

---

# RED CARD – СПЕЦИФИКА СТАНДАРТА PSI DSS ДЛЯ RHEL

ФЕДОР КУЛИШОВ  
POSITIVE TECHNOLOGIES



POSITIVE / TECHNOLOGIES®

---

## ОГЛАВЛЕНИЕ

|    |  |    |
|----|--|----|
| 1  | Введение   | 3  |
| 2  | Что такое PSI DSS и зачем он нужен   | 4  |
| 3  | Требование 1. Установить и обеспечить функционирование межсетевых экранов для защиты данных о держателях карт  | 6  |
| 4  | Требование 2. Не использовать пароли и другие системные параметры, заданные производителем по умолчанию        | 12 |
| 5  | Требование 3. Обеспечить безопасное хранение данных о держателях карт  | 19 |
| 6  | Требование 4. Обеспечить шифрование данных о держателях карт при их передаче через сети общего пользования     | 24 |
| 7  | Требование 5. Использовать и регулярно обновлять антивирусное программное обеспечение                          | 26 |
| 8  | Требование 6. Разрабатывать и поддерживать безопасные системы и приложения                                     | 28 |
| 9  | Требование 7. Ограничить доступ к данным платежных карт в соответствии со служебной необходимостью             | 29 |
| 10 | Требование 8. Назначить уникальный идентификатор каждому лицу, имеющему доступ к информационной инфраструктуре | 43 |
| 11 | Требование 10. Контролировать и отслеживать любой доступ к сетевым ресурсам и данным о держателях карт         | 55 |
| 12 | Требование A1. Поставщики услуг с общей средой должны защищать среду данных платежных карт                     | 65 |
| 13 | Исследовательский центр Positive Research  | 70 |

# 1 Введение

---

В данной статье будет рассмотрена настройка стандартной Linux-системы (с учетом стандартного ПО, поставляющегося в комплекте с дистрибутивом) для соответствия стандарту PCI DSS на примерах RHEL 5 и Fedora Core 12. Для каждого требования стандарта будут даны рекомендации по настройке системы, как на основе существующих технических стандартов (CIS, NIST, SANS), так и на основе опыта настройки подобных систем.

В качестве целевого Linux-дистрибутива был выбран RedHat Enterprise Linux, т.к. он, наряду с Novell SUSE Enterprise Linux Server/Desktop, пользуется наиболее широкой поддержкой производителей аппаратного и программного обеспечения и имеет множество опций коммерческой поддержки, из-за чего широко применяется в бизнес-сообществе, в том числе среди компаний, ведущих операции с пластиковыми картами. FC12 служит базой для будущего RHEL6.

**Примечание.** Данная публикация представляет лишь одну из точек зрения на проблему аудита соответствия требованиям PCI DSS. Аудиторы, проверяющие конкретные системы обработки данных, могут иметь свое видение безопасных настроек, не всегда совпадающее с рекомендациями этой статьи, т.к. некоторые пункты стандарта допускают множество толкований. Однако данная работа может послужить отправной точкой для получения RH-систем, не только удовлетворяющих требованиям стандарта, но и настроенных безопасно.

## 2 Что такое PCI DSS и зачем он нужен

Основными разработчиками стандарта PCI DSS (Payment Card Industry Digital Security Standard) являются платежные системы Visa и Master Card. Как указано в официальном документе «Требования и процедура аудита безопасности», «... стандарт безопасности данных индустрии платежных карт (PCI DSS) разработан в целях упрощения внедрения и распространения мер по обеспечению безопасности данных о держателях карт. В основе данного стандарта лежат 12 требований, сгруппированных таким образом, чтобы упростить процедуру аудита безопасности. Данный стандарт предназначен для использования аудиторами при проверке торгово-сервисных предприятий и поставщиков услуг на соответствие его требованиям» (в соответствие с [1]).

### Цель данной статьи

Стандарт предъявляет множество технических и организационных требований, например:

- Проверить, что весь иной входящий и исходящий трафик явно запрещен (1.2.1.b, техническое)
- Получить и проверить документацию, подтверждающую, что наборы правил пересматриваются как минимум раз в полгода (1.1.6.b, организационное)

Требования предъявляются ко **всей инфраструктуре** (относящейся к обработке данных пластиковых карт) проверяемого объекта. В то же время, существуют как общепризнанные (такие, как семейство CIS, SANS, NIST), так и различные внутрикорпоративные стандарты информационной безопасности, где даются четкие рекомендации по технической настройке **определенных целевых систем**. При этом установление соответствия между требованиями PCI DSS и, например, требованиями CIS для конкретной ОС представляется нетривиальной задачей.

Дело осложняется тем, что часто требования PCI DSS «размыты», то есть с первого взгляда не очевидно, какие подсистемы необходимо под них настраивать, дадут ли полученные системные настройки полное покрытие требований и существуют ли вообще техническая возможность выполнить конкретное требование стандарта. С другой стороны, CIS и другие технические стандарты защиты оперируют конкретными параметрами системы (например, для UNIX-систем в CIS приводятся даже сценарии, выполняющие требуемые настройки). Тем не менее, при более глубоком изучении стандарта PCI DSS оказывается, что многие, поначалу казавшиеся неконкретными, требования могут быть технически удовлетворены. Предлагаемая Вашему вниманию статья ставит своей целью сближение абстрактного «аудиторского» языка стандарта PCI DSS с четкими терминами системных администраторов



и технических специалистов по информационной безопасности. В качестве целевой системы выступают дистрибутивы RHEL5 и Fedora 12, причем последняя рассмотрена как база для выходящего скоро RHEL6 (доступного в бета-версии с апреля 2010 года). Структура статьи в целом повторяет структуру PCI DSS, для каждого пункта стандарта приводятся соответствующие технические рекомендации.

## Публикации по теме

Попытки конкретизировать требования PCI DSS применительно к определенным операционным системам предпринимались уже неоднократно. Примерами могут служить публикации [2], [3] и [4]. Однако ни в одной из работ не были приведены полные и детальные настройки ОС; анализ требований не покрывал технической части стандарта целиком и в основном ограничивался несколькими главами PCI DSS, либо был слишком общим и не затрагивал конкретных настроек ОС (команд, конфигурационных файлов, списка применимого ПО).

## Общие для всех глав положения

Здесь и далее будем считать, что на целевой системе RHEL установлено только ПО из стандартной поставки; исключения из этого правила оговариваются отдельно. Предполагается, что узлы под управлением RHEL выступают либо в роли серверов, либо в роли рабочих станций сети обработки данных; серверы RHEL также могут выполнять некоторые функции в DMZ. Настройка межсетевых экранов между DMZ и сетью обработки данных, а также между DMZ и сетью Интернет детально не рассматривается, т.к. в крупных организациях Linux-узлы (с установленными дистрибутивами общего назначения) обычно не выполняют функции маршрутизаторов и шлюзов безопасности.

Многие требования являются сугубо организационными или их применение для самой операционной системы нелогично: например, многие пункты главы 3 описывают порядок хранения данных платежных карт, которые в реальной жизни отражаются в базе данных, то есть на уровне прикладного, а не системного, ПО. Если в тексте не упомянуто требование, это означает, что оно организационное или не применимо к данной платформе. Для краткости изложения требования PCI DSS часто приводятся в сокращенном виде.

**ВАЖНО.** Несмотря на то, что настройка ОС и прикладного ПО, несомненно, важна для безопасной обработки данных, немалую часть стандарта PCI DSS занимают требования к документации и внедрению. Более того, во многих технических требованиях стандарта указано, что определенные механизмы должны быть не только настроены, но и внедрены. Это означает, что документации тоже стоит уделить внимание, хотя бы для того, чтобы успешно пройти аудит на соответствие стандарту PCI DSS. Рисунок 2 – получение конфигурации устройства через память

## 3 Требование. Установить и обеспечить функционирование межсетевых экранов для защиты данных о держателях карт

---

### Краткое содержание

Глава начинается с сугубо организационных требований, технические требования приводятся с пункта 1.1.5.b. Как следует из описания, большая их часть требует настройки Netfilter и при необходимости — tcp\_wrappers. Однако в стандарте CIS для RHEL упоминается только настройка tcp\_wrappers (пункт 3.2), а Netfilter в текущей редакции (v.1.1.2 от 06.2009) упоминается лишь один раз, в части запуска служб при старте системы; настройки правил фильтрации трафика там не оговариваются.

Далее, возникает вопрос, к какому типу трафика отнести VPN-туннели до смежных центров обработки данных: к локальному, DMZ или Интернет-трафику. Более того, в следующих главах стандарта не уделяется должного внимания VPN-каналам. Также не совсем ясно, допускает ли стандарт передачу Интернет-трафика от узлов сети обработки данных через прокси-серверы в DMZ.

Проанализировав требования, можно сделать следующий вывод о рекомендуемой архитектуре сети:

1. В DMZ должны быть расположены серверы приложений (именно приложений, а не СУБД!), доступные для сторонних клиентов, которые обращаются (через DNAT/VPN) к серверам баз данных, стоящим в отдельной защищенной сети.
2. На граничных маршрутизаторах (между DMZ и сетью Интернет и между защищенной сетью и DMZ) настроены строгие политики фильтрации трафика.
3. Строгие правила фильтрации трафика также настроены на всех узлах защищенной сети.
4. Доступ в сеть Интернет из защищенной сети разрешен только на DMZ-узлы.

**1.1.5.b Выявить разрешенные небезопасные сервисы, протоколы и порты, проверить их необходимость, а также то, что механизмы защиты документированы и внедрены, путем изучения стандартов конфигурации межсетевых экранов и маршрутизаторов и настроек каждого сервиса.**

Основной смысл этого требования — избавиться, где это возможно, от прикладных протоколов, передающих трафик (как данные, так и информацию аутентификации) в открытом виде. Мешать исполнению этого требования будут:

1. telnet – впрочем, в Linux он уже долгое время не ставится «из коробки»;
2. ftp – вместо него необходимо использовать sftp, если отсутствует четкая формулировка в пользу сохранения FTP;
3. «Большая почтовая тройка» pop3/imap/smtp – в большинстве случаев заменяются на свои «S»-аналоги;
4. Различные веб-ориентированные механизмы доступа, например, к почте, если трафик в них не шифруется (например, SquirrelMail, трафик клиентов которого при соответствующих настройках можно пустить через порт 443);
5. Сервисы, запускаемые обычно из xinetd (echo, discard, tftp и т.п.) — их необходимость на 90% систем может быть поставлена под вопрос.

На этом шаге настройки межсетевых экранов (в стандартной поставке это Netfilter и tcp\_wrappers) не рассматриваются, требования к ним будут приведены ниже.

Приблизительный аналог данного требования в CIS — главы «Minimize Boot Services» и «Minimize Xinetd Network Services» (номера 3 и 4).

### **1.2.1.a Проверить, что входящий и исходящий трафик ограничен только необходимыми для среды данных о держателях карт соединениями и что ограничения документированы**

Собственно, данное требование и относится к настройкам системных и прикладных межсетевых экранов. При этом в нем не уточняется, на каких узлах сети должен фильтроваться трафик; будем считать, что под это требование подпадают настройки как маршрутизаторов/шлюзов безопасности, так и серверов, рабочих станций и портативных компьютеров пользователей, имеющих доступ в сеть обработки платежной информации.

Для выполнения этого требования необходимо создать четкие ACCEPT-правила, желательно по принципу «один сервис — одно правило» (из-за особенностей Netfilter на один сервис будет в действительности приходиться два правила — по одному в цепочках INPUT и OUTPUT или оба в FORWARD, если целевой узел — маршрутизатор), в **iptables** и при необходимости дополнить их настройками **tcp\_wrappers** в **/etc/hosts.allow**.

Очевидно, межсетевой экран должен работать на старте системы. В большинстве случаев это решается включением сервиса iptables на старте системы:

#### **chkconfig iptables on**

Тем не менее, в некоторых случаях системному администратору может быть удобнее поместить список правил фильтрации в сценарий, запускаемый при старте системы. Этот

сценарий может вызываться, например, из `/etc/rc.d/rc.local`. Удобство такого подхода — сценарий всегда перед глазами, при необходимости можно быстро изменить и применить правила. Недостаток описан в требовании 1.2.2.

