The RSBAC library

The RSBAC library

The RSBAC introduction

Amon Ott

Stanislav Ievlev

Heinrich W. Klöpping

The RSBAC introduction
by Amon Ott, Stanislav Ievlev, and Heinrich W. Klöpping

Audience: This book is intended for use by experienced and skilled Unix professionals who wish to install, configure and use RSBAC.

Approach: This book resulted from a project founded on June 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. It provides an introduction to *Rule Set Based Access control* (RSBAC). If you are new to RSBAC, this book is the place to start reading. It provides an overview of what RSBAC is and how it can be employed. It is aimed at both potential users of RSBAC and programmers that would like to enhance the software by writing their own modules - or even changing the software itself. This book also introduces its companions: *The RSBAC cookbook*, *The RSBAC reference manual*, *The RSBAC programmers cookbook* and *The RSBAC programmers reference manual*.

To learn where the latest version of this book can be downloaded or read please refer to Section 6.2.

Sources: Our sources of information were (Open Source) material on the Internet, several books, practical experience of the authors , research and programming work done by the authors and others. We try to give credit where due, but are fallible. We apologize.

Caution

While every precaution was made in the preparation of this book, we can assume no responsibility for errors or omissions. When you feel we have not given you proper credit or feel we may have violated your rights or when you have suggestions how we may improve our work please notify us immediately so we can take corrective actions.

Organization of this book: This book has been organised in three parts:

I. part I - An introduction to RSBAC
II. part II - An introduction to the other books
III. part III - An introduction to the RSBAC documentation project

This book was written using the DocBook V3.1/SGML documentation standard.

Dedication

> And I said, "You can stop if you want with the Z
> Because most people stop with the Z - but not me!
> In the places that I go there are things that I see
> That I never could spell if I stopped with the Z.
> I'm telling you this because you're one of my friends
> My alphabet starts where your alphabet ends!
>
> You'll be sort of surprised what there is to be found
> Once you go beyond Z and start poking around"
>
> —Dr Seuss *On Beyond Zebra*

Table of Contents

List of Tables

List of Figures

Preface

Linux has always been a very stable and trustworthy operating system - even more so in comparison with it's closed-source alternatives. Driven by closed source vendors' questionable license policies, security risks, bugs and vendor-lock, more and more IT-managers choose the Linux alternative. Linux also has a good reputation -- as have other Unices -- when it comes to security. That may or may not be due to the blatant lack of proper security in other "operating systems". However, your data is arguably *not* safe on a standard Linux system. Linux is susceptible to security breaches, malware and programming bugs too.

Hence, a number of workarounds and extensions have been written. One of the most popular (and in my not so humble opinion one of the most elegant and stable ones) is *Rule Set Based Access Control*. I have been working with RSBAC since 1999 and have been impressed by what Amon wrote ever since. However, the lack of a good cookbook for it struck me as one of the major hurdles on the road to its acceptance. For I am convinced that it deserves such broad acceptance given its qualities.

So, we set out to write such a cookbook. And here it is. It still is a work in progress, and unless the nature of security related work suddenly changes probably will be so indefinitely. The authors wanted to create a useful book that would guide you through the seemingly awkward process of understanding, installing and maintaining RSBAC. This first version of our book originated from various materials, amongst them the introduction to RSBAC written by Stanislav Ievlev, the original documentation written by the author of RSBAC, Amon Ott, and a sequel of four articles I wrote for the Dutch Linux Magazine. We had to write many additional chapters from scratch and many hours were spent researching and trying actual security configurations. We finally made it.

It is up to you to judge our efforts, and, hopefully, improve our work. To quote the laudated Dr David A. Lien, author of the excellent (1977) TRS80 Level I BASIC beginners manual: "Sit back, relax, read slowly as though savoring a good novel, and above all, let your imagination wander. I'll supply you with all the routine facts and techniques you need. The real enjoyment begins when your imagination start the creative juices flowing and a computer becomes a tool in your own hands. You become its master - not the other way around.".

To the many people we unintentionally forgot to give credit where due, from the bottom of our hearts: **we thank you.**

Henk Klöpping, December 2002

I. Introducing a safer system

The first part of this book gives a generic overview of security related problems and introduces RSBAC.

Chapter 1. An introduction to security issues

1.1. Abstract

Rule Set Based Access Control (RSBAC) is a Unix (currently Linux) based security framework, that consists of kernel enhancements and related patches available for recent Linux kernel versions. It controls access to computer resources. The proper use of RSBAC renders many commonly used workarounds for Unix security superfluous and enhances others. RSBAC offers high granularity and flexibility and can be extended using a well defined API by means of kernel(-like) modules.

1.2. UNIX security related problems

Unix was not created with security in mind[1]. During its first 20 years Unix was mainly used by computer professionals, and computer crime was a rarity. The need to exchange information prevailed over the need to protect it and so the need for more than just a rudimentary security system was often absent[2]. The popular belief that (any) Unix is a 'secure operating system' may well stem from the fact that mainstream operating systems that were introduced later proved to be notoriously *unsafe*.

On a Unix (Linux) system access to system objects like memory and files is granted to *processes*. These processes are initiated by (or on behalf of) system users. A process typically loads a program to perform its tasks.

A task can be performed safely if the process has *just enough* rights. If a process has more rights its program may gain access to classified system objects. By clever use of known bugs or backdoors in the program code a cracker could even manipulate classified objects, which in turn may result in loss of data and performance penalties.

To prevent this, some kind of system needs to be put in place that gives processes "just enough" rights. On an average Unix system this defaults to a crude mechanism: Unix file permissions. This model lets the kernel decide whether or not a process may access a system object (memory, device, disk etc.) based upon persistent lists of permissions, that are part of the filesystem. The kernel checks these lists to determine if access should be granted or denied.

This only provides a crude granularity: for files we have lists that can specify access for a user, a predefined group of users (in which everybody has exactly the same rights) and "everybody else". We only can specify read-, write- and execute permissions. That's it. If you want to grant one user reading rights, another user reading and writing rights and yet another user execution rights - and everybody else no rights at all - you're stuck. You are forced to solve this at the application level.

The low granularity of this security model often causes Unix processes to have too many rights. If more granularity is needed one has to resort to adding security related code into the program source code. But what if you do not have access to the source code? Or what if the programmer made a mistake? A running process can be manipulated i.e., by employing some bug. The situation is worsened by the fact that Unix permissions are often not properly configured.

Then there is the system user named ’root’. It’s a very remarkable user: any process that acts on behalf of that user can in effect do anything it wants, including manipulating system data to mask its actions. The obvious solution - do not grant ’root’ privileges to processes anymore - can not be implemented, since access to many system objects depends on root rights. For example access to one of the privileged ports, let’s say port 80. The **http** daemon that uses that port is *required* to have root rights. Properly written daemons will revoke that right as soon as possible, but that’s not mandatory and can not be enforced by the kernel. As far as the kernel is considered any process that is started on behalf or ’root’ can do whatever it wants.

But Joe Average User may also inherit too many rights. On a standard Unix system an average user is allowed to create files and can allow other users to gain access to them. "Of course" you may say. But let’s assume Joe is a human resource manager whose files are highly confidential. On many instances the Joe’s in this world are not even aware of file permissions. The default permissions well may allow other users in his group to read his files. Anyway, on a Unix system at least ’root’ will always be able to read and alter his files.

Allowing Joe to access his own files seems fair enough. But the standard Unix security model assumes that it is Joe who is allowed to grant access to this file to other users too. That assumption is often incorrect. And when we are considering the fierce European privacy laws and regulations, the Unix security model is seriously flawed to say the least.

1.3. Workarounds and extensions

The flaws of the Unix safety model have been known for a long time. However, modern Unices like Linux are easily extended. A wealth of security-related patches, extensions and enhancements have been created for Linux. And the Linux kernel itself underwent a number of security related improvements like support for POSIX capabilities.

Since most Unix (Linux) computers are linked to a public network nowadays -- for example to the Internet -- even more security enhancements are necessary to protect the system. Some examples include:

Linux capabilities

> ’Linux capabilities’ allow finer grained distribution of ’root’ rights. Instead of having to grant or deny *all* rights, the regular ’root’ rights are split up and can be granted individually to a program or process. There are not many programs yet that are capable of handling these capabilities[3].

Extra file attributes

> A number of Linux filesystems allow setting of extra permission flags, like ’open for append’. Check out the **chattr** manual pages. Alas, ’root’ is still able to revoke these rights at will. Some versions of the kernel do not support validation of these attributes.

Hardening

> Toolsets like ’bastille’ *(http://www.bastille-linux.org)* (http://www.bastille-linux.org) will help closing many commonly misused security holes. I wholeheartedly recommend its use. However, none of the

granularity problems are solved, nor is the omnipotence of 'root' challenged. Some hardening toolsets are only available for a limited number of distributions.

Intrusion detection

Intrusion detectiontools like 'tripwire' *(http://www.tripwire.org)* (http://www.tripwire.org) help in finding unintended changes on your system. However the aforementioned problems are not tackled at all. Intrusion detection is very useful but does not do anything to protect your system.

Network security extensions

Well known programs like **tcpd** help to prevent unwanted network access to your system. Kernel mechanisms and accompanying configuration software like **iptables** and **ipchains** are very useful and can enhance your security significantly. But once someone gains access to your system, either legally or not, the old security related problems roar their ugly heads again.

chrooted processen

Sometimes daemons or other untrusted processes are run from within a **chroot** jail. This prevents an intruder who employs a bug or backdoor within the daemon from doing too much damage on your system, but he or she still can do damage within the **chroot**ed jail. And if this happens to be your **ftp** server, and the intruder succeeds in attaching worms or virusses to files intended for download, a lot of harm can be done in your name. And of course, the protection is limited to the **chroot**ed daemon. Other processes may still threaten your security. The 'root' user is still able to access everything on your system.

Encryption

One of the better ways to prevent someone to read your data is to encrypt it. Encryption adds the benefit of preventing someone to steal one of your backup tapes and read confidential data from it. But to be able to read your data, you need to decrypt it. And since root also is allowed to connect to your processes (for example using '**strace**') chances are that 'root' can determine the password/passphrase you use to encrypt your private key. Also, 'root' is able to destroy your data by simply altering it or removing it.

Configurationtools

Configurationtools like **PIKT** or **cfengine** assist with the configuration of conventional system parameters. They will check your system and correct flaws they find. But they also may require a process to run as 'root' to enable it to (re)configure your system. A system that can assist you in keeping your system well-configured is indeed very nice to have, but the core problems are not solved by these programs.

Work-arounds and extensions made Linux a better and safer system. But every new security related problem may require yet another extension or work-around. Every work-around or extension introduces new risks and hence the reweighting of risks against each other (some additional background can be found in Appendix B). Administering a secure system riddled with various extensions and work-arounds can easily become a nightmare. Only a few specialists are available that really dig these tools and fully understand their configuration and inner workings. RSBAC will solve these problems once and for all and additionally will eliminate root's omnipotence.

1.4. Alternatives for RSBAC

Much work has recently been done to enhance security on the kernel level of Open Source operating systems. For example TrustedBSD (which is a project that aims at adding security extensions to the FreeBSD kernel), LOMAC extensions for Linux *(http://freshmeat.net/projects/lomac)* (http://freshmeat.net/projects/lomac), POSIX.1e ACL's *(http://acl.betbits.at/)* (http://acl.betbits.at/)) and the aforementioned Linux Capabilities.

For some time the American NSA (National Security Agency) has been working on SELinux - which is an addition to the Linux operating system that strongly resembles the RSBAC extensions for Linux. However it is based on a different theoretical framework. Still, SELinux is similar with RSBAC in many ways. Check out *http://www.nsa.gov/selinux/faq.html* (http://www.nsa.gov/selinux/faq.html) for additional information.

Those of you who want to dig into the differences of these two systems may want to read a discussion between Amon Ott en Stephen Smalley (SELinux). Check out the thread that starts on *http://www.nsa.gov/selinux/list-archive/0344.html* (http://www.nsa.gov/selinux/list-archive/0344.html) or *http://www.rsbac.org/oldarchives/rsbac.2001/date/article-248.html* (http://www.rsbac.org/oldarchives/rsbac.2001/date/article-248.html).

SELinux has been offered tot the Open Source community as a prototype on 1.1.2000. RSBAC already had a positive track-record since 1996. Both solutions stem from (academic) research, both of them are fully Open Source. The similiarities will undoubtetly lead to mutual adaptation of algorithms and code. It is my fondest desire that both systems will gain from each other. For now, my humble opinion is that RSBAC still holds the advantage over SELinux: it sustains a high quality and stability and has been around for many years. It’s famous for its modularity and extensibility which has lead to the availability of security models that no other system can offer, for example the Privacy Model.

Notes

1. More information about standard Unix security can be found at http://secinf.net/info/unix/usec/unix_information.htm (http://secinf.net/info/unix/usec/unix_information.htm)
2. Not withstanding the fact that even in the late 80’s and early 90’s of the last century secure Unices have been available that satisfied the (now superfluous) C2/B1 “ Orangebook (http://csrc.ncsl.nist.gov/secpubs/rainbow/std001.txt)” levels.
3. Version 1.2.0 of RSBAC supports Linux capabilities.

Chapter 2. RSBAC Introduction

2.1. RSBAC overview

The RSBAC project started as its author's master thesis in November 1996 at Hamburg University. In the spirit of academic and Open Source principles it bases on earlier work of various researchers, amongst them Abrams and LaPadula [1]. Abrams et al designed a generic framework for access control (GFAC - Generalized Framework for Access Control) which Amon Ott adapted for Linux. All source code has been written independently. No companies or governments have steered or influenced its creation. All of its sourcecode has been published under the terms and conditions of the GPL.

On a running Unix (Linux) system programs/processes are not able to bypass the kernel if they want to use system resources like disks, io-ports and memory. Hence the only way to enforce security policies has been (and needs to be) within the kernel code. That reasoning lead to the rise of a number of security extension on the standard Linux kernel, amongst whom RSBAC takes a special place. RSBAC offers a sound framework that bases on a generic and generally well accepted theoretical model. It allows the inclusion of modules that implement one or more security enforcement models.

The RSBAC environment uses an object oriented model, derived from the generic GFAC model. The GFAC model allows definition of the functionality of security related software on the abstract level. With other words: the GFAC model can be used for all operating systems. It defines an abstract view, for example: "a subject can issue a request to an object which will be denied or granted", or "within this model there needs to be a component that decides and another one that enforces that decision".

RSBAC is a GFAC implementation for Unix. It honours the GFAC model but is more specific: "the subject (proces) that loads the program `/a/b/c/` is not allowed to send a request of type `READ_WRITE_OPEN` to the `FILE` object '`/tmp/a`" or "the component that enforces the decision is named AEF, the component that decides will be know as the ADF and both will be implemented as part of the kernel code".

On an even less abstract level (RSBAC for Linux) this translates into code that implements "if a process that loaded the program `/a/b/c` tries to execute the system call `open('tmp/a', O_RDWR)` this will be denied." and "the AEF is implemented as a series of patches for the kernel".

2.2. Architecture

Figure 2-1. RSBAC components

Figure 2-1 shows a schematic overview of the architecture. The GFAC model has been translated into Linux kernel code that decides which calls are allowed (the ADF) and another block of code that enforces these decisions (the AEF). The AEF is by nature operating system specific and therefore it was kept as small as possible, as one of RSBAC's design goals was portability. The ADF contains most of the code and was written as platform independent as possible. RSBAC should be easy to port to (for example) FreeBSD or proprietary

Unices like HP-UX, Solaris and AIX. RSBAC uses kernel memory to implement a storage area for its status data, the ACI. Each block is described in more detail next:

2.2.1. ACI Access Control Information

This block contains all functionality that takes care of the (internal) RSBAC administration like locking information and linked lists. This data will be kept in kernel memory. Communication between AEF and ADF is mostly done using the ACI data. The code in this block also keeps track of the status of calls. If so configured, the data in the ACI will be flushed to disk regularly. Data will be stored in special directories. For each filesystem that needs to be mounted one such directory will be created. These directories are either named `/rsbac` (up to version 1.2.0) or `/rsbac.dat` (version 1.2.0), for example: `/var/rsbac.dat` or `/usr/rsbac.dat` etc. Note, however, that the ACI does not store decision rules: these are hardcoded as decision modules. Each module forms one rule set.

2.2.2. AEF Access control Enforcement Facility

This block comprises of functionality to intercept and possibly deny Linux kernel calls to processes. The AEF will fetch status-data from the ACI and will communicate with the ADF (described later) to learn whether or not it needs to grant access to the system call. The AEF will either deny or grant the call accordingly and will notify the ADF.

2.2.3. ADF Access control Decision Facility

This code receives a notification from the AEF when a process issued a kernel system call. Any related data will be fetched from the ACI and a chain of decision modules will be traversed. Each module will independently decide whether or not the call is acceptable, based on the model it implements and the parameters and context of the call. At the end of the chain some modules may have denied access, while others may have granted it. The ADF will make the final decision, normally it uses a restrictive algorithm: if only one module rejected the call, the call will be rejected. Next, the ADF communicates its decision to the AEF. The ADF will be called by the AEF again after completion of a granted system call and the ACI will be updated accordingly.

In short: the AEF intercepts calls and either allows or denies them, the ADF is the part where the decisions are made and the ACI is used for storage of status data and configuration items.

