# Novell AppArmor (2.0.1) Quick Start

Этот документ поможет вам понять концепцию работы основы Novell AppArmor - содержимого профилей AppArmor. Вы узнаете, как создавать или изменять профили AppArmor. Вы можете создавать профили или управлять ими тремя различными способами. Самый удобный интерфейс к AppArmor обеспечивается с помощью модулей AppArmor для YaST, которыми можно пользоваться в графическом или ncurses виде. Точно такая же функциональность обеспечивается при использовании командной строки AppArmor или при редактировании профилей в текстовом редакторе.

## 1.0 Режимы AppArmor

**complain/learning**

В жалующемся или обучающемся режиме определяются нарушения правил профиля AppArmor, такие как, доступ к файлам со стороны профильных программ, не разрешенный профилем. Нарушения разрешены, но тут же регистрируются. Это метод удобен для разработки профилей и используется инструментами AppArmor для создания профилей.

Для активизации обучающего режима вручную (при использовании командной строки), добавьте флаг наверх профиля таким образом, чтобы `/bin/foo` стало `/bin/foo=(complain)`.

**enforce**

Загрузка профиля в принудительном режиме предписывает политике, определенной в профиле, отправлять сообщения о нарушениях в *syslogd*.

## 2.0 Запуск и остановка AppArmor

Используйте команду **rcapparmor** с одним из следующим параметров:

**start**

Загружает модуль ядра, монтирует *securityfs*, анализирует и загружает профили. Профили и заключение начинают применяться к любому приложению, после того, как оно будет запущено. Процессы, запущенные ранее старта AppArmor продолжают быть неограниченными.

**stop**

Размонтируется *securityfs*, профили становятся недействительными.

*Translate by Hrafn (hrafngunnarson@gmail.com)*

**reload**

Перезагружаются профили.

**status**

Если AppArmor запущен, выводится информация о количестве загруженных профилей, и в каком режиме они работают.

Используйте команду `rcaaeventd` для управления событиями, регистрирующимися с помощью **aa-eventd.** Используйте опции **start** и **stop** для переключения статуса **aa-eventd** и проверьте его статус с помощью **status**.

## 3.0 Инструменты командной строки AppArmor

**autodep**

Примерные основные требования профиля AppArmor. **autodep** создает приблизительные профиль для программы или рассматриваемого приложения. Итоговый профиль называется приблизительным по той причине, что совсем необязательно содержит все входы профиля, необходимые для заключения программы должным образом.

**complain**

Устанавливает профиль AppArmor в обучающий режим.

**enforce**

Переводит профиль в принудительный режим из обучающего.

**genprof**

Генерирует профиль. При запуске вы должны указать программу для профиля. Если указан не полный путь до программы, **genprof** ищет переменную `$PATH`. Если профиля не существует, **genprof** создает его, используя **autodep**.

**logprof**

Управляет профилями AppArmor. **logprof** - это диалоговый инструмент, использующийся для рассмотрения вывода обучающего или жалующегося режимов, найденных во входах syslog AppArmor и генерирования новых входов в профилях AppArmor.

**unconfined**

Выводит список процессов с портами tcp и udp, которые не имеют загруженных профилей AppArmor.

# 4.0 Методы профилирования

**Автономное профилирование**

Используется genprof. Подходит для профилирования небольших приложения.

**Систематическое профилирование**

Подходит для  профилирования большого количества программ за один раз и для профилирования приложений, которые могут быть запущены всегда.

Чтобы использовать систематическое профилирование, сделайте, как показно ниже:

1. Создайте профили для ваших программ, котрые составляют ваше приложение (**autodep**).
2. Поместите нужные профили в обучающий или жалующийся режим.
3. Развивайте ваше приложение
4. Проанализируйте логи (**logprof**)
5. Повторите шаги 3-4
6. Отредактируйте профили
7. Вернитесь в принудительный режим
8. Повторно просканируйте профили (**rcapparmor restart**)

# 5.0 Обучающий режим

Используя genprof, logprof или YaST в обучающем режиме, вы получаете несколько вариантов выбора для продолжения:

**Allow**

Предоставить доступ.

**Deny**

Запретить доступ.