Косвенное следствие этого требования — отключение протокола IPv6 в случае, если он не используется для передачи данных. В системах на базе RHEL для этого необходимо запретить загрузку модуля `ipv6.ko`, добавив следующую запись в файл внутри каталога `/etc/modprobe.d` (например, `/etc/modprobe.d/blacklist.conf`):

```
install ipv6 /bin/true
```

В результате модуль поддержки IPv6 не будет загружаться, даже если этого потребует какое-либо приложение или часть ОС (например, `ip6tables`). Команда

```
modprobe ipv6
```

также откажется загружать модуль; единственным способом загрузки при таких настройках останется использование `insmod`.

Подобные действия можно выполнить и для других сетевых протоколов, не использующихся в системе (SCTP, ...).

#### **1.2.1.b Проверить, что весь иной входящий и исходящий трафик явно запрещен**

Решается установкой стандартных политик межсетевого экрана через `iptables` в DROP:

```
iptables -P INPUT DROP  
iptables -P OUTPUT DROP  
iptables -P FORWARD DROP,
```

а также путем запрета всего неразрешенного трафика в `tcp_wrappers` с помощью следующей записи в `/etc/hosts.deny`:

```
ALL: ALL
```

#### **1.2.2 Проверить, что конфигурационные файлы маршрутизаторов синхронизированы, например, рабочие конфигурационные файлы и конфигурационные файлы, используемые при перезагрузке маршрутизатора, имеют одинаковую безопасную конфигурацию**

Формулировка, приведенная в тексте требования, вполне ясна. Однако стоит отметить, что это требование также применимо к настройке Netfilter на серверах и рабочих станциях.

Вы можете проверять на совпадение вывод команды `iptables-save` и файл правил `/etc/sysconfig/iptables`. Процедуру несложно оптимизировать при помощи текстовых фильтров и `md5sum`.

Необходимо помнить, что если правила фильтрации трафика описаны не в стандартном сценарии, запускаемом при старте системы (см. п. 1.2.1.a), то автоматизировать процедуру будет гораздо сложнее.

**Tcp\_wrappers** в подобной проверке синхронизации настроек не нуждается.

### **1.3.2 Проверить, что входящие Интернет-соединения ограничены только адресами, находящимися в DMZ**

Потребуется настройка серверов (см. п. 1.2.1.а) и маршрутизатора/шлюза безопасности на границе сети обработки данных и DMZ. Таким образом, трафик будет фильтроваться и на шлюзе, и на узлах сети обработки данных.

Неочевидное следствие требования — при соответствующих настройках граничных сетевых устройств (маршрутизаторов/шлюзов безопасности) допускается DNAT-переброска трафика, исходящего с узлов DMZ, в сеть обработки данных, которая, как будет показано далее, не должна иметь реальную IP-адресацию.

### **1.3.3 Проверить, что отсутствуют прямые входящие и исходящие маршруты между сетью Интернет и средой данных о держателях карт**

Формально требование является частью 1.2.1.а, однако имеет более широкие последствия.

Отсюда вытекает следующее:

1. В DMZ требуется настроить серверы обновлений для всего ПО из сети обработки данных, т.к., с одной стороны, узлы должны регулярно обновляться, а с другой стороны — устанавливать обновления напрямую из сети Интернет нежелательно.

2. Подобная сеть не должна иметь реальную IP-адресацию (она ей не требуется); кроме того, в п.1.3.8 имеется явный запрет на реальную адресацию.

Из текста требования не ясно, допускается ли использование прокси-серверов для доступа в сеть Интернет узлов защищенной сети. Если это необходимо для бизнеса, то прямого запрета нет.

### **1.3.5 Проверить, что исходящий трафик из среды данных о держателях карт имеет доступ только к IP-адресам, расположенным в DMZ**

Формально является частью требований 1.2.1.а и 1.3.3, однако не дает ответа на вопрос, как расценивать VPN-трафик между сетями обработки данных, расположенными на разных площадках и соединяющихся через VPN-туннель, проходящий через Интернет.

### **1.3.7 Проверить, что базы данных расположены во внутренней сети, отделенной от DMZ**

Данная настройка тривиальна, однако проверка в условиях большой DMZ-сети может быть затруднена.

Первым шагом проверки может быть запуск nmap с узла внутри DMZ:

```
nmap -sS -sV -T5 dmz-net/dmz-mask
```

```
nmap -sU -sV -T5 dmz-net/dmz-mask
```

Также можно (и нужно!) запустить сканирование узлов DMZ извне (подобная проверка отвечает пункту 11.2 PCI DSS, однако официально должна проводиться организацией со статусом ASV).

Может оказаться, что СУБД разрешено прослушивать только loopback-интерфейс или только локальный UNIX-сокеты; кроме того, доступ к порту может быть закрыт межсетевым экраном. В этом случае потребуется провести анализ вручную или использовать специализированный сканер (способный авторизоваться на целевых узлах) для выявления запущенных баз данных.

С этим пунктом можно сопоставить требование CIS 4.18 с доработками по остальным базам данных (Oracle, IBM DB2 и т.д.).

### **1.3.8 Для нескольких межсетевых экранов и маршрутизаторов проверить работоспособность механизма трансляции адресов для предотвращения раскрытия внутренних адресов**

Из этого требования явно следует, что сеть обработки данных не должна иметь реальных IP-адресов, т.к. в любом случае будет использоваться NAT.

### **1.4.a Проверить, что на все мобильные и принадлежащие сотрудникам компьютеры, имеющие прямой доступ в сеть Интернет и используемые для доступа к локальной сети организации, установлены персональные межсетевые экраны.**

Netfilter входит в стандартную поставку, необходимо только проверить настроенные в нем правила фильтрации трафика аналогично требованиям пункта 1.2.1.a.

В этом и следующем пункте подразумеваем, что RHEL/Fedora установлены на рабочие ноутбуки пользователей. В таком случае излишними будут и организационные меры защиты, предотвращающие вынос подобных устройств с данными и внешних носителей информации за контролируемый периметр; такие меры рассматриваются в пунктах 9.6 — 9.9 стандарта PCI DSS.

### **1.4.b Проверить, что настройки персонального межсетевого экрана выполнены в соответствии со стандартами организации и не могут быть изменены пользователем.**

Применение правил фильтрации трафика не должно вызвать трудностей. Настройки же привилегий пользователей, напротив, требуют более тщательного подхода и будут рассмотрены далее в главе 7.

Для выполнения последних двух требований (1.4.a и 1.4.b) также целесообразно будет специальным образом настроить граничные межсетевые экраны, чтобы вне зависимости от изобретательности пользователей (имеющих привычку искать способы избавиться от навязанных свыше ограничений) трафик до подобных узлов был жестко ограничен.

#### **Список источников:**

1. Стандарт безопасности данных индустрии платежных карт (PCI DSS). Требования и процедура аудита безопасности (Версия 1.2.1, июль 2009) – [www.pcidss.ru](http://www.pcidss.ru)
2. <http://www.pcilinux.com>
3. <http://rnc2.com/regulatory-compliance/pcidss/logging-for-pci-dss-compliance/>
4. <http://www.ibmssystemsmag.com/aix/augustseptember08/coverstory/21121p1.aspx>

## 4 Требование. Не использовать пароли и другие системные параметры, заданные производителем по умолчанию

### Краткое содержание

Требования этой главы имеют намного больше общего с CIS, нежели требования предыдущей. Соответствующие пункты CIS RHEL – главы 3,4, п. 2.3, 8.2, 9.4, 11.2, SN.8. Однако в дополнение к настройкам самой ОС здесь придется воспользоваться и внешними средствами – сканерами портов и средствами подбора паролей.

### 2.1 Всегда следует менять установленные производителем настройки по умолчанию перед установкой системы в сетевую инфраструктуру

Речь идет о паролях для сетевых устройств и точек Wi-Fi доступа, строках доступа SNMP и т.п. Требование подразумевает ручную проверку (без которой в большинстве случаев не обойтись), однако есть шанс избежать заметной части трудоемких работ, используя специализированный сетевой или оффлайн-инструмент подбора паролей.

В качестве сетевого инструмента подбора паролей может быть использован THC Hydra, словари перебора которого можно дополнить по своему усмотрению. Недостаток Hydra состоит в необходимости вручную указывать адреса и порты; при желании этот инструмент можно посредством сценариев связать с Nmap, распознающим запущенные сервисы на сканируемых узлах.

Кроме того, существуют и оффлайн-инструменты подбора паролей, имеющие более высокое быстродействие, чем сетевые. В случае Linux такой инструмент должен будет скачать локальную базу хэшей (/etc/shadow), определить используемый алгоритм (authconfig --list, обычно md5 или sha512) и провести атаку на хэши уже на локальном узле, с которого запущено сканирование. Например, подобный функционал реализован в сканере MaxPatrol (от компании Positive Technologies). Самый известный открытый оффлайн-взломщик паролей – John the Ripper, в последнее время популярность набрал также RainbowCrack, способный вести перебор еще и на GPU.

Разумеется, требуется исключить использование пустых паролей, для чего нужно проверить отсутствие таковых в `/etc/shadow`, после чего запретить авторизацию по пустым паролям в `ssh`, добавив в `/etc/ssh/sshd_config` следующую строку:

### PermitEmptyPasswords no

Значение «no» установлено для этого параметра по умолчанию, поэтому следует лишь убедиться, что в файле настроек не указано значение «yes».

Запрет на использование пустых паролей есть в CIS, п. 9.2. Кроме того, п. 11.4 предусматривает включение `pat_cracklib` для анализа паролей при их создании или смене.

**ВАЖНО.** Стоит отметить, что процедуры сканирования PCI ASV запрещают интенсивный перебор паролей (чтобы не создавать чрезмерный трафик и не подвергать пользователей угрозе блокировки), речь идет только о проверке настроек по умолчанию. В RHEL-системах отсутствуют учетные записи с заранее определенными именами пользователей и паролями (исключение составляет анонимный FTP в тех случаях, когда такой доступ должен быть запрещен), поэтому формально нет необходимости применять инструменты подбора паролей. Тем не менее, если есть возможность безопасно применить взлом паролей путем перебора для исследования стойкости паролей – его стоит применить.

## 2.2.a Изучить стандарты конфигурации всех системных компонентов. Проверить, что стандарты конфигурации учитывают положения общепринятых отраслевых стандартов (SANS, NIST, CIS)

Потребуется настроить целевые узлы в соответствии с требованиями общепринятых (например, CIS) либо корпоративных стандартов (которые зачастую составляются на основе общепринятых). Для оценки соответствия настроек целевого узла заданным требованиям существует множество коммерческих сканеров (MaxPatrol, Symantec SCCS, Nessus, Acunetix и т.п.), которые могут заметно упростить трудоемкую процедуру проверки.

Однако не стоит слепо полагаться на стандарты и автоматизированные средства; ошибки возможны везде, потому не следует пренебрегать ручным анализом.

Соответствует

**Дополнительные настройки безопасности:**  
Количество попыток входа до блокировки учетной записи

**Краткое описание**  
Рекомендуется настроить блокировку учетной записи после трех последовательных неудачных попыток зарегистрироваться в системе.

**Контроль**

**Результаты проверки**

| Проверка  | Результат |
|---|-----------|
| Параметр <code>deny=3</code> установлен в файле <code>/etc/pam.d/ssh</code>                   | Да        |
| Параметр <code>deny=3</code> установлен в файле <code>/etc/pam.d/system-auth</code>           | Да        |
| Параметр <code>unlock_time=1800</code> установлен в файле <code>/etc/pam.d/ssh</code>         | Да        |
| Параметр <code>unlock_time=1800</code> установлен в файле <code>/etc/pam.d/system-auth</code> | Да        |

**Рис. 1 – Блокирование пользователей в FC12 (MaxPatrol)**



Например, CIS RHEL v.1.1.2 в пункте SN.8 предписывает блокировать пользователей после определенного числа неудачных попыток входа в систему. Рекомендуемые настройки подходят только для RHEL4 с модулем `ram_tally`, для 5й версии они теряют смысл, т.к. при неизменном имени модуля он имеет уже другие параметры; кроме того, вместе со старой доступна и новая версия модуля — `ram_tally2`. При этом стандарт официально поддерживает 5ю версию. В новых системах, в частности Fedora 12, настройки отличаются даже от RHEL 5 (настройка `ram_tally2` одинаковая, но FC12 не может использовать `ram_tally`).

В сканере MaxPatrol корректно реализована поддержка всех современных RHEL-систем.

### **2.2.1 Для нескольких системных компонентов проверить, что выполняется правило "один сервер - одна основная функция". Например, веб-сервер, сервер СУБД и DNS-сервер следует размещать на разных компьютерах**

Для выполнения этого требования можно воспользоваться любой из существующих технологий виртуализации. Для RHEL 5 официально поддерживается Xen, однако в RHEL 6 от него отказались в пользу KVM. Оба решения предлагают похожий функционал и отличаются лишь реализацией; так, в качестве основной причины отказа от Xen в новых версиях дистрибутива была названа сложность его реализации на уровне ядра ОС и неясные перспективы включения исходного кода Xen в официальную версию ядра Linux.

