Hardening systems with AppArmor

GOLDEN CAGE

www.photocase.com

After penetrating a remote system, intruders might think they are home and dry, but AppArmor spoils the fun, locking the miscreants in a virtual cage. **BY RALF SPENNEBERG**

Nobody's perfect – and this is particularly true of software. Any non-trivial application will have its fair share of programming errors. Intruders exploit these errors, taking control of the software, and making the program do things the developer never envisaged. The situation starts to become critical if the application has privileges that are different from the privileges of the attacker.

For example, the *ping* command requires root privileges in order to send the special packet formats that it needs. But it is theoretically possible for the process to misuse its root privileges to cause all kinds of trouble. Although *ping* is a well-behaved program, an attacker capable of hijacking the tool would have unrestricted access to the rest of the system.

AppArmor [1] changes this. Instead of allowing the root program unrestricted access to the system, it assigns limits, attempting to achieve a balance between effectiveness and complexity. AppArmor uses simple and transparent mechanisms to provide a high standard of protection, which is similar to Systrace. It does not attempt to compete with more complex systems such as SELinux or RSBAC, but then again, configuring these complex alternatives requires much more skill on the part of the system administrator.

Listing 1: Building an AppArmor-capable kernel

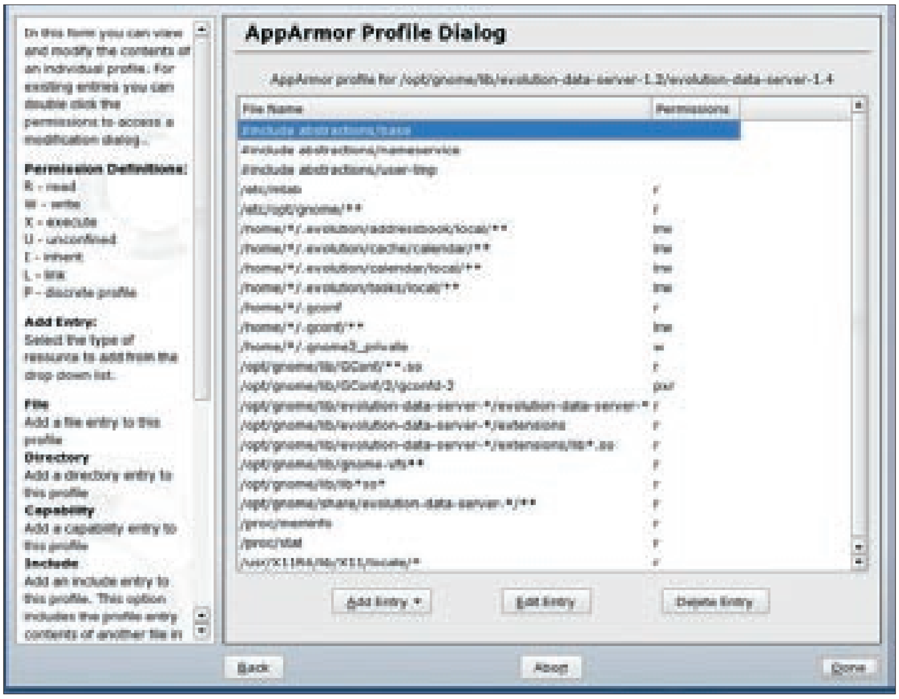
```
tar xjf linux-2.6.15.tar.bz2
cd linux-2.6.15
patch -p1 <../aa_2.0-2.6.15.patch
patch -p1 <../aa_namespace_sem-2.6.15.patch
make oldconfig
make bzImage
make modules
make modules_install
make install
rmdir /subdomain
ln -s /sys/kernel/security/subdomain /subdomain
```

Hardened

Despite all the additional obstacles, the first rule of security is to avoid vulnerabilities. On your AppArmor-protected computer – on any system for that matter – you should disable services you do not need, patch in good time, and use a carefully-crafted configuration. This leaves AppArmor to deal with previously unknown vulnerabilities and accompanying zero-day exploits.

AppArmor monitors what files an application accesses, and what kind of access this is; at the same time, it governs the use of root privileges. Depending on the kernel version, Linux can distinguish between 29 different capabilities (see *man 7 capabilities*). For example, *CAP_KILL* refers to root's ability to terminate any process, and *CAP_NET_RAW* to the ability to create arbitrary network packets.