2.3. RSBAC terminology

The terminology used within the documentation and software may require some clarification. The terminology is used regularly within RSBAC logfiles too. RSBAC makes use of three important concepts: *Subjects*, *Objects* and *Requests*. An instance of an Object is called a *Target*, object classes are sometimes called *Target Types*.

- on a Linux system a RSBAC subject is equivalent to a process that wants to perform some type of action on an object (for example memory or a file);

- an object (i.e. memory, disk) will be an instance of an object class. Sometimes an object class is referred to as a *target type* and the instance of an object class is sometimes referred to as a 'target'. Hence the term 'target' and 'object' are equivalents. For example the file `/etc/passwd` is an object (target) of type (target type, object class) FILE;
- and finally a request is an abstraction for an action to be performed by a subject on an object (target). There are many dozens of requests, most of them are valid within more than one object class.

An example: a process wants to open the file `/etc/passwd`. The process has PID 1899 and has loaded the program stored in file '`/usr/bin/vi`'. Now `/etc/passwd` is an object of class `FILE` - or alternately we could say it's a target of target type `FILE`.

Using the RSBAC terminology we would say that a SUBJECT (the process) with `caller_pid` 1899 and `caller_prog_name vi` issued a REQUEST of type `READ_WRITE_OPEN` for OBJECT (`target_type`) `FILE` with tid (target id) `/etc/passwd`.

2.4. RSBAC Objects and Target Types

Apart from the targettype `FILE` there are types `DIR` (directories), `FIFO` (named pipes), `SYMLINK` (symbolic links), `DEV` (devices, denoted by their major and minor number), `IPC` (shared memory, semaphores etc.), `USER` (systemusers), `PROCESS` (processes) and `SCD` (System Control Data) Version 1.2.0 adds `NETDEV` (network devices), `NETTEMP` (network templates) and `NETOBJ` (network objects). The class `SCD` (system control data) contains all objects that relate to the entire system, like the real time clock, the systems name or the systems domainname or raw access to kernel memory. There are some requests that do not relate to one of the defined object classes, for example the request to add a kernel module. In these cases the placeholder objecttype '`NONE`' is used.

2.5. RSBAC Requests

A request is an abstraction for an action that a subject wants to perform on an object. A request can be valid within many object classes. For example `READ_OPEN`, which is a valid request for object classes (target types) `FILE`, `FIFO`, `DEV` and `IPC` object. Let's assume a process want to use shared memory. To gain access the process will issue the system call '`shmat()`'. Using the RSBAC terminology we would say this is "a `READ_OPEN` request on a target type `IPC`". We could request the same on a target type FILE, probably the AEF will have intercepted the system call '`open()`'.

After installation of RSBAC the file `/usr/src/linux/Documentation/rsbac/html/targetsrequests.htm`. will be present. It contains the detailed list of all intercepted systemcall's and the related object classes. For now I will give just a few examples of system calls intercepted by RSBAC:

`CHANGE_OWNER`: set UID on this object
`DELETE`: remove this object
`READ_OPEN`: open this object for reading
`SEND_SIGNAL`: send a signal to this object
`CLONE`: fork a process
`LINK_HARD`: make a hard link

`MOUNT`: mount a device on a directory
`REMOVE_FROM_KERNEL`: remove a module from the kernel

2.6. *Inheritance of RSBAC settings

[PBU] An important aspect of RSBAC is inheritance. Normally in UNIX the permissions of one file are independent from other files or directories. In RSBAC, they can depend on one of the directories leading to the file or directory. The default setting in RSBAC is to inherit the settings from the parent directory, which in turn can inherit settings from its parent directory, and so on, until the root directory (`/`) is reached. This mechanism is called *inheritance*.

The consequence is that changing the settings of a directory automatically changes the settings of files and directories below that directory. This provides an efficient way to administer the security of large numbers of files.

[* add an example of `FF no_execute` on `/home`, which prevents programs from being executed in users' directories]

[* also some text about when files do not inherit from their parent directory. I.e. by setting stuff explicitly. Perhaps also add that inheritance is on per-module basis, so setting FF stuff does not affect RC stuff and vice versa. Maybe I left out some special conditions which affect inheritance, which could be added too]

2.7. The security officer

To configure your modules you can use the utilities provided in the RSBAC distribution. By default these utilities can only be executed by a special user - the security officer. By default UID 400 is used for the security officer but you can opt for another UID during RSBAC installation/compilation.

Note that even 'root' needs processes to change its UID to 400 - and on a RSBAC system a process does not have these rights by default, not even the processes that run with UID 0. The only way to become security officer is to log in as such. In practice this is implemented by offering an SSH connection to the security officer and/or by allowing him to log in over the console.

Many people will start wondering now: 'what's the difference? Whether one has a Superuser with UID 0 who can acccess anything or a security officer with UID 400 who can configure access for anything to anybody, including himself?' The difference lies in the amount of effort needed to obtain these rights. The standard Unix kernel contains code that gives the user with UID 0 almost unlimited access to system resources. And any process that needs to run under an alternate UID needs 'root' rights to do so - like the http daemon we mentioned before, which needs root rights to access port 80 before (non mandatorily!) switching to a less dangereous UID. If there exists a security hole in a setuid binary which communicates with the outside world it may be exploited by a cracker to manipulate your system at will. Once he has root rights, he can do whatever he wants. *And a firewall will not protect you against that*.

On a RSBAC system it is impossible to become the security officer via a hole in a daemon (unless the security officer allowed it, of course). No daemon should ever run setuid 'security officer'. No deamon is allowed to change to that UID, regardless whether or not it has 'root' privileges. Not one user has the right to switch to UID 400. Not even 'root'. On a properly configured RSBAC system the only way to become security officer is by logging in as such using a console or secure encrypted connection. To become security officer on a tightly secured RSBAC system you need physical access to the system. Of course you need to know the password. Nobody - not even root - can become security officer, unless that same security officer allowed it.

You may observe that even on a RSBAC system 'root' might change the security officers password and log in as security officer himself. Yes, that would be a possibility, given of course that 'root' is allowed to change the password and shadow files. You can take additional measures, for example use additional authentication methods. Or can forbid write access to the password- and shadow files except for special programs, that are accesible only to (non-root) system administrators. On some of my own systems it is impossible to access the security officers account, unless you have a special hardware key plugged into the system. And not even 'root' is permitted to change the software that checks for the key. The point here is: you should be careful not to create the illusion of security. A badly configured RSBAC system does not really add much to your security.

On some systems you might even need to resolve to the situation in which the security officer is not allowed to work at all on the system. To alter the configuration on such systems you need to physically reboot and start up a special maintenance kernel to reconfigure the system.

Take good notice of the fact that RSBAC was designed to protect a *running* system. When you have physical access to a system such a system can not be properly secured unless additional measurements are taken. By simply unplugging the power cord, for example, even your RSBAC protected system will suffer from a denial of service attack (sic). For example, on a laptop, you really should make sure that you take additional provisions to protect your data, for example encrypted filesystems and hardware protection schemes.

2.8. The RSBAC implementation for Linux

RSBAC for Linux consists of a number of patches for the kernel and a number of utilities. The patches enable us to connect to our framework and the utilities are needed to define and configure the security models.

The kernel patches intercept system-calls and reroute them through a chain of decision making modules. If all modules accept the execution of the intercepted system call the request is accepted and the system call will be executed. If on the other hand just one module refuses the call no other module can revert that decision and the call will not be executed. Denial of a call will be logged to allow auditing.

Each RSBAC security module implements a security model. You can mix and match security models at will to implement your security *policy* by simply loading the modules you need. oes When a process issues a request for a system call it is rerouted through the chain of RSBAC modules. Each module determines whether or not the call is allowed by applying the rules of its model within the scope of the configuration you provided. If some rule forbids execution of the call the RSBAC framework refuses its execution. Only if *all* modules permit the execution of the call it will be executed. Any module can deny access to a call, but never undo the denial of another module. This is called a 'restrictive design' and is an important part of the philosophy behind the RSBAC framework and hence its implementations. Denials will be properly logged. RSBAC provides extensive logging facilities, both to standard Unix (Linux) logging facilities and to a RSBAC specific secured circular buffer.

The RSBAC distribution comes with a set of ready to use modules along with the necessary tools to configure them. Just load the modules you need to provide the proper security level. If you need more you can also design and write your own modules and add them within the framework. The RSBAC system offers a flexible, extensible solution that enables extensive elimination of security risks.

Now, how does a module interact with the outside world? How is it configured? An few examples will clarify this. One of the modules implements additional attributes for (parts of) the filesystem (`FF`). For example this module could be used to deny execution of files in and under a certain directory. Another module verifies if a process is allowed to change its UID (the `AUTH` module). This module enforces that a program/process only has limited access to other user identities. On a regular system, if a program is owned by 'root' and has its setuid bit set it can change in and out of *any* user identity. By using the `AUTH` module you can restrict its access to a limited number of identities. Note that the program/process has no control at all over this: it can issue the system call to change into an identity but that call simply fails (which will be logged) if you have not explicitly allowed it.

Though you can load entire stacks of modules that often is not necessary at all. By loading a basic set of modules you can already achieve a major improvement of your systems security. The more complex models and modules (for example the Privacy Model `PM`) are used in very specific and exceptional situations. In general it is wise to restrict yourself to the use of modules whose functionality you fully understand.

Notes

1. La Padula, L. J., Rule Set Modeling of a Trusted Computer System, Essay, in: Information Security: An Integrated Collection of Essays, Hrsg.: Abrams, M. D., Jajodia, S., Podell, H. J., IEEE Computer Society Press, 1995 (also available on the Internet, see *http://www.acsac.org/secshelf/09.pdf* (http://www.acsac.org/secshelf/book001/09.pdf)).

Chapter 3. The RSBAC models

3.1. The modules/models provided with RSBAC

If you employ RSBAC you have to try to limit yourself to using the modules/models your really need - and, most important: that you really understand. The 'common' needs are covered quite well by using a combination of the `AUTH`, `RC` en `FF` models. I will describe some of the most used models and modules below.

All other models (`ACL`, `MAC`, `FC`, `SIM`, `PM`, `MS`, `CAP` en `REG`) are specifically used for uncommon needs. I will not describe them in this introduction but you can find full descriptions of these models in the section *RSBAC Models* in *The RSBAC reference manual*.

Some of these lesser used models serve as practical demonstrations for some security theory. Some of them overlap in functionality. Others are complex and hard to administer. The `FC` model on the other hand is regarded upon as hard to employ because it allows only low granularity. The RSBAC `PM` model allows a correct implementation of the rules of the strict European privacy regulations and laws in automated systems. To the best of my knowledge it is the only implementation of that functionality existing today.

The wealth of available models can easily confuse the novice. In itself the number of ready-to-use modules makes a strong point for the software: you can freely choose from all of them. Models you do not understand (let alone use) can be left out during installation. They will not use any of your precious resources.

Another important module is the `REG` module. It is not a conventional module but offers an interface annex framework to create your own security modules. Modules built using the `REG` module can be loaded as if they were regular kernel modules. The RSBAC source code contains a number of examples how to use the `REG` module. If you have any special security needs that can not be satisfied by one or more of the existing modules you can roll your own. Very powerful.

3.1.1. The `AUTH` module

The `AUTH` module is a mandatory part of RSBAC. Its provides functionality needed by all other modules. `AUTH` allows subjects to change their identity - in other words to `AUTH`enticate themselves. The Linux implementation of the `AUTH` module handles requests from a process to change its UID. Whether or not a call to change UID should be accepted is decided by inspecting the configuration data stored in the ACI.

Of course the actual system call can only be executed by a running *process* but administration of these rights is done by allowing *programs* to execute `AUTH` rights. The configuration data is normally kept in non-volatile memory (actually a part of your hard disk), the ACI. If such a program is loaded within a process the rights of that program are inherited by the process itself. So when that process requests a change of UID the `AUTH` module will allow it. By default *no program/process has rights to change its UID*[1].

`AUTH` permissions can be issued on two levels: either a process/program is allowed to assume any UID it wants to (`auth_may_setuid`) or you have to provide the module with a list of UID's a program may assume (its 'capabilities'). The functionality both methods offer is basically the same: setting `auth_may_setuid` can be thought of as setting all possible capabilities for a UID.

You can configure the `AUTH` module using the command line utility '**auth_set_cap**'. By default this program will only accept calls from the security officer (defaults to UID 400).

3.1.2. The FF module

The File Flags model allows you to set additional attributes for files, fifo's and directories. These attributes are enforced upon all subjects (processes) on the system. As always the ACI is used to store the additional attributes. If you set additional attributs for an object the following nine bits of data will be stored in the ACI (the value within brackets indicates which bit is used): `read_only` (1), `execute_only` (2), `search_only` (4), `write_only` (8), `secure_delete` (16), `no_execute` (32), `no_delete_or_rename` (64), `add_inherited` (128) and `append_only` (256). What they do can mostly be determined from their names, but I will elaborate on some of these:

- if you set the `secure_delete` flag for an object this signifies that the actual data in a file will be overwritten (using NULL-characters) if you delete the file. This makes it much more difficult to reconstruct what has been in the file. RSBAC is mostly filesystem independent and should thus work with any local filesystem. It has been tested (and works with) ext2, ext3, reiserfs (without inode number checking and `secure_delete`), xfs, minix and (v)fat[2].
- by setting the `add_inherited flag` you specify that the object will inherit the attributes of its parent directory. Be careful when setting this attribute: it may result in inaccessible files, for example if a directory has the `read_only` attribute set and you specify `write_only` en `add_inherited` for a file in that directory.

Most rights are inherited - in other words: by setting a flag on a directory the flag also is set for all objects (files/directories/fifo's) within that directory. This does not hold true for the `add_inherited` and `secure_delete` bits.

To set a fileflag you can use the command line utility **attr_set_fd**, for example:

```
$ attr_set_fd FD ff_flags 1 /a/b/c
```

Alas you need to calculate the bitmask yourself if you use the **attr_set_fd** command. For example to set both '`secure_delete`' (16) and '`no_execute`' (32) you would have to use 48 as fourth argument. Alternately you may set the `FF` flags using the command **attr_set_file_dir**, which allows the use of add/remove flags. To query the current file attributes you can use the utilitiy **attr_get_fd** and to reset/remove attributes you need to use **attr_rm_fd**. Again: by default only the security officer can alter the file flag permissions.

3.1.3. The RC module

The Role Compatibility model closely resembles the way people think. People will typically describe security issues in terms like for example: 'an operator should only be allowed to read logfiles'. The operator represents a role in a proces. The `RC` model also uses the 'roles' concept. A role can be seen as a label that belongs to a set of tasks a user may perform, like 'webmaster' or 'operator'. Such a role comes with a number of permissions to gain access to (certain) (classes of) objects.

The objects are categorised in 'types', which in turn can be given meaningful names like 'logfiles' or 'webdocuments'. A 'role' is granted access to certain 'types', for example: the 'operator' is allowed to access objects of type 'logfile', a 'webmaster' is allowed to access objects of type 'webpage'. What type of access a certain 'role' is granted can be specified too, so let's say that an 'operator' is allowed to access 'logfiles' - but he is only allowed to *read* them. Within the `RC` model we distinguish between objects of class `FD` (Files/Directories), objects of class `IPC`, of class `DEV`, `SCD` and `PROCESS`.

By using the `RC` model a subject is only allowed acces to certain forms of access to objects by assuming a 'role'. To do so, the subject needs authorisation. If authorised, the subject gains access to any object that is 'compatible' with his role. Any subject (process) is granted an initial role - and that role is inherited by any subject that is generated by it. On a Linux box a subject is a process and its rights are inherited by its children. So, when for example a user (Mary Model) logges in her login process will automatically assume the default role assigned to her, let's assume it is named 'rolemodel'. Now, all processes spawned by Mary's login are able to excersize any rights 'compatible' with the role 'rolemodel'.

The `RC` model allowes a subject to change its role. To do so, the subject needs to issue a special system call or it needs to load a program that has a 'forced role' set. More about 'forced roles' later. You can only change roles if the roles are 'compatible' - changing roles is only allowed if the security officer allowed it. Furthermore changing from role A into role B does not imply that you are allowed to change back from role B into role A. In other words: role B can be incompatible with role A while role A is compatible with role B.

Be careful: there is no mandatory relation between the UID of a subject (process) and its `RC` role. It might be that a process has the role 'operator' set and - assuming that the role 'operator' allows it - that process is allowed to change its UID but it still will be in its initial 'operator' role. To change to the default role for a UID you need to have your software perform an additional system call.

It is possible to set an attribute for a program to indicate it has a 'forced role': if any process loads that program the process will obtain the rights for that role. This somewhat resembles setting a set-uid bit on a program. It allows you to create programs that have access to certain objects by allowing a certain role to access the objects and consequently force that role upon the program.

Executables that do not have a role forced upon them by default inherit the role of the process that loads the program. Alternately you can set a flag to indicate that the program should inherit the default role of its UID. The program should have the setuid bit set and you should configure the `AUTH` module to allow that UID change.

These concepts are best illustrated using an example. We assume we want to create an anonymous ftp server. Such a server is typically run from within a chrooted jail which - even on a non RSBAC system - prevents us from many security breaches. We can increase our servers security even more by using the `RC` model.