Также можно добавить, что Xen не поддерживает режим «перегрузки», при котором виртуальным машинам предоставляется больше процессорных ядер, чем их физическое количество на сервере виртуализации. Такой режим часто применяется для тестирования, разработки и хостинга, когда на виртуальные серверы приходится небольшая нагрузка. В таком случае стоит использовать KVM, а не Xen.

Кроме Xen и KVM, упоминается также проект OpenVZ, не включенный в дистрибутивы Linux, однако широко применяющийся для VPS-хостинга и являющийся основой для коммерческого продукта Virtuozzo. OpenVZ позволяет более гибко настраивать выделение ресурсов для виртуальных машин. Из недостатков стоит отметить спектр гостевых ОС: в настоящий момент поддерживается только Linux.

Существует множество коммерческих (или закрытых) средств виртуализации, среди которых можно упомянуть VMWare ESX, Virtuozzo и Microsoft Hyper-V.

### **2.2.2 Для выборки из нескольких системных компонентов проверить включенные сервисы и протоколы. Проверить, что ненужные или небезопасные сервисы и протоколы выключены или их использование обосновано и документировано**

Формулировка, приведенная в тексте требования, вполне ясна; более того, пункты требования выполняются при соблюдении требований CIS к запуску демонов на целевых узлах (главы 3, 4 CIS). Суть большинства этих требований заключается в использовании команды

**`chkconfig daemon_name on | off,`**

которая для каждого выбранного сервиса включает или отключает его загрузку при старте

ОС на predetermined уровнях запуска (для самостоятельных сервисов и компонентов xinetd синтаксис одинаков).

По пп. 2.2.2 и 2.2.4, в дополнение к выполнению требований CIS, можно отключить ненужные модули ядра, например, поддержку пока что экзотических сетевых протоколов, таких как SCTP, аналогично примеру из пункта 1.2.1.a, а также модули отсутствующего на данном сервере/рабочей станции/ноутбуке оборудования.

### 2.2.3. Следует настроить параметры безопасности системы таким образом, чтобы исключить возможность некорректного использования системы.

Формулировка требования недостаточно конкретна, однако можно сделать предположение, что в него входит настройка защитных механизмов уровня ядра. Кроме того, настройка безопасных параметров ядра важна для защиты всей системы, однако это единственный пункт требований, где данный механизм может быть применим.

Первым механизмом защиты ядра является защита от переполнения буфера, реализуемая в основном за счет случайных адресов стека приложения, а также случайных адресов загрузки библиотек и приложений (общепринятый термин – Address Space Layout Randomization, ASLR [1]). В официальной версии ядра Linux подобные механизмы начали внедряться, начиная с версии 2.6.12, полностью функционал был реализован в версии 2.6.25 [2].

Однако еще до выхода указанных версий ядра дистрибутивы RHEL и Fedora Core имели встроенный механизм ExecShield ([3], [4]), реализующий подобные функции. Убедиться в его наличии можно при помощи команды:

```
[root@support ~]# sysctl -a | grep -E 'randomize | shield'  
  
kernel.randomize_va_space = 1  
  
kernel.exec-shield = 1
```

Указанные переменные ядра должны иметь ненулевые значения. Отключать эту возможность следует только в особых случаях, например, при тестировании ПО или разработке средств безопасности. Для этого в файл /etc/sysctl.conf необходимо добавить строки:

```
kernel.randomize_va_space = 0  
  
kernel.exec-shield = 0
```

После этого либо перезагрузите ОС, либо примените эти параметры без перезагрузки:  
**sysctl -p**

Дополнительным элементом защиты от атак переполнения буфера является запрет на выполнение кода в стеке, накладываемый на уровне тела приложения или библиотеки. В Linux за эти действия отвечает утилита **execstack**, способная разрешать или запрещать выполнение кода в стеке для конкретных программ и библиотек, а также запрашивать у

приложений и библиотек необходимость выполнения кода в стеке [5].

Приложения и библиотеки с потенциально исполняемым стеком могут быть найдены следующим способом:

```
[root@support ~]# echo BINARIES;; for dir in /usr/libexec `echo $PATH | sed -e 's:/ /g'`; do execstack -q ${dir}/* 2>/dev/null; done | grep -Ev ^\|-; echo LIBS;; find /lib /usr/lib -type f -name '*.so' -exec execstack -q {} \; 2>/dev/null | grep -v ^\|-
```

BINARIES:

```
? /sbin/mount.vmhgfs
? /usr/sbin/vmware-checkvm
? /usr/sbin/vmware-guestd
? /usr/sbin/vmware-tools-upgrader
? /usr/sbin/vmware-vmtoolsd
? /usr/bin/vmware-hgfsclient
? /usr/bin/vmware-xferlogs
```

LIBS:

```
X /usr/lib/vmware-tools/lib32/libvmGuestLibJava.so
X /usr/lib/vmware-tools/lib32/libvmGuestLib.so
```

Также следует настроить сетевые параметры ядра, описанные в CIS (пп. 5.1, 5.2, SN.3 версии 1.1.2) и отвечающие за маршрутизацию, ICMP-протокол и защиту от DoS-атак. В указанном стандарте приведены детальные настройки ядра, нет необходимости переносить их в данную статью.

Возможно, к этому пункту требований также относится использование SELinux, детальное рассмотрение настроек которого выходит за рамки данной работы. Особенности его настройки приведены в [6] и [7].

#### **2.2.4 Для нескольких системных компонентов проверить, что ненужная функциональность выключена. Проверить, что включенные функции документированы и безопасно настроены**

Настройка системы согласно этому требованию во многом аналогична п. 2.2.2, однако ее можно дополнить отключением поддержки пользовательских внешних носителей (usb-дисков и firewall-устройств). Для этого требуется создать и заполнить следующие файлы:

**/etc/modprobe.d/disable-usb:**

```
install usb-storage /bin/true
```

**/etc/modprobe.d/disable-fireware:**

```
install fireware_ohci /bin/true
```

Если впоследствии внешнее USB- или Fireware-устройство потребуется для резервного копирования, то администратор сможет загрузить соответствующий модуль ядра с помощью команды `insmod`, недоступной для непривилегированных пользователей.

Кроме того, если защищаемый узел поддерживает WiFi и/или WiMAX, которые в условиях данной компании не нужны для выполнения рабочих задач, их можно отключить в BIOS, а также удалить модули ядра:

```
rm -f /lib/modules/`uname -r`/kernel/drivers/net/wi{max,reless}/*
```

**Примечание.** Удаление модулей ядра требуется производить после каждого обновления ядра [8].

### **2.3 Для нескольких системных компонентов проверить, что для защиты удаленного административного доступа применяются криптографические механизмы**

Для выполнения требования нужно:

1. Отключить запуск демона `telnet` (который и так по умолчанию не устанавливается), а также поддержку `r`-команд и демонов (`rlogin`, `rsh`, `rcp`), которых также нет в современных дистрибутивах.

2. Отключить небезопасные опции SSHD, установив в файле `/etc/ssh/sshd_config` параметры:

```
Protocol 2  
RhostsRSAAuthentication no  
IgnoreRhosts yes  
PermitEmptyPasswords no
```

Впрочем, все опции (кроме, возможно, первой) по умолчанию, как и требуется, отключены, поэтому в большинстве случаев потребуется не изменение параметров, а проверка, что им не присвоены другие (небезопасные) значения.

3. При использовании LDAP-аутентификации необходимо защитить трафик между клиентом и сервером от перехвата. Если шифрование сеанса связи не производится, злоумышленник сможет перехватить хэши паролей, передаваемые клиенту сервером; кроме того, при смене пароля новый пользовательский пароль передается на сервер LDAP в открытом виде. Чтобы этого избежать и удовлетворить требование стандарта, можно выполнить действия из [9] (пример был приведен для клиента CentOS5 и сервера FC12) со следующими изменениями:

3.1 Режим шифрования на сервере в `/etc/openldap/slapd.conf` указывается строкой

## TLSCipherSuite HIGH:MEDIUM:+TLSv1:+SSLv3

3.2 Во время генерации пары ключей для LDAP-сервера нельзя указывать парольную фразу (challenge). Это требование приведено в официальной документации (man slapd.conf).

3.3 На стороне клиента можно указать прямой путь к файлу сертификата, а не к каталогу (в файле настройки LDAP-аутентификации **/etc/ldap.conf**):  
**TLS\_CACERT /etc/openldap/cacerts/rf12.crt**

3.4 Кроме того, Вы можете настроить для клиента более низкие значения таймаута соединения и другой тип соединения (bind\_policy в

**/etc/ldap.conf**):  
**timelimit 10**  
**bind\_timelimit 4**  
**bind\_policy soft**

4. При наличии FTP-демонов нужно убедиться, что суперпользователь не может входить в систему по FTP. У стоящего в системе по умолчанию **vsftpd** в каталоге **/etc/vsftpd/** существует 2 файла, определяющие пользователей, которым запрещен такой вход, — **ftpusers** и **user\_list**. Необходимо поместить запись «root» в **ftpusers**, после чего проверить значение параметра **userlist\_deny** в файле **/etc/vsftpd/vsftpd.conf**. Если параметр установлен в YES (по умолчанию), то запись «root» должна быть в файле **user\_list**, если NO — записи «root» там быть не должно.

5. Также стоит предусмотреть и предотвратить другие, в основном экзотические, ситуации, например проверку почты суперпользователя через SquirrelMail без использования HTTPS.

### Список источников:

1. [http://en.wikipedia.org/wiki/Address\\_space\\_layout\\_randomization](http://en.wikipedia.org/wiki/Address_space_layout_randomization)
2. [http://kernelnewbies.org/Linux\\_2\\_6\\_25](http://kernelnewbies.org/Linux_2_6_25)
3. <http://www.cyberciti.biz/faq/what-is-rhel-centos-fedora-core-execshield/>
4. <http://www.redhat.com/magazine/009jul05/features/execshield/>
5. <http://www.tenouk.com/Bufferoverflowc/Bufferoverflow5.html>
6. <http://www.redhat.com/docs/manuals/enterprise>
7. [http://www.redhat.com/docs/en-US/Red\\_Hat\\_Enterprise\\_Linux/6-Beta/](http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6-Beta/)
8. <http://people.redhat.com/sgrubb/files/hardening-rhel5.pdf>
9. [http://vuksan.com/linux/LDAP\\_authentication\\_under\\_Linux.html](http://vuksan.com/linux/LDAP_authentication_under_Linux.html)

## 5 Требование. Обеспечить безопасное хранение данных о держателях карт

---

### Краткое содержание

В главе 3 множество требований к хранению данных платежных карт, заметная часть которых относится к базам данных и серверам приложений, а не к ОС. Интерес представляют требования к шифрованию данных, многие из которых можно реализовать стандартными средствами из поставки RHEL. Стандарт CIS не регламентирует вопросы шифрования данных.

#### **3.4.1.a Если применяется шифрование на уровне диска, проверить, что логический доступ к файловой системе реализован при помощи механизма, независимого от механизмов разграничения доступа операционной системы**

Суть требования состоит в том, что обращение к расшифрованным данным должно было возможно только при знании ключа; отсюда следует, что любые процессы и пользователи (даже администраторы системы) не смогут прочесть и корректно изменить такие данные без знания ключей расшифрования.

У всех распространенных механизмов шифрования файловых систем Linux (cryptsetup, cryptsetup + LUKS, EncFS, eCryptFS) расшифрованная файловая система (ФС) логически ничем не отличается от обычной ФС и имеет те же атрибуты доступа, ACL и т.п. С помощью этого реализуется свойство прозрачности файловых операций. Однако, даже с корректно настроенными правами доступа к данным, суперпользователь root или процессы с UID=0 смогут получить доступ к любым данным после их расшифрования владельцем ключа, а значит, данное требование PCI DSS не выполняется для всех механизмов шифрования файловых систем Linux.

Настройки SELinux, разрешающие доступ к расшифрованным данным только заранее указанным пользователям, не помогут в выполнении данного требования, так как

- ситуация будет зависеть от SELinux, который является механизмом разграничения доступа ОС;
- даже при настроенных SELinux-политиках суперпользователь все равно сможет их изменить и получить доступ к расшифрованным данным.

