevel}.d/ (you could do this manually yourself) . You could as well chmod 644 /etc/init.d/portmap, but that gives an error message when booting. You can also strip off the "start-stop-daemon" part in the /etc/init.d/portmap shell script. Имейте в виду, что переход с telnet на ssh с использованием других протоколов, передающих незакрытый текст,

в ЛЮБОМ случае не увеличит вашу защиту! Лучше было бы удалить ftp, telnet, pop, imap, http и установить вместо них соответствующие им сервисы, поддерживающие шифрование. Вы должны подумать над переходом на SSL-версии этих сервисов: ftp-ssl, telnet-ssl, pop-ssl, https ... Большинство из вышеприведенных советов применимы к каждой системе Unix.

5.3 Полезные заплаты для ядра

Для ядра существуют заплатки, значительно повышающие защиту системы. Вот некоторые из них:

- OpenWall patch by Solar Designer. Это полезный набор ограничений для ядра, таких как ограниченные ссылки, FIFOs в /tmp, ограниченный /proc, специальная обработка файловых дескрипторов, неисполняемый стек в пользовательской области и многое другое. Домашняя страница: http://www.openwall.com/linux/

- LIDS - Linux intrusion detection system. Авторы: Huagang Xie & Philippe Biondi. Эта заплата облегчает процесс создания защищенной системы Linux. Вы можете вводить ограничения для каждого процесса, давать им права писать и читать файлы, или удалять, умолчания, возможность читать файлы. К тому же вы можете также установить возможности для определенных процессов. Несмотря на то, что это пока бета-версия, это отличный инструмент для администратора-параноика. Домашняя страница: http://www.lids.org

- POSIX Access Control Lists (ACLs) for Linux. Эта заплата добавляет в ядро Linux списки управления доступом (access control lists), мощный метод для ограничения прав для файлов. Домашняя страница: http://acl.bestbits.at/

- Linux trustees. This patch adds a decent advanced permissions system to your Linux kernel. All the objects are stored in the kernel memory, which allows fast lookup of all permissions. Домашняя страница: http://www.braysystems.com/linux/trustees.html (http://www.braysystems.com/linux/trustees.html)

- International kernel patch. This is a crypt-oriented kernel patch, therefore you have to pay attention to your local laws regarding the use of cryptography. It basically ad ds use of encrypted file systems. Домашняя страница: http://www.kerneli.org

- SubDomain. A kernel extension to create a more secure and easier to setup chroot environment. You can specify the files needed for the chrooted service manually and do not have to compile the services statically. Домашняя страница: http://www.immunix.org/subdomain.html (http://www.immunix.org/subdomain.html)

- UserIPAcct. This is not really a security related patch, but it allows you to create quotas for the traffic on your server per user. And you can fetch statistics about the user traffic. Домашняя страница: http://ramses.smeyers.be/useripacct (http://rsmeyers.3ti.org/useripacct)

5.4 Genius/Paranoia Ideas, what you could do

This is probably the most unstable and funny section, since I hope that some of the "duh. that sounds crazy"-ideas might be realized. Following here you will find some - well, it depends on the point of view whether you say they are genius, paranoid, crazy or secure - ideas to increase your security rapidly but you will not come unscathed out of it.

- Playing around with PAM. As said in the phrack 56 PAM article the nice thing with PAM is that "You are limited only by what you can think of." It is true. Imagine суперпользователь login only possible with fingerprint or eyescan or cryptocard (hmm, why did I do an OR conjunction and not AND here).
- Fascist Logging. I would say everything we talked about logging above is "soft logging". If you want to perform real logging, get a printer with fanfold paper and log everything hard by printing on it. Sounds funny, but it's reliable and it cannot be removed.
- CD distribution. This idea is very easy to realize and extremely secure. Create a hardened debian distribution, a damned good firewall, make an ISO of it and burn it on CD. Make it bootable. Upshot of all this is a ro whole distribution with about 600 MB space for services and the fact to make it impossible for intruders to get read write access on this system. Just make sure every data which should get written, gets written over the wires. Anyway, the intruder cannot change firewall rules, routing entries or start own daemons (he can, but reboot and he has to hack into your system again to change them).
- Switch module capability off. When you disable the usage of kernel modules at kernel compile time many kernel based back doors are impossible to implement, since most of them are based on installing modified kernel modules.

Глава 6

After the compromise

6.1 General behavior

If you really want to clean up residual wastes, you should remove the compromised host from your network and re-install the OS from scratch. This might not have any effect if you do not know how the intruder got root. In this case you must check everything: firewall/file integration/loghost logfi les and so on.