To do so we create a role 'ftpserver'. We also add a new objecttype: the FD objecttype 'publicfile'. The program that implements our ftp server is given the default role 'ftpserver', which implies that all processes that load this program and al their children will inherit the privileges and restrictions for that role. Next we configure the toplevel directory for our ftp server to be of type 'publicfile' and by doing so imply that all subdirectories and files will also be of that type.

Then we configure the role 'ftpserver' and restrict its rights to reading and searching objecttypes 'publicfile'. Our ftp server also needs rights to access our network, so we need to define an additional object type: an IPC object type 'ftpnet'. The role 'ftpserver' will be granted the CREATE and DELETE rights on objects of type 'ftpnet' and for the role 'ftpserver' the default IPC type is set to 'ftpnet'.

By configuring it like this, it is impossible for the ftpserver to execute or delete any file in its subdirectories. Even when the ftpserver runs with superuser privileges and has bugs the size of Paris no harm can be done by it - the RC model prevents this.

If for whatever reason you want to be able to execute a file from within the ftp server - for example a copy of `/bin/ls` - you can create another role, let's say 'ftp-executable', which can be a copy of the role 'ftpserver', but with additional rights to enable it to execute files. By granting the copy of `/bin/ls` a forced role 'ftp-executable' the ftp server process that loads that program will inherit the proper rights to execute the program.

The tools to configure the `RC` module are **rc_copy_role**, **rc_get_item** and **rc_set_item**. Again, these tools can only be used by the security officer by default. The RSBAC documentation contains an good example too on how to configure a (more) secure webserver.

Notes

1. Which can be very frustrating for novice RSBAC users: after they have installed RSBAC and rebooted their new kernel they find out that they are not able to log on into their system anymore - by default the program `/bin/login` is not allowed to change UID either. However, there is a good workaround available to solve that problem: you can start up a newly installed RSBAC kernel with the `rsbac_auth_enable_login` option, which will allow `/bin/login` to change UID so you can log in. Note that this requires physical access to the system.
2. To be precise: secure truncation should work on all filesystems, but secure delete requires changes in the filesystem code. Since for example xfs is not part of 2.2 or 2.4 kernels, secure delete is not supported for it. In general it is considered dangereous to use secure delete on journalled filesystems. Please consider using encryption instead.

Chapter 4. RSBAC installation

4.1. RSBAC installation

This paragraph marks a milestone: by now you should have learned enough about RSBAC to be able to decide whether or not you need its functionality. If you are sure you do not need more than the standard security features a modern Unix (like Linux) has to offer, I thank you for your interest and bid you farewell. For those of you still with me by now: our next step will be to actually install RSBAC.

You will need to patch the Linux kernel to implement RSBAC. This will only work if you have the original kernel source code. At the time of this writing (may 2003) the stable version of RSBAC was 1.2.1 for kernels 2.2.21-25 and 2.4.19-20. Older kernels are supported too. Amon Ott tries his very best to support the newest development kernels too, check the RSBAC website to learn more. If for any reason you need to patch a kernel that is not supported I suggest you contact Amon Ott.

The software consists of a patch, the code for the kernel extensions that is hooked up to the kernel and an archive with administrative software. The RSBAC website (*http://www.rsbac.org* (http://www.rsbac.org)) provides a good source of additional documentation and links to other related materials. If you are eager to start you might want to read the relevant section in this book or read the relevant section on the RSBAC website. (*http://www.rsbac.org/instadm.htm*).

Start by installing the source code for one of the supported Linux kernels in `/usr/src/linux`. If you are not used to building your own kernels, for example because you use precompiled distributions, you may want to build a 'clean' kernel first to get a grip on the procedure. To learn how to compile kernels from source you might want to consult the kernel-HOWTO (*http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html* (http://www.linuxdoc.org/HOWTO/Kernel-HOWTO.html)). If you were succesful in booting your freshly build kernel and if that kernel provides the functionalities you want, you should save the kernel configuration in a safe place.

As reparation for the actual installation of RSBAC you could fetch the patch and accompanying software and create the user 'security officer'. Use any UID you want (except 0) - the default is 400, but you are able to override that during installation.

II. Introduction to the other RSBAC books

The second part of this book introduce the reader to other volumes of this bookset.

Chapter 5. Introduction to the other RSBAC books

5.1. Dummy section

Dummy text.

5.2. Dummy section

Dummy text.

III. An Introduction to the RSBAC documentation project

The third part of this book explains how the RSBAC documentation project is run. It explains how you can participate and what tools you need to translate the DocBook source code into other formats, for example into HTML.

Chapter 6. Introduction to The RSBAC documentation project

6.1. The RSBAC documentation project

The RSBAC documentation project was founded on June, 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. The projects target is to deliver and maintain a set of (Open Sourced) manuals. The audience for the manuals are skilled Unix professionals or Unix power users. The manuals were targeted to enable the audience to install, configure and use Rule Set Based Access control (RSBAC). They provide general information about security, references to other works, and instructions to install, configure and use RSBAC.

This part of *The RSBAC introduction* contains information about the way our project is structured, and provides information on how to write and publish documentation. It also contains a section about the tools we used to prepare this book and how to install these. This section is only relevant for those that want to participate in our project. Section 6.4 contains useful information for people that want to convert the DocBook sources into other formats.

6.2. First things first

For a proper understanding of our project you should learn a number of things first. You should know that the RSBAC library ..

- is written by *authors*, formatted by *editors*, proofread by *proofreaders* and divided into *chunks*. And that all these terms and more are explained in Section 6.3.
- is written using the DocBook V3.1 standard (SGML, using XML conventions where possible);
- is published under the GNU FDL;
- can be downloaded from the RSBAC CVS repository:

  ```
  $ cvs -d anoncvs@devel.altlinux.org:/cvs/rsbac co docs
  password: cvs
  ```

  This includes sources for the documentation and related code. *Uploading* materials is restricted to Editors and the Project Leader;
- can be read on-line at *http://books.rsbac.org*;
- is discussed by Proofreaders, Authors, Editors and the Project Leader using a closed mailing list, maintained by the Project Leader. The list can be found at *http://mail.rsbac.org/mailman/listinfo/rsbac-book* (http://mail.rsbac.org/mailman/listinfo/rsbac-book).

6.3. Project jargon

To be able to participate in (discussion about) our project you need to know a bit about the lingo we speak and the structure of our project. Please read all of the following subsections carefully. If you set out to be either an Editor, Author or Proofreader you consequently should read the corresonding section: Section 6.5, Section 6.7 or Section 6.4.

If after that you still do not understand how you may participate in this project, please address the Project Leader (you wil find his address in a later section), who will cheerfully answer your questions and update this document if necessary.

6.3.1. Chunks

To enable distribution of work over many people and to prevent versioning conflicts that CVS might not be able to resolve, this book will be divided in "chunks". A "chunk" is just a block of text in the book. Chunks are written by *AUTHORS* but are checked into CVS by *EDITORS*. Note, that the Editor for a chunk may or may not be the Author of that chunk.

6.3.2. Proofreaders

Anybody that can download versions of this book can become a 'Proofreader'. Proofreaders have access to the otherwise closed RSBAC-book mailing list (and its archive). To become a Proofreader one has to prove to one of the Authors or Editors to be worthy of that name by proofreading at least one chunk of the book and pointing out at least one error in it. If the Author or Editor acknowledges and corrects the error, the Project Leader is informed by the Author or Editor that has accepted the correction. The Project Leader will grant a Proofreader access to the closed mailing list and will see to it that the Proofreaders name is listed in new versions of the book in Section 6.8 and Section 6.9. Learn more about editors by studying Section 6.5.

6.3.3. Editors

Editors ensure that the materials they upload into CVS are consistent with the DocBook standard, and - in as much as is possible for DocBook 3.1 (SGML) documents - adhere to the guidelines for writing DocBook/XML documents. An Editor should never upload a chunk into CVS that does not comply with these standards. Editors will need to download other Editors contributions regularly and will have to test the total integrity of the book by converting the DocBook source code into HTML or Postscript, using one of the scripts provided in the CVS repository. If errors are found in chunks outside their jurisdiction, they will promptly inform the responsible Editor, so he or she can correct his chunk. Learn more about editors by studying Section 6.5.

6.3.4. Authors

Authors ensure that the content of the chunk is correct: an Author writes the actual content and is responsible for the validity of that content. Note, that it is entirely possible to be an Author without knowing anything about the DocBook standard nor CVS. Note, that it is entirely possible to be an Editor without knowing how to write proper educational texts. Authors will need to download other Authors contributions regularly (which of course only is possible if an Editor has uploaded these contributions first). They will proofread other Authors texts. If they find errors in chunks outside their jurisdiction, they will promptly inform the responsible Author, so he or she can correct his chunk. Authors will be appointed Editors by the Project Leader and deliver their work to the Editor. Learn more about authors by studying Section 6.7.

6.3.5. Translators

Anybody that can download versions of this book and is able to read and fully understand the English original and is fluent in (at least) one other language can become a 'Translator'. Translators have access to the otherwise closed RSBAC-book mailing list (and its archive). To become a translator you have to translate a part of the book into another language and send the Project Leader proof of that - for example the URL of the website where the translation can be found. Any translation is without the responsibility of the authors and should always be available under the same license as the original. The Project Leader will grant a Translator access to the closed mailing list and will see to it that the Translators name is listed in new versions of the book in Section 6.8 and Section 6.9. Learn more about translators by studying Section 6.6.

6.3.6. Delegation

Authors are allowed to delegate parts of their chunks to other Authors. Editors are allowed to delegate part of their chunks to other Editors. Delegation of *EDITORS* should be reported to the Project Leader. The delegating Editor will split the chunk into proper subchunks. The projectleader will reflect these changes into the Master delegation file. If the delegated work has been finished, the delegating Editor will reassemble the subchunks into one chunk and inform the Project Leader, who will again update the Master file to reflect the change. Delegation of Authors is transparant to the other project members: the Author will have to reassemble the delegated work and hand over his work as a whole to his Editor.

6.3.7. The Master Editor

Any problems regarding the technical consistency of the book that could not be resolved within a week by the Editors will become the problem of the Master Editor. The Master Editor will investigate the problem and try to pinpoint the Editor responsible for the problem. If just one Editor causes the problem, the Master Editor will respectfully inform that Editor of the nature of the problem and will offer assistence to that Editor to help him resolve the problem. If the problem is caused by 2 or more Editors the Master Editor will inform all Editors involved and appoint an Editor to resolve the matter. Since the Master Editor can be an Editor, he may be able to appoint himself. Additionally, the Master Editor is responsible for the integral quality of the DocBook code used in releases of the book.

6.3.8. The Master Author

Any problems regarding the correctness of the content of the book that could not be resolved within a week by the Authors will become the problem of the Master Author. The Master Author will investigate the problem and try to pinpoint the Author responsible for the problem. If just one Author causes the problem, the Master Author will respectfully inform that Author of the nature of the problem and will offer assistence to that Author to help him resolve the problem. If the problem is caused by 2 or more Authors the Master Author will inform all Authors involved and appoint an Author to resolve the matter. Since the Master Author can be an Author, he may be able to appoint himself. Additionally, the Master Author is responsible for the integral quality of the content in releases of the book.

6.3.9. The Project Leader

The Project Leaders guides the project. He issues and discusses guidelines and procedures. He resolves problems that can not be resolved by either the Master Author or Master Editor. He also acts as the spokesperson for the project and is the maintainer of this file and maintains the closed mailinglist for Proofreaders, Editors and Authors.

6.3.10. Handles

Often, people can be identified uniquely by their (full) names. However, sometimes that does not apply, for example when we have two people with the same name. To enable us to uniquely identify each Project Member, he/she is appointed a *handle* by the Project Leader. All handles ever issued are listed in this book in Section 6.8. Currently the handles are sequences of three capital letters - which means that we can uniquely identify 17576 people. The Project Leader will try to assign you a handle that matches your name (e.g. your initials) or you can suggest a handle. If the handle is available it will be appointed to you. Else, the first free handle that is in the suggested range will be appointed to you.

6.3.11. The TODO list

Writing chapters, reviewing them, maintaining support software, creating websites ... all that and more are *tasks* within our project. To maintain an overview of the tasks and to enable prioritisation of them, the Project Leader is responsible for maintaining a TODO list. That list is accessible for everyone with CVS access (discussed before in Section 6.2) and can be found in the subdirectory `/project/`. It should be kept in a strict format, which is documented in the TODO file itself. As you may have guessed: the list is also used by a number of programs, hence the strict adherence to a standard format. These programs are part of the CVS tree too and can be found under `/project/software`. One of these programs `/project/software/scan/genmail` generates weekly remainders for the Project Members and mails these to them.

6.3.12. Tasks and Items

A task is a set of (related) work that should be performed in order to help create a (better) set of books. A task typically is created if one of the Project Members signals that something needs to be done. Usually the task is discussed on the mailing list first. If it is not something that requires immediate attention and if the general feeling is that the task should be performed, it is entered (by the Project Leader) on the TODO list. Each task will then become an *item* on the TODO list. Items are formalised descriptions of tasks. They have unique numbers to enable unobfuscated identification (the *item number*), somebody will sooner or later take responsibility for it so we have to have a placeholder for his/her *handle*, a priority needs to be given to it (a number from 1-9: 1 suggests very high priority) and we have to track the date it was entered and the date it was completed. A description of the task is of course part of the item too.

6.3.13. Events

Apart from *tasks* we also talk about *events* sometimes. An event is a task that needs to be completed with utmost priority - in fact: it can not wait for completion long enough to enter in as an item in our TODO list. Often the person that handles an event already handled it even before being able to report about it on the mailing list. Events are and should be exceptional. An example of such an event is the correction of a syntax error in one of the DocBook source files. Under normal circumstances Editors test the correctness of the entire set (including the modifications) *before* commiting changes to CVS. If for some reason that was not done an event occurs: the person that is responsible for uploading the erroneous version should immediately correct it.

6.3.14. *Grabbers

[* Grabbers are unassigned items/tasks TBW]

6.3.15. *WIP indicators

[* TBW]

6.4. How to be a Proofreader

6.4.1. Your mission

TBW

6.4.2. The procedures you have to know

TBW

6.4.3. What tools do you need?

TBW

6.4.4. Installation of tools

TBW

6.4.5. Configuration of tools

TBW

6.4.6. Testing

TBW

6.5. How to be an Editor

6.5.1. Your mission

TBW

6.5.2. The procedures you have to know

TBW

6.5.3. What tools do you need?

TBW

6.5.4. Installation of tools

TBW

6.5.5. Configuration of tools

TBW

6.5.6. Testing

TBW

6.6. How to be a Translator

6.6.1. Your mission

TBW

6.6.2. The procedures you have to know

TBW

6.6.3. What tools do you need?

TBW

6.6.4. Installation of tools

TBW

6.6.5. Configuration of tools

TBW

6.6.6. Testing

TBW

6.7. How to be an Author

6.7.1. Your mission

TBW

6.7.2. The procedures you have to know

TBW

6.7.3. What tools do you need?

TBW

6.7.4. Installation of tools

TBW

6.7.5. Configuration of tools

TBW

6.7.6. Testing

TBW

6.8. List of project workers

Table 6-1. List of RSBAC Project Members

Handle	Name	E-mail
AOT	Ott, Amon	ao at rsbac dot org
HIY	Hideki, Yamane	henrich at iijmio-mail dot jp
HWK	Kloepping, Heinrich Wilhelm (Henk)	henk at rsbac dot org
JYS	Smith, Joey	joey at joeysmith dot com
PBU	Busser, Peter	peter at trusteddebian dot org
SIE	Ievlev, Stanislav	inger at altlinux dot org

6.9. List of roles of project members

Table 6-2. Roles of the RSBAC Project Members

Role	Started	Ended	Abb.
Project Leader	2000.06.28		HWK

Role	**Started**	**Ended**	**Abb.**
Master Author	2000.06.28		AOT
Author	2000.06.28		HWK
	2000.06.28		AOT
	2000.06.28		SIE
Editor	2000.06.28		HWK
	2000.06.28		SIE
	2000.06.28	2002.06.09	AOT
Proofreader	2003.06.17		PBU
	2003.09.01		JYS
Translator (Japanese)	2003.09.13		HIY

Appendix A. GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the

publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies

to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under

the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Appendix B. How safe do you want to be?

Knowing how to use tools like RSBAC is just one piece of the security puzzle. The first step on the road to a more secure system is awareness. You need to be *aware* of the risks and weigh the benefits and costs of matching countermeasures. Do you really need a more secure system?

You may argue that many people use Linux for a hobby and/or for private use and that very few people would be interested in breaking in on such systems. You may argue that you are off-line most of the times. But nowadays even private users have a lot of valuable data stored on their systems and most of these systems are (regularly) connected to the Internet. For example in the Netherlands it is fairly common to do your banking on-line (over the Internet). The network connection is encrypted but information about your transaktions is stored unencrypted in the browser cache. Your private (e-)mail probably is stored somewhere on your Linux box, as is data about your credit cards and possibly confidential materials belonging to your clients or your employer. Your unprotected Linux system could be a worthwile target for malicious crackers. And for some virtual burglars the incentive is not even to steal your data. A fair number of people likes to break into other peoples systems "because they are there". Many of them are not even mischievous, but they may want to use your system to provide them with extra resources or to mask their identity with yours, for example by using your system to send anonymous e-mail. In that process they may intentionally weaken your security (create backdoors) which may create additional opportunities for real crackers. Accepting these risks can result in substantial damage and costs.

