Денис Колисниченко

Grsecurity — ваша личная охрана

За безопасность нужно платить, а за ее отсутствие — расплачиваться. В этой статье речь будет идти о системе управления доступом, которая является дополнительным барьером для злоумышленника.

Сегодня мы займемся безопасностью вашей Linux. Конечно, она и так считается одной из самых надежных операционных систем, но все же и ее взламывают. Мы рассмотрим одну из самых популярных систем управления доступом — Grsecurity. Почему наш выбор пал именно на нее, ведь Grsecurity — далеко не единственная система управления доступом, существуют также LIDS, SELinux и RSBAC.

Самой строгой из всех перечисленных систем считается SELinux, но она и самая сложная в настройке, поскольку работа с ней подразумевает наличие у администратора определенного багажа знаний. Если вам нужно описание SELinux, в Интернете есть много HOWTO по ее установке на русском языке (http://dkws.narod.ru/howto.html).

Про LIDS написано много статей, руководств, ее описание можно встретить во многих книгах по Linux. И это неудивительно: LIDS была, наверное, самой популярной системой безопасности до появления SELinux. В любом случае найти хорошее описание LIDS особого труда не составит. Grsecurity — очень мощная система, но не так хорошо описанная, как LIDS и SELinux. Вот именно поэтому мы ее и рассматриваем.

Может, у вас возник закономерный вопрос: если SELinux — самая строгая система безопасности, то зачем же рассматривать Grsecurity и LIDS? Ведь проще сразу установить SELinux. Конечно, это так, но SELinux — самая строгая только при условии правильной настройки. Если ее недостаточно умело отрегулировать, например, если у администратора не хватает опыта, то толку от этой системы не будет. Напротив, при правильной настройке (которая намного проще) системы Grsecurity и LIDS обеспечивают примерно такой же уровень безопасности, что и SELinux.

Прежде чем приступить к рассмотрению Grsecurity, поговорим о том, что такое вообще система управления доступом и как она должна работать.

Управление доступом — зачем оно нужно?

Что же не устраивало разработчиков систем управления доступом в обычной Linux? В этой ОС есть два типа пользователей — администратор и все остальные. Права обычных пользователей можно ограничить и с помощью штатных средств операционной системы. Но что может делать администратор — вы сами знаете. Если злоумышленник завладеет паролем администратора (как он это сделает — это уже другой вопрос), то получит полную власть над системой. А вот

если на вашем компьютере установлена система управления доступом, то она не позволит ему сделать ничего, в крайнем случае он сможет произвести только те операции, которые не причинят системе ощутимого вреда.

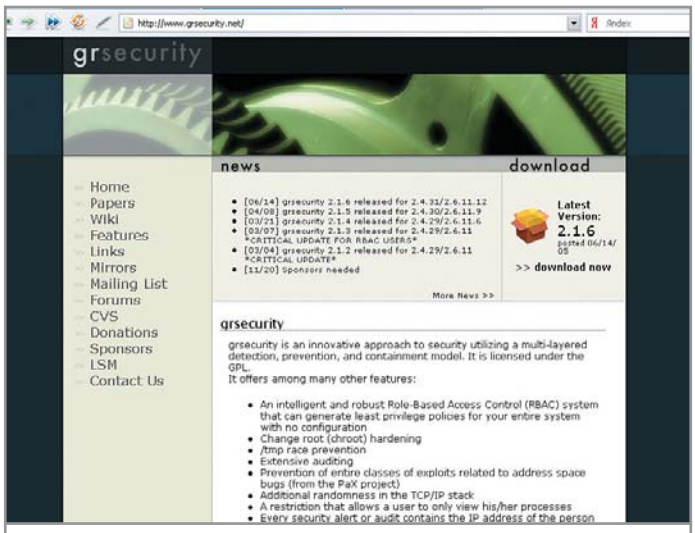
Теперь вернемся к обычным пользователям. Их права ограничиваются, как правило, только на доступ к файлам. Также можно задать ограничение на использование системных ресурсов — дискового пространства (квоты), процессорного времени, установить максимальное число процессов. И все. Система управления доступом (СУД) может запретить пользователю выполнять кое-какие действия, то есть вы можете задать список программ, с которыми вправе работать тот или иной пользователь, а остальные приложения при попытке их запуска будут блокироваться. Во многих случаях даже обычным пользователям часто предоставляются чрезмерные полномочия. Например, зачем пользователю, который зарегистрировался в системе только для чтения почты, возможность компиляции исходного кода или запуска фоновых демонов? Также СУД решает проблему SUDO (когда отдельным пользователям предоставляются все полномочия root) — администратор может определить для каждого пользователя действия, которые он может выполнять с полномочиями root.

В Unix вы можете использовать одну из следующих моделей:

- Дискретное управление доступом (Discretionary Access Control, DAC). С данной системой мы все давно знакомы. Это обычная система пользователей и прав доступа, которая присутствует по умолчанию во всех Unix-системах. Здесь доступ к объекту (например, файлу) регулируется его владельцем с помощью прав доступа. Владелец может контролировать не только тех пользователей, которым положен доступ к объекту, но и режимы доступа (например, это могут быть чтение, запись или выполнение).
- Принудительное управление доступом (Mandatory Access Control, MAC). В этой системе доступ к объекту контролирует не его владелец, а администратор системы. Здесь власть администратора гораздо больше, чем в предыдущей модели. На практике в чистом виде MAC не существует — это только теоретическая модель, но на ее основе строятся некоторые системы управления доступом.
- Модель «тип-домен» (Domain Type Enforcement, DTE) основана на MAC, но с учетом концепции минимальных привилегий: процессу должны быть предоставлены минимально необходимые привилегии. В DTE объекты (файлы) формируют типы, а субъекты (процессы) — домены. Таблица DDT (Domain Definition Table — таблица определения домена) описывает, как домены и типы могут взаимодействовать друг с другом.

Кроме этих трех моделей в Unix часто используются списки управления доступом (Access Control List, ACL), управления доступом на базе ролей (Role-based Access Control, RBAC), возможности и модули защиты (Linux Security Modules, LSM).