С другой стороны, если истолковать это требование как «механизмы разграничения доступа к данным и механизмы шифрования данных должны производиться различными компонентами ОС», то стандартная для Linux подсистема шифрования данных удовлетворяет этим требованиям.

Однако количество гипотез относительно истинного смысла на этом не ограничивается. Возможно, требование предъявляется только к аппаратным устройствам дискового шифрования. В этом случае требование 3.4.1.a не будет предъявляться, и организация сможет использовать без ограничений любую подходящую технологию программного шифрования (в т.ч. стандартную LUKS).

Рассматривая это требование, следует принять решение, нужно ли использовать дисковое шифрование вообще (это касается жестких дисков серверов/рабочих станций или внешних массивов). На практике шифрование этих носителей необходимо в следующих случаях:

1. Ценные данные хранятся на ноутбуке, который может покинуть пределы контролируемого периметра (или периодически их покидает). Данные могут касаться не только платежных карт, но и архитектуры сети, учетных записей для входа на серверы и т.п.
2. Неисправные жесткие диски передаются в ремонт сторонней организации (что на практике происходит в большинстве случаев), которая при желании имеет шанс восстановить с них какую-либо конфиденциальную информацию и использовать ее в своих целях.
3. Возникает необходимость перевезти между площадками настроенное и рабочее оборудование с данными (серверы, дисковые массивы и т.п.) с помощью третьих лиц (транспортных компаний и т.п.), полный контроль над действиями которых невозможен или затруднен.

**3.4.1.c Проверить, что данные о держателях карт на съемных носителях хранятся только в зашифрованном виде. Примечание: Часто шифрование на уровне всего диска не зашифровывает информацию на съемных носителях, поэтому может потребоваться шифровать данные на съемных носителях отдельно.**

Для шифрования данных на съемных носителях можно применить как шифрование отдельных файлов (eCryptFS, OpenSSL), так и всего диска (LUKS). eCryptFS по умолчанию не входит в состав рассматриваемых дистрибутивов, поэтому более детально остановимся на OpenSSL и LUKS.

Шифрование отдельных файлов можно выполнить с помощью openssl, например

**openssl enc -aes-256-cbc -in input.file -out output.file**

При расшифровании дополнительно указывается ключ «-d»:

**openssl enc -aes-256-cbc -d -in output.file -out input.file.decrypted**

Однако такой способ подходит в основном для хранения резервных копий данных («зашифровал и забыл») ввиду трудоемкости и непрозрачности доступа к зашифрованным данным.

Более универсальный сценарий — шифрование на уровне диска в режиме ядра, которое можно настроить следующим образом:

1. Заполнить целевой раздел псевдослучайными данными (пусть это будет /dev/sdb1):

**dd if=/dev/urandom of=/dev/sdb1**

2. Настроить шифрование пустого дискового раздела

**cryptsetup -y -c aes --key-size=256 luksFormat /dev/sdb1**

При этом потребуется дважды ввести пароль шифрования.

3. Создать файловую систему на зашифрованном разделе, для этого открыть его в ручном режиме:

**cryptsetup luksOpen /dev/sdb1 topsecret**

(здесь нужно ввести пароль, до «закрытия» устройства он больше запрашиваться не будет)  
После этой операции в системе появится устройство с расшифрованными данными **/dev/mapper/topsecret**, обращаться к которому можно как к обычному дисковому разделу или логическому тому.

**mkfs.ext3 -m0 /dev/mapper/topsecret**

И затем проверить, что операция завершилась успешно:

**mkdir /mnt/topsecret**

**mount /dev/mapper/topsecret /mnt/topsecret**

**ll /mnt/topsecret**

Все, созданная зашифрованная ФС готова к обычному чтению/записи.

**ВАЖНО.** Следует соответствующим образом настроить права доступа к расшифрованным данным — при помощи установки стандартных rwx-прав, ACL файловой системы, а при

необходимости и SELinux. Также следует помнить о том, что суперпользователь при желании сможет получить любой доступ к данным, если они расшифрованы.

Если использование этой ФС носит эпизодический характер, то п.4 можно пропустить и каждый раз вводить 2 команды:

```
cryptsetup luksOpen /dev/sdb1 topsecret  
mount /dev/mapper/topsecret /mnt/topsecret
```

По завершении работы с зашифрованным устройством его нужно логически удалить:

```
cryptsetup luksClose topsecret
```

4. Для автоматического монтирования ФС при загрузке ОС (актуально скорее для дисковых разделов, а не внешних устройств), необходимо добавить записи в файлы:

```
/etc/crypttab:  
topsecret /dev/sdb1 none luks,check=ext2,retry=5
```

Здесь опция «**none**» означает, что пароль вводится с клавиатуры (иначе задает путь к файлу, содержащему ключ шифрования, см. **man crypttab**), последнее поле содержит опции шифрования (так, **retry=5** задает максимальное количество запросов пароля).

**ВНИМАНИЕ!** В случае, когда раздел описан в **/etc/crypttab** (неважно, присутствует ли эта ФС в **/etc/fstab**), ОС будет пытаться его расшифровать при запуске (если при этом пароль вводится с клавиатуры, то ОС будет ждать его ввода и без этого не станет загружаться – такое поведение для серверов крайне нежелательно). Если расшифрованный раздел не должен постоянно существовать в системе, настройка **/etc/crypttab** не нужна.

```
/etc/fstab:  
/dev/mapper/topsecret /mnt/topsecret ext3 noauto,defaults 0 0
```

5. В случае непредвиденных ситуаций (например, при отключении питания) расшифрованное устройство (здесь это **/dev/mapper/topsecret**) исчезает из системы; без повторного ввода пароля расшифрование раздела получить доступ к данным будет невозможно.

6. Чтобы убедиться в том, что данные хранятся в зашифрованном виде, можно проверить, что указанный дисковый раздел зашифрован при помощи LUKS:

```
cryptsetup isLuks /dev/sdb1 && echo OK | | echo fail
```

В случае, когда применяется шифрование отдельных файлов при помощи OpenSSL или других подобных средств, такая проверка не может быть выполнена автоматически.

Единственное «но» - нужно удостовериться, что не был использован NULL-алгоритм шифрования:

**grep -a -q pattern /dev/sdb1 && echo fail || echo OK,**

где pattern — любая часть содержимого незашифрованного файла.

### **3.5.2 Изучить системные конфигурационные файлы, убедиться, что ключи хранятся в зашифрованном виде и ключи шифрования ключей хранятся отдельно от ключей шифрования данных.**

При использовании LUKS это требование соблюдается, т.к. ключи шифрования данных хранятся в зашифрованном виде (они шифруются на основании вводимого пользователем пароля) на соответствующем зашифрованном разделе ФС — то есть отдельно от ключей шифрования ключей.

Проверить использование LUKS-шифрования для дискового раздела можно, например, так:

**cryptsetup isLuks /dev/sdb1 && echo OK || echo fail**

Здесь /dev/sdb1 — зашифрованный раздел (а НЕ имя расшифрованного устройства в /dev/mapper), который может быть дисковым разделом или логическим томом.

## **6** Требование. Обеспечить шифрование данных о держателях карт при их передаче через сети общего пользования

---

### Краткое содержание

В главе приводятся требования по шифрованию данных, аналогов которым нет в CIS. Ключевые моменты – использование средств создания VPN-каналов, поддержка шифрования веб-трафика и использование сертификатов.

**4.1 Для защиты данных о держателях карт во время передачи их через общедоступные сети следует использовать стойкие криптографические алгоритмы и протоколы, такие как SSL/TLS и IPSEC. Примерами общедоступных сетей, на которые распространяются требования PCI DSS, являются:**

- **Интернет;**
- **Беспроводные технологии;**
- **GSM;**
- **GPRS.**

Требования этого пункта можно разделить на две группы. К первой относится шифрование трафика между клиентом и сервером, выполняемое, например, в режиме доступа клиента к сайту банка или при оплате услуг по пластиковой карте через Интернет. Во вторую группу входят VPN-соединения между компонентами системы при их удаленном взаимодействии: это может быть сеанс связи между главным отделением и филиалом банка, отделением банка и платежным терминалом, а также между различными центрами обработки данных одной и той же организации. Рассмотрим вкратце требования обеих групп.

Для организации удаленного доступа клиента к банковской системе через Интернет чаще всего применяется HTTPS, с информацией по базовой настройке которого в Apache можно ознакомиться в [1]. Корректная работа с сертификатами и шифрованием трафика часто является критически важной для обеспечения защиты данных; этому вопросу отводится заметное место в документации проекта Apache, рассмотрение которой, однако, выходит за рамки данной статьи.

Для организации VPN-канала в крупных организациях часто применяются решения на базе сетевого оборудования, при этом для серверов и рабочих станций наличие зашифрованного

туннеля незаметно. Если необходимо создать VPN-туннель на основе Linux-сервера, как правило, применяются средства [2] или [3].

**Список источников:**

1. <http://httpd.apache.org/docs/2.0/ssl/>
2. <http://www.openswan.org>
3. <http://www.strongswan.org>

## 7 Требование. Использовать и регулярно обновлять антивирусное программное обеспечение

---

### Краткое содержание

Практически все известные производители антивирусного ПО выпускают продукты для Linux, как для рабочих станций, так и для серверов (файл-серверы, почтовые серверы, антивирусные шлюзы для прокси-серверов). Единственным открытым антивирусом для Linux (и не только для него) является ClamAV. Его настройки будут рассмотрены ниже в этой главе.

Аналогичные требования CIS – п. 12.

#### **5.1 Для нескольких системных компонентов, включая все типы операционных систем, подверженных воздействию вирусов, проверить, что используется антивирусная защита (если подходящая антивирусная технология существует).**

Из-за распространенности Linux-систем аудиторы могут отнести RHEL к операционным системам, подверженным воздействию вирусов. В этом случае будет не лишним установить ClamAV, хотя объективно его стоит использовать лишь для выполнения формальных требований стандарта: он заметно уступает коммерческим продуктам по быстродействию, возможностям эвристического обнаружения (которые отсутствуют) и удобству использования для конечных пользователей. Преимущество ClamAV состоит в его универсальности, т.к. в поставку помимо сканера файловой системы обычно включается сервер сканирования, часто используемый в составе почтовых и прокси-серверов.

Самый быстрый способ установить ClamAV – выполнить команду

**yum -y install clamav clamav-update**

#### **5.1.1 Для нескольких системных компонентов проверить, что антивирусное программное обеспечение обеспечивает защиту от всех известных форм вредоносного программного обеспечения, включая шпионские и рекламные программы.**

Защита от вредоносных рекламных программ (более известных как AdWare) актуальна для рабочих станций и браузеров; опасность таких программ для серверов неочевидна. Более

важное требование – защита от шпионского ПО, в том числе от руткитов; в этом случае требуется дополнительно установить пакет Rootkit Hunter:

### **yum install rkhunter**

**5.2.c Для нескольких системных компонентов, включая все типы операционных систем, подверженных воздействию вирусов, проверить, что автоматическое обновление антивирусного программного обеспечения и периодические проверки включены.**

По умолчанию пакет clamav-update сразу после установки добавляет задачу обновления баз в cron при помощи задания /etc/cron.d/clamav-update, которое раз в 3 часа проверяет обновления антивирусных баз. Пакет rkhunter добавляет свое задание (ежедневное сканирование) в /etc/cron.daily/rkhunter, однако возможность регулярного обновления баз в него не заложена.

Для выполнения требования необходимо создать cron-задание для clamscan, чтобы периодически выполнялась проверка файловых систем на наличие вредоносного ПО.

**5.2.d Для нескольких системных компонентов проверить, что включено протоколирование событий антивирусного программного обеспечения и журналы протоколирования сохраняются в соответствии с требованием 10.7 PCI DSS.**

Сканер файловой системы clamscan по умолчанию выводит результат работы на консоль; ему можно указать путь к файлу отчета с помощью опции "-l". Разумно расположить эти файлы в каталоге /var/log либо передавать консольный отчет сканера на вход утилите logger, которая записывает эти сообщения через syslog, чтобы затем переслать журналы регистрации на удаленный сервер.

Rootkit Hunter по умолчанию хранит журналы регистрации событий в каталоге /var/log/rkhunter.

## 8 Требование. Разрабатывать и поддерживать безопасные системы и приложения

---

**6.1.a Для нескольких системных компонентов и программного обеспечения проверить, установлены ли актуальные обновления безопасности, выпущенные производителем.**

Дистрибутивы RedHat имеют в своем составе демона yum-updatesd, проверяющего через определенные промежутки времени наличие обновлений и при соответствующих настройках устанавливающего их. Поведение демона задается в конфигурационном файле /etc/yum/yum-updatesd.conf, сценарий запуска - /etc/init.d/yum-updatesd.