In the case of the ping command, it would be perfectly OK for AppArmor to allow the use of *CAP_NET_RAW*, but to deny the use of *CAP_KILL*. This would



In this form you can view and modify the contents of an individual profile. For existing entries you can double click the permissions to access a modification dialog.

Permission Definitions:
R - read
W - write
X - execute
U - unconfined
I - inherit
L - link
P - discrete profile

Add Entry:
Select the type of resource to add from the drop down list.

File
Add a file entry to this profile
Directory
Add a directory entry to this profile
Capability
Add a capability entry to this profile
Include
Add an include entry to this profile. This option includes the profile entry contents of another file in

AppArmor Profile Dialog

AppArmor profile for /opt/gnome/lib/evolution-data-server-1.2/evolution-data-server-1.4

File Name	Permissions
#include abstractions/base	
#include abstractions/nameservice	
#include abstractions/user-tmp	
/etc/mtab	r
/etc/opt/gnome/**	r
/home/*/.evolution/addressbook/local/**	lrw
/home/*/.evolution/cache/calendar/**	lrw
/home/*/.evolution/calendar/local/**	lrw
/home/*/.evolution/tasks/local/**	lrw
/home/*/.gconf	r
/home/*/.gconf/**	lrw
/home/*/.gnome2_private	w
/opt/gnome/lib/GConf/**.so	r
/opt/gnome/lib/GConf/2/gconfd-2	pxr
/opt/gnome/lib/evolution-data-server-*/evolution-data-server-*	r
/opt/gnome/lib/evolution-data-server-*/extensions	r
/opt/gnome/lib/evolution-data-server-*/extensions/lib*.so	r
/opt/gnome/lib/gnome-vfs**	r
/opt/gnome/lib/lib*so*	r
/opt/gnome/share/evolution-data-server-*/**	r
/proc/meminfo	r
/proc/stat	r
/usr/X11R6/lib/X11/locale/*	r

Figure 1: AppArmor maintains a profile for each protected application.

prevent an intruder from killing other processes.

AppArmor on Linux

Novell's SLES 9 and Suse Linux 10.0 distributions have the AppArmor system integrated out of the box. AppArmor was not free at the time (see the "Immunix" box). Following the release of AppArmor under the GPL, Novell has now announced that it will be integrating AppArmor with OpenSUSE 10.1. If you prefer not to wait while this happens, you can use OpenSUSE 10.0, however, the installation is quite complex. Among other things you will need to modify and rebuild the kernel, so the update is not recommended for unexperienced users.

AppArmor RPMs for OpenSUSE 10.0 are available at Novell Forge [3]. Although Suse/Novell built the RPMs for OpenSUSE 10.1 Alpha, they also work on OpenSUSE 10.0. The installation follows the normal steps, *rpm -ivh packetname.rpm*. The kernel also requires AppArmor support. Novell has the required patches at [4]; the patches are designed for the original version 2.6.15 kernel [5]. To build an AppArmor-capable kernel, load both the original kernel and the patches *aa_2.0-2.6.15.patch* and *aa_namespace_sem-2.6.15.patch*. Then follow the steps in Listing 1.

It is also possible to install AppArmor on non-Suse systems, such as Debian or Fedora. However, this involves compiling the source code archives, and doing without a GUI, as a GUI would mean you running Yast 2.

Starting and Stopping

Suse has GUI-based controls for enabling AppArmor. Launch Yast, and select *AppArmor* in the left column. Then launch the AppArmor control bar on the right. This is where you can check the current AppArmor status and enable

Immunix

Novell acquired Immunix mid-1995. Immunix has specialized in the development of security solutions for years. The company's modified GCC, known as StackGuard, compiles applications in a way that prevents many types of buffer overflow exploit. To do this, StackGuard uses a so-called canary. This early-warning system generates random numbers when a program is launched. Before each call to a sub-program, it stores canary values on the stack. If the value has changed when the program returns, it quits the program, suspecting a buffer overflow. (The term canary comes from mining, where miners used canaries to warn them of carbon monoxide build-up.)

Immunix also led the development of the LSM interface (Linux Security Modules [2]) in kernel 2.6. This interface allows kernel modules to monitor security-critical events at various locations. Some security systems use LSM, such as LIDS (Linux Intrusion Detection System), and SELinux (Security Enhanced Linux). The latter was developed by NSA (National Security Agency, USA) and implements a MAC (Mandatory Access Control) system, which lets admins define detailed policies for access permissions. This restrictive set of policies can even monitor and restrict the superuser, root, and all of root's activities.