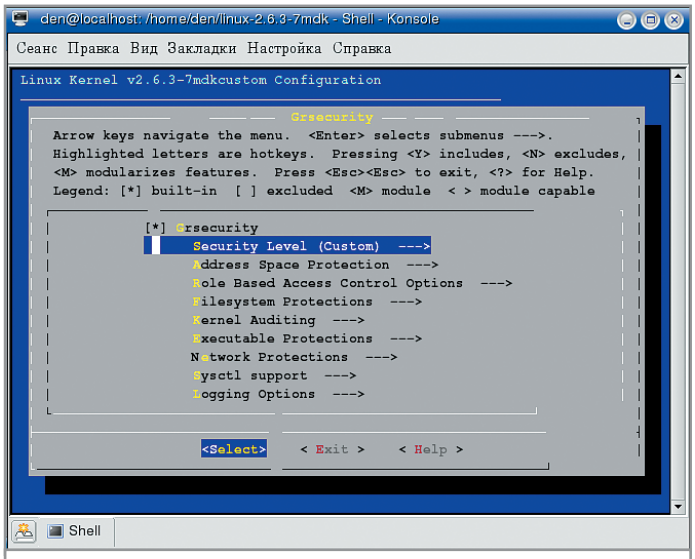
С ACL вы наверняка не раз сталкивались при конфигурировании системы. Поэтому перейдем сразу к RBAC. В этой системе всем пользователям назначены одна или несколько ролей, которые он может выполнять в системе. Роль — это действия, которые разрешены пользователю. Например, администратор может назначить пользователя, ответственного за интернет-сервер — обычно это веб-мастер. Ему администратор разрешает изменение файла конфигурации, чтение протоколов, запуск, остановку и перезапуск сервера Apache, а также изменение каталога /var/www/html (как правило, это DocumentRoot).



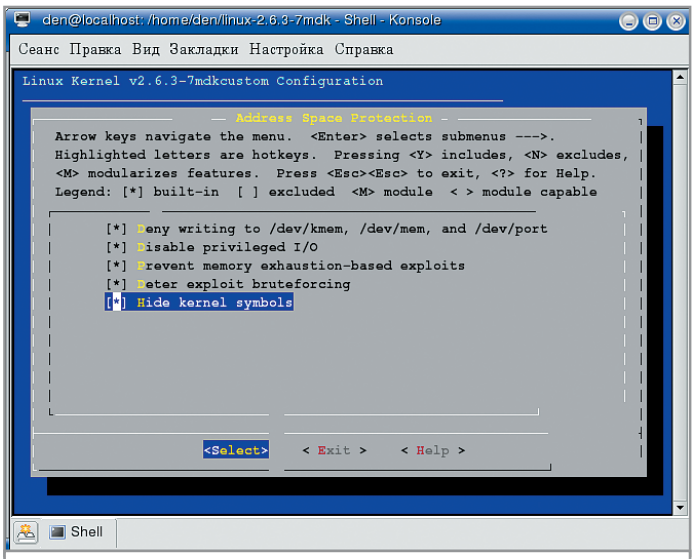
Домашняя страница проекта Grsecurity — www.grsecurity.net

Возможности (capabilities) появились давно — начиная с версии ядра 2.1, но мало кто с ними работал. Каждая возможность — это какое-то одно привилегированное действие пользователя root, например CAP_KILL — право «убивать» процессы других пользователей. Некоторым пользователям (которые должны выполнять администраторские функции) в зависимости от их нужд назначаются отдельные списки возможностей, то есть определяются действия, которые они могут выполнять от имени root. На данный момент Linux поддерживает 28 возможностей, семь из которых соответствуют POSIX-стандарту. Полный их список приведен в файле /usr/include/linux/capability.h.

Модули защиты (LSM) — это сравнительно молодое средство управления доступом. Его история началась с разработки различных систем управления доступом. Все они реализовывались в виде патчей ядра Linux. Но из-за отсутствия централизованной координации в результате получилось, что у каждого проекта защиты был свой патч для ядра, часто не совместимый с другими патчами. В прошлом Линус Торвальдс категорически отвергал все эти патчи, но в 2001 году в ответ на технологию SELinux, представленную NSA (Агентство национальной безопасности) на Linux Kernel Summit, он отметил, что для включения в ядро он будет рассматривать более обширную систему безопасности. Так появился проект LSM (Linux Security Modules). Его цель — предоставить разработчикам систем защиты общий (единый) интерфейс для реализации проектов, основанных на ядре. Реализация такой системы позволит меньше зависеть от ядра и не будет требовать его перекомпиляции, как в случае с патчами. Все LSM-модули используют для взаимодействия с ядром стандартный интерфейс, который не будет изменяться с выходом следующего релиза (не версии) ядра.



Опции ядра Grsecurity



Опции защиты памяти

Модули защиты LSM — это не система управления доступом, фактически они представляют собой интерфейс, созданный для взаимодействия различных систем управления доступом и ядром. LSM предоставляет только структуру для реализации системы безопасности, воплощением же самой системы занимаются модули. Такой подход снимает проблему несовместимости между патчами отдельных разработчиков: LSM — один для всех, а модули могут быть совершенно разными у каждого разработчика.

В Linux 2.6 поддержка LSM обеспечивается по умолчанию (ее нужно только включить). Для того чтобы это сделать, в меню «Security Options» активируйте опцию «Enable Different Security Models».

Введение в Grsecurity

По понятным причинам мы не сможем рассмотреть все возможности Grsecurity, но надеемся, что приведенной ниже информации вполне достаточно для того, чтобы вы смогли установить и настроить эту систему.

Основная цель Grsecurity — сократить до минимума конфигурацию системы, поскольку сложные системы часто настраиваются неправильно, что приводит к возникновению потенциальных дыр в системе безопасности.

Вот некоторые усовершенствования, вносимые Grsecurity в вашу систему:

- улучшенное chroot-окружение, не позволяющее процессу выйти за его пределы;
- OpenBSD-рандомизация TCP ISN (номер последовательности TCP-пакета) и PID (Process ID);
- ACL (MAC);
- RBAC;
- защита стека PaX.