Для запуска этого демона при старте ОС необходимо выполнить:

**chkconfig yum-updatesd on**

## 9 Ограничить доступ к данным платежных карт в соответствии со служебной необходимостью

---

### Краткое содержание

Оба технических требования, приведенные в этом разделе, достаточно сложны в реализации, настройка ОС под каждое из них затрагивает множество подсистем. Аналоги для требования 7.2.1 в CIS RHEL – это пункты 8.1, 8.2, 8.5, 8.8, 9.2, 9.8, 9.11, SN.7 и SN.11, однако они покрывают не все механизмы разграничения доступа, существующие в системе. Для требования 7.1.1 частичные аналоги в CIS – пункты 7.2 и 7.6.

#### 7.1.1 Доступ пользователям предоставлен только к тем данным, которые необходимы им для выполнения своих должностных обязанностей.

Данное требование слишком размыто. К нему при желании можно отнести любые настройки доступа, начиная от членства в группах и заканчивая политиками SELinux. Рассмотрим наиболее распространенные системные средства контроля доступа.

I. Важной частью разграничения привилегий является контроль **доступа к объектам файловой системы** (например, к файлам баз данных, если база хранится в ФС, а не на raw device). Доступ может предоставляться как традиционным для UNIX **rwх-способом**, так и с помощью списков контроля доступа **ACL (Access Control List) на уровне файловой системы**.

Для выявления некорректных прав доступа (не те флаги rwх, неверные владельцы и группы) удобнее всего применить стандартную команду **find** (с ключами **-perm**, **-user** и **-group**), однако она не позволяет вести поиск по ACL.

Механизм ACL применяется для гибкого разграничения прав доступа, когда применения стандартного способа «владелец : группа : остальные» оказывается недостаточно. Этот механизм заметно расширяет возможности контроля доступа к файлам и каталогам, однако обладает рядом недостатков (которые будут рассмотрены ниже). Для использования ACL файловая система ext3/ext4 должна быть смонтирована с опцией "acl"; XFS в этой опции не нуждается.

**ВАЖНО.** RHEL5 не поддерживает создание и управление файловыми системами XFS; ОС может работать только с заранее созданными XFS [1]. Тем не менее, необходимые утилиты можно получить, связавшись с менеджером [1], либо установить их самостоятельно [2 и 3]

(при этом неясно, как изменятся условия технической поддержки такой системы). В дистрибутиве RHEL6 ожидается полная поддержка XFS наряду с ext4 [4].

Для установки расширенных прав доступа применяется команда **setfacl**, для просмотра – **getfacl**.

Так, команда

**setfacl -m u:user1:rwX /path/to/file**

добавит в список доступа к файлу /path/to/file пользователя "user1" с правами rwX.

Просмотреть (рекурсивно) ACL файлов и каталогов можно с помощью команды:

**getfacl -sR /path/to**

где ключ «-s» подавляет вывод информации о файлах и каталогах, не имеющих ACL.

В листинге объекты с ACL можно отличить по символу «+» в поле прав доступа (выделено желтым):

```
[root@rf12 ~]# ls -lF m*
-rw-r--r--. 1 root root 91620 Jun 10 15:48 mkinitrd-6.0.93-1.fc12.i686.rpm
-rwxrwx---+ 1 root root 7 Jun 30 15:46 my.sh*
```

При этом в отображении традиционных прав доступа возникает ошибка, причем не только у команды ls, но и у find, stat, и т.п. Как можно видеть, ls неправильно отображает права доступа группы: на самом деле, для файла my.sh установлены права доступа 700; даже find «думает», что для указанного файла установлены права 770 (stat допускает ту же ошибку):

```
[root@rf12 ~]# find ./ -type f -perm 770 -print
./my.sh
```

При этом файл my.sh действительно имеет права доступа 700 (содержимое поля "group::---"):

```
[root@rf12 ~]# getfacl my.sh
# file: my.sh
# owner: root
# group: root
user::rwx
user:user:rwx
group::---
mask::rwx
other::---
```

В этом можно легко убедиться на примере. По логике ls/find/stat, пользователь из группы users должен обладать правами на выполнение этого сценария, на самом же деле он не сможет его запустить:

```
[root@rf12 ~]# cp -p my.sh /home/superadmin/  
[root@rf12 ~]# chown root:users /home/superadmin/my.sh  
[root@rf12 ~]# su - superadmin  
[superadmin@rf12 ~]$ id -Gn  
users wheel  
[superadmin@rf12 ~]$ ll  
total 8  
-rwxrwx---+ 1 root users 7 Jul 1 17:58 my.sh  
[superadmin@rf12 ~]$ ./my.sh  
-bash: ./my.sh: Permission denied
```

### Примечание.

Команды файловых операций для сохранения ACL требуют применения дополнительных опций (выделены желтым):

```
cp -p file-with-acl new-file-with-acl  
tar --acls /path/to/archive.tar files-with-acl  
rsync -auvA my.sh /home/user/
```

Также стоит помнить, что ACL будут сохраняться только на тех ФС, которые поддерживают эту опцию.

Кроме того, ACL уже установленной программы будет утерян после ее обновления (выделено желтым):

```
[root@rf12 security]# setfacl -m u:user:rw /sbin/rdisc  
[root@rf12 security]# getfacl -s /sbin/rdisc  
getfacl: Removing leading '/' from absolute path names  
# file: sbin/rdisc  
# owner: root  
# group: root  
user::rwx  
user:user:rw-  
group::r-x  
mask::rwx  
other::r-x  
[root@rf12 security]# rpm --quiet -U --force /mnt/cdrom/Packages/iputils-20071127-  
9.fc12.i686.rpm  
[root@rf12 security]# getfacl /sbin/rdisc  
getfacl: Removing leading '/' from absolute path names  
# file: sbin/rdisc  
# owner: root  
# group: root  
user::rwx
```



```
group::r-x  
other::r-x
```

Подводя итог по ACL, стоит отметить следующие недостатки данного механизма:

1. Отсутствие поддержки ACL у стандартных утилит, путаница между ACL и гvx-правами доступа (find, ls, stat). Команда **find** не позволяет работать с ACL; если необходимо контролировать права доступа к файлам и каталогам с учетом ACL, приходится использовать рекурсивный обход каталогов с подавлением вывода стандартных гvx-правил доступа (**getfacl -sR**) и сценарий собственной разработки (awk, perl, python и т.д.).

2. Необходимость использования дополнительных ключей команд или внесения alias'ов в **/etc/bashrc**:

```
alias cp='cp --preserve=all'
```

```
alias rsync='rsync -X'
```

```
alias tar='tar --xattrs'
```

Однако в этом случае (если используются настройки по умолчанию) затираются пользовательские alias'ы (из \$HOME/.bashrc); так, например, для root значение cp='cp -i' будет заменено на cp='cp --preserve=all'. Для исправления этой ситуации придется дополнительно редактировать файл /etc/bashrc.

3. Некорректная или злонамеренная установка ACL ставит под удар защиту всей системы. Если для ограничения доступа достаточно стандартного гvx-механизма, то рекомендуется принудительно отключить ACL, указав опцию **noacl** в /etc/fstab для файловых систем ext2/ ext3/ ext4.

4. Невозможность отключения этого механизма для XFS; в связи с этим XFS-разделам необходимо уделять особое внимание.

II. Список **SUID/SGID приложений** должен постоянно контролироваться. Добиться этого можно сравнительно простыми способами, например:

```
find / \( -path /proc -o -path /sys -o -path /srv -o -path /dev -o -path /selinux \) -  
prune -o -type f \( -perm -4000 -fprintf /root/suid.log "%p\n" -o -perm -2000 -fprintf  
/root/sgid.log "%p\n" \)
```

В этом примере все найденные SUID-приложения отражаются в файле /root/suid.log, все SGID-программы – в /root/sgid.log. Каталоги, не содержащие реальных файлов (/proc, /sys, /srv, /dev, /selinux), исключаются из поиска.

Также рекомендуется настроить контроль целостности найденных исполняемых файлов или, по крайней мере, следить за тем, чтобы пользователи не могли изменить содержимое этих файлов.

Кроме того, все сторонние файловые системы (CIFS, NFS, внешние носители) необходимо монтировать с опцией "nosuid", чтобы избежать выполнения небезопасных программ из неконтролируемого источника.

Аналоги этих требований приводятся в CIS RHEL v.1.1.2 в пунктах 7.2 и 7.6.

III. Следует также обратить внимание на систему **SELinux**. Несмотря на сложность настройки и использования, она позволяет очень гибко настраивать доступ определенных пользователей и приложений к данным. Кроме того, система поддерживает категоризацию

данных, то есть разделение их не только по степени конфиденциальности, но и по предметной области (режим MLS). Рассмотрению настроек SELinux может быть посвящена отдельная обширная статья; впрочем, у RedHat уже есть полная документация по этому вопросу [5, 6 и 7].

### 7.2.1 Изучить настройки системы и документацию изготовителя, убедиться, что система контроля доступа включает в себя покрытие всех системных компонентов.

Если в предыдущем пункте речь шла о контроле доступа к данным, то здесь дело касается разграничения доступа пользователей. Рассмотрим все распространенные системные механизмы ограничения прав пользователей. В примерах будут фигурировать листинги команд, выполняемых от имени root, пользователь user (пользователь, для которого настраиваются те или иные разрешения) и пользователь superadmin (который, несмотря имя, этими разрешениями обладать не будет).

I. В первую очередь следует убедиться в том, что не используются **устаревшие способы аутентификации**, например, поддержка .rhosts в PAM (CIS 8.1) и .netrc (CIS 9.8).

II. **Все механизмы доступа пользователей к системе** должны постоянно контролироваться. Например, не следует выдавать реальный login shell (bash, sh и т.д.) сугубо почтовым пользователям. Если в системе установлен демон FTP, необходимо проверить его настройки как для анонимных, так и для локальных пользователей.

III. В случае, когда недоверенный пользователь может получить физический доступ к локальной консоли критически важных серверов и/или рабочих станций, следует не только использовать пароль BIOS и отключить в BIOS возможность загрузки с внешних носителей, но и установить пароль для входа в **меню GRUB** или, по крайней мере, в режим **single-user**.

Зачем это нужно?

Необходимость использовать пароль GRUB связана, например, с тем, что GRUB позволяет читать файлы с разделов диска с правами root при условии, что сам GRUB может работать с ФС данного раздела (выполняется как минимум на ext2/ext3). Таким образом, пользователь, имеющий физический доступ к узлу, может во время его включения войти в меню загрузки и считать с диска произвольный файл. Однако GRUB, разумеется, не сможет прочесть файл из зашифрованной ФС, а также в случае, если ФС создана на логическом томе.

```
[root@rf12 etc]# grub --no-floppy
... [ приветствие пропущено ] ...
grub> cat (hd0,0)/grub/grub.conf
cat (hd0,0)/grub/grub.conf
# grub.conf generated by anaconda
... [ пропущена часть файла ] ...
    initrd /initramfs-2.6.31.5-127.fc12.i686.PAE.img
[Hit return to continue]
grub> quit
quit
```

Кроме того, войдя в меню GRUB, можно будет указать уровень загрузки, в том числе однопользовательский режим (single-user mode), когда по умолчанию на tty1 запускается bash от имени root без запроса пароля. Развитие событий в случае атаки злоумышленника может быть крайне негативным (хорошо, если все ограничится экстренной проверкой регламента резервного восстановления).

Защитить локальную консоль можно с помощью следующих действий.

4. Выполнить команду

#### **grub-md5-crypt**

В ответ на приглашение дважды ввести желаемый пароль и сохранить полученный хэш.

5. В файле **/boot/grub/grub.conf** ПЕРЕД всеми командами «title» добавить строку:  
**password --md5 сохраненный\_хэш**

6. После этого пользователь сможет выбрать загрузку любой из систем, перечисленных в title, однако без указания пароля GRUB невозможно будет изменить опции загрузки, в том числе редактировать runlevel.

7. Чтобы отключить режим single-user mode, не требующий ввода пароля, в новых дистрибутивах, построенных на event-загрузке, необходимо в файле **/etc/event.d/rcS-sulogin** в блок «script... end script» добавить следующую строку:

#### **exec /sbin/sulogin**

Строку «exec /bin/bash» при этом следует закомментировать или удалить.