However, countermeasures have a price tag too and often introduce other risks. Your system may be more difficult to administer or may show signs of reduced performance. There may be a (too) steep learning curve involved. Misconfiguration of a protection system may result in a system that in fact is *less* secure than before, with the added risk of a user that feels he is safe and hence is more careless with private data then ever.

In the professional world you will need to weigh the risks first. Often this process of weighting risks results in a formal security policy. The security policy should be communicated and accepted before deciding about solutions, procedures and tools. In a professional environment this process is repeated regularly: am I still safe? Are there new risks? How do I counteract? But even if you are "just" a private user it does not harm to think before you act and adopt (part of) the policies companies use.

This book does not deal with security policies extensively. A handbook that is a guide to setting computer security policies and procedures for sites that have systems on the Internet is available on the Internet itself *(http://www.faqs.org/rfcs/rfc1244.html)* (http://www.faqs.org/rfcs/rfc1244.html). It lists issues and factors that a site must consider when setting security policies. One of the basic approaches listed is:

- look at what you are trying to protect;
- look at what you need to protect it from;
- determine how likely the threats are;
- implement measures which will protect your assets in a cost-effective manner;
- review the process continuously, and improve things every time a weakness is found.

Of course, protecting your systems involves more than protecting them against unwanted access. You also will need to consider environmental disasters (floods, fire, storms, lightning), failing hardware, power brown-outs and black-outs, back-ups and a wealth of other continuity- and availability related issues far beyond the scope of the RSBAC bookset.

RSBAC is just a tool - a very useful one, but nevertheless: just a tool. It was designed to help you prevent your systems from unwanted access, but in that process it may restrict the use of your system in general. You need to be willing to accept the extra administration and the sometimes steep learning curve that comes with using RSBAC. Always ask yourself: "Why do I accept putting up with restrictions - is the risk involved worth my effort?". In practice, many times the answer should be “yes”, but it’s your decision. Make sure it is well-considered.

Bibliography

[Albitz01] John Doe and Wei Ping, *A fake bibliography entry (first edition)* Tinkerman, 2002. ISBN 0-555-55555-4.

The RSBAC programmers reference manual

Amon Ott

Stanislav Ievlev

Heinrich W. Klöpping

The RSBAC programmers reference manual
by Amon Ott, Stanislav Ievlev, and Heinrich W. Klöpping

Audience: This book is intended for use by experienced and skilled Unix professionals that wish to understand and possibly modify the RSBAC source code.

Approach: This book resulted from a project founded on June 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. We aimed at providing a reference manual for C-programmers. Every function used in the source code of *Rule Set Based Access Control* (RSBAC) has been documented in this book. This book will be accompanied by four other books: *The RSBAC cookbook*, *The RSBAC reference manual*, *The RSBAC programmers cookbook* and *The RSBAC introduction*.

To learn where the latest version of this book can be downloaded or read please refer to Section 6.2 in *The RSBAC introduction*.

Sources: Our sources of information were (Open Source) material on the Internet, several books, practical experience of the authors and others and research and programming work done by the authors. We try to give credit where due, but are fallible. We apologize.

Caution

While every precaution was made in the preparation of this book, we can assume no responsibility for errors or omissions. When you feel we have not given you proper credit or feel we may have violated your rights or when you have suggestions how we may improve our work please notify us immediately so we can take corrective actions.

Organization of this book: This book has been organised in four parts:

I. part I

II. part II

III. part III

IV. part IV

This book was written using the DocBook V3.1/SGML documentation standard.

Dedication

And I said, "You can stop if you want with the Z
Because most people stop with the Z - but not me!
In the places that I go there are things that I see
That I never could spell if I stopped with the Z.
I'm telling you this because you're one of my friends
My alphabet starts where your alphabet ends!

You'll be sort of surprised what there is to be found
Once you go beyond Z and start poking around"

—Dr Seuss *On Beyond Zebra*

Table of Contents

Preface

Linux has always been a very stable and trustworthy operating system - even more so in comparison with it’s closed-source alternatives. Driven by closed source vendors’ questionable license policies, security risks, bugs and vendor-lock, more and more IT-managers choose the Linux alternative. Linux also has a good reputation -- as have other Unices -- when it comes to security. That may or may not be due to the blatant lack of proper security in other “operating systems”. However, your data is arguably *not* safe on a standard Linux system. Linux is susceptible to security breaches, malware and programming bugs too.

Hence, a number of workarounds and extensions have been written. One of the most popular (and in my not so humble opinion one of the most elegant and stable ones) is *Rule Set Based Access Control*. I have been working with RSBAC since 1999 and have been impressed by what Amon wrote ever since. However, the lack of a good cookbook for it struck me as one of the major hurdles on the road to its acceptance. For I am convinced that it deserves such broad acceptance given its qualities.

So, we set out to write such a cookbook. And here it is. It still is a work in progress, and unless the nature of security related work suddenly changes probably will be so indefinitely. The authors wanted to create a useful book that would guide you through the seemingly awkward process of understanding, installing and maintaining RSBAC. This first version of our book originated from various materials, amongst them the introduction to RSBAC written by Stanislav Ievlev, the original documentation written by the author of RSBAC, Amon Ott, and a sequel of four articles I wrote for the Dutch Linux Magazine. We had to write many additional chapters from scratch and many hours were spent researching and trying actual security configurations. We finally made it.

It is up to you to judge our efforts, and, hopefully, improve our work. To quote the laudated Dr David A. Lien, author of the excellent (1977) TRS80 Level I BASIC beginners manual: “Sit back, relax, read slowly as though savoring a good novel, and above all, let your imagination wander. I’ll supply you with all the routine facts and techniques you need. The real enjoyment begins when your imagination start the creative juices flowing and a computer becomes a tool in your own hands. You become its master - not the other way around.”.

To the many people we unintentionally forgot to give credit where due, from the bottom of our hearts: **we thank you.**

Henk Klöpping, December 2002

I. Bogus part

Bogus introductory text.

Chapter 1. bogus chapter title

1.1. bogus section title

Some bogus text.

Appendix A. GNU Free Documentation License

GNU Free Documentation License
Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the

publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies

to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under

the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliography

[Albitz01] John Doe and Wei Ping, *A fake bibliography entry (first edition)* Tinkerman, 2002. ISBN 0-555-55555-4.

The RSBAC programmers cookbook

Amon Ott

Stanislav Ievlev

Heinrich W. Klöpping

The RSBAC programmers cookbook
by Amon Ott, Stanislav Ievlev, and Heinrich W. Klöpping

Audience: This book is intended for use by experienced and skilled Unix professionals that wish to understand and possibly modify the RSBAC source code.

Approach: This book resulted from a project founded on June 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. We aimed at providing a *cookbook* for programmers -- a book filled with hands-on instructions and examples of enhancing and modifying the C-sources of *Rule Set Based Access control* (RSBAC). This book will be accompanied by four other books: *The RSBAC cookbook*, *The RSBAC reference manual*, *The RSBAC programmers reference manual* and *The RSBAC introduction*.

To learn where the latest version of this book can be downloaded or read please refer to Section 6.2 in *The RSBAC introduction*.

Sources: Our sources of information were (Open Source) material on the Internet, several books, practical experience of the authors and others and research and programming work done by the authors. We try to give credit where due, but are fallible. We apologize.

Caution

While every precaution was made in the preparation of this book, we can assume no responsibility for errors or omissions. When you feel we have not given you proper credit or feel we may have violated your rights or when you have suggestions how we may improve our work please notify us immediately so we can take corrective actions.

Organization of this book: This book has been organised in four parts:

I. part I

II. part II

III. part III

IV. part IV

This book was written using the DocBook V3.1/SGML documentation standard.

Dedication

> And I said, "You can stop if you want with the Z
> Because most people stop with the Z - but not me!
> In the places that I go there are things that I see
> That I never could spell if I stopped with the Z.
> I'm telling you this because you're one of my friends
> My alphabet starts where your alphabet ends!
>
> You'll be sort of surprised what there is to be found
> Once you go beyond Z and start poking around"
>
> —Dr Seuss *On Beyond Zebra*

Table of Contents

Preface

Linux has always been a very stable and trustworthy operating system - even more so in comparison with it's closed-source alternatives. Driven by closed source vendors' questionable license policies, security risks, bugs and vendor-lock, more and more IT-managers choose the Linux alternative. Linux also has a good reputation -- as have other Unices -- when it comes to security. That may or may not be due to the blatant lack of proper security in other "operating systems". However, your data is arguably *not* safe on a standard Linux system. Linux is susceptible to security breaches, malware and programming bugs too.

Hence, a number of workarounds and extensions have been written. One of the most popular (and in my not so humble opinion one of the most elegant and stable ones) is *Rule Set Based Access Control*. I have been working with RSBAC since 1999 and have been impressed by what Amon wrote ever since. However, the lack of a good cookbook for it struck me as one of the major hurdles on the road to its acceptance. For I am convinced that it deserves such broad acceptance given its qualities.

So, we set out to write such a cookbook. And here it is. It still is a work in progress, and unless the nature of security related work suddenly changes probably will be so indefinitely. The authors wanted to create a useful book that would guide you through the seemingly awkward process of understanding, installing and maintaining RSBAC. This first version of our book originated from various materials, amongst them the introduction to RSBAC written by Stanislav Ievlev, the original documentation written by the author of RSBAC, Amon Ott, and a sequel of four articles I wrote for the Dutch Linux Magazine. We had to write many additional chapters from scratch and many hours were spent researching and trying actual security configurations. We finally made it.

It is up to you to judge our efforts, and, hopefully, improve our work. To quote the laudated Dr David A. Lien, author of the excellent (1977) TRS80 Level I BASIC beginners manual: "Sit back, relax, read slowly as though savoring a good novel, and above all, let your imagination wander. I'll supply you with all the routine facts and techniques you need. The real enjoyment begins when your imagination start the creative juices flowing and a computer becomes a tool in your own hands. You become its master - not the other way around.".

To the many people we unintentionally forgot to give credit where due, from the bottom of our hearts: **we thank you.**

Henk Klöpping, December 2002

I. Bogus part

Bogus introductory text.

Chapter 1. bogus chapter title

1.1. bogus section title

Some bogus text.

Appendix A. GNU Free Documentation License

GNU Free Documentation License
Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the

publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies

to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under

the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliography

[Albitz01] John Doe and Wei Ping, *A fake bibliography entry (first edition)* Tinkerman, 2002. ISBN 0-555-55555-4.

The RSBAC reference manual

Amon Ott

Stanislav Ievlev

Heinrich W. Klöpping

The RSBAC reference manual
by Amon Ott, Stanislav Ievlev, and Heinrich W. Klöpping

Audience: This book is intended for use by experienced and skilled Unix professionals that wish to install, configure and use RSBAC.

Approach: This book resulted from a project founded on June 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. We aimed at providing a reference manual for users of *Rule Set Based Access control* (RSBAC), in which every command or configuration item has been listed and explained. This book will be accompanied by four other books: *The RSBAC cookbook*, *The RSBAC programmers cookbook*, *The RSBAC programmers reference manual* and *The RSBAC introduction*.

To learn where the latest version of this book can be downloaded or read please refer to Section 6.2 in *The RSBAC introduction*.

Sources: Our sources of information were (Open Source) material on the Internet, several books, practical experience of the authors and others and research and programming work done by the authors. We try to give credit where due, but are fallible. We apologize.

Caution

While every precaution was made in the preparation of this book, we can assume no responsibility for errors or omissions. When you feel we have not given you proper credit or feel we may have violated your rights or when you have suggestions how we may improve our work please notify us immediately so we can take corrective actions.

Organization of this book: This book has been organised in four parts:

I. part I - RSBAC theory
II. part II - RSBAC manual pages
III. part III - TBW
IV. part IV - TBW

This book was written using the DocBook V3.1/SGML documentation standard.

Dedication

> And I said, "You can stop if you want with the Z
> Because most people stop with the Z - but not me!
> In the places that I go there are things that I see
> That I never could spell if I stopped with the Z.
> I’m telling you this because you’re one of my friends
> My alphabet starts where your alphabet ends!
>
> You’ll be sort of surprised what there is to be found
> Once you go beyond Z and start poking around"
>
> —Dr Seuss *On Beyond Zebra*

Table of Contents

List of Tables

List of Figures

List of Examples

Preface

Linux has always been a very stable and trustworthy operating system - even more so in comparison with it's closed-source alternatives. Driven by closed source vendors' questionable license policies, security risks, bugs and vendor-lock, more and more IT-managers choose the Linux alternative. Linux also has a good reputation -- as have other Unices -- when it comes to security. That may or may not be due to the blatant lack of proper security in other "operating systems". However, your data is arguably *not* safe on a standard Linux system. Linux is susceptible to security breaches, malware and programming bugs too.

Hence, a number of workarounds and extensions have been written. One of the most popular (and in my not so humble opinion one of the most elegant and stable ones) is *Rule Set Based Access Control*. I have been working with RSBAC since 1999 and have been impressed by what Amon wrote ever since. However, the lack of a good cookbook for it struck me as one of the major hurdles on the road to its acceptance. For I am convinced that it deserves such broad acceptance given its qualities.

So, we set out to write such a cookbook. And here it is. It still is a work in progress, and unless the nature of security related work suddenly changes probably will be so indefinitely. The authors wanted to create a useful book that would guide you through the seemingly awkward process of understanding, installing and maintaining RSBAC. This first version of our book originated from various materials, amongst them the introduction to RSBAC written by Stanislav Ievlev, the original documentation written by the author of RSBAC, Amon Ott, and a sequel of four articles I wrote for the Dutch Linux Magazine. We had to write many additional chapters from scratch and many hours were spent researching and trying actual security configurations. We finally made it.

It is up to you to judge our efforts, and, hopefully, improve our work. To quote the laudated Dr David A. Lien, author of the excellent (1977) TRS80 Level I BASIC beginners manual: "Sit back, relax, read slowly as though savoring a good novel, and above all, let your imagination wander. I'll supply you with all the routine facts and techniques you need. The real enjoyment begins when your imagination start the creative juices flowing and a computer becomes a tool in your own hands. You become its master - not the other way around.".

To the many people we unintentionally forgot to give credit where due, from the bottom of our hearts: **we thank you.**

Henk Klöpping, December 2002

I. RSBAC theory

The first part of this book describes the theory behind RSBAC.

Chapter 1. Introduction

1.1. Introduction

TBW

Chapter 2. RSBAC Models

2.1. Mandatory Access Control (MAC)

2.1.1. Bell-La Padula

2.1.1.1. Definitions

The Bell and La Padula Model describes access by active entities, called subjects, to passive entities, called objects. One entity can, depending on type of access, be in both roles.

From the distinction between read and write access four modes of access can be distinguished: neither read nor write (execute, e), read only (read, r), write only (append, a) and read-write (write, w). The set of all access types is named *A*.

2.1.1.2. System States

Each current access of a subject S[i] to an object O[j] in mode x is treated as a triple (S[i], O[j], x). All these triples together form the set of current accesses b.

Objects are structured according to the Father-Son-Principle and build a hierarchy *H* of one or more hierarchically ordered, independent trees.

All authorized accesses of all subjects to all objects are held in the matrix *M*. Each cell *M[i,j]* of *M* thus contains a subset of *A* with authorized accesses of S[i] to O[j].

A security level is a pair (`Security Classification`, `Set of Categories`). A security classification is a value out of a hierarchy, e.g. `public`, `confidential`, `secret`, `top secret`. A category is a formal assignment to a work area.

One entity with security level (S[1],C[1]) dominates another entity with (S[2],C[2]), if S[1]>=S[2] and C[1] is a superset of C[2]. The property *dominates* over all entities builds a partial order *D*.

The assignment of security levels to subjects and objects, the classification function *F*, is a triple (f[S], f[O], f[C]) of security level assignment functions. `f[S](S[i])` is the maximum security level of subject S[i], `f[O](O[j])` the security level of object O[j] and `f[C](S[i])` the current security level of subject S[i]. Thus for subjects `maximum` and `current` security level are distinguished.

For all S[i] f[S](S[i]) must always dominate f[C](S[i]).

A state z of the model is a tuple (b, *M*, *F*, *H*). A system is a sequence of (request, decision, next state) with initial state z[0].

2.1.1.3. Security Properties

The first property to be maintained is the simple security property (no read-up). This property states that a subject S[i] may have read access to an object O[j] ((S[i],O[j],r) or (S[i],O[j],w) is a current access), if S[i] dominates O[j].

Figure 2-1. Illegal information flow in Bell-La Padula model.

To prevent copying of an object to a lower security level by a malicious subject, the **-property* (no-write-down) must be maintained. This property states: If a subject S[i] has current read access to an object O[1] and current write access to an object O[2], then O[1] must be dominated by O[2] (O[1] has a lower security level than O[2]). Thus information flow is restricted to upwards.

As a strict accordance to the *-property would significantly reduce the usability of the system, some subjects can be marked as trusted subjects without *-property restriction.

Access control by the matrix *M* of authorized accesses by subjects to objects is called *discretionary access control*, its security property is called *ds-property*. The current access must always be in the set of authorized access in its matrix cell.

All properties and security levels must be mandatorily enforced by the system.Every property is added to the other ones and can never reduce system security. A state that fulfils all properties is called secure.