Whereas the current Novell/Suse distribution has kernel support for the SELinux program, it does not have the policies needed to make this work.

The AppArmor system is also by Immunix. Novell has positioned AppArmor as a simple and effective alternative to SELinux. Where SELinux is a comprehensive solution that requires a very complex configuration, AppArmor simply targets individual applications and critical events. At the end of January 2006, Novell released the source code for AppArmor under the GPL and immediately published the code [3] on its own website.

Profiles

The AppArmor package contains profiles for the following servers:

- Postfix
- Apache (in prefork mode)
- Squid
- OpenSSH server
- NTP server
- Name Service Caching Daemon (ncsd)
- Identd
- Protocol services Klogd and Syslogd

Profiles are also available for a variety of client programs:

- Acrobat Reader
- Ethereal
- Opera
- Firefox
- Evolution
- Gaim
- Realplayer
- Man
- Netstat
- Ping
- Traceroute

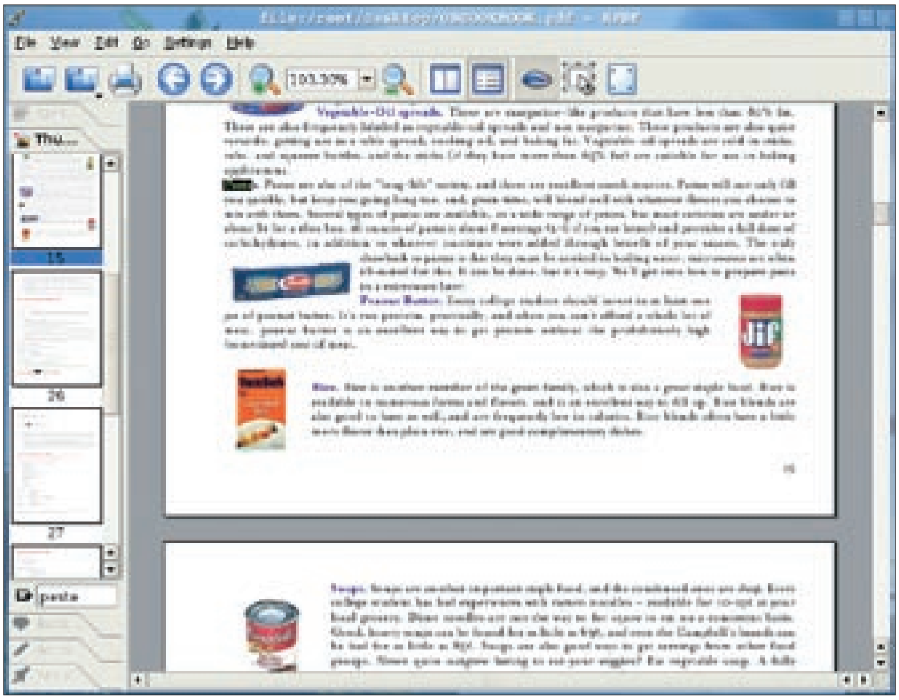


Figure 2: Kpdf displaying a PDF document. If the document is from an attacker, the attacker could exploit a vulnerability in the PDF

AppArmor. You can use the command line if you prefer: enter *rcsubdomain start* and *rcsubdomain stop* (working as root).

To allow AppArmor to work, the tool must be running before the protected applications are launched. This is why AppArmor is launched at boot time. AppArmor also needs a profile file in */etc/subdomain.d*for each application it is to protect.

Self-Protection

Novell has profiles for a large number of critical commands (see the "Profiles" box). I will be using the Kpdf file viewer (Figure 2) to show you how simple it is to generate a profile, thanks to AppArmor's learning mode.

In the last few years, a number of programming errors have been discovered in various PDF viewers, such as Xpdf and Kpdf. An attacker who knows about these errors could craft a PDF to inject and run malevolent code and thus take control of the PDF viewer.

Tip: Testing AppArmor

When you use the wizard to create a profile, do not include the print command. This means that the wizard will not have the function in the profile. When you run Kpdf later, you will note that everything works as expected, but that you can't print.

To add Kpdf to the list of programs monitored by AppArmor, launch Yast 2, and select the profile wizard below *AppArmor*. Start by entering the name of the application and its full path. If you do not know the path, you can type *which kpdf* to find it. The path on Suse Linux is */opt/kde3/bin/kpdf*.