Для установки Grsecurity вам придется перекомпилировать ядро — Grsecurity реализована виде патча ядра (судя по всему, в ближайшее время Grsecurity поддерживать LSM не будет), поэтому вам нужно загрузить патч для вашего ядра с сайта, расположенного по адресу www.grsecurity.net, пропатчить и перекомпилировать ядро. Кстати, пропатчить ядро можно с помощью команды:

```
/usr/src]# patch -p0 < имя_файла_патча
```

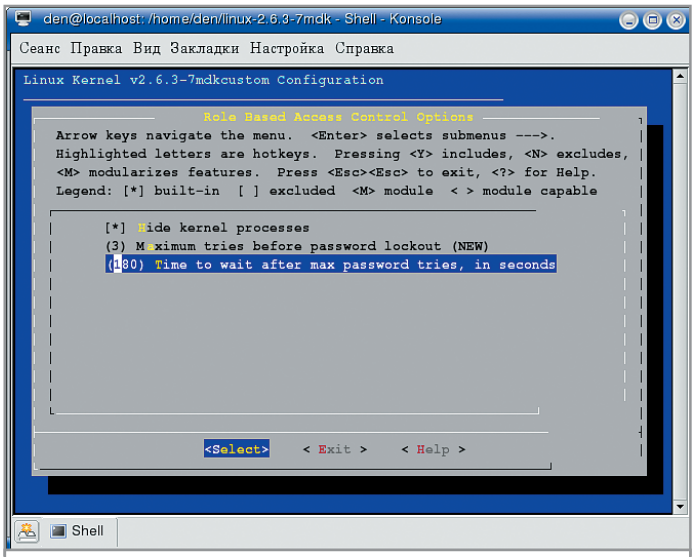
Имя файла патча мы специально не указываем, чтобы вы «по образу и подобию» не скачали патч для ядра автора. Вам нужно скачать патч для своего ядра — номер версии патча должен совпадать с версией ядра.

После применения патча запустите команду menuconfig (make menuconfig). В меню «Security Options» появятся опции Grsecurity. Включите их и перекомпилируйте как обычно ядро. Не исключено, что на упомянутом сайте не окажется патча именно для вашей версии. Тогда вам нужно будет загрузить с сайта www.kernel.org исходные коды того ядра, которое можно использовать с Grsecurity.

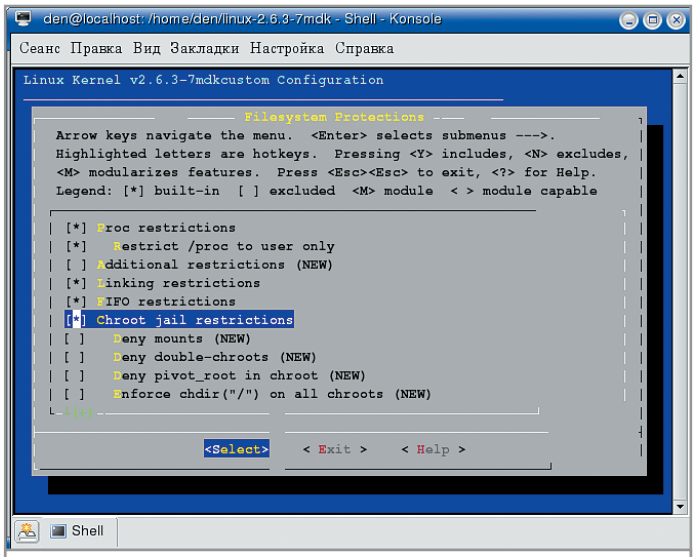
Мы рассмотрим почти все опции ядра Grsecurity, но сначала хотелось бы сделать одно важное замечание о группах пользователей. С помощью групп Grsecurity позволяет управлять ограничениями, которые накладываются на права входящих в группу пользователей. Идентификаторы групп (GID) могут выбираться произвольно, но по умолчанию Grsecurity использует идентификаторы в диапазоне от 1001 до 1005 включительно. Например, для того чтобы запретить пользователям создавать сетевые соединения, может применяться опция ограничения сокетов — по умолчанию для этой возможности используется GID 1004. Пользователи, входящие в группу с GID 1004, не смогут создавать сетевые соединения. Чтобы добавить пользователя в данную группу, необходимо ввести следующую команду:

```
# usermod -G users,1004 den
```

Для быстрой настройки Grsecurity вы можете выбрать один из трех уровней безопасности: низкий (low), средний (medium) и высокий (high). Подсказка подробно объяснит, какие функции будут включены в том или ином случае. Кроме того, можно настроить пользовательский (custom) уровень и установить все опции вручную. Мы, например, пойдем по пути «максимального сопротивления» и выберем именно пользовательский уровень.



Опции RBAC



Защита файловой системы

Address Space Protection (Защита адресного пространства)

Опции данной группы используются для защиты памяти. Вы можете включить все опции кроме CONFIG_GEKERNSEC_IO, поскольку она не совместима с XFree86.

Deny writing to /dev/kmem, /dev/mem and /dev/port. Символьные устройства /dev/mem и /dev/kmem позволяют пользователям уровня root непосредственно читать и записывать соответственно в системную память и память ядра. Процесс записи сам по себе обычно очень опасен, поскольку злоумышленник может использовать запись для загрузки модулей ядра прямо в память, даже если администратор отключил поддержку загружаемых модулей. Устройство /dev/port предоставляет прямой доступ к портам ввода/вывода, что также нежелательно. Данная опция не совместима с VMWare, поэтому, если вы используете этот эмулятор, не включайте ее.

Disable Privileged I/O. Данная опция позволяет отключить привилегированный ввод/вывод, то есть отключает системные вызовы ioperm() и iopl(). Опция не совместима с XFree86.

Remove Address from /proc/<pid>/[maps|stat]. Псевдофайловая система /proc предоставляет много информации о запущенных процессах. Злоумышленник может использовать ее в своих целях, поэтому желательно отключить меппинг памяти. При этом вам нужно использовать PaX. О том, что это такое, вы сможете прочитать на сайте http://pax.grsecurity.net. Пока не включайте эту опцию. Сделаете это тогда, когда узнаете, что такое PaX.