2.1.1.4. Decision Rules

The three properties lead to the following rules for access control decisions. A current access (S[i],O[j],x) is only granted, if the following conditions are met:

1. *ss-property*: S[i] dominates O[j], if x = r or x = w (x contains read access).
2. **-property*: S[i] is trusted or
    a. O[j] dominates current level of S[i], if mode = a
    b. level of O[j] is equal to current level of S[i], if mode = w
    c. current level of S[i] dominates level of O[j], if mode = r
3. *ds-property*: x is in cell *M[i,j]* of matrix *M* of authorized accesses

2.1.1.5. Functions

To use the defined security system state transition functions are necessary. These functions must provably change a secure state to another secure state, according to the decision rules. So induction can be used to proof every reachable state to be secure.

A complete set of those functions can be as follows:

- Change set b of current accesses:
  - get-access(): add triple to set b
  - release-access(): remove triple from b
- Change matrix *M* of authorized accesses:
  - give-access-permission(): add access mode to a cell of *M*
  - rescind-access-permission(): remove access mode from a cell of *M*
- Change classification function *F*:
  - change-object-level()
  - change-current-level()
- Change object hierarchy *H*:
  - create-object(): create leaf object
  - delete-object-group(): remove object with all subobjects from tree

2.1.1.6. Evaluation

The Bell-La Padula model only treats confidentiality aspects. Integrity, availability and privacy of data are not protected. E.g., a subject on lowest security level can delete all data in all its categories, if it is not discretionally protected. Attacks like this can also happen without the user’s knowledge, just think of malware or mistakes. Especially discretionary access control is liable to be attacked by malware.

The concept of trusted subjects which can only be implemented as users or user processes leads to further possibilities of attack by use of high level user accounts.

This model should only be used without additional protection, if confidentiality is the only issue or if data can be easily restored.

2.1.2. Unix System V/MLS

The model of mandatory access control used in RSBAC is mostly the same as in Unix System V/MLS, Version 1.2.1. This operating system was developed in 1989 by the National Computer Security Center of the USA with classification B1/TCSEC.

Unix System V/MLS implements the Bell-La Padula model with some smaller changes, e.g. the *ds-property* is replaced by `Unix style access control`. Security levels with classification categories are maintained, *simple-security-property* and **-property* are enforced. In contrary to `Bell-La Padula` writing is only permitted on the same level.

Bell-La Padula defined four modes of access:

- execute, `E`
- read, `R`
- write, `W`
- append,, `A`

This Unix system adds ten more modes, which mostly cover parts of the above modes:

- search in directory, `S`
- overwrite, `O`
- create, `C`
- link, `L`
- unlink (delete), `U`
- read file/i-node status, `St`
- change status, `Ch`
- send signal/kill, `K`
- read IPC,, `Ripc`
- write IPC, `Wipc`

Subjects are processes, which inherit their owner's security level. Four types of objects are defined:

- File
- Directory
- Interprocess Communication Channel (IPC-Channel)
- System Control Data (SCD)

Table 2-1. Request Condition for access

Request	Condition for access
R/S/E	S >= O
W(O/A)	S = O
C/L/U	S = Od
St	S >= O
Ch	S = O

Request	**Condition for access**
Ripc	S >= O
Wipc	S = O
K	S = O

Table 2-1> shows a summary of access control conditions. S means subject, O object, Od directory object and >= and = stand or *dominates* and *has same level*. Read and write on directories mean access to entries, open is not possible.

2.1.3. The *RSBAC MAC* implementation

The Unix System V/MLS model has been changed to fit into the *RSBAC* access request scheme, which knows more than 30 types of access. Also, write-up is implemented in the original way, so that you can always write to all higher levels.

> **Note:** From version 1.1.1 onwards, writing is only allowed on the same level.

Since administration depends on the role security officer, role based functions had to be added. These restrict all changes to the classification of subjects and objects and role assignments (setting of *MAC* attributes) to security officers.

The security_level attributes used in RSBAC are what is usually called security classifications. Categories, limited to a number of 64 for efficiency reasons, have been added in RSBAC 1.0.8.

> **Note:** From 1.0.9b, the number of security_levels has been increased to 253 (0-252, 8 Bit minus 3 special values).

The current security level (classification) and the current category set of a process are automagically adjusted as needed, if the mac_auto flag is on, which is the default value. However, mac_auto is turned off as soon as the process actively sets its current level or category set.

**-property* enforcement is done with upper and lower bounds, called min_write and max_read. These values are reset only on execution of another program, not at process forking/cloning time or closing of files, because only new execution empties the process memory space.

> **Note:** Please note that until version 1.1.0, all write accesses, e.g. creating a file in a DIR (CREATE on DIR target), lead to the min_write boundary to be adjusted. This can lead to very limited access. Therefore, from version 1.1.1 onwards, the once-only write accesses `CREATE` and `DELETE` do not adjust the min_write boundary, while `MOUNT`, `APPEND_OPEN`,`READ_WRITE_OPEN`, `WRITE_OPEN` and `TRACE` still do.

Devices are treated similar to files with security levels and categories, and all properties are enforced. However, these checks can be turned off (attribute mac_check), because the system might become unusable otherwise.

The MAC file/dir attributes security_level and mac_categories can be inherited from the parent dir. For the security level the value to indicate inheritance from parent is 5 (4 is used internally), for categories it is the empty set (all bits 0). From version 1.0.9b, the security_level special values have been raised by 249, now being 254 and 253.

Stanislav Ievlev and me added a *MAC* option called *MAC-Light* to make the *MAC* module easier to use. Changes are:

1. File/Dir/Fifo object creation is always granted
2. Every user may mount, if levels are sufficient (used to be limited to system administrators)

The *MAC* model should be used, if you need a conceptually proven model for confidentiality. However, it is quite difficult to use in a typical Linux environment.

2.2. Functional Control (*FC*)

The role based model of functional control assigns one role to each user, e.g. general user, security officer or system administrator. Every object gets a category, e.g. general, security or system object.

The security officer states which roles are compatible with which object categories, or in other words, users in which roles can access objects in which categories. The security system enforces these settings.

The file/dir/fifo attribute object_category can be inherited from the parent dir.

The functional control model can in the simple version that is implemented in RSBAC only protect system data and security relevant data, but it already enforces separation of duties between the two special roles. An extension to more and more flexible roles could build this to a strong model. Without distinction between different access modes this model should only be used as part of a combined system.

FC can be easily expressed with RC model, so it is kind of obsolete. The RC default settings are very similar to this model.

You should use this model only to get experience as a base for other, more powerful models.

2.3. Security Information Modification (SIM)

This role based model protects data of type security information. Only users with role security officer get write access to those objects.

The file/dir attribute data_type can be inherited from the parent dir.

Like the functional control this model should only be used in combination with others. Otherwise the security relevant information can still be protected against tampering by system administrators, what is more than Unix style access control can.

SIM can be easily expressed with RC model, so it is kind of obsolete. You should use this model only to get experience as a base for other, more powerful models.

2.4. Simone Fischer-Huebner's Privacy Model (*PM*)

This model was presented on the 17th National Computer Security Conference in Baltimore, USA, in 1994 by its developer Simone Fischer-Huebner. It follows the rules of the Federal German Privacy Law and the EU directive on privacy.

The model and its implementation in RSBAC are described in detail in our paper for "NISS 98 Conference".

The model focus lies on privacy. Confidentiality, integrity and availability are maintained for personal data and transaction procedures by the definition of necessary accesses.

System control data like general settings or authentification information can only be protected by declaring them as personal data. If this is not possible for some data, they cannot be protected.

This model should be used for storage and processing of personal data. To protect system data without the administration overhead of treating them as personal data, another model, e.g. FC, SIM, RC or ACL, should be used.

Use *PM* model, if you want to process personal data and need adequate protection.

2.5. Malware Scan (*MS*)

This is not really an access control model, but rather a system protection model against malware. Execution, reading and transmission of malware infected files can be prevented.

This model should be used, if data, especially programs, are transferred from other systems to prevent a widespread malware infection. However, only malware known by the scanner algorithm can be be detected. On

Linux this is the bliss virus in variants A and B and a few DOS viruses. Platform independent macro or java viruses will have to be included later.

Currently, this is only a working demonstration model, because too few viruses are detected. However, it is rather simple to add more scan strings, if you want to. From version 1.2.0, there is a generic interface to add other scanning engines.

For more details see our paper on “Approaches to Integrated Malware Detection and Avoidance” for The Third Nordic Workshop on Secure IT Systems (Nordsec’98)

2.6. File Flags (*FF*)

This model defines some access flags for files, fifos, symlinks and dirs. Currently, the following flags are supported:

Table 2-2. FF rights

Flag	Checked for	Notes
execute_only	FILE, FIFO, SYMLINK	
search_only	DIR	
read_only	FILE, FIFO, SYMLINK, DIR	
write_only	FILE, FIFO, SYMLINK	
secure_delete	FILE	File is blanked on delete and truncate (ext2, ext3, msdos/vfat, minix only)
no_execute	FILE	
no_delete_or_rename	FILE, FIFO, SYMLINK, DIR	new in 1.1.1, not inherited
append_only	FILE, FIFO, SYMLINK	new in 1.1.2, write accesses are limited to APPEND_OPEN and WRITE, read accesses are allowed
add_inherited	FILE, FIFO, SYMLINK, DIR	not inherited

These flags are checked on every access to the given target types. Only users in system_role `security officer` can change the flags.

The `add_inherited` flag is special: If set, the parent dir’s flags are added (or’d) to the target’s own flags. Inheritance is on by default.

Warning

The flags `no_delete_or_rename` and `add_inherited` cannot be inherited, they must always be set explicitely!

Please note that the attributes are independent from each other and restrictive: All attributes that are set are applied, e.g. `execute_only` and `no_execute` together (or `read_only` and `write_only`) lead to no access.

Flags that are only checked for some target types are ignored for the other ones. This can be used to set e.g. `search_only` and `execute_only` on a dir - you can SEARCH (not READ!) in the dir and EXECUTE files in it, but nothing else.

Example 2-1. Preventing a group of files against reading

Set `write_only` on a logging dir. All log files created in that dir inherit the `write_only` flag, thus the log can never be read unless the flag is removed

Example 2-2. Create a group of append-only files

Set `append_only` on a logging dir. All log files created in that dir inherit the `append_only` flag, thus the log can be read, but writing can only append to the file, unless the flag is removed. Add flag `write_only`, if the files should not be read either.

Example 2-3. Preventing execution of files from a directory

Set `no_execute` on /home. All executables below that dir inherit this flag, thus no user can execute files from her home directory, unless the flag is removed.

Example 2-4. Prevent moving of a directory

Set `no_delete_or_rename` on `/home`. User home dirs below can be added, removed and individually protected, but the parent dir `/home` cannot be moved or replaced to fake other home dirs for most users.

File Flags should be used, if you need global access settings which are valid for all users.

2.7. Role Compatibility (*RC*)

2.7.1. Role Compatibility (*RC*) - Until v1.0.9-pre3

> **Warning**
>
> This is the description of a rather old RC version. Please go to the new description for recent RSBAC versions!

This is a powerful and flexible role based model. It defines 64 `RC types` for each target type (File/Dir, Device, Process, Inter Process Communication, System Control Data) and 64 `RC roles` to access them. The RC module has been added in *RSBAC* version 1.0.8.

Please also see the RC model description in the "RC-Paper".

2.7.1.1. Role and Type Entries

Each `role entry` has the following fields:

Table 2-3. RC Role entry fields

Role Entry Field	Type/Values	Description
name	string of 15 chars	Name of role
role_comp	64 bit vector	Compatible roles (1 Bit per role) = roles process may change to without chown (setuid)
type_comp_fd	64 bit vector	Compatible file/dir types (1 Bit per type)
type_comp_dev	64 bit vector	Compatible device types
type_comp_process	64 bit vector	Compatible process types
type_comp_ipc	64 bit vector	Compatible IPC types
type_comp_scd	64 bit vector	Compatible SCD types
admin_type	none, system or role admin	Role for RC role/type administration
def_fd_create_type	File/dir type number or special value	Type of new files/dirs
def_process_create_type	Process type number or special value	Type of new (forked/cloned) processes
def_process_chown_type	Process type number or special value	Type of process after a chown (setuid)
def_process_execute_type	Process type number or special value	Type of process after execute (start of a new program)
def_ipc_create_type	IPC type number or special flag	Type of new IPC channels

The `admin_type` entry denotes the *RC administration rights*: none = none, system admin = read-only, role admin = full. They do not give any other access rights, this is done with compatibility settings only.

Type entries have name fields (15 char strings). Only file/dir type entries have an additional boolean value `type_fd_need_secdel`, which indicates a need of secure deletion/truncation of files of this type.

SCD types are fixed and represent one area of accessible system data each. They are also used for administration rights, like adding modules or setting system time. SCD compatibility means accessability of the SCD facility.

2.7.1.2. RC attributes

Each target apart from user targets gets a `rc_type attribute` to indicate its type. For files and dirs this field can hold the special value `type_inherit_from_parent` to signal inheritance of the attribute.

User entries get an `rc_def_role` attribute, which is used to determine the process's initial role after each CHOWN (setuid).

Process entries also have an rc_role for the current role.

File/Dir entries have a field `rc_force_role` to specify a forced role, if this file is executed. This mechanism works similar to the setuid or setgid field in Unix file systems. forced role can also have one of the special values.

2.7.1.3. Special values

The special values mentioned above are as follows:

Table 2-4. RC Role special values

Role Special Value	Meaning
role_inherit_user	use user's (process owner's) rc_def_role
role_inherit_process	use current rc_role of process (keep role)
role_inherit_parent	use current rc_role of parent dir or process

Table 2-5. RC Type special values

Type Special Value	Meaning
type_inherit_process	use current rc_type of process (keep role)
type_inherit_parent	use current rc_type of parent dir or process
type_no_create	creation of process/IPC is not allowed
type_no_execute	execution of other programs is not allowed
type_use_new_role_def_create	for process chown (setuid): use `def_process_create_type` of the new role

2.7.1.4. Initial Configuration

hen started without role definitions, three pre-defined roles are set up: *General User* (0), *Role Admin* (1) and *System Admin* (2). The pre-defined role settings are derived from the hard-wired settings in the *FC* module.

When started without type definitions, one type per target is set up. This is the default type *General*, which is also used as default value for all type attributes.

As usual in *RSBAC*, user root (0) has `rc_def_role` 2 and user 400 has `rc_def_role` 1 as predefined value in the default useraci contained in the admin tool package.

> **Note:** The pre-defined roles are normal roles designed to get you going. They can be changed like all other roles! You may easily lock yourself out, if you change them without knowing exactly what you are doing.
>
> Still, the maintenance mode will allow you to modify roles, if you turned maintenance support for RC policy data on at kernel configuration.

To be secure, test your configurations with different role numbers and use `rc_copy_role` to copy them, if necessary.

2.7.2. Role Compatibility (*RC*) - From v1.0.9-pre4 onwards

The older RC model has been much changed. Until v1.1.2, it defines 64 RC types for each target type (File/Dir, Device, Process, Inter Process Communication, System Control Data) and up to 64 RC roles to access them. The original RC module has been added in RSBAC version 1.0.8. From v1.2.0, the number of roles and types is only limited by their 32 bit interger index number.

RC is a role based model: Every user has a default role, which is inherited by all his/her processes. Based on the current role, access to objects of certain types is granted or denied. The role can be changed by a change of the process owner, by the process via system call (only "compatible" roles allowed) or by execution of a specially marked executable (using initial_role or force_role, need not be "compatible").

Creation of new objects is a special case in every access control model. Here, every role has entries for the types of new objects, as well as entries for type setting behaviour on execution and change of process owner. Additionally, all these entries have a "no_create" special value, which disallows such requests.

Please also see the RC model description in the (rather old) "RC-Paper".

2.7.2.1. Role and Type Entries

Up to v1.1.2, each role entry has the following fields:

Table 2-6. RC Role entry fields

Role Entry Field	Type/Values	Description
name	string of 15 chars	Name of role
role_comp	64 bit vector	Compatible roles (1 Bit per role) = roles process may change to without chown (setuid)
type_comp_fd	Array of 64 vectors of 64 bit each (64x64 Bit Matrix)	Compatible file/dir types by request types (1 Bit per type and request) = True/False value, which type may be accessed with which request

Role Entry Field	Type/Values	Description
type_comp_dev	Array of 64 vectors of 64 bit each (64x64 Bit Matrix)	Compatible device types by request types (1 Bit per type and request)
type_comp_process	Array of 64 vectors of 64 bit each (64x64 Bit Matrix)	Compatible process types by request types (1 Bit per type and request)
type_comp_ipc	Array of 64 vectors of 64 bit each (64x64 Bit Matrix)	Compatible IPC types by request types (1 Bit per type and request)
type_comp_scd	Array of 64 vectors of 64 bit each (64x64 Bit Matrix)	Compatible SCD types by request types (1 Bit per type and request)
admin_type	none, system or role admin	Role for RC role/type administration
def_fd_create_type	File/dir type number or special value	Type of new files/dirs (use no_create to disallow creation, needs CREATE right for chosen type)
def_process_create_type	Process type number or special value	Type of new (forked/cloned) processes (use no_create to disallow creation, eeds CREATE right for chosen type)
def_process_chown_type	Process type number or special value	Type of process after a chown (setuid) (use no_chown to disallow creation)
def_process_execute_type	Process type number or special value	Type of process after execute (start of a new program) (use no_execute to disallow creation)
def_ipc_create_type	IPC type number or special flag	Type of new IPC channels (use no_create to disallow creation, needs CREATE right for chosen type)

From v1.2.0, all compatibility and admin/assign role settings are placed into separate list, but the other entries are unchanged. Also, NETDEV, NETOBJ and NETTEMP type compatibilities have been added to support the new network targets.