**Glob**

Изменить путь до директории путем включения всех фалов предложенной директории.

**Glob w/Ext**

Измените оригинальный путь к директории, сохраняя расширение имени файла. Это позволит программе обращаться в предложенной директории ко всем файлам, которые заканчиваются указанным расширением.

**Edit**

Разрешить редактирование выделенной линии. Новая (отредактированная) линия появится в конце списка. Эта опция называется **New** в инструментах командной строки **logprof** и **genprof**.

**Abort**

Прервать logprof или YaST. При этом произойдет потеря всех изменений в правилах, правила останутся неизмененными.

**Finish**

Закрыть logprof или YaST. Сохраняются все изменения, сделанные в правилах, и правила изменяются.

## 6.0 Примерный профиль

```
# a variable definition

@{HOME} = /home/*/ /root/

# a comment about foo.
/usr/bin/foo {
  /bin/mount           ux,
  /dev/{,u}random      r,
  /etc/ld.so.cache     r,
  /etc/foo.conf        r,
  /etc/foo/*           r,
  /lib/ld-*.so*        mr,
  /lib/lib*.so*        mr,
  /proc/[0-9]**        r,
  /usr/lib/**          mr,
  /tmp/foo.pid         wr,
  /tmp/foo.*           lrw,
  /@{HOME}/.foo_file   rw,

  # a comment about foo's subprofile, bar.
  ^bar {
    /lib/ld-*.so*        mr,
    /usr/bin/bar         px,
    /var/spool/*         rwl,
  }
}
```

# 7.0 Структура профиля

Профили – это простые текстовые файлы в директории `/etc/apparmor.d`. Они состоят из нескольких частей: **#include**, **capability entries**, **rules** и **«hats»**.

## 7.1 #include

Эта секция профиля AppArmor, обращающаяся к файлу включения, определяет права доступа для программ. Используя включение, вы можете предоставить программе доступ к пути директории и файлам, которые также требуются другим программам. Используя включения может сократить размер профиля. Это хороший способ выбрать включения, когда это будет предложено.

Чтобы помочь вам в профилировании ваших приложений, AppArmor обеспечивает три класса #includes: абстракции (abstractions), программные блоки (program chunks)и переменные (variables).

Абстракции - #includes, которые сгруппированы, как общие прикладные задачи. Эти задачи включают в себя доступ к механизмам аутентификации, доступ к определению имени сервиса, общим граифческим требованиям и к учетным системным записям, например, база, консоли, Kerberosclient, perl, почта пользователя, временные файлы пользователя, аутентификация, bash и сервисы разрешения имен.

Программные блоки – это контроль доступа для определенных программ, которым системный администратор, возможно, хотел бы управлять, основываясь на локальной политике. Каждый блок используется одной программой.

Используя переменные, вы можете создавать такой дизайн своих профилей, что они могут быть использованы в различной окружающей среде. Изменения содержимого переменной делаются только в определении переменной, в то время, как профиль, содержащий переменную, остается нетронутым.

## 7.2 Capability entries (POSIX 1.e)

Завяления capabilities – это просто слово «capabilities», завершающее название **POSIX 1.e capabilities**, как описано в man-странице `capabilities (7)`.

## 7.3 Правила: Главные опции для файлов и директорий

| **Опция** | Файл |
|---|---|
| чтение | r |
| запись | w |
| ссылка | l |

## 7.4 Правила: Определение прав на выполнение

Для исполняемых программ, которые могут быть вызваны из ограниченных программ, профиль создающий инструменты, спросит вас о необходимом способе, котрый также напрямую будет отражен непосредственно в профиле:

| **Опция** | Файл | Описание |
|---|---|---|
| Inherit | ix | Оставаться в таком же (родительском) профиле |
| Profile | px | Требуется, что существовал профиль для отдельно исполняемой программы. Нет очистки окружения |
| Profile | Px | Требуется, что существовал профиль для отдельно исполняемой программы. Есть очистка окружения |
| Unconstrained | ux | Выполняется программа без профиля. По причинам безопасности избегайте выполнение программ в добровольном или неограниченном режиме. Нет очистки окружения |
| Unconstrained | Ux | Выполняется программа без профиля. По причинам безопасности избегайте выполнение программ в добровольном или неограниченном режиме. Этот метод использует очистку окружения |
| Allow Executable Mapping | m | Позволяется вызов PROT_EXEC с использованием mmap(2) |

Внимание: Запуск в режиме **ux**.