Hide Kernel Symbols. Скрывает символьную информацию ядра. Пользователи уровня root не смогут получить информацию о загруженных модулях и символах ядра, что позволяет защитить систему от атак памяти (например, переполнения буфера). Включите эту опцию.

Role Based Access Control (RBAC) Options (Опции RBAC)

При конфигурировании RBAC доступны следующие опции:

Hide Kernel Processes. Скрывает процессы ядра от обычных пользователей, но не от администратора. Включите эту опцию.

Maximum Tries Before Password Lockout. Устанавливает количество попыток возможного ввода неправильного пароля. По умолчанию число попыток равняется трем. Если пользователь троекратно введет неправильный пароль, учетная запись будет заблокирована на определенный период времени, указанный в следующей опции.

Time to Wait After Max Password Tries, in Seconds. По умолчанию время блокировки составляет 30 секунд, по истечении которых пользователь опять сможет вводить пароль. Не устанавливайте слишком большое число, поскольку злоумышленник сможет использовать его для блокировки учетной записи.

Filesystem Protection (Защита файловой системы)

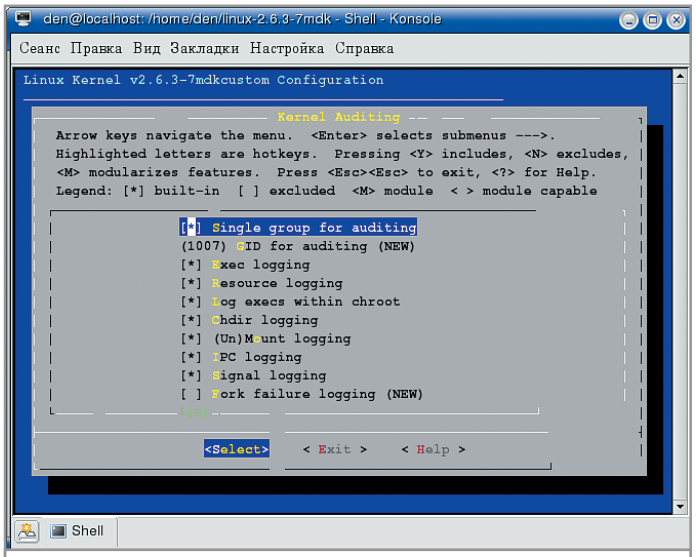
Proc Restrictions. Опция позволяет управлять доступом к /proc двумя способами.

- **Restrict /proc to User Only**: обычные пользователи не смогут просматривать /proc-информацию о процессах других пользователей, а также данные о сети и ядре.
- **Allow Special Group**: создается специальная группа, которой можно просматривать /proc-информацию. GID этой группы должен быть указан в опции «GID for Special Group» (CONFIG_GRKERNSEC_PROC_GID).

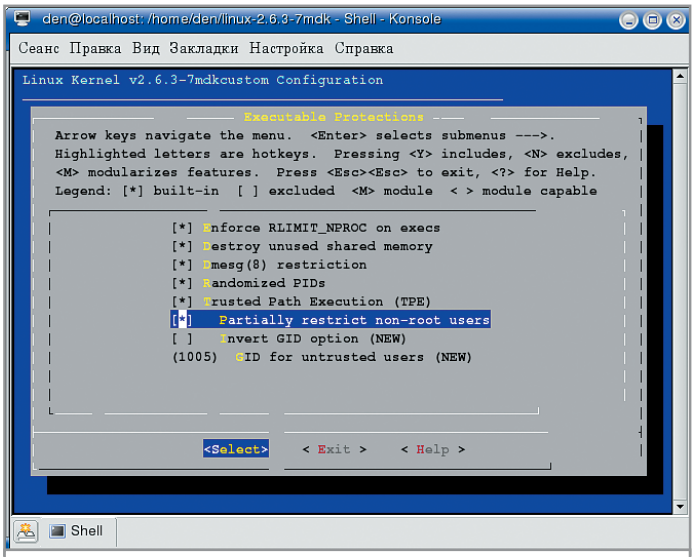
Additional Restrictions. Опция добавляет дополнительные ограничения, запрещающие пользователям читать информацию о процессоре и устройствах (запрещает доступ к файлам /proc/cpuinfo и /proc/devices). Включите эту опцию: чем меньше информации окажется у злоумышленника, тем сложнее ему будет взломать систему.

Linking Restrictions. Если вы включите эту опцию, пользователь не сможет создавать жесткие ссылки на файлы, которые ему не принадлежат, а также переходить по символической ссылке, которая принадлежит другому пользователю.

FIFO Restrictions. FIFO (First In, First Out) — специальный тип потока, используемого для обмена данными между процессами. Включение этой опции запретит пользователям делать записи в FIFO из не принадлежащих им каталогов.



Аудит ядра



Защита исполняемых файлов

chroot Restrictions. chroot-окружение используется для организации «песочницы» — отдельной области файловой системы, к которой и только к которой процесс, запущенный непосредственно в «песочнице», может получить доступ. По сравнению с OpenBSD/FreeBSD chroot-окружение Linux довольно слабое (процесс, получивший права root или запущенный с этими правами, может легко выйти за его пределы), поэтому данная опция позволяет его несколько укрепить. Рассматриваемые далее опции помогут создать дополнительные ограничения для chroot-окружения. Мы, в свою очередь, можем только порекомендовать включить все опции, связанные с chroot-окружением.

Kernel Auditing (Аудит ядра)

Опции данного меню позволяют протоколировать определенные системные вызовы, например execve() и fork().

Single Group for Auditing. Grsecurity может протоколировать действия пользователя (в частности, exec, chdir, mount/unmount и IPC). Будут протоколироваться действия не всех пользователей, а только тех, что входят в группу, GID которой указан в опции GID for Auditing (CONFIG_GRKERNSEC_AUDIT_GID). Чтобы протоколировать действия конкретного пользователя, его нужно добавить в эту группу.

Exec Logging. Протоколирует все системные вызовы execve(), осуществленные пользовательским процессом. Вы сможете отслеживать все программы, которые запускаются пользователем.