The admin_type entry denotes the global RC administration rights for roles, types and RC specific attributes: none = no access, system admin = read-only, role admin = full. They do not give any object access rights, this is done with compatibility settings only.

Type entries have name fields (15 char strings). Only file/dir type entries have an additional boolean value type_fd_need_secdel, which indicates a need of secure deletion/truncation of files of this type.

SCD types are fixed and represent one area of accessible system data each. They are also used for administration rights, currently only auth_administration. SCD compatibility means accessability of the SCD facility. Additionally, the special SCD target ’other’ is used to control requests with target type NONE.

2.7.2.2. RC attributes

Each target apart from user targets gets a rc_type attribute to indicate its type. For files and dirs this field can hold the special value type_inherit_from parent to signal inheritance of the attribute.

User entries get an rc_def_role attribute, which is used to determine the process's initial role after each CHOWN (setuid).

Process entries also have an rc_role for the current role and a field rc_force_role to keep the executed program's rc_force_role value.

File/Dir entries have a field rc_force_role to specify a forced role, if this file is executed. This mechanism works similar to the setuid or setgid field in Unix file systems. The forced role can also have one of the special values (see below). The forced role value is copied into the process attributes for further use on CHOWN requests.

From version 1.1.2, FD entries also have a field rc_initial_role. This setting determines, which role will be used directly after start of execution. At the next setuid (CHANGE_OWNER on this process), it will be replaced by the value in rc_force_role.

rc_initial_role can also have the special (and default) value role_use_force_role, in which case the value from rc_force_role is also taken as initial value. This is the same behaviour as before this field was added.

For NETTEMP targets, there are two RC type entries: rc_type and rc_type_nt. rc_type is inherited to the NETOBJ targets covered by this template, rc_type_nt is used for access to the template itself.

2.7.2.3. Special Values

The special values mentioned above are as follows:

Table 2-7. RC Role special values

Role Special Value	Meaning
role_inherit_user	use user's (process owner's) rc_def_role on CHOWN and EXECUTE (default forced role until 1.0.9a-pre2)
role_inherit_process	keep current rc_role of process on CHOWN and EXECUTE
role_inherit_parent	inherit value from parent object, e.g. parent DIR
role_inherit_up_mixed	keep current rc_role of process on EXECUTE, but use new process owner's rc_def_role on CHOWN (default forced role from 1.0.9a-pre3)
role_use_force_role	valid in rc_initial_role only, default value. Use the value from rc_force_role.

Table 2-8. RC Type special values

Type Special Value	Meaning
type_inherit_process	use current rc_type of process (keep role)
type_inherit_parent	use current rc_type of parent dir or process
type_no_create	creation of process/file/dir/IPC is not allowed
type_no_execute	execution of other programs is not allowed

Type Special Value	Meaning
type_no_chown	change owner of process is not allowed
type_use_new_role_def_create	for process chown (setuid): use `def_process_create_type` of the new role

2.7.2.4. Initial Configuration

From version 1.2.1, the following behaviour can be turned off by the global kernel boot parameter rsbac_no_defaults, which is useful for complete restore of a previous configuration from a backup script.

When started without role definitions, three pre-defined roles are set up: *General User* (0), *Role Admin* (1) and *System Admin* (2). The pre-defined role settings are derived from the hard-wired settings in the *FC* module.

When started without type definitions, three types per target are set up. These are the default type *General* (0), which is also used as default value for all type attributes, the type *Security* (1) and the type *System* (2). Only *General* is actually used in the default setting, but example compatibilities are set up for all types.

As usual in *RSBAC*, user root (0) has rc_def_role 2 and user 400 has rc_def_role 1 as predefined value in the default user ACI settings.

> **Note:** The pre-defined roles are normal roles designed to get you going. They can be changed like all other roles! You may easily lock yourself out, if you change them without knowing exactly what you are doing. It is mostly the best choice to copy a role first and modify the new one.
>
> Still, maintenance and soft mode will allow you to modify roles, if you turned maintenance or soft mode support for *RC* policy on at kernel configuration.

To be safe, test your configurations with different role numbers and use rc_copy_role to copy them, if necessary.

To follow the least-privilege rule, you can make the default user role 0 a dummy and set the role for every user explicitely. This way nobody gains anything by adding new users

2.7.2.5. Time limits (from 1.2.0 onwards)

From version 1.2.0, you can set time-to-live values for compatible admin and assign roles as well as for all type compatibility settings. After the given time, the entry is deleted and further access is denied.

You can set ttl values with parameters `-t`, `-T` and `-D` in rc_set_item admin tool.

> **Warning**
>
> All ttl settings depend on the correct system time. You should take special care that it is always correct, if you are using this feature!

2.7.2.6. Advanced Administration with separation of duties (from 1.0.9a-pre2 onwards)

Newer RC versions contain a sophisticated model of separation of duty. For this, the following additions have been made:

- New role vector admin_roles: Which roles a user in this role may administrate. Several role settings are further restricted by other rights, e.g. `role_comp` and `type_comp_xx`.
- New role vector assign_roles: Which roles a user in this role may read and assign to users and processes (process only, if MODIFY_ATTRIBUTE is allowed), and which compatible roles she may assign to administrated roles (if assign_roles contains all roles involved).
- Further restriction on changing user/process role: the old role must also be contained in your assign_roles vector. This way, a partial role assigner must always stay within a limited set of roles, and cannot affect users and processes in other roles.
- These new vectors may only be changed by old style Role Admins. If you set them at the beginning, and then remove all Role Admins, this separation is forever fixed (well, unless booting Maint kernel or switching RC off).
- New type access rights:

  `ADMIN`

  : Set/delete name, set need_overwrite for FD types

  `ASSIGN`

  : Assign this type to objects. You also need `MODIFY_ATTRIBUTE` on the old object type.

  `ACCESS_CONTROL`

  : Change normal (old) access rights to this type for your administrated roles

  `SUPERVISOR`

  : Change these new special rights to this type for your administrated roles

  If no role has SUPERVISOR right to a type, the separation is forever fixed (again unless booting Maint kernel, or someone still has old style admin-type Role-Admin)

- Old roles and types (from older RC version) are automatically updated on the first boot of the new version (except from 1.1.2 to 1.2.0). On update, Role Admins simply get everything new fully allowed. Caution: If you copy such an updated role, it also gets everything set for all roles and types!

  System Admins get assign right for their own role only, which means they are allowed to read their own role settings, but not to change anything.

  So you could reboot with new version, reset old admin_type to none for all roles and thus get your current administration settings fixed.

- The old style admin_type Role_Admin still works as before, to keep things simple for normal use.

2.7.2.7. When to use RC model

This model should be used whenever subjects and objects can be easily grouped into roles and types, if you need a strong separation of duty, or if you need access control based on programs, not only on users. The role and type abstraction makes administration much easier than individual settings!

2.8. Authentification Module (*AUTH*)

2.8.1. Basics

This module can be seen as a support module for all others. It restricts `CHANGE_OWNER` on process targets (setuid) for a process: the request is only granted, if the process has either the `auth_may_setuid` flag set or the target user ID is in its capability set. The `auth_may_setuid` flag and the capability set are inherited on execute from the program file.

In other words: No setuid() variant system call will be allowed for any process running any program, unless the process (e.g. inherited from the program file) has either the flag auth_may_setuid set or an *AUTH* capability for the target uid.

A program file's capabilities can be set, if all modules grant a `MODIFY_ATTRIBUTE` request for `A_auth_add_f_cap` or A_auth_remove_f_cap. Process capabilities can only be added by other processes that have the `auth_may_set_cap` flag set, which is also inherited from their executed file.

This way an enforcement of daemon based authentification is possible, as well as a restriction of system daemons to a set of user IDs.

> **Warning**
>
> If enabled without a login program having auth_may_setuid or a capability set and without a capability setting daemon, you will not be able to login to your system! Use kernel parameter *rsbac_auth_enable_login* in emergencies or at the first boot to set auth_may_setuid for `/bin/login`.

With *AUTH* enabled, system daemons which setuid to a normal user account, like `portmap`, `mysql` or whatever, need an explicit capability to do so. They will only be able to use uids you granted.

2.8.2. *AUTH* attributes

All Processes have two auth flags: `auth_may_setuid` and `auth_may_set_cap` with the above meanings. These are inherited by all child processes.

All files have the same two flags, which replace those of the process on execute of the file.

Files and processes also have capability sets with the same inheritance rules. Adding to and removing from capability sets is restricted by different access control schemes for processes and files: process caps can be set by any process, which has the auth_may_set_flag set, whoever may be the owner of the process. File flags are set on behalf of users by any program.

2.8.3. Special Values

Up to version 1.0.9a, file capability sets can have the special value 65533 (UID -3), which is replaced by the owner of the process at the time the set is copied (execute time). Thus the process can come back to the original user ID after changing it. This value has been changed to 4294967292 (-3 again) in version 1.0.9b, which supports 32 Bit user ids.

2.8.4. Initial Configuration

Initially, nothing is allowed. Thus you have to enable the login of an *AUTH* security officer (uid 400 by default), who can do everything necessary. This can also be done with a maintenance kernel, setting appropiate values for all program files involved.

As a shortcut, you can use the kernel parameter *rsbac_auth_enable_login*, which set `auth_may_setuid` (full rights to change process owner) for `/bin/login`. This behaviour is hardcoded in the *AUTH* module data structures.

2.8.5. Administration

AUTH only restricts its administration, if this is enabled in the kernel configuration Administration is limited to users with system_role security_officer, and all involved attributes are protected.

Every module can further restrict the AUTH administration, if it depends on proper authentification. See the configuration menu help pages for more info.

Administration is mostly based on the file and process attributes and the capability sets mentioned above. See “instadm.htm” for more details.

2.8.6. When to use AUTH model

This model should always be used, because it controls which users can work on the system.All other models depend on proper user identification, which can be enforced with AUTH model.

2.9. Access Control Lists Module (*ACL*)

2.9.1. Basics

Access Control Lists specify, which subject (user, RC role or ACL group) may access which object (of an object type) with which requests (usual RSBAC requests and ACL specials).

If there is no ACL entry for a subject at an object, the rights to the parent object are inherited. Inheritance can be restricted by inheritance masks.

On top of the inheritance hierarchy there is a default *ACL* for each object type (`FILE`, `DIR`, ...).

To change the ACL of an object, you need the special right Access Control for this object. The special right *Forward* allows to give somebody else the rights you have, but you cannot revoke them afterwards.

Special right *Supervisor* includes all other rights, can never be masked out (unless allowed in kernel configuration) and can only be set by users who already have it. This right is set for user 400 in those default ACLs, which cannot be successfully read from disk at boot time, e.g. because of new installation.

All object types are supported. IPC, USER and PROCESS objects only have the common default ACL, which is always inherited to all objects of this type - these objects are too short-lived to administrate useful individual ACLs.

As of version 1.0.9a, there are an unchangeable ACL group Everyone (group 0), which by definition contains all possible users, as well as user defined groups.

Group management allows every user to define global and private groups without restriction. Global groups can be used by every user, private ones only by the group owner. Also, the group owner is the only one allowed to add or remove group members or to change the group settings name, owner and type. Since the owner can be changed, groups are transferable (thus making them possibly unaccessible for the original owner). This feature might become optional in future releases, because it can be used for attacks.

Group rights are added to user and role rights. As there is no global group administrator, very user can do the job of maintaining a sensible group structure, e.g. user secoff.

Just to mention: Similarities to other PC network systems may not be accidential... ;-)

2.9.2. Time limits (from 1.2.0 onwards)

From version 1.2.0, you can set time-to-live values for all ACL entries and group memberships. After the given time, the entry is deleted and further access rights change according to the remaining settings.

You can set ttl values with parameters `-t`, `-T` and `-D` in the ACL admin tools or through `rsbac_acl_menu` and `rsbac_acl_group_menu`.

> **Warning**
>
> All ttl settings depend on the correct system time. You should take special care that it is always correct, if you are using this feature!

2.9.3. When to use ACL model

This model should be used whenever you have individual subjects and objects, which cannot easily be grouped into roles and types for RC model, or if you need strong (possibly iscretionary) access control with individual user groups. However, individual ACLs can be confusing. Try to use time-to-live settings for all temporary changes.

2.10. Linux Capabilities (*CAP*)

2.10.1. Basics

In Linux kernels, all root special rights are grouped in so-called *Posix Capabilities*, e.g for network administration or full file access. The *RSBAC CAP* module allows to define a minimum and maximum capability set for both users and programs. Program settings override user settings, and minimum settings override maximum settings.

This module can be used to:

- restrict rights of programs run by root
- add root rights to normal users or programs run by them

in the standard Linux way. It is thus the only *RSBAC* module which directly interferes with existing Linux access control.

2.10.2. When to use CAP model

This model should be used whenever you have to do something which is usually forbidden by standard Linux access control, or if you have to run a root daemon, but want to restrict its rights in the rough Posix capability scheme.

CAP is specially useful to give *RSBAC* administrators, who are not root, read access to all directories so that they can administrate there despite insufficient Linux access modes.

Note: If you only want to partially disable Linux access control for filesystem objects for all users, you might consider to use the generic RSBAC functionality provided for this purpose through the “Allow disabling of Linux filesystem access control” kernel config option.

2.11. JAIL

2.11.1. Basics

All Linux kernels provide the chroot system call to confine a process in a subdirectory. However, it still has all usual privileges, specially when run as root. Also, there are several ways to break out if the chroot environment.

The JAIL module provides a new call rsbac_jail, which makes a chroot call (with chdir("/")) and adds further restrictions on the calling process and all subprocesses. Some of these restrictions can be turned off by flags to the syscall or the rsbac_jail command line wrapper, these are marked with an * in the following list. The rsbac_jail system call also takes the allowed IP-Address for binding (may be 0.0.0.0 for any) as parameter.

Processes in a jail may not:

- Add or remove kernel modules.
- Shutdown or reboot the system.
- Mount or umount filesystems.
- Create sockets of other types than UNIX and INET (IPv4).
- Use other INET (IPv4) addresses than given (optionally, the ANY address 0.0.0.0 can be silently changed to the given address).
- Create INET raw sockets.
- Access IPC objects outside this jail.
- Create device special files (to prevent unwanted device accesses).
- Signal, trace or get status from processes outside this jail.
- Change Linux file modes to include suid or sgid flags.
- Set rlimits.
- Modify settings of any non-rlimit SCD or NETDEV target.
- Access RSBAC attributes.
- Access RSBAC Network Templates.
- Switch off Linux DAC.
- Switch RSBAC modules, softmode or log settings.

The *JAIL* module provides a superset of the *FreeBSD* jail functionality (except individual kernel level hostnames).

All processes in jails are listed in /proc/rsbac-info/jails, if *RSBAC* proc support has been enabled.

2.11.2. When to use *JAIL* module

Use this module for simple service encapsulation, where chroot environments are applicable, but insufficient. Please do not forget that objects within the environment must be protected separately, e.g. with FF flag read_only. As usual, the JAIL module only places further restrictions, so all other modules can be used.

Chapter 3. RSBAC Targets and requests

3.1. Targets

RSBAC restricts access by subjects to objects. The subjects are always processes, acting on behalf of a user with certain attributes, like system_role etc. Objects in *RSBAC* are called (Access) *Targets*. They are grouped in *Target Types*. The following types are defined:

Table 3-1. RSBAC targets

Type	Comment
FILE	Files, including device special files. Identified by device and inode number.
DIR	Directories, identified by device and inode number.
FIFO	(new in v1.1.1) FIFO special files
DEV	Devices, identified by type (char or block), major and minor number
IPC	InterProcess Communication: Semaphores (sem), Messages (msg), Shared Memory shm), Sockets (sock) and FiFo (fifo, removed in 1.1.1).
SCD	System Control Data: Objects affecting the whole system. This target type is the only one with a fixed number of objects, identified by number (see Table 3-2>).
USER	Users as objects, mostly for access control information (*ACI*).
PROCESS	Processes as objects.
NETDEV	Network Device, identified by name.
NETTEMP	Network Template, identified by index number. Access control: access to template itself, RC Administration: access to values/settings for both template and NETOBJ, ACL administration: Default ACLs for NETOBJ.
NETOBJ	Network Object, identified by internal pointer to struct socket. Attribute values mostly inherited from NETTEMP settings.
NETTEMP_NT	ACL administration only, ACL entries for NETTEMP objects themselves.
NONE	No object associated with this request. In some models (*RC*, *ACL*) this is internally changed into *SCD* target “other”.
FD	(Only in user space): Let the command line tool decide between types FILE and DIR

System Control Data (SCD) targets are these:

Table 3-2. RSBAC SCD Targets

Type	Comment
time_strucs	System timer
clock	System time and date
host_id	Host name
net_id	Domain name
ioports	Access Control for direct hardware access
rlimit	Setting process resource limits
swap	Control of swapping
syslog	System log
rsbac	*RSBAC* data in `/proc`
rsbaclog	*RSBAC* own log
kmem	Direct access to kernel memory via proc or device
other	`MODIFY_SYSTEM_DATA` for sysctl, otherwise only internal in *RC* and *ACL*: Substitute for target `NONE`.
auth_administration	(only in RC and ACL) AUTH model administration
network	General networking, like routing, arp etc. (Devices are protected as NETDEV targets!).
firewall	Firewall settings, packet filter etc.