Как можно чаще избегайте запуска программ в режиме ux. Программа, запущенная в ux-режиме, не только никак не защищена AppArmor, но и дочерние процессы наследуют определенные переменные режима от родителя, которые могут повлиять на выполнение потомка и создать опасность нарушения безопасности.

За более подробной информацией по поводу других режимов выполнения файлов обратитесь на man-страницу `apparmor.d (5)`. За более конкретной информацией о setuid и setgid очистки окружения ображайтесь к man-странице `ld.so (8)`.

## 7.5 Правила: Пути и Globbing

| Glob | Описание |
|---|---|
| * | Замена любого количества знаков, кроме /. |
| ** | Замена любого количества знаков, включая /. |
| ? | Замена любого одиночного знака, кроме /. |
| [ abc ] | Замена одного знака a, b или c |
| [ a-c ] | Замена одного знака a, b или c |
| { ab, cd } | Расширение к одному правилу. для соответствия ab, и к другому, для соответствия cd |

## 7.6 Hats

Профиль AppArmor представляет собой политику безопасности для конкретной программы или процесса. Он применяется к исполняемой программе, но если какой-то части программы требуется отличные права доступа, чем другой части программы, программа может изменить «шляпы» (**hats**) для использования другого контекста безопасности, котрые отличается от прав доступа основной программы. Она известна, как «шляпа» (**hats**) или подпрофиль.

Профиль может иметь произвольное количество подпрофилей, но только двух уровней: подпрофиль не может иметь подподпрофиль.

Особенность **AppArmor ChangeHat** может использоваться приложениями для обращения к hats или подпрофилям в процессе выполнения. В настоящее время пакеты **apache2-mod_apparmor** и **tomcat_apparmor** используют **ChangeHat** для обеспечения заключения подпроцесса веб-сервера Apache и контейнера сервлета Tomcat.

# 8.0 Полезные дополнения

## 8.1 Автодокументация

Инструмент «sitar» собирает всю доступную информацию о конфигурации вашей системы и создает всестороннюю системную документацию. Он может использоваться для документирования всех новых и измененных профилей.

## 9.0 Логирование и аудит

Все, что происходит с AppArmor, регистрируется с использованием интерфейса аудита системы (**auditd** записывает логи в `/var/log/audit/auditd.log`). В довершение этой инфраструктуры уведомление о событии может быть настроено. Настройте эту фичу, используя YaST. Она основано на уровнях строгости , согласно `/etc/apparmor/severity.db`. Частота и способ уведомления (например, e-mail) тоже может быть настроено.

Если **auditd** не запущен, AppArmor записывает логи в файл системных логов, расположенный в `/var/log/messages`, посредством `LOG_KERN`.

Используйте YaST для создания сообщений в форматах CVS и HTML.

## 10.0 Директории и файлы

**/sys/kernel/security/apparmor/profiles**

Виртуальный файл, отображающий загруженный в данное время набор профилей.

**/etc/apparmor/**

Расположение конфигурационных файлов AppArmor.

**/etc/apparmor.d/**

Расположение профилей, в имени пути которых заменяется «/» на «.» (это не касается корневого каталога /), что делает профили более легкими для управления. Например, профиль для программы `/usr/sbin/ntpd` назван `usr.sbin.ntpd`.

**/etc/apparmor.d/abstractions/**

Место расположения абстракций.

**/etc/apparmor.d/program-chunks/**

Место расположения программных блоков.

**/proc/*/attr/current**

Рассматривается статус заключения процесса и профиль, который используется для заключения процесса. Команда `ps auxZ` автоматически восстанавливает эту информацию.

## 11.0 Больше информации

Чтобы узнать больше о проекте AppArmor, зайдите на домашнюю страницу проекта http://en.opensuse.org/AppArmor. Больше информации о концепции и настройке AppArmor можно найти в **Novell AppArmor Administration Guide**.

*Translate by Hrafn (hrafngunnarson@gmail.com)*