Launch the application, and work with it for a while. Make sure you use all of Kpdf's features. But also make sure that an attack is impossible during the learning phase. AppArmor will later allow all the features that Kpdf uses now. After running through the full list of functions, you should quit the application. You can now analyze the recorded results in the profile wizard. Todo so, select *Scan system log for AppArmor events* (Figure 3).

Child Processes

After completing the event analysis, which can take a few minutes, the wizard asks you if you would like to allow all of these access types, suggesting an action in each case. If the monitored program calls another program, for example, the profile wizard gives you the following choices:

- Inherit: The same restrictions as with Kpdf are applied to the new application *kdialog*.
- Profile: This application has its own profile.
- Unconfined: AppArmor will not monitor this program.
- Deny: Stop the new application from launching

As Kpdf uses the *kdialog* program to open and close files, *Unconfined* is an option. As this gives the helper program complete freedom, the wizard warns about possible vulnerabilities (Figure 4). It might be better to create a profile for KDialog to restrict the program to accessing PDF files only.

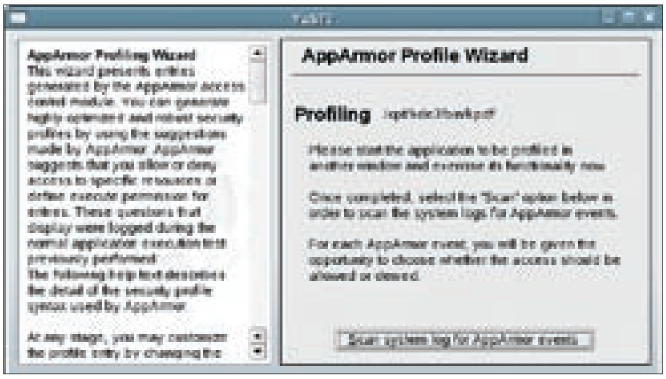
File Access

After making a decision for each application that Kpdf calls, the wizard asks about the files used by Kpdf. You can choose *Allow* to permit access to most files. The wizard has an include directive for certain files.

Listing 2: Including abstractionsv

```
# vim:syntax=subdomain
# Last Modified: Sun Jan 22 10:16:55 2006
/opt/kde3/bin/kpdf flags=(complain) {
  #include <abstractions/authentication>
  #include <abstractions/base>
  #include <abstractions/bash>
  #include <abstractions/gnome>
  #include <abstractions/kde>
  #include <abstractions/nameservice>
  #include <abstractions/user-write>

  / r,
  /etc r,
  /etc/X11/.kstylerc.lock rw,
  /etc/X11/.qt_plugins_3.3rc.lock rw,
  /etc/X11/.qtrc.lock rw,
  /etc/exports r,
  /etc/rpc r,
...
}
```



AppArmor Profiling Wizard
This wizard presents entries generated by the AppArmor access control module. You can generate highly optimized and robust security profiles by using the suggestions made by AppArmor. AppArmor suggests that you allow or deny access to specific resources or define execute permission for entries. These questions that display were logged during the normal application execution test previously performed.
The following help text describes the detail of the security profile syntax used by AppArmor.

At any stage, you may customize the profile entry by changing the

AppArmor Profile Wizard

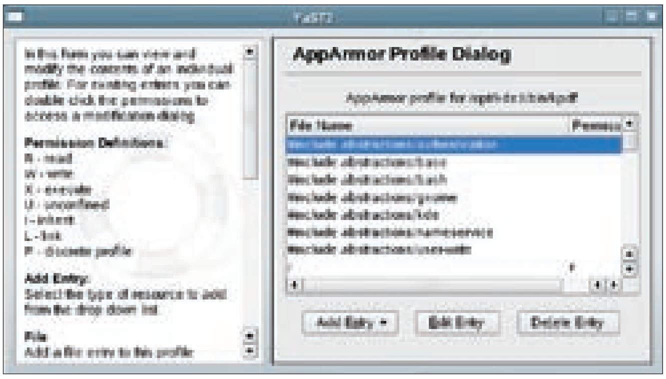
Profiling /opt/kde3/bin/kpdf

Please start the application to be profiled in another window and exercise its functionality now.

Once completed, select the "Scan" option below in order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the opportunity to choose whether the access should be allowed or denied.

Figure 3: AppArmor recording events for analysis.



In this form you can view and modify the contents of an individual profile. For existing entries you can double click the permissions to access a modification dialog.

Permission Definitions:
R - read
W - write
X - execute
U - unconfined
I - inherit
L - link
P - discrete profile

Add Entry:
Select the type of resource to add from the drop down list.