3.2. Requests

Before access to a target is granted, a request call to the *Access Control Decision* facility (*ADF*) is performed. Based on the request type and the target, access can be granted or denied.

RSBAC requests and the system calls they are issued from are listed in the following table. Please note that some requests are only issued under certain conditions, e.g. `EXECUTE` from `mmap()` only, if mapping request is for `EXEC` mode. Also, some calls depend on the kernel configuration settings, e.g. *RSBAC* net support.

Some calls are done from common helper functions, e.g. `do_fork()`. Those functions that also perform the `rsbac_adf_set_attr()` notification call for the request are marked with an *.

Additionally, some requests provide extra data with kernel internal attribute types. These attributes are: `A_group`, `A_sockaddr_p`, `A_signal`, `A_mode`, `A_nlink`, `A_switch_target`, `A_mod_name`, `A_request`, `A_ms_segment`, `A_trace_request`, `A_auth_add_f_cap`, `A_auth_remove_f_cap`, `A_auth_get_caplist`, `A_prot_bits`. Please have a look into `include/rsbac/types.h` for the respective data types.

Table 3-3. RSBAC Requests

Request	Description	Valid Target Types	System calls and funtions
ADD_TO_KERNEL	Add a kernel module	NONE	create_module(NONE), init_module(NONE)

Request	**Description**	**Valid Target Types**	**System calls and funtions**
ALTER	Change IPC control information	IPC	msgctl(IPC), shmctl(IPC)
APPEND_OPEN	Open to append	FILE, DEV, IPC	open(FILE,DEV)*, msgsnd(IPC)*, sendto(IPC)*, sendmsg(IPC)*
CHANGE_GROUP	Change active group	IPC,PROCESS,NONE	setgid(PROC), [a] setregid(PROC),setresgid(PROC),setgroups(PRO setfsgid(NONE) (for DAC only), shmctl(IPC), msgctl(IPC)
CHANGE_OWNER	Change owner	FILE, DIR, FIFO, IPC, PROCESS, NONE	chown(FILE, DIR, FIFO), lchown(FILE, DIR, FIFO), fchown(FILE, DIR, FIFO), setuid(PROC)*, setreuid(PROC)*, setresuid(PROC)*, setfsuid(NONE) (for DAC only), shmctl(IPC), msgctl(IPC)
CHDIR	Change working directory	DIR	chdir(DIR), fchdir(DIR), chroot(DIR)
CLONE	Fork/clone a process	PROCESS	fork(PROC)*, vfork(PROC)*, clone(PROC)*
CLOSE	Close opened file etc. Should always be granted.	FILE, DIR, FIFO, DEV, IPC, NETOBJ	close(FILE, DIR, FIFO, DEV, IPC, NETOBJ)*, shmdt(IPC)*, msgrcv(IPC)*, msgsnd(IPC)*, send(IPC)*, sendto(IPC)*, sendmsg(IPC)*, recv(IPC)*, recvfrom(IPC)*, recvmsg(IPC)*
CREATE	Create object	DIR(where), IPC, NETTEMP, NETOBJ	creat(DIR, IPC)*, open(DIR, IPC)*, mknod(DIR)*, mkdir(DIR)*, symlink(DIR)*, shmget(IPC)*, msgget(IPC)*, socket(IPC)*, accept(IPC)*, rsbac_net_temp(NETTEMP), socket(NETOBJ)

Request	**Description**	**Valid Target Types**	**System calls and funtions**
DELETE	Delete object	FILE, DIR, FIFO, IPC	unlink(FILE, DIR, FIFO)*, rmdir(DIR)*, msgctl(IPC)*, shmctl(IPC)*, shutdown(IPC)*, close(IPC)*, rsbac_net_temp(NETTEMP)
EXECUTE	Execute a file (until v1.1.2: also map library file/other code executable, see `MAP_EXEC`)	FILE	exec()*
GET_PERMISSIONS_DATA	Read Unix permissions (mode)	FILE, DIR, FIFO	access(FILE, DIR, FIFO)
GET_STATUS_DATA	Get status (stat() etc.)	FILE, DIR, FIFO, IPC, SCD, NETDEV	open_port(SCD) (`/dev/kmem` etc.), open_kcore(SCD) (`/proc/kcore`), stat(FILE, DIR, FIFO, IPC), newstat(FILE, DIR, FIFO, IPC), lstat(FILE, DIR, FIFO, IPC), newlstat(FILE, DIR, FIFO, IPC), fstat(FILE, DIR, FIFO, IPC), newfstat(FILE, DIR, FIFO, IPC), stat64(FILE, DIR, FIFO, IPC), lstat64(FILE, DIR, FIFO, IPC), fstat64(FILE, DIR, FIFO, IPC), statfs(FILE, DIR, FIFO), fstatfs(FILE, DIR, FIFO), rsbac_stats(SCD), rsbac_check(SCD), rsbac_stats_pm(SCD), rsbac_stats_rc(SCD), rsbac_stats_acl(SCD), rsbac_log(SCD), (access to *RSBAC* proc-files(SCD)), dev_ioctl(NETDEV), arp_ioctl(NETDEV), ip_mroute_setsockopt(SCD network), firewalling code (SCD firewall)
LINK_HARD	Hard link	FILE, DIR, FIFO	link(FILE, DIR, FIFO)
MODIFY_ACCESS_DATA	Change access information, e.g. time, date	FILE, DIR, FIFO	utimes(FILE, DIR, FIFO)
MODIFY_ATTRIBUTE	Change an *RSBAC* attribute value	All target types	(specific request needed for various security models)

Request	Description	Valid Target Types	System calls and funtions
MODIFY_PERMISSIONS_DATA	Change Unix permissions	FILE, DIR, FIFO, SCD	ioperm(SCD), iopl(SCD), chmod(FILE, DIR, FIFO), fchmod(FILE, DIR, FIFO)
MODIFY_SYSTEM_DATA	Change system settings	SCD, NETDEV	stime(SCD), settimeofday(SCD), adjtimex(SCD), sethostname(SCD), setdomainname(SCD), setrlimit(SCD), syslog(SCD), sysctl(SCD), swapon(SCD), swapoff(SCD), rsbac_log(SCD), dev_ioctl(NETDEV), arp_ioctl(NETDEV), ip_mroute_setsockopt(SCD network), firewalling code (SCD firewall)
MOUNT	Mount a filesystem	DIR, DEV	mount(DIR, DEV) (separate mount notification for data structures)
READ	Read from DIR or NETTEMP.Optional: read from other objects	DIR, NETTEMP (optional: FILE, FIFO, DEV, IPC, NETOBJ)	read(FILE, FIFO, DEV, IPC, NETOBJ)*, readv(FILE, FIFO, DEV, IPC)*, pread(FILE, DEV, IPC)*, readdir(DIR), open(DIR), rs-bac_net_temp(NETTEMP)
READ_ATTRIBUTE	Read *RSBAC* attribute value	All target types	(specific request needed for various security models)
READ_OPEN	Open for read	FILE, FIFO, DEV, IPC	open(FILE, FIFO, DEV, IPC)*, shmat(IPC)*, msgrcv(IPC)*, recv(IPC)*, recvfrom(IPC)*, recvmsg(IPC)*
READ_WRITE_OPEN	Open for read and write	FILE, FIFO, DEV, IPC	open(FILE, FIFO, DEV, IPC)*, shmat(IPC)*, bind(IPC)*, connect(IPC)*, listen(IPC)*
REMOVE_FROM_KERNEL	Remove kernel module	NONE	delete_module(NONE)
RENAME	Rename	FILE, DIR, FIFO	rename(FILE, DIR, FIFO) (*RSBAC* identification not changed by rename!)

Request	Description	Valid Target Types	System calls and funtions
SEARCH	Lookup in dir or symlink from inside kernel for access with full path	DIR, SYMLINK	(internal functions: lookup_dentry(DIR), path_walk(DIR), lookup_hash(DIR), follow_symlink(SYMLINK))
SEND_SIGNAL	Send a signal	PROCESS	kill(PROC)
SHUTDOWN	Shutdown/reboot system	NONE	reboot(NONE)
SWITCH_LOG	Change RSBAC log settings	NONE	rsbac_adf_log_switch(NONE)
SWITCH_MODULE	Switch decision module on/off	NONE	rsbac_switch(NONE)
TERMINATE	End of calling process, for attribute cleanup. Should always be granted.	PROCESS	exit(PROC)
TRACE	Trace a process	PROCESS	ptrace(PROC) (architecture dependent)
TRUNCATE	Truncate	FILE	open(FILE)*, truncate(FILE)*,ftruncate(FILE)*, truncate64(FILE)*, ftruncate64(FILE)*
UMOUNT	Umount a filesystem	DIR, DEV	umount(DIR, DEV) (separate umount notification for data structures)
WRITE	Write to a DIR, SCD or NETTEMP. Used for object moving to target dir. Optional: write to file etc.	DIR, SCD (optional: FILE, FIFO, DEV, IPC-sock)	write(FILE, FIFO, IPC, DEV, NETTEMP)*, writev(FILE, FIFO, IPC, DEV)*, pwrite(FILE, IPC, DEV)*, rename(DIR), rsbac_write(SCD), rsbac_net_temp(NETTEMP)
WRITE_OPEN	Open for write	FILE, FIFO, DEV, IPC	open(FILE, FIFO, DEV, IPC)*
MAP_EXEC	Map a library from a file (target `FILE`) or other code (target `NONE`) for execution.	FILE, NONE	mmap(FILE) (`EXEC` mode), mprotect(FILE, NONE) (`EXEC` mode), uselib(FILE)
BIND	Bind network address and port (if applicable) to local socket, bind to network device	NETDEV, NETOBJ	dev_ioctl(NETDEV), bind()*
LISTEN	Listen on a local socket	NETOBJ	listen()*
ACCEPT	Accept a connection from a remote network endpoint	NETOBJ	accept()*

Request	Description	Valid Target Types	System calls and funtions
CONNECT	Connect to remote network endpoint	NETOBJ	connect()*
SEND	Send to remote network endpoint	NETOBJ	send()*, sendmsg()*, sendto()*
RECEIVE	Receive from remote network endpoint	NETOBJ	recv()*, recvmsg()*, recvfrom()*
NET_SHUTDOWN	Shutdown channel of local socket	NETOBJ	shutdown()
Notes: a. `PROC` means `PROCESS`			

Please remember that some models (*RC*, *ACL*) internally change `NONE` targets to *SCD* target “other” for access control.

II. Part II: RSBAC manual pages

The second part of this book contain manual pages for the various RSBAC commands.

Chapter 4. RSBAC Manual pages

4.1. RSBAC manual pages

acl_mask

Name

`acl_mask` — View or set an object's mask for right inheritance

Synopsis

attr_set_fd `[-vrpsbn]` `[-V`*`version`*`]` `{rights}` `{target-type}` `[file/dirname(s)]`

Description

Using this utility you can view or set an object's general ACL mask.

Options

`-v`

verbose output

`-r`

recurse into subdirs

`-p`

print right names

`-s`

set mask, not get

`-b`

backup mode

`-n`

list valid SCD names

`-V` *`version`*

supply RSBAC integer `version` number for upgrading `rights`

list of space-separated right names (requests and ACL specials), also request groups: R (read requests), RW (read-write), SY (system), SE (security), A (all) and S (ACL special rights)

`target-type`

One of possible *RSBAC* target types: FILE, DIR, FIFO, SYMLINK, DEV, SCD, NETDEV NETTEMP_NT, NETTEMP, NETOBJ or FD. (FD: lets program decide between FILE, DIR, FIFO and SYMLINK, no DEV)

acl_rm_user

Name

`acl_rm_user` — remove all groups and memberships of a user

Synopsis

acl_rm_user [`-y`] {`user`}

Description

Remove everything in ACL related to a user. It is very useful to run this utility after **userdel** command to get rid of old settings.

Options

`-y`

remove without asking

`user`

name or id of the user

attr_back_fd

Name

`attr_back_fd` — Backup *RSBAC* attributes from filesystem objects

Synopsis

attr_back_fd [`-vrnmia`] [`-o` *`target-file`*] [`file/dirname(s)`]

Description

You should use `attr_back_fd` to backup *RSBAC* attributes of filesystem objects. This program should be called by a user with full attribute read access, e.g. root with all modules off. You can also create special settings (e.g. special role for *RC*) for modules you use in your system. Symlinks are not followed.

Options

`-v`

be verbose

`-r`

walk recursively into subdirs

`-m`

ignore `ms_scanned`

`-i`

use *MAC* non-inherit values as default values

`-o` *`target-file`*

write to *`target-file`*, not stdout

`-a`

list attributes and values

attr_back_user

Name

`attr_back_user` — Backup *RSBAC* attributes for users

Synopsis

attr_back_user [`-aAv`] [`-o` *`target-file`*] [`username(s)`]

Description

You should use `attr_back_user` to backup *RSBAC* attributes of system users. This program should be called by a user with full attribute read access, e.g. the default user 400. You can also create special settings (e.g. special role for *RC*) for modules you use in your system.

Options

`-v`

be verbose

`-a`

process all known user accounts

`-o` *`target-file`*

write to *`target-file`*, not stdout

`-A`

list attributes and values

attr_get_fd

Name

`attr_get_fd` — fetch *RSBAC* attributes from files or directories

Synopsis

attr_get_fd [`-vrnea`] {`module`} {`target-type`} {`attribute`} [`file/dirname(s)`]

Description

You need `attr_get_fd` to get *RSBAC* attribute values for filesystem objects. There are different attributes for different *RSBAC* modules. Check appropriate documentation about possible attributes for module you want to administrate or use `-a` option to see full list.

Options

`-v`

verbose program output

`-e`

show effective (maybe inherited) value instead of real values. See appropriate documentation about difference between effective and real access rights.

`-r`

walk recursively into subdirs

`-n`

list all requests. Full target and request lists are in the “Targets and Requests” chapter of the *RSBAC* documentation.

`-a`

list attributes and values

`module`

One of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH.

`target-type`

One of the possible *RSBAC* target types, e.g., FILE, DIR, FIFO, SYMLINK, DEV or FD (FD: all of FILE, DIR, FIFO and SYMLINK, but no DEV))

attr_get_file_dir

Name

`attr_get_file_dir` — fetch *RSBAC* attribute values for filesystem objects

Synopsis

attr_get_file_dir [`-epcRa`] [`-n` *`target`*
] {`module`} {`target-type`} {`file/dirname(s)`} {`attribute`} [`request`]

attr_get_file_dir [`-epcRa`] [`-n` *`target`*
] {`module`} {`target-type`} {`file/dirname(s)`} {`attribute`} [`position`]

attr_get_file_dir {`list_category_nr`}

Description

You need `attr_get_file_dir` to get *RSBAC* attribute values of filesystem objects. There are different attributes for different *RSBAC* modules. Check appropriate documentation about possible attributes for module you want to administrate or use `-a` option to see the full list.

Options

`-e`

show effective (maybe inherited) value instead of the real value. See appropriate documentation about the difference between effective and real values.

`-p`

print names of used requests

`-n` *`target`*

list all requests for *`target`*

`-c`

list all Linux capabilities

`-R`

list all *RES* resource names

`-a`

list all attributes and values

`-A`

list attributes and values

`module`

One of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC, AUTH or RES.

`target-type`

One of the possible *RSBAC* target types, e.g., FILE, DIR, FIFO, SYMLINK or DEV.

attr_get_process

Name

`attr_get_process` — get *RSBAC* attributes on the selected process

Synopsis

attr_get_process [`-pna`] {`module`} {`pid`} {`attribute`} [`bit-no`]

Description

If you want to get *RSBAC* attribute on some process, you may use `attr_get_process` utility. Check appropriate documentation about possible attributes and values for module you want to administrate or use `-a` option to see full list.

Options

`-p`

print all request names

`-n`

list all request names

`-a`

list attributes and values

`module`

one of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH

`bit-no`

additional parameter for vector based attributes, like `categories` and `log_program_based`. Value 0 means no, and 1 means yes.

attr_get_up

Name

`attr_get_up` — get *RSBAC* attributes on the selected user or process

Synopsis

attr_get_up [`-a`] {`module`} {`target-type`} {`attribute`} [`user(s)/proc-no`]

Description

If you want to get *RSBAC* attribute on some process or user, you may use `attr_get_up` utility. Check appropriate documentation about possible attributes and values for module you want to administrate or use `-a` option to see full list.

Options

`-a`

list attributes and values

`module`

one of the possible *RSBAC* modules, e.g., EN, MAC, FC, SIM, PM, MS, FF, RC or AUTH

`target-type`

USER or PROCESS

attr_get_user

Name

`attr_get_user` — get *RSBAC* attributes of the selected user

Synopsis

attr_get_process [`-enblcRa`] {`module`} {`user`} {`attribute`} [`position`|`request-name`]

Description

If you want to get *RSBAC* attribute of some user, you may use `attr_get_user` utility. Check appropriate documentation about possible attributes and values for module you want to administrate or use `-a` option to see full list.