Resource Logging. Протоколирует попытки превысить лимит ресурса, например максимальное число процессов.

Log execs Within chroot. Протоколирует вызовы execve() в пределах chroot-окружения.

Chdir Logging. Протоколирует вызовы chdir().

(Un)Mount Logging. Протоколирует монтирование и размонтирование файловых систем.

IPC Logging. Протоколирует разделяемую память, семафоры и очереди сообщений — все это относится к взаимодействию между процессами.

Signal Logging. Данная опция протоколирует поступление важных сигналов, например SIGSERV (segmentation Fault), которые могут сообщать о попытке взлома какого-либо приложения.

Fork Failure Logging. Протоколирует неудачные попытки системного вызова fork(). Как правило, вызов fork() проваливается, если превышен лимит ресурса. Это может быть признаком атаки типа fork bomb, когда злоумышленник вызывает огромное количество fork, чтобы помешать системе запустить законный процесс.

Time Change Logging. Протоколирует изменения, происходящие в системных часах.

Executable Protection (Защита исполняемых файлов)

Enforce RLIMIT_NPROC on exec. RLIMIT_NPROC (конфигурируется через /etc/limits) позволяет ограничить ресурсы пользователя. По умолчанию они проверяются только во время вызова fork(). Включение данной опции позволяет производить проверку ресурсов во время вызова execve().

Dmesg Restriction. Включение данной опции позволяет выполнять программу dmesg только пользователю root.

Randomized PIDs. ID процессов будет назначаться не последовательно, а произвольно, что не позволит злоумышленнику предсказать PID демонов и других процессов.

Trusted Path Execution. Данная опция не позволит пользователю случайно запустить исполняемый файл (возможно, троян), записанный каким-либо другим пользователем. То есть ему разрешается выполнение программ только из тех каталогов, владельцем которых является непосредственно пользователь root.

Network Protection (Защита сети)

Larger Entropy Pools. Данная опция позволяет увеличить в два раза размер пулов, которые используются Grsecurity и многими другими приложениями. Данная опция должна быть обязательно включена.

Truly Random TCP ISN Selection. Если вы включите эту опцию, то стандартный способ генерации ISN, используемый в вашей операционной системе Linux, будет заменен по-настоящему случайным способом генерации ISN, который используется обычно в OpenBSD.

Randomized IP IDs. При включении данной опции рандомизации подвергнутся ID пакетов, которые передаются системой. По умолчанию Linux просто увеличивает ID каждого следующего пакета.

Randomized TCP Source Port. Опция, отвечающая за то, что генерирование номеров портов исходящих TCP-соединений будет происходить случайным образом.

Randomized RPC XIDs. Рандомизация идентификатора транзакций (XID), который используется протоколом RPC. В Linux значение XID просто увеличивается, что позволяет очень просто предсказать следующий XID.

Sockets Restrictions. С помощью трех подопций можно задать различные ограничения сокетов:

- **Deny Any Sockets to Group**: после включения данной подопции пользователи указанной группы (вам нужно упомянуть ее GID) не смогут связаться с другими узлами;
- **Deny Client Sockets to Group**: в этот раз пользователи данной группы не смогут выступать в роли клиентов, следовательно, для них невозможно будет установить соединение с другими узлами сети;
- **Deny Server Sockets to Group**: наиболее полезно к этой группе отнести пользователей, от имени которых запускаются различные сетевые сервисы, например FTP-сервер или Apache.

Последние две опции особенно сильно отражаются на FTP-сервере: при запрещении клиентских сокетов для передачи может использоваться только пассивный режим, а при запрещении серверных сокетов должен использоваться только активный режим.

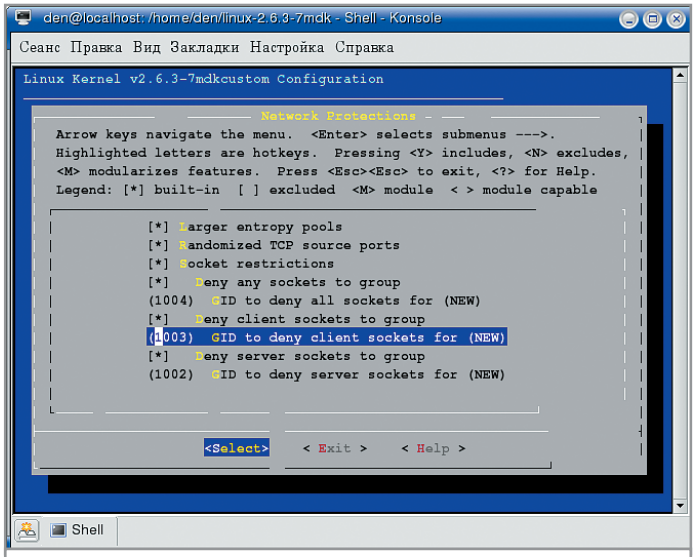
Sysctl Support

Если данная опция включена, большинство ограничений Grsecurity может быть включено/выключено с помощью файловой системы /proc (/proc/sys/kernel/grsecurity/) безо всякой перекомпиляции ядра. Но мы не рекомендуем вам включать эту опцию, поскольку она может стать слабым местом в защите вашей сети.

Конфигурация ядра уже завершена. Теперь перекомпилируйте его, и ваша система будет готова к настройке Grsecurity. Управление доступом реализуется с помощью ACL, но вы не сможете найти опции ядра, связанные с управлением доступом, поскольку конфигурирование ACL осуществляется с помощью утилиты gradm. Ее можно скачать по адресу www.grsecurity.net.

Структура ACL

ACL (список управления доступом) находится в файле /etc/grsec/acl. Можно также самостоятельно включать в него дополнительные списки, используя C-стиль подключения заголовочных файлов (<include>), но рекомендуется хранить все списки в одном файле.