**ВНИМАНИЕ!** После такой настройки в Fedora Core 12 из-за ошибки в sulogin невозможно вводить команды в однопользовательском режиме; кроме того, даже bash в single-user mode периодически не отображает на экране вводимые символы.

8. Для дистрибутивов, загрузка которых построена на /etc/inittab, требуется добавить в **/etc/inittab** строку:  
**~~:S:respawn:/sbin/sulogin**

В третьем поле этой строки может быть указано как respawn, так и wait, в зависимости от того, какое поведение ожидается от системы в однопользовательском режиме. В первом случае ОС будет оставаться на уровне запуска 1 даже после выхода из оболочки; во втором

– после выхода из bash будет загружен уровень по умолчанию. Примечательно то, что первое поле может быть любой строкой длиной не более 4 символов (а не только “~~”), не конфликтующей с другими записями в /etc/inittab.

IV. Далее следует осуществить **настройку su**. Необходимо убедиться, что выполнять su могут только члены группы wheel; для этого проверьте настройки файла **/etc/pam.d/su**, в

котором должны присутствовать следующие строки (опция `use_uid` нужна для применения эффективного, а не первоначального UID):

```
auth      sufficient  pam_rootok.so
auth      required  pam_wheel.so use_uid
```

При этом в файле **не должна** присутствовать (или должна быть закомментирована) строка, позволяющая членам группы `wheel` не вводить пароль целевого пользователя (в том числе `root`) при выполнении `su`:

```
auth      sufficient  pam_wheel.so trust use_uid
```

После этого добавьте всех пользователей, которые должны иметь возможность выполнять команду `su`, в группе `wheel`:

```
usermod -G wheel $username
```

V. Далее необходимо проверить **настройки sudo**. Сосредоточимся на основных опциях конфигурационного файла `/etc/sudoers`, править который можно как напрямую, так и с помощью команды **visudo** (рекомендуется использовать второй вариант). Формат этого файла общий для всех UNIX-систем (как для Linux, так и для Solaris, AIX, и т.д.). Дополнительно рекомендуется ознакомиться со справочным руководством (`man sudoers`).

1. Настройки для пользователей и групп записываются в виде строки:

```
username hostname = [([user][:group])] [modifier:] cmd1[, cmd2]...  
[, [modifier:] ...]
```

Здесь:

- **[...]** – означает, что параметр не является обязательным;
- **[, [modifier:] ...]** – означает, что в одной строке может быть записано несколько правил для одного и того же пользователя, при этом структура правил будет неизменна;
- **username** – имя пользователя, имя группы (записывается с символом “%” в начале строки, например `%wheel`);
- **hostname** – имя узла, для которого выполняется данное правило (чаще всего `ALL`);
- **([user][:group])** – от имени каких пользователей (или членов какой группы) можно выполнять команды. Допустимый синтаксис:
  - `(user)`,
  - `(:group)`,
  - `(user:group)` – можно выполнять от имени этого пользователя и членов этой группы,
  - `(ALL)` – от имени любого пользователя (в том числе `root`),
  - также допустимы так называемые «alias» – списки пользователей или групп.

Если параметр не указан, то команда может быть выполнена от имени любого пользователя.

**ВАЖНО.** Указание только группы в виде (:group) возможно не для всех версий sudo. Так, версия 1.7.1 (FC12) поддерживает такую опцию, а версии 1.6.8 (RHEL5) и 1.6.9 (SLES11) – нет;

- **modifier** – ключевые слово, например:
  - NOPASSWD – разрешить выполнять эти команды без ввода своего пароля,
  - PASSWD – требовать ввод пароля пользователя при выполнении команды (поведение по умолчанию),
  - NOEXEC – запретить использование тактики, известной как «escape to shell», когда, например, пользователь, которому было предоставлено право на выполнение **vi**, может из **vi** перейти в shell с UID, под которым запущен **vi** (поведение по умолчанию),
  - EXEC – разрешить программам использовать механизм «escape to shell»;
- **cmd** – путь до команды и при необходимости ее аргументы (параметр может быть записан в виде регулярного выражения). Допустимы следующие виды записей:
  - cmd1, cmd2, ... – список команд (возможно, с аргументами),
  - cmd1, !cmd2, ... – cmd1 можно выполнять, cmd2 выполнять нельзя,
  - также разрешается использовать alias'ы команд (подробнее об этом см. п.3).

2. Таким образом, допустимые записи в /etc/sudoers могут иметь вид:

- **user ALL=(ALL) NOPASSWD: ALL** – разрешить пользователю user выполнять все команды от имени всех пользователей без ввода своего пароля;
- **user ALL=(root) /bin/kill [[:print:]]\*** – разрешить пользователю выполнять от имени root команду **kill** с любыми аргументами (без аргументов запрещено);
- **superadmin ALL=(wheel) /bin/date, /bin/su [[:print:]]\*, /sbin/shutdown -r, /usr/bin/id [[:print:]]\*, (root) /bin/kill [[:print:]]\*** – пользователь superadmin может выполнять от имени группы wheel команды date, su (только с аргументами, без аргументов запрещено), перезагружать компьютер, выполнять id (только с аргументами), а также запускать **kill** от имени root с аргументами.

**ВАЖНО.** При такой настройке из-за особенностей механизмов авторизации пользователь сможет выполнять команду **su без** указания пароля целевого пользователя (в том числе и root), так как пароль у него будет запрошен один, а не два раза.

3. Следует проверить опции Defaults, определяющие поведение по умолчанию. Так, интерес представляет опция

## Defaults targetpw

Если используется эта опция, то для выполнения команды от имени целевого пользователя необходимо будет указать не свой (поведение по умолчанию), а его пароль. Такая опция устанавливается по умолчанию, например, на SLES 11. В этом случае поведение sudo будет похоже на поведение su, однако механизм sudo более гибок в настройке, нежели su.

**Примечание.** В описанном случае, если пользователь будет выполнять какую-либо команду от имени группы (новые версии sudo), то ему придется ввести пароль root, а не свой пароль, с помощью команды

## sudo -g groupname cmd

**ВАЖНО!** Неверно введенный пароль пользователя (если не указана директива **Defaults targetpw**) или принудительный обрыв ввода пароля (Ctrl+C) приравнивается к некорректному вводу учетных данных, что влечет за собой увеличение счетчика неудачных

попыток входа (pam\_tally и pam\_tally2; подробнее об этом в разделе 8) и может привести к блокировке пользователя. Если указана директива targetpw, то увеличивается счетчик не текущего, а целевого пользователя, что может привести к крайне негативным последствиям. В связи с этим рекомендуется не злоупотреблять опцией targetpw.

4. Рекомендуется настроить продолжительность беспарольного выполнения команд. По умолчанию между последовательными процедурами ввода пароля проходит 5 минут; в это время пользователь может выполнять команды с помощью sudo без ввода пароля. Длительность промежутка определяется директивой

## Defaults:username timestamp\_timeout=N

Здесь username – имя пользователя (или ALL для всех), N – количество минут (0 – отсутствие интервала, пользователи будут вводить пароль каждый раз; отрицательное значение – бесконечно долгий промежуток).

**ВАЖНО.** Указанный интервал времени отсчитывается от момента выполнения последней команды sudo. Таким образом, если таймаут равен 1 минуте, а пользователь вводит sudo-команды раз в 55 секунд, то ему не придется вводить пароль каждый раз заново. Но если очередная команда sudo выполняется после предыдущей через промежуток времени, превышающий интервал timestamp\_timeout, то пароль придется ввести заново. Если в течение периода timestamp\_timeout пользователь выйдет из системы, а затем снова зарегистрируется в ней, то он сможет выполнять sudo-команды без ввода пароля.

5. Отметим, что для того, чтобы иметь возможность выполнить sudo, пользователь не должен быть заблокирован. Например, если пользователь user был заблокирован из-за последовательности неудачных попыток входа и при этом открыта его сессия, то из нее он не сможет выполнить sudo, даже если соответствующее правило имеется в /etc/sudoers. Однако если правило sudoers допускает для user выполнение команды без пароля (NOPASSWD:), то попытка будет успешной.



б. Совместное использование **su + sudo** позволяет гибко настраивать административный доступ. Например, если пользователь `user2` может выполнять `sudo` со своим паролем (или без него), то при удаленном входе в систему ему понадобится ввести всего один пароль. В связи с этим не рекомендуется предоставлять таким пользователям возможность удаленного входа (т.е. такие пользователи не должны быть указаны в директиве **AllowUsers** файла **/etc/ssh/sshd\_config**). При использовании `su+sudo` пользователь `user1` может удаленно войти в систему, переключиться на `user2` (**su – user2**) и выполнять команды через `sudo` от имени `user2`; в этом случае для выполнения `sudo`-команд ему потребуется указать уже два пароля.

VI. Отдельно следует проверить **уникальность UID и непустые пароли**.

Сценарий для выполнения первой задачи достаточно прост:

```
awk -F: '{print $1, $3}' /etc/passwd | sort -k2 -n | uniq -d -f1
```

Результат его выполнения будет либо пуст (это значит, что условие уникальности UID выполнено), либо будет представлять собой список записей вида «пользователь UID» для всех пользователей, у которых были обнаружены не уникальные идентификаторы.

Для решения второй задачи используется следующий сценарий:

```
awk -F: '($2 == "") {print $1;}' /etc/shadow
```

Результат выполнения этого сценария будет либо пуст (условие отсутствия пустых паролей выполнено), либо будет представлять собой список пользователей с пустыми паролями. Кроме того, рекомендуется удалить аргумент «nullok» у модуля `pam_unix.so` из файлов настройки PAM в `/etc/pam.d`.

VII. Часто бывает необходимо проконтролировать **членство пользователей в группах**. Например, для Oracle члены системных групп `oracle` и `oinstall` имеют больше привилегий, чем рядовые пользователи. Следить нужно не только за содержимым **/etc/group**, но и за первичными группами пользователей (с помощью команды **id** или в четвертом поле в **/etc/passwd**).

**Внимание.** Кроме первичной и вторичных групп в `/etc/passwd` и `/etc/group`, следует также проверять файл **/etc/security/group.conf** и настройки PAM в `/etc/pam.d/`. При входе пользователя в систему модуль `pam_group.so` временно предоставляет ему членство в указанных группах и не лишает его этого членства после истечения отведенного времени. Эта проблема аналогична описанной в требовании 8.5.6, однако она не исправляется с помощью сценария.

VIII. Начиная с ядра 2.6.24, в Linux появилась полноценная поддержка так называемых **capabilities** – административных полномочий, которые ранее были целиком сосредоточены у `root`, а теперь могут быть выданы определенному приложению и/или пользователю.

Механизм был реализован на основе черновика стандарта POSIX1.e. В настоящий момент имеется 35 административных полномочий (в описании **man capabilities**). Они определяют такие возможности, как работа с raw-сокетами, которая необходима, например, утилитам ping и tcpdump (CAP\_NET\_RAW) (для работы без capabilities для утилиты ping установлен SUID-флаг), произвольное изменение владельцев и групп файлов (CAP\_CHOWN), отправка процессам произвольных сигналов (CAP\_KILL) и т.п. За реализацию данного механизма разграничения прав доступа отвечает грм-пакет libcap.

На практике это позволяет избавиться от SUID-привилегий при запуске утилит passwd, ping и т.п. [8, 9 и 10], а также частично избежать использования sudo. При этом, в отличие от традиционных механизмов SUID/SGID/sudo, процесс получает не все административные привилегии, а только те, которые ему необходимы для выполнения определенных действий. Предположим, например, что пользователю user необходимо запускать сниффер tcpdump, для которого требуется привилегия CAP\_NET\_RAW. Чтобы реализовать такую возможность, не устанавливая SUID-бит на приложение, необходимо выполнить следующие действия (пример приведен для FC12, как для дистрибутива с версией ядра >= 2.6.24).

**Примечание.** В RHEL5 задать привилегии для исполняемого файла нельзя. Предлагаемые обертки (утилиты susar и exessar) не выполняют своей главной задачи: они не дают пользователю возможность выполнять определенные команды с указанным набором привилегий. Более того, setrcaps не реализует заявленный функционал и не позволяет изменить привилегии для уже запущенного процесса.

1. Для файла /usr/sbin/tcpdump устанавливаются effective- и inheritable-привилегии (подробнее см. man):

**setcap cap\_net\_raw=ei /usr/sbin/tcpdump**

2. Система настраивается для выдачи привилегий пользователям при входе в систему. В файле **/etc/pam.d/system-auth** должно быть указано (выделенная строка):

```
auth    required    pam_env.so
auth    required    pam_cap.so
```

Затем при необходимости те же действия выполняются для файла **/etc/pam.d/sshd** (если в нем не указано «auth include system-auth», в т.ч. на FC12); приведенная строка должна содержаться в том же блоке, что и другие «auth required» (как правило, не ниже третьей строки).