Options

`-e`

show effective (maybe inherited) value, not real

`-n`

numeric value

`-b`

both names and numbers

`-l`

list all users

`-c`

list all Linux capabilities. This option is useful only for *CAPS* module

`-R`

list all RES resource names. This option is useful only for *RES* module

`-a`

list attributes and values

`module`

one of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH

`mac_[min_]categories`

has vector type. Works with additional parameter position: 0=no, 1=yes

`log_user_based`

has vector type. Works with additional parameter request-name: 0=no, 1=yes

attr_rm_fd

Name

`attr_rm_fd` — remove (reset into default values) all *RSBAC* attributes of file or directory

Synopsis

attr_rm_fd [`-vr`] {`target-type`} [`file/dirname`]

Description

Remove (reset into default values) all attributes of the selected file or directory. It is usually not necessary to run this utility, because attributes of deleted objects are automatically removed.

Options

`-v`

verbose output

`-r`

recurse into subdirs

`target-type`

Valid *RSBAC* target type, on of the: FILE, DIR, FIFO, SYMLINK, DEV or FD (FD: lets program decide between FILE, DIR, FIFO and SYMLINK, no DEV)

attr_rm_file_dir

Name

`attr_rm_file_dir` — remove (reset into default values) all *RSBAC* attributes of file or directory. It is usually not necessary to run this utility, because attributes of deleted objects are automatically removed.

Synopsis

attr_rm_file_dir {`target-type`} [`file/dirname`]

Description

Remove (reset into default values) all attributes of selected file or directory.

Options

`target-type`

FILE, DIR, FIFO, SYMLINK or DEV

attr_rm_user

Name

`attr_rm_user` — remove all attributes of a user

Synopsis

attr_rm_user {`user...`}

Description

Remove all attributes related to a user. It is advisable to run this utility after e.g. the **userdel** command to get rid of old settings, because the kernel does not know about valid user accounts.

Options

`user`

name or UID of the user

attr_set_fd

Name

`attr_set_fd` — set *RSBAC* attributes on the selected file or directory

Synopsis

attr_set_fd [`-vrna`] [`-V`*`version`*] {`module`} {`target-type`} {`value`} [`file/dirname(s)`]

Description

If you want to change *RSBAC* attribute for some file or directory, you can use the `attr_set_fd` utility. Check appropriate documentation about possible attributes and values for the module you want to administrate or use `-a` option to see full list.

Options

`-v`

verbose output

`-r`

recurse into subdirs, attributes will only be set for targets of the given type

`-n`

list all requests

`-a`

list attributes and values

`-V` *`version`*

supply RSBAC integer `version` number for upgrading

`module`

One of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH.

`target-type`

One of the possible file object target types: FILE, DIR, FIFO, SYMLINK, DEV or FD. FD target type lets the system decide between FILE, DIR, FIFO and SYMLINK, but no DEV.

attr_set_process

Name

`attr_set_process` — set *RSBAC* attributes on the selected process

Synopsis

attr_set_process [`-pamA`] {`module`} {`process-id`} {`attribute`} {`value`}

Description

If you want to change *RSBAC* attribute on some process, you may use `attr_set_process` utility. Check appropriate documentation about possible attributes and values for module you want to administrate or use `-A` option to see full list.

Options

`-p`

Print resulting request names.

`-a`

Add, not set. This is usefull for attributes like process' *MAC* category.

`-m`

Remove, not set

`-A`

list attributes and values

`module`

One of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH.

attr_set_user

Name

`attr_set_user` — set *RSBAC* attributes on the selected user

Synopsis

attr_set_user [`-pamAV`] {`module`} {`user`} {`attribute`} [`position`] {`value`}

attr_set_user [`-pamAV`] {`module`} {`user`} {`log_user_based`} [`request-list`]

Description

If you want to change *RSBAC* attribute for some user, you may use `attr_set_user` utility. Check appropriate documentation about possible attributes and values for module you want to administrate or use `-A` option to see full list.

Options

`-p`

Print resulting requests.

`-a`

Add, not set. This is useful for attributes like users' *MAC* category set.

`-m`

Remove, not set

`-A`

list attributes and values

`module`

One of the possible *RSBAC* modules, e.g., GEN, MAC, FC, SIM, PM, MS, FF, RC or AUTH.

linux2acl

Name

`linux2acl` — convert linux groups and rights to *RSBAC ACL* groups and rights

Synopsis

linux2acl [`-vrgGpPn`] {`file/dir/scdname(s)`}

Description

`linux2acl` creates an ACL model administration script from existing Linux groups and filesystem object rights. Should be used, if Linux filesystem access control is meant to be replaced by ACL and disabled via `rsbac_dac_disable` (s.a.).

See "Installation and Administration" guide about compile time kernel options.

Options

`-v`

use verbose in scripts

`-r`

recurse into subdirs

`-g`

also create group entries with members

`-G`

only create group entries with members

`-p`

print right names

`-P`

use private groups

`-n`

use numeric user ids where possible

net_temp

Name

`net_temp` — work with network templates

Synopsis

net_temp [`-vbsnluda`] [`-V` *`version`*] {`function`} {`id`} [`set-param`]

net_temp [`-vbsnluda`] {`list_temp_{names,nr}`}

net_temp [`-vbsnluda`] {`list_template`} {`id`}

Description

work with network templates

Options

`-v`

verbose output

`-l`

list available functions

`-b`

backup mode

`-s`

scripting mode

`-n`

take number as address

`-u`

take string as address

`-d`

take DNS name as address and convert to IP address

`-a`

list all templates in detail

`-V` *`version`*

supply *RSBAC* integer version number for upgrading

pm_create

Name

`pm_create` — create some files in a particular *PM* class

Synopsis

pm_create {`class`} {`mode`} [`filename(s)`]

Description

This program will create files with PM class `class` and Linux access rights `mode`. Please note that in PM model, the create right depends on the desired class of the new object.

See appropriate *RSBAC* documentation for *PM* module details.

Options

`class`

number of the PM class

`mode`

access rights of new files

pm_ct_exec

Name

`pm_ct_exec` — start program with some PM current task

Synopsis

pm_ct_exec {`task-nr`} {`prog`} [`args`]

Description

This program will set `rsbac_pm_current_task` to `task-nr` and then execute prog via execvp().

See appropriate *RSBAC* documentation for *PM* module details.

Options

`task-nr`

number of the PM task

`program`

program to run

rc_copy_role

Name

`rc_copy_role` — copy one *RC* role with all settings into another

Synopsis

rsbac_copy_role {`from_role`} {`to_role`}

Description

During *RC* module administration, this command allows to copy an existing role with all its associated rights. This functionality is useful to e.g. split one role into two, or to create testing.

To be secure, test your configurations with different role numbers and use `rc_copy_role` to copy (create) them, if necessary.

rc_role_wrap

Name

`rc_role_wrap` — execute some program with different *RC* role

Synopsis

rc_role_wrap [`-v`] {`new_role`} {`program`} [`args`]

Description

Sometimes, you need to run some program with different *RC* roles. Note that `new_role` must be one of the compatible roles of the role requesting the change.

A *RC force role* or *RC initial role* attribute value on the executable file often provides a better solution.

Options

`-v`

verbose output

rsbac_check

Name

`rsbac_check` — trigger consistency checking

Synopsis

rsbac_check {`correct`} {`check_inode`}

Description

request the data structures to be checked for consistency. This can also reduce list sizes, because unnecessary entries and those with negative time-to-live are deleted. It is advisable to run this command regularly, e.g. once per week.

Options

`correct`

correction mode: 0 - do not correct errors, 1 - correct errors, 2 - correct more.

`check_inode`

checking mode: 0 - do not check inode numbers, 1 - also check inode numbers.

rsbac_jail

Name

`rsbac_jail` — put program into *RSBAC* jail

Synopsis

rsbac_jail [`-vilnrao`] {`path`} {`IP`} {`prog`} [`args`]

Description

All *Linux* kernels provide the chroot system call to confine a process in a subdirectory. Unfortunately, this does not protect the system from root processes, and it can be broken out of. The *JAIL* module extends the chroot system call functionality to provide a superset of the *FreeBSD* jail functionality (except individual kernel level hostnames).

This program will put the process into a jail with chroot to `path`, ip address `IP` and then execute `prog` with `args`.

See appropriate *RSBAC* documentation about for *JAIL* module details.

Options

`-v`

verbose program output

`-i`

allow access to `IPC` outside this jail

`-l`

allow jailed processes to change their rlimits

`-n`

allow all network families, not only `UNIX` and `INET` (`IPv4`)

`-r`

allow `INET` (`IPv4`) raw sockets (e.g. for ping)

`-a`

auto-adjust `INET` any address `0.0.0.0` to jail address, if set

`-o`

additionally allow to/from remote `INET` (`IPv4`) address `127.0.0.1`

rsbac_stats

Name

`rsbac_stats` — write *RSBAC* status info into syslog

Synopsis

rsbac_stats

Description

This program requests the kernel to write the current *RSBAC* status to the syslog at level `KERN_INFO`. You can see results in system logs (or logs of rklogd) later.

This feature may be useful in some scripts

rsbac_write

Name

`rsbac_write` — request the modified data structures to be written to disk

Synopsis

rsbac_write

Description

RSBAC has two locations of *ACI* (Access Control information): on disks and in the computer's RAM. During system boot *RSBAC* ACI is read from disk and placed into appropriate data structures in kernel memory. When you change some attributes, you really made changes in RAM storage. *RSBAC* syncs modified data from RAM to disk every few seconds (2 secs by default, see kernel config). So, you can loose your changes on some unexpected system damage.

Use **rsbac_write**, if you want to write the changed *RSBAC* attributes to disk immediately, or to get it written at all, if you disabled automatic writing in kernel config. Please note that the data will also be written when the respective filesystem gets umounted.

switch_adf_log

Name

`switch_adf_log` — switch general log subsystem settings

Synopsis

switch_adf_log [`-ntbgs`] [`-V` *`version`*] {`request`} [`target`] [`value`]

Description

RSBAC has general log settings and log settings per user or process. Using `adf_switch_log` you can change general log settings for all types of *RSBAC* requests.

See appropriate *RSBAC* documentation about *ADF* subsystem and possible requests. You can also use `-n` to see possible requests and `-t` to see possible targets.

Options

`-n`

list all requests

`-t`

list all target types

`-b`

backup log level settings

`-g`

get not set

`-s`

scripting mode

`-V` *`version`*

supply RSBAC integer version number for upgrading

`request`

one of possible RSBAC requests or `ALL` for all requests.

`value`

0, 1 or 2. Amon, please, describe this values.

`target`

One of the possible RSBAC targets or leave out for `ALL` targets.

switch_module

Name

`switch_module` — switch *RSBAC* module on/off or enable/disable global or individual module “soft mode”

Synopsis

switch_module [`-s`] {`module`} {`value`}

Description

Switch decision modules on or off, if enabled by kernel configuration. Parameters are modules name and 0 or 1. Necessary permissions are module dependent. This tool can also switch “soft mode” on or off.

See appropriate *RSBAC* documentation about “soft mode”.

Options

`-s`

switch module’s individual softmode, not the whole module

`module`

name of RSBAC module, e.g., *RC* or *ACL*

`value`

1 corresponds to “turn on”, and 0 - “turn off”

Appendix A. GNU Free Documentation License

GNU Free Documentation License
Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the

publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies

to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under

the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliography

[Albitz01] John Doe and Wei Ping, *A fake bibliography entry (first edition)* Tinkerman, 2002. ISBN 0-555-55555-4.

The RSBAC cookbook

Amon Ott

Stanislav Ievlev

Heinrich W. Klöpping

The RSBAC cookbook
by Amon Ott, Stanislav Ievlev, and Heinrich W. Klöpping

Audience: This book is intended for use by experienced and skilled Unix professionals that wish to install, configure and use RSBAC.

Approach: This book resulted from a project founded on June 28th, 2002 by Amon Ott, Stanislav Ievlev and Henk Klöpping. We aimed at providing a *cookbook* -- a book filled with hands-on instructions and examples of the installation, configuration and use of *Rule Set Based Access control* (RSBAC). This book will be accompanied by four other books: *The RSBAC reference manual*, *The RSBAC programmers cookbook*, *The RSBAC programmers reference manual* and *The RSBAC introduction*.

To learn where the latest version of this book can be downloaded or read please refer to Section 6.2 in *The RSBAC introduction*.

Sources: Our sources of information were (Open Source) material on the Internet, several books, practical experience of the authors and others and research and programming work done by the authors. We try to give credit where due, but are fallible. We apologize.

Caution

While every precaution was made in the preparation of this book, we can assume no responsibility for errors or omissions. When you feel we have not given you proper credit or feel we may have violated your rights or when you have suggestions how we may improve our work please notify us immediately so we can take corrective actions.

Organization of this book: This book has been organised in four parts:

I. context and background information
II. theory of RSBAC (including security models)
III. obtaining and installing RSBAC
IV. administering RSBAC

This book was written using the DocBook V3.1/SGML documentation standard.

Dedication

> And I said, "You can stop if you want with the Z
> Because most people stop with the Z - but not me!
> In the places that I go there are things that I see
> That I never could spell if I stopped with the Z.
> I'm telling you this because you're one of my friends
> My alphabet starts where your alphabet ends!
>
> You'll be sort of surprised what there is to be found
> Once you go beyond Z and start poking around"
>
> —Dr Seuss *On Beyond Zebra*

Table of Contents

Preface

Linux has always been a very stable and trustworthy operating system - even more so in comparison with it's closed-source alternatives. Driven by closed source vendors' questionable license policies, security risks, bugs and vendor-lock, more and more IT-managers choose the Linux alternative. Linux also has a good reputation -- as have other Unices -- when it comes to security. That may or may not be due to the blatant lack of proper security in other "operating systems". However, your data is arguably *not* safe on a standard Linux system. Linux is susceptible to security breaches, malware and programming bugs too.

Hence, a number of workarounds and extensions have been written. One of the most popular (and in my not so humble opinion one of the most elegant and stable ones) is *Rule Set Based Access Control*. I have been working with RSBAC since 1999 and have been impressed by what Amon wrote ever since. However, the lack of a good cookbook for it struck me as one of the major hurdles on the road to its acceptance. For I am convinced that it deserves such broad acceptance given its qualities.

So, we set out to write such a cookbook. And here it is. It still is a work in progress, and unless the nature of security related work suddenly changes probably will be so indefinitely. The authors wanted to create a useful book that would guide you through the seemingly awkward process of understanding, installing and maintaining RSBAC. This first version of our book originated from various materials, amongst them the introduction to RSBAC written by Stanislav Ievlev, the original documentation written by the author of RSBAC, Amon Ott, and a sequel of four articles I wrote for the Dutch Linux Magazine. We had to write many additional chapters from scratch and many hours were spent researching and trying actual security configurations. We finally made it.

It is up to you to judge our efforts, and, hopefully, improve our work. To quote the laudated Dr David A. Lien, author of the excellent (1977) TRS80 Level I BASIC beginners manual: "Sit back, relax, read slowly as though savoring a good novel, and above all, let your imagination wander. I'll supply you with all the routine facts and techniques you need. The real enjoyment begins when your imagination start the creative juices flowing and a computer becomes a tool in your own hands. You become its master - not the other way around.".

To the many people we unintentionally forgot to give credit where due, from the bottom of our hearts: **we thank you.**

Henk Klöpping, December 2002

I. Context and Background

Before you start tinkering with your systems, you need to think things through. You need to set policies or at least think a bit about security. That requires some knowledge about security related concepts. This part introduces you to a number of these concepts and provides most of the information you need to be able to find the proper balance between safety and usability for your systems. After reading this part you will be able to set a security policy and know the implications of that.

Chapter 1. bogus chapter title

1.1. bogus section title

Some bogus text.

II. Theory of RSBAC

RSBAC bases on academic work. The theoretical framework that layed the foundation for the RSBAC implementation is discussed and a number of alternative studies are briefly introduced. Next the RSBAC implementation is dicussed: the framework and its components are introduced, as is the theory of security models. In the last chapters of this part all security models that RSBAC currently supports are thouroughly discussed.

Chapter 2. bogus chapter title

2.1. bogus section title

Some bogus text.

III. Obtaining and installing RSBAC

This part informs you how to obtain the RSBAC source code and how to compile, test and install RSBAC. It will guide you through the process of installation and initial configuration step by step.

Chapter 3. bogus chapter title

3.1. bogus section title

Some bogus text.

IV. Administering RSBAC

In this part we provide various examples of possible combination of RSBAC models and how to administer them. A number of example systems is introduced and we give hands-on instructions on how to configure a secure webserver, a secure Samba server and a server where you can safely store privacy sensitive data. For the most commonly used models details are provided on how to administer them[1]. Most administrative tools are introduced here along with many (tested) examples of their use.

Notes

1. the less commonly used models are not mentioned here, but will be discussed in separate papers and *The RSBAC reference manual*.

Chapter 4. bogus chapter title

4.1. bogus section title

Some bogus text.

Appendix A. GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the

publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies

to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under

the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will

automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
> Permission is granted to copy, distribute and/or modify this document
> under the terms of the GNU Free Documentation License, Version 1.1
> or any later version published by the Free Software Foundation;
> with the Invariant Sections being LIST THEIR TITLES, with the
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
> A copy of the license is included in the section entitled "GNU
> Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliography

[Albitz01] John Doe and Wei Ping, *A fake bibliography entry (first edition)* Tinkerman, 2002. ISBN 0-555-55555-4.