Защита сети

Структура ACL выглядит следующим образом:

```
Subject <путь к процессу> <необязательные режимы процесса>
{
        <файл объект> <необязательные режимы объекта>
        [+|-] <возможность>
        <имя ресурса> <«мягкий» лимит> <«жесткий» лимит>
        connect {
                <ip>/<сетевая маска>:<порт_от>-<порт_до>
<тип> <протокол>
}
        bind {
                <ip>/<сетевая маска>:<порт_от>-<порт_до>
<тип> <протокол>
        }
        <имя ресурса> <«мягкий» лимит> <«жесткий» лимит>
}
```

Рассмотрим пример ACL для демона печати cupsd:

```
subject /usr/sbin/cupsd o {
        /                                       h
        /etc/cups/certs
        /etc/cups/certs/0                       wcd
        /etc/group                              r
        -CAP_ALL
        +CAP_CHOWN
        +CAP_DAC_OVERRIDE
        bind            disabled
        connect   disabled
}
```

Ограничения накладываются на объекты /etc/group, /etc/cups/certs/0 и / путем указания режимов объекта. Для каталога /etc/cups/certs режим не задан.

Режимы субъекта (процесса)

В Grsecurity субъектом называется файл или процесс, к которому применяется ACL (в предыдущем примере — /usr/sbin/cupsd). Режимы субъекта позволяют управлять его поведением (см. таблицу 1).

К сожалению, рассмотреть PaX нам не позволяет объем данной статьи, но вы сможете познакомиться с ним на сайте, расположенном по адресу www.grsecurity.net.

Режимы объектов

Объектом является файл или каталог. Запомните, что если для обозначения режима используется прописная буква, это означает протоколирование, если строчная — определяемое действие.

Для объектов могут использоваться режимы, перечисленные в таблице 2.

Возможности

Список доступных возможностей был приведен чуть выше. В дополнение к ним в Grsecurity используется псевдоним CAP_ALL, обозначающий все возможности. Обычно его применяют для запрета всех возможностей, а впоследствии для разрешения необходимых. Например, сейчас мы запретим все возможности, а потом разрешим возможность изменения владельца файлов:

```
-CAL_ALL
+CAP_CHOWN
```

IP ACL

Grsecurity представляет концепцию IP ACL: ограничения IP-адресов, протоколов, типов сокетов, связанных с процессом:

```
connect {
        <ip>/<сетевая маска>:<порт_от>-<порт_до>
<тип> <протокол>
}
        bind {
                <ip>/<сетевая маска>:<порт_от>-<порт_до>
<тип> <протокол>
        }
```

В обоих случаях синтаксис идентичен: IP-адрес, соответствующая ему маска (если маска не указана, подразумевается /32), после этого следует диапазон портов (младший и старший порты разделяются дефисом). Например, диапазон 0–65 535 используется по умолчанию. Тип сокета может быть следующим: sock, dgram, raw_sock или any_sock. Последним идет протокол. Можно использовать любой протокол, указанный в файле /etc/protocols.

В следующем примере мы ограничим соединения с субъектом в сети 192.168.1.x, разрешим только stream-сокеты в непривилегированном режиме:

```
connect {
192.168.1.1/24:1025-65535       stream          tcp
}
```

Если же вы не хотите управлять ни bind, ни connect, можно их отключить:

```
bind        disabled
connect     disabled
```

Таблица 1. Режимы субъекта

Режим	Поведение
b	Включает учет процесса
d	Защищает псевдофайлы / proc /<pid>/fd и /proc/<pid>/mem этого процесса
h	Скрытый процесс; его могут «увидеть» только процессы, для которых установлен режим v
k	Может «убивать» процессы, защищенные режимом p
l	Включает режим обучения для процесса
o	Отменяет ACL-наследование (см. дальше)
p	Защищает процесс; его может «убить» только процесс в режиме k или другой процесс, принадлежащий этому субъекту
r	Удаляет ограничения ptrace
v	Может «видеть» скрытые процессы
A	Защищает разделяемую память этого процесса; доступ к разделяемой памяти могут получить только процессы, принадлежащие этому субъекту
C	Если процесс «забил тревогу», он будет «убит». Если с этим процессом связан какой-то IP-адрес, также будут «убиты» все процессы, относящиеся к этому IP-адресу
K	Если процесс «забил тревогу», он будет «убит»
T	Не позволяет процессу выполнить любой троянский код
P	Отключает PaX-функцию PEGEEXEC для этого процесса
S	Отключает SEGMEXEC (PaX)
M	Отключает MPROTECT (PaX)
R	Отключает RANDMMAP (PaX)
G	Отключает EMUTRAP (Pax)
X	Включает RANDEXEC (Pax)

Ограничение ресурсов

Ограничение ресурсов процесса задается в следующем виде:

```
<имя ресурса> <«мягкий» лимит> <«жесткий» лимит>
```

С доступными ресурсами вы можете ознакомиться в таблице 3. Почти все эти ресурсы вы уже могли использовать в файле /etc/limits. Для некоторых из них нужно задать время. По умолчанию оно задается в миллисекундах, что не очень удобно. Можно использовать единицы s (секунды), m (минуты) или d (дни). Для ресурсов, где лимит нужно задавать в байтах, удобно использовать единицы K (1000 байт), M (1 000 000 байт) и G (1 000 000 000 байт). Если ограничений не накладывается, нужно указать строку unlimited.

Примеры ограничения ресурса:

```
RES_FSIZE       200M    300M
RES_NPROC       2       3
RES_NOFILE      10      10
```

Наследование субъекта

Grsecurity поддерживает наследование ACL для упрощения его самого. Наследование применяется ко всем субъектам, за исключением тех, которые не были специально отмечены флагом o. Лучше всего продемонстрировать наследование на примере:

```
/ {
        /
        /bin rx
        /etc r
        /tmp rw
        -CAP_ALL
        connect disabled
        bind disabled
}
```

```
/usr/loca/bin/someapp {
        / h
        /etc rw
        /var h
        +CAP_SETUID
}
```

При обработке второго ACL Grsecurity проверит существование ACL для каждого компонента пути (в этом примере для /usr/localbin, /usr/local, /usr и /). В данном случае ACL существует только для субъекта /. Его ACL будет унаследован субъектом /usr/local/bin/someapp.