File
Add a file entry to this profile

AppArmor Profile Dialog

AppArmor profile for /opt/kde3/bin/kpdf

File Name	Permiss
#include abstractions/authentication	
#include abstractions/base	
#include abstractions/bash	
#include abstractions/gnome	
#include abstractions/kde	
#include abstractions/nameservice	
#include abstractions/user-write	

Figure 5: Yast 2 lets you edit an AppArmor profile.

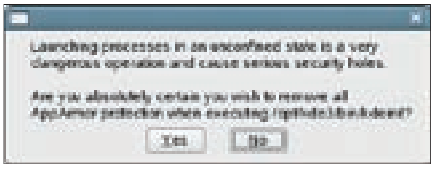
Many applications need access to the KDE configuration files. Instead of allowing access to every single file, and thus unnecessarily bloating the profile, you can simply add a profile template to your profile. To do so, select the *#include <abstractions/kde>* line. Profile templates are referred to as abstractions in AppArmor-speak.

AppArmor comes with a collection of additional abstractions, for the Bash shell and DNS name resolution, for example. After answering all the prompts, you are taken back to the profile wizard welcome screen. The profile is stored below */etc/subdomain.d/opt.kde3.bin.kpdf* (see Listing 2 for an excerpt). You can now close the profile wizard and quit the application.

Fine Tuning

If the application fails to perform as expected, just relaunch the profile wizard and repeat the learning process. The wizard first analyzes the existing profile and then updates the profile on changes. Any entries you added manually using a text editor are kept. After each manual change, you need to relaunch AppArmor to tell the tool to load the profile. As an alternative, you could use Yast 2, and either choose to update a profile or select the icon with the pen to edit a profile (Figure 5).

As network services are constantly exposed to danger, Novell gives you the *unconfined* program, which discovers the network services running on your system and displays their AppArmor status. The output in Listing 3 shows that this system is running CUPS and that the RPC portmapper is not being monitored. Novell does not have profiles for these services.



Launching processes in an unconfined state is a very dangerous operation and can cause serious security holes.

Are you absolutely certain you wish to remove all AppArmor protection when executing /opt/kde3/bin/kdeinit?

Figure 4: When you select Unconfined, the wizard warns you about the potential security risk.

Over the next few weeks and months, you can expect Novell to keep on churning out profiles. If you are interested in keeping track of developments, check out the mailing list at [6], and drop by the AppArmor homepage [1] occasionally.

Well-Protected

AppArmor monitors critical applications. A program is only permitted to access specific files and call specific commands. If the application has a security hole that might allow an attacker to launch a shell or other commands with the victim's privileges, AppArmor steps in to protect the system. The application runs in a kind of sandbox, or jail, and is incapable of breaking out.

AppArmor cannot prevent vulnerabilities, but it can prevent attackers from exploiting them. This effectively protects users from the symptoms of new, previously unknown attacks. AppArmor is thus highly recommended for programs that are accessible via the network or that handle data from untrusted sources, such as emails, images, videos, or office documents. ■

Listing 3: Displaying AppArmor status

```
# unconfined
7988 /usr/lib/postfix/master
   confined by '/usr/lib/postfix/
   master (enforce)'
7988 /usr/lib/postfix/master
   confined by '/usr/lib/postfix/
   master (enforce)'
8025 /usr/sbin/cupsd not
   confined
8025 /usr/sbin/cupsd not
   confined
8081 /sbin/portmap not
   confined
8081 /sbin/portmap not
   confined
8109 /usr/sbin/sshd confined
   by '/usr/sbin/sshd (enforce)'
```

INFO

[1] AppArmor: *http://www.opensuse.org/AppArmor*

[2] LSM: *http://lsm.immunix.org*

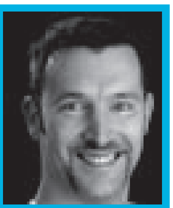
[3] AppArmor packages: *http://forge.novell.com/modules/xfcontent/downloads.php/apparmor/Stable/*

[4] Kernel patches for AppArmor: *http://forge.novell.com/modules/xfcontent/downloads.php/apparmor/Development/*

[5] Kernel repository: *http://www.kernel.org*

[6] AppArmor mailing list: *http://forge.novell.com/mailman/listinfo/apparmor-general*

THE AUTHOR



Ralf Spenneberg is a freelance Unix/Linux trainer, consultant, and author. Ralf's business, OpenSource Training, offers training and consultancy services. Ralf has published a number of books on the topics of intrusion detection and virtual private networks.