3. Пользователю user после входа в систему выдается привилегия CAP\_NET\_RAW; для этого необходим файл **/etc/security/capabilities.conf**:

```
cap_net_raw      user
none           *
```

Здесь в первой строке user получает привилегию, а во второй строке запрещается выдача привилегий всем остальным пользователям.



4. После входа в систему пользователь может запускать сниффер, т.к. у него есть все необходимые для этого привилегии:

```
[user@rf12 ~]$ getpcaps $$  
Capabilities for `24591': = cap_net_raw+i  
[user@rf12 ~]$ getcap `which tcpdump`  
/usr/sbin/tcpdump = cap_net_raw+ei  
[user@rf12 ~]$ tcpdump -vnni lo  
tcpdump: listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes
```

5. Все остальные (непривилегированные) пользователи не смогут запустить сниффер:

```
[superadmin@rf12 ~]$ tcpdump -vnni lo  
tcpdump: lo: You don't have permission to capture on that device  
(socket: Operation not permitted)
```

Данный механизм имеет два недостатка, общие с ACL. Так, стандартные файловые утилиты нуждаются в дополнительных ключах для сохранения привилегий [10]:

```
cp --preserve=all acl_cap/tcpdump ./  
rsync -auX acl_cap/tcpdump ./
```

Утилита **tar** (по крайней мере, версии 1.22-8 в FC12) не позволяет сохранять POSIX capabilities в архивы.

Кроме того, файловые привилегии исчезают у приложений после обновления ПО:

```
[root@rf12 security]# setcap cap_net_raw=eip /sbin/rdisc  
[root@rf12 security]# getcap /sbin/rdisc  
/sbin/rdisc = cap_net_raw+eip  
[root@rf12 security]# rpm --quiet -U --force /mnt/cdrom/Packages/iputils-20071127-9.fc12.i686.rpm  
[root@rf12 security]# getcap /sbin/rdisc  
[root@rf12 security]#
```

Итак, механизм привилегий имеет следующие недостатки:

1. Необходимость указывать дополнительные ключи для команд или вносить alias'ы в /etc/bashrc.
2. Удаление файловых привилегий при обновлении ПО.
3. Несмотря на важность этого механизма, он недостаточно документирован (отсутствие man-страниц по команде getpcaps и файлу настройки capability.conf), с ним знакомы далеко не все ИТ-специалисты. Механизм достаточно сложен в настройке и имеет множество



«подводных камней» [10 и 11], из-за чего его использование удобно скорее для нарушителей (в том числе при создании бэкдоров), нежели для администраторов.

4. Неправильная настройка привилегий программ представляет большую угрозу для безопасности всей системы, сопоставимую с таковой для SUID-приложений [11]. При этом механизм SUID более отлажен и лучше документирован.

5. Capabilities не могут быть установлены для shell-сценария, в отличие от SUID/SGID (впрочем, в некоторых ОС, например, в AIX, сценарий не может иметь и привилегии SUID/SGID).

IX. **Способ входа в систему** – еще одно средство контроля доступа. Модуль pam\_access.so (файл настройки /etc/security/access.conf) позволяет указать, каким пользователям и группам следует предоставлять доступ к системе в зависимости от источника попытки входа (имени терминала, адреса узла, имени домена или сети с маской). Например, чтобы доступ по ssh имел только пользователь user (на время забудем, что для этого также может применяться директива AllowUsers из /etc/ssh/sshd\_config), причем только с определенного узла, в /etc/security/access.conf необходимо указать:

```
+ : user : 192.168.6.50
```

```
# Полный запрет по умолчанию ДОЛЖЕН идти последней строкой!
```

```
- : ALL : ALL
```

X. Кроме того, можно контролировать **время входа в систему**. Для этого существует PAM-модуль pam\_time.so, конфигурационный файл которого - /etc/security/time.conf. Если указать в данном файле:

```
sshd;*;user;!A12000-0830
```

это будет означать, что к службе sshd (первое поле) со всех консолей (второе поле) пользователю user (третье поле) будет запрещен (символ «!») доступ во все дни недели ("A1") с 20:00 до 08:30. Конфигурационный файл содержит довольно подробные комментарии.

Для того, чтобы некоторая служба использовала описанный механизм, необходимо подключить к нему pam\_time.so в соответствующем файле настроек из каталога /etc/pam.d. Для SSHD это будет файл /etc/pam.d/sshd:

```
...  
account required pam_nologin.so  
account required pam_time.so  
...
```

Если требуется настроить общесистемный доступ, то в time.conf в поле службы указывается значение «\*»; если настройки касаются всех пользователей, то в третьем поле также

указывается значение «\*»; после этого необходимо добавить в файл **/etc/security/system-auth** следующую строку:

```
...  
account required pam_unix.so  
account required pam_time.so  
...
```

**ВНИМАНИЕ!** Как честно предупреждают заголовки файлов из /etc/pam.d/, автоматизированные средства настройки PAM могут затереть внесенные администратором изменения.

#### Список источников:

1. [https://bugzilla.redhat.com/show\\_bug.cgi?id=521173](https://bugzilla.redhat.com/show_bug.cgi?id=521173)
2. <http://phaq.phunsites.net/2008/02/04/enabling-reiserfs-xfs-jfs-on-redhat-enterprise-linux/>
3. <http://www.john-isaac.com/2009/01/xfs-support-for-rhel5/>
4. <http://press.redhat.com/2010/04/21/red-hat-enterprise-linux-6-beta-available-today-for-public-download/>
5. <http://www.redhat.com/docs/manuals/enterprise>
6. [http://www.redhat.com/docs/en-US/Red\\_Hat\\_Enterprise\\_Linux/6-Beta/](http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6-Beta/)
7. [http://docs.fedoraproject.org/en-US/Fedora/13/pdf/SELinux\\_FAQ/Fedora-13-SELinux\\_FAQ-en-US.pdf](http://docs.fedoraproject.org/en-US/Fedora/13/pdf/SELinux_FAQ/Fedora-13-SELinux_FAQ-en-US.pdf)
8. Использование разрешений POSIX в Linux – <http://posix.ru/freenotes/linux/51>
9. Разрешения POSIX: root без привилегий – <http://posix.ru/freenotes/linux/93>
10. POSIX Capabilities & File POSIX Capabilities – <http://www.friedhoff.org/posixfilecaps.html>
11. [http://www.sevagas.com/IMG/pdf/exploiting\\_capabilities\\_the\\_dark\\_side.pdf](http://www.sevagas.com/IMG/pdf/exploiting_capabilities_the_dark_side.pdf)

## 10 Требование. Назначить уникальный идентификатор каждому лицу, имеющему доступ к информационной инфраструктуре

---

### Краткое содержание

Большая часть требований, описанных в этой главе, относится к парольной политике Linux, хорошо описанной в CIS.

### 8.3 Убедиться в использовании двухфакторной аутентификации при удаленном доступе сотрудников путем наблюдения за подключающимся удаленно к внутренней сети сотрудником (например, администратором сети).

Базовым примером двухфакторной аутентификации при удаленном доступе является применение ssh-сертификатов. Для входа на удаленный сервер требуется не только предоставить файл («у меня есть»), но и ключ его расшифровки («я знаю»). Ключевая пара может храниться не только в виде файла, но и на смарт-карте.

Ключевая пара генерируется командой `ssh-keygen`, например:

```
ssh-keygen -t rsa -b 4096 -f /home/user/.ssh/remoteuser_at_remotehost
```

При этом в целевом каталоге (по умолчанию это `$HOME/.ssh/`) появятся два файла — файл, имя которого было задано (файл закрытого ключа), и файл с расширением `.pub` (открытый ключ). По умолчанию (для протокола 2) в зависимости от выбранного алгоритма создается файл `id_rsa[.pub]` или `id_dsa[.pub]`. Для кэширования расшифрованных ключей применяется утилита `ssh-agent`.

Парольная фраза закрытого ключа должна быть не короче 5 символов, но ключевая пара может быть создана с пустой парольной фразой. Более того, стандартные средства (в

отличие от пользовательских паролей) не позволяют задать требования к парольной фразе. Тем не менее, по файлу закрытого ключа можно определить, задана ли для него парольная

фраза (вторая строка закрытого ключа должна содержать запись «ENCRYPTED»), выполнив следующие действия от имени *root*:

```
for key in `find /home -type f \( -regex '.*/.ssh/.*' -a ! -name '*known_hosts*'
-a ! -name '*.pub' \)`;
do
found="`awk '(NR == 2) && /ENCRYPTED/ {print;}' "$key"`";
[ -n "$found" ] && echo "Key $key is OK" || echo "Key $key lacks its
passphrase";
done
```

Кроме описанного выше способа, для SSH существуют и другие реализации двухфакторной аутентификации, в том числе построенные на USB-токенах и смарт-картах.

Мировыми лидерами в этой области, поддерживающими Linux, являются компании Aladdin и Rainbow; на российском рынке также представлена продукция фирмы «Актив». Впрочем, указанные решения не входят в стандартную поставку RHEL-дистрибутивов и поэтому здесь не рассматриваются.

**8.4.a Для нескольких системных компонентов изучить файлы паролей и убедиться в том, что пароли нечитаемы при передаче и хранении.**

**8.4.b Для поставщиков услуг: изучить файлы паролей, убедиться в том, что клиентские пароли зашифрованы.**

Механизм локальной аутентификации Linux по умолчанию предполагает использование хэшированных (MD5, SHA-1, SHA-512) или шифрованных (DES, Blowfish и т.п.) паролей. Убедиться в том, что они действительно применяются, можно с помощью команды (на FC12):

```
[root@blacknet pam.d]# authconfig --test
...
shadow passwords are enabled
password hashing algorithm is sha512
...
```

**Примечание.** RHEL4 при этом отобразит на экране псевдографическое меню, в котором необходимо выбрать пункт «Cancel», после чего команда отобразит настройки в консоли аналогично приведенному выше примеру.

Соответствующие настройки хранятся в файле **/etc/sysconfig/authconfig**, за это отвечают следующие две строки:

```
USESHADOW=yes  
PASSWDALGORITHM=sha512
```

**Внимание.** Следует помнить о том, что:

а) алгоритм хэширования указывается не только здесь, но и в файлах настройки PAM, находящихся в **/etc/pam.d/** (как правило, это аргумент модуля **pam\_unix.so**);

б) в случае смены алгоритма хэширования паролей необходимо следить за синхронизацией конфигурационных файлов.

Кроме того, важно следить за содержимым файла **/etc/login.defs** (используемого при манипуляциях с пользователями), в котором также задается алгоритм хэширования паролей.

### **8.5.3 Убедиться в том, что для первого входа в систему пользователю устанавливается уникальный первоначальный пароль, который изменяется при первом входе в систему.**

Стандартные механизмы Linux не предусматривают возможности такой настройки. Другие UNIX-системы могут уже изначально отвечать этому требованию; например, в AIX при смене пользовательского пароля администратором для пользовательского пароля устанавливается флаг «Изменено администратором», и при следующем заходе пользователь должен будет сменить пароль.

Однако такое же поведение можно настроить и для Linux; для этого при создании пользователя необходимо выполнить следующие команды:

```
useradd [options] newuser  
passwd newuser  
usermod -L newuser  
chage -d 0 newuser  
usermod -U newuser
```

После этого *newuser* будет обязан при следующем входе в систему сменить пароль в соответствии с настроенной парольной политикой (подробнее об этом см. ниже в главе 8). Команды *usermod* (блокировка/разблокировка) применяются для того, чтобы пользователь не успел войти в систему с помощью первичного пароля, назначенного ему администратором.

Чтобы автоматизировать создание пользователей, можно написать для команды *passwd* обертку, которая будет выполнять «*chage -d0*» для указанного пользователя после сеты

пароля, если команда была вызвана суперпользователем *root*. При этом необходимо помнить о том, что если обертка располагается в одном каталоге с *passwd*, то очередное обновление ПО уничтожит данные изменения.

#### **8.5.4 Выбрать несколько уволенных за прошедшие шесть месяцев сотрудников и проанализировать списки контроля доступа, убедиться в том, что их учетные записи заблокированы.**

Данное требование выполняется просто. Для выбранных пользователей необходимо проверить, что (один из вариантов):

а) запись удалена из системы (нет такого пользователя в */etc/passwd*) либо

б) запись заблокирована (второе поле файла */etc/shadow* начинается с символа «!», за которым идет хэш пароля, или целиком состоит из «!!»).