Предыдущий пример может быть переписан так:

```
/ {
        /
        /bin rx
        /etc r
        /tmp rw
        -CAP_ALL
        connect disabled
        bind disabled
}
/usr/local/bin/someapp {
        /h
        /bin rx
        /etc r
        /tmp rw
        -CAP_ALL
        +CAP_SETUID
        connect disabled
        bind disabled
}
```

Возникший конфликт двух ролей (например, в первом случае для /etc используется режим r, а во втором — режим rw) будет решен в пользу субъекта (то есть здесь будет использоваться режим /etc rw).

Роли

Одна из новейших функций Grsecurity — RBAC — предоставляет администратору дополнительный контроль над субъектами. В данной реализации RBAC ключевым понятием является роль: один или более субъектов могут выполнять какую-либо роль — вы можете объявить один объект несколько раз для разных ролей.

Роль описывается следующим образом:

```
role <имя роли> <необязательные режимы роли>
```

Роль default можно использовать, чтобы применить ACL к пользователям, у которых нет роли, например:

```
role default
subject / {
        /                       r
        /opt                    rx
        /home                   rwxcd
        /mnt                    r
        /dev
        /dev/grsec              h
        /dev/urandom            r
        /dev/random             r
        -CAP_ALL
        connect 192.168.1.0/24:22 stream tcp
        bind    0.0.0.0 stream dgram tcp udp
}
role admin sA
subject   / r {
        / rwcdmxi
}
```

В данном примере мы запретили пользователям локальной сети регистрироваться на удаленных машинах с помощью SSH и наложили ограничения на доступ к некоторой части файловой системы. Роль admin перезаписывает субъект /, устанавливая более слабые ограничения.

Для дополнительной безопасности можно также определить роль role_allow_ip, разрешающую соединения с определенного адреса:

```
role_allow_ip <ip>/<сетевая маска>
```

Например:

```
role_allow_ip 192.168.10.1/32
```

Автоматическая генерация ACL

Для автоматического создания файла правил Grsecurity предоставляет режим обучения. Даже если вы предпочитаете создавать файлы правил вручную, мы рекомендуем использо-

Таблица 2. Режимы управления объектами

Режим	Описание
a	Объект может быть открыт для добавления информации. Информация помещается в конце объекта
c	Разрешает объекту создавать каталоги
d	Разрешает удаление каталога
h	Скрывает объект
i	Если устанавливается на исполняемый файл — ACL субъекта будет унаследован во время выполнения этого файла
m	Объекту можно создавать SUID/SGID-файлы
p	Нельзя использовать ptrace для этого объекта
r	Объект может быть открыт для чтения
s	Не протоколировать отклоненные попытки доступа к объекту
t	Объект можно просмотреть с помощью ptrace, но нельзя изменить
w	Объект может быть открыт для записи или добавления данных
x	Разрешает выполнение объекта
A	Протоколирование успешного добавления информации в объект
C	Протоколирование успешного создания каталога
D	Протоколирование успешного удаления каталога
I	Протоколирование успешного наследования ACL
R	Протоколирование успешного чтения данных
M	Протоколирование успешного создания SUID/SGID-файла
W	Протоколирование успешной записи
X	Протоколирование успешного выполнения

Таблица 3. Доступные ресурсы

Режим	Описание
RES_AS	Ограничение адресного пространства (в байтах)
RES_CORE	Максимальный размер файла core (в байтах)
RES_CPU	Максимальное процессорное время (в мс)
RES_DATA	Максимальный размер данных (в байтах)
RES_FSIZE	Максимальный размер файла (в байтах)
RES_LOCKS	Максимальное число блокировок файлов
RES_MEMLOCK	Максимальное число блокировок памяти (в байтах)
RES_NOFILE	Максимальное число открытых файлов (минимально допустимое число открытых файлов равно 3 — для STDOUT, STDIN и STDERR)
RES_NPROC	Максимальное число процессов
RES_RSS	Максимальный RSS (в байтах)
RES_STACK	Максимальный размер стека (в байтах)

вать режим обучения — он послужит фундаментом для вашего собственного файла.

Если у вас еще не установлена программа gradm, сейчас самое время это сделать (для установки используются команды ./configure; make; make install). После того как вы напишете «make install», вас попросят ввести пароль, который будет использоваться для администрирования ACL-системы. Не вводите пароль пользователя root!

Для запуска автоматического обучения запустите gradm со следующими опциями:

```
# gradm -F -L /etc/grsec/learning.log
```

Во время обучающего режима все действия системы будут запротоколированы в файл /etc/grsec/learning.log. После завершения обучающего режима Grsecurity обработает протокол и на его основании сгенерирует ACL.

В обучающем режиме, так же как и в случае со systrace, важно поработать с приложением, которое вы хотите защитить (с субъектом), в самых разных режимах, чтобы система Grsecurity запротоколировала абсолютно все действия, которые может выполнять субъект. Например, для веб-сервера вы должны не только запросить HTML-страницу, но и разные сценарии, написанные на языках PHP и Perl. Также не забудьте запустить сценарий, работающий с MySQL. Для веб-браузера вам нужно будет посетить различные сайты, не обделяйте вниманием также HTTPS-ресурсы и сайты со скриптами (JavaScript, Jscript и т. д.).

Во время обучающего режима не производите действий администратора, например запуск и остановку сервиса, модифицирование учетной записи, установку и удаление программного обеспечения, иначе они запишутся в протокол и будут считаться разрешенными.

Таблица 4. Режимы роли

Режим	Описание
A	Роль администратора, различные ограничения, например на использование ptrace, снимаются
g	Роль группы
G	Разрешается использование утилиты gradm
l	Режим обучения роли
N	Для этой роли не требуется аутентификация
s	Специальная роль, для которой не применяются ACL
T	Обучение TPE (Trusted Path Execution)
u	Пользовательская роль

Обучающий режим должен длиться целый день (также будут запротоколированы задачи cron). После этого отключите контроль доступа с помощью следующей команды:

```
# gradm -D
Password:
```

Введите пароль, который вы указали при установке программы. Чтобы gradm сгенерировал ACL по полученному протоколу, используйте опцию -O:

```
# gradm -F -L /etc/grsec/learning.log -O /etc/grsec/acl
Beginning full learning 1st pass...done.
Beginning full learning role reduction...done.
Beginning full learning 2nd pass...done.
Beginning full learning subject reduction for user root...done.
Beginning full learning subject reduction for user den...done.
Beginning full learning subject reduction for user snmps...done.
Beginning full learning 2rd pass...done.
Beginning full learning object reduction for subject /...done.
Beginning full learning object reduction for subject /bin/bash...done.
Beginning full learning object reduction for subject /bin/cat...done.
...
Beginning full learning object reduction for subject /...done.
Beginning full learning final pass...done.
```

Создание ACL — довольно утомительная и рутинная задача, поэтому даже опытным пользователям Grsecurity мы рекомендуем использовать встроенный режим обучения для создания начального ACL. Далее мы рассмотрим непосредственно создание ACL для демона sshd.

После многочасовой работы Grsecurity в режиме обучения был получен следующий ACL:

```
subject /usr/sbin/sshd o {
        /                               h
        /bin                            h
        /bin/bash                       x
        /dev                            h
        /dev/log                        rw
        /dev/ptmx                       rw
        /dev/pts
        /dev/pts/1                      rw
        /dev/tty                        rw
        /etc                            r
        /etc/grsec                      h
        /lib                            rx
        /proc                           h
        /proc/1817
        /proc/1819
  /usr                                  h
  /usr/lib/libcrack.so.2.7              rx
  /usr/lib/libglib-1.2.so.0.0.10         rx
  /var                                  h
  /var/empty/sshd
  /var/log
  /var/log/lastlog                      rw
  /var/log/wtmp                         w
```

```
/var/run/utmp                    rw
/root
-CAP_ALL
+CAP_DAC_OVERRIDE
+CAP_SETGID
+CAP_SETUID
+CAP_SYS_CHROOT
+CAP_SYS_TTY_CONFIG
bind 0.0.0.0/32:0 dgram ip
connect 192.168.9.100/32:53 dgram udp
}
```

Задать правила для /proc/<число> мы не можем, поскольку число (PID) будет изменяться при каждом запуске процесса. Поэтому, чтобы удалить записи для /proc/<число>, необходимо написать следующее:

```
/proc
```

Обратите внимание: вы должны снять флаг h с /proc. По аналогии, терминал не всегда будет pts1, поэтому нужно обеспечить rw-доступ (чтение/запись) ко всем /dev/pts:

```
/dev/pts        rw
```

Правило bind сейчас разрешает привязку к любому порту и любому интерфейсу. Учитывая, что мы настраиваем демон SSH, можно явно указать порт — 22:

```
bind 0.0.0.0/32:22 dgram ip
```

Инструкцию connect также необходимо подкорректировать. В ней перечислены адреса компьютеров, с которыми субъект может соединяться (но не те, которые могут соединяться с субъектом!). Наш автоматически сгенерированный ACL содержит правило, разрешающее субъекту соединяться с сервером имен (192.168.9.100). Однако ведь серверов имен может быть несколько. Обычно их два, а наше правило разрешает соединение только с одним сервером — с тем, с которым соединялась программа во время режима обучения. Если этот сервер недоступен, программа будет обращаться к другому серверу имен, указанному в конфигурационных файлах системы, но не сможет найти нужную информацию о нем, поскольку второй сервер не указан в ACL. Указать два (или более) сервера можно так:

```
connect {
        10.0.0.1:53 dgram udp
        10.0.0.2:53 dgram udp
        }
```

Если серверов больше двух и вам лень указывать их все, можно использовать следующую инструкцию, разрешающую подключение к любому серверу имен (порт 53):

```
connect 0.0.0.0/32:53 dgram up
```

Не забывайте, что SSH-демону может понадобиться соединение с демоном AUTH (IDENT), работающим по порту 113. В момент обучения такого соединения не произошло, поэтому в автоматически сгенерированном ACL и слова нет о IDENT. Исправим это:

```
connect {
        192.168.0.0/8:113 dgram ip
        10.0.0.0/24:113 dgram ip
        }
```

Как и в предыдущем случае, можно разрешить соединение с любым IDENT-сервером:

```
connect 0.0.0.0/32:133 dgram up
```

Окончательный листинг ACL для субъекта /usr/sbin/sshd выглядит следующим образом:

```
subject /usr/sbin/sshd o {
        /                        h
        /bin                     h
        /bin/bash                x
        /dev                     h
        /dev/log                 rw
        /dev/ptmx                rw
        /dev/pts                 rw
        /dev/tty                 rw
        /etc                     r
        /etc/grsec               h
        /lib                     rx
        /proc                    h
/usr                             h
/usr/lib/libcrack.so.2.7         rx
/usr/lib/libglib-1.2.so.0.0.10   rx
/var                             h
/var/empty/sshd
/var/log
/var/log/lastlog                 rw
/var/log/wtmp                    w
/var/run/utmp                    rw
/root
-CAP_ALL
+CAP_DAC_OVERRIDE
+CAP_SETGID
+CAP_SETUID
+CAP_SYS_CHROOT
+CAP_SYS_TTY_CONFIG
bind 0.0.0.0/32:22 dgram ip
connect {
192.168.9.100/32:53 dgram udp
192.168.9.200/32:53 dgram udp
192.168.0.0/16:113 dgram udp
}
}
```

Однако это еще не все. В нашем примере используется режим субъекта o. Можно задать несколько дополнительных режимов, например: h — чтобы скрыть процесс от ненужных глаз (но тогда не забудьте «спрятать» объект /var/run/sshd.pid); A — чтобы защитить разделяемую память процесса; T — для защиты от троянов; X — для активации RANDEXEC. Учитывая новые режимы, первая строка описания субъекта будет выглядеть так:

```
subject /usr/sbin/sshd ohATX {
```

Также желательно ограничить сбои процесса, чтобы их было не более двух в течение часа:

```
RES_CRACH        2        60m
```

Надеюсь, что данная статья поможет вам при настройке Grsecurity. Любые вопросы можете смело направлять по адресу электронной почты: dhsilabs@mail.ru. |