Если пользователь не удален, а заблокирован, то можно проверить, имеются ли на диске данные, принадлежащие этому пользователю. Хотя эта проверка и выходит за пределы требования, она может быть весьма полезна:

```
for bu in $blockedusers; do
    find / \( -path /proc -o -path /sys -o -path /srv -o -path /dev -o -path
/selinux \) -prune \
    -o -user $bu -printf "%u %p\n";
done
```

#### **8.5.5 Убедиться в том, что неактивные более 90 дней учетные записи удаляются или блокируются.**

Прямой реализации данного механизма в Linux нет, однако можно решить эту проблему, используя сроки смены пароля. Для этого уже созданным пользователям назначается максимальный срок действия пароля, равный 90 дням (*-M90*), и отводится нулевой срок (*-I0*) на смену пароля после его устаревания:

```
chage -I0 -M90 -m7 -W14 user
```

В данном примере также указан минимальный интервал между изменениями пароля (7 дней), а также момент начала отправки пользователю предупреждений о смене пароля (за 14 дней до окончания срока действия пароля).

Чтобы эти настройки применялись для новых пользователей, требуется отредактировать следующие файлы:

**/etc/default/useradd:**

INACTIVE=0

**/etc/login.defs:**

PASS\_MAX\_DAYS 90

PASS\_MIN\_DAYS 7

PASS\_WARN\_AGE 14

**Внимание.** Важно понимать, в чем состоит отличие между этими двумя конфигурационными файлами. /etc/default/useradd относится ТОЛЬКО к команде useradd, а/etc/login.defs – ко всем командам над пользователями. При работе с useradd часто требуется править ОБА файла.

Для изменения значений по умолчанию также можно воспользоваться командой **useradd -D [-параметр значение]**

Впрочем, данная команда позволяет изменить только содержимое /etc/default/useradd.

### **8.5.6 Убедиться в том, что учетные записи, используемые поставщиками для удаленной поддержки, заблокированы и активируются только на время выполнения работ, в течение которого непрерывно контролируются.**

Для пользователей сторонних организаций (интеграторов, соисполнителей, сторонних разработчиков) следует применить механизмы ограничения доступа, основанные на анализе:

- а) способа входа в систему (откуда пользователь может входить в систему);
- б) времени входа в систему (в какое время он может это делать – например, только в рабочее время или только на выходных).

Оба пункта описаны в требовании 7.2.1 (IX, X), однако здесь стоит внести следующие дополнения:

1. Учетные записи пользователей сторонних организаций удобнее всего объединить в одну группу (или набор групп), так как это упростит их настройку.

2. Доступ по умолчанию для таких пользователей должен быть запрещен («что явно не разрешено – то запрещено»).

3. По умолчанию учетные записи должны быть заблокированы (см. п. 8.5.4.2).

4. Для контроля над действиями сторонних пользователей требуется настроить подсистему аудита уровня ядра auditd, которая будет подробно описана в главе 10.

Предположим, что члены групп `partners` и `support` имеют право входить в систему только в будние дни с 9 до 18 часов из сети `a.b.c.d/w.x.y.z`. В этом случае правила доступа к системе могут быть записаны в следующем виде:

```
/etc/pam.d/{system-auth,sshd} :  
auth          required    pam_access.so nodefgroup  
...  
account       required    pam_time.so
```

Отдельная настройка `sshd` не требуется, если в нем указано «`auth include system-auth`».

Параметр «`nodefgroup`» нужен, чтобы указывать имена групп в скобках и не путать их с пользователями.

```
/etc/security/access.conf:  
- : (support) (partners) : ALL EXCEPT a.b.c.d/w.x.y.z  
Или (если настроена строгая политика доступа для всех учетных записей):  
+ : (support) (partners) : a.b.c.d/w.x.y.z  
... (настройки остальных пользователей) ...  
- : ALL : ALL
```

`/etc/security/time.conf` – файл не позволяет задавать системные группы, необходимо явно указывать все учетные записи сторонних пользователей (предположим, это пользователи `user1`, `user2`, `user3`):

```
sshd;pts*;user1 | user2 | user3;Wk0900-1800
```

Из синтаксиса правил видно, что `time.conf` имеет более «запретительный» характер, т.к. при нахождении записи о пользователях модуль будет предоставлять доступ ТОЛЬКО в то время, которое указано в настройках.

**Примечание.** Текущая реализация `pam_time` ограничивает только время входа в систему, но не длительность сессии. Таким образом, если член группы `support` войдет в систему в 17:59, то дальше он сможет работать неограниченно долго. Чтобы реализовать функцию ограничения длительности сессии, можно написать дополнительные сценарии, которые будут отслеживать процессы указанных пользователей и принудительно завершать их в «неположенное» согласно `time.conf` время.

### 8.5.8.a Для нескольких системных компонентов проверить списки учетных записей пользователей и проверить следующее:

- стандартные учетные записи заблокированы или не используются;
- разделяемые учетные записи для функций администрирования и иных критичных функций не существуют;
- разделяемые и стандартные учетные записи не используются для администрирования какого-либо системного компонента.

Для выполнения этого требования необходимо запретить вход в систему пользователю *root*, после чего можно будет отслеживать действия каждого администратора; в противном случае не удастся выяснить, кто именно произвел какие-либо действия от имени *root*.

Требуемая настройка затрагивает как минимум два конфигурационных файла – */etc/ssh/sshd\_config* и */etc/security/access.conf*.

Под стандартными учетными записями могут пониматься как учетные записи, используемые приложениями (*daemon*, *bin*, *nobody*, *ftp* и т.п.), так и учетные записи с легко предсказуемыми именами (*admin*, *user*, *test* и т.д.). Проще всего проверить факт блокировки, проанализировав второе поле в файле */etc/shadow* – если значение равно «\*» или начинается с «!», то соответствующий пользователь заблокирован. Также пользователям может быть запрещен вход в систему по причине просроченного пароля, превышения лимита неудачных попыток входа или на основании ограничения времени доступа (*/etc/security/time.conf*).

**8.5.9 Для нескольких системных компонентов проверить настройки и убедиться в том, что пользователь должен менять пароль не реже одного раза в 90 дней. Для поставщиков услуг изучить внутренние процессы и клиентскую документацию, убедиться в том, что клиентский пароль должен меняться регулярно и у клиентов есть инструкция о том, когда и при каких обстоятельствах пароль должен быть изменен.**

Требование соответствует CIS п.9.3. Как и в п.8.5.5 PCI DSS, для каждого пользователя необходимо выполнить:

**chage -I0 -M90 -m7 -W14 \$USERNAME**

После этого в файлы конфигурации системы вносятся изменения для применения указанных параметров к вновь создаваемым пользователям (если не переопределять ключи запуска команды *useradd*).

***/etc/default/useradd:***

*INACTIVE=0*

***/etc/login.defs:***

*PASS\_MAX\_DAYS 90*



PASS\_MIN\_DAYS 7  
PASS\_WARN\_AGE 14

**8.5.10 Для нескольких системных компонентов проверить настройки и убедиться в том, что длина пароля не менее семи символов. Для поставщиков услуг изучить**

**внутренние процессы и клиентскую документацию, убедиться в том, что к клиентским паролям предъявляется требование минимальной длины.**

В файле `/etc/pam.d/system-auth` у модуля `pam_cracklib.so` должен быть указан аргумент:  
**minlen=7**

Кроме того, следует убрать аргумент «nullok» у модуля `pam_unix.so` в файлах из каталога `/etc/pam.d`, т.к. данный параметр разрешает использование пустых паролей. Аналог в CIS – пункт 11.4.

**Примечание 1.** Проверить длину уже созданных паролей невозможно. Тем не менее, вместо этого можно обязать всех пользователей сменить пароли (несложный сценарий с вызовом «**chage -d0 \$USERNAME**» для каждого найденного в системе пользователя), предварительно установив параметр `minlen`.

**Примечание 2.** *Root* может задать для пользователя пароль, не удовлетворяющий требованиям сложности.

**8.5.11 Для нескольких системных компонентов проверить настройки и убедиться в том, что пароль должен содержать как цифровые, так и буквенные символы. Для поставщиков услуг изучить внутренние процессы и клиентскую документацию, убедиться в том, что пароль должен содержать как цифровые, так и буквенные символы.**

Настройка системы в соответствии с данным требованием аналогична п. 8.5.10, но здесь используются другие аргументы:

**dcredit=-1 lcredit=-1 ucredit=-1**

Данная строка определяет, что пароль должен содержать не менее одного заглавного символа, не менее одного строчного символа и не менее одной цифры. Для данного модуля PAM невозможно указать количество букв напрямую (только отдельно строчные и заглавные).

Аналог CIS – пункт 11.4. Также см. примечания к п. 8.5.10.

**8.5.12 Для нескольких системных компонентов проверить настройки и убедиться в том, что при изменении новый пароль должен отличаться от четырех предыдущих. Для поставщиков услуг изучить внутренние процессы и клиентскую документацию, убедиться в том, что при изменении новый пароль должен отличаться от четырех предыдущих.**

Требуется создать файл **/etc/security/opasswd**, установить на него права доступа 600, а затем указать в файле **/etc/pam.d/system-auth** параметр для модуля **pam\_unix.so** в строке «password ... pam\_unix.so ...»:

**remember=4**

После этого хэши всех устанавливаемых пользователями паролей будут сохраняться в указанном файле.

Стандарт CIS RHEL v1.1.2 не дает рекомендаций по хранению истории паролей.

**Примечание.** Если пароль пользователя меняет *root*, новое значение в указанный файл не записывается.

**8.5.13 Для нескольких системных компонентов проверить настройки и убедиться в том, что учетная запись пользователя блокируется после максимум шести неудачных попыток входа. Для поставщиков услуг изучить внутренние процессы и клиентскую документацию, убедиться в том, что учетная запись клиента временно блокируется после максимум шести неудачных попыток входа.**

Для каждого поколения RHEL – 4, 5, 6 (FC12) – настройки будут отличаться. Рассмотрим их в порядке 4, 6, 5.

Для RHEL4 файл **/etc/pam.d/system-auth** должен содержать следующие строки:

- **auth required pam\_tally.so onerr=fail no\_magic\_root**

Строка должна быть помещена ПЕРЕД строкой

```
auth required /lib/security/$ISA/pam_deny.so
```

- **account required pam\_tally.so deny=5 no\_magic\_root reset**

Строка должна быть помещена ПЕРЕД строкой

```
account required /lib/security/$ISA/pam_permit.so
```

Указано именно 5 неудачных попыток входа, т.к. логика работы модуля добавляет 1 к заданному значению.

В RHEL4 настройки PAM для sshd по умолчанию берутся из системных, поэтому редактировать их не нужно.

В FC12 (и в RHEL6) настройка производится следующим образом:

Файл **/etc/pam.d/system-auth** должен содержать следующую строку (выделено жирным шрифтом):

```
auth    required    pam_env.so
auth    required    pam_tally2.so onerr=fail deny=6 unlock_time=1800
```

Строка должна идти в блоке «auth required», т.к. порядок следования модулей важен. Время снятия блокировки указывается в секундах; если параметр unlock\_time не указан, то снять блокировку можно только в ручном режиме.

Чтобы блокировка работала и при удаленном входе в систему, файл /etc/pam.d/sshd должен содержать следующую строку:

```
auth include          system-auth
или запись, аналогичную приведенной выше.
```

Для RHEL5 допустимо использование как pam\_tally, так и pam\_tally2. В первом случае необходимо задать опции аналогично pam\_tally2 для RHEL6, в тех же конфигурационных файлах и с теми же ограничениями на положение строки в файле:

```
auth    required    pam_tally.so onerr=fail deny=6
```

(новая версия модуля pam\_tally не прибавляет 1 к указанному значению)

Во втором случае (pam\_tally2) настройки полностью совпадают с настройками pam\_tally2 для RHEL6.

**Внимание.** Настройки, указанные в пункте SN.8 стандарта CIS RHEL v1.1.2, не подходят для RHEL5 и FC12.

**Примечание.** При использовании настроек по умолчанию неверно введенный пароль для sudo приравнивается к неудачной попытке входа, однако в случае неверного ввода пароля при выполнении passwd счетчик не увеличивается.

Кроме того, если *user1* попытается выполнить переключение на заблокированного *user2*:

#### **su - user2**

счетчик неудачных попыток входа *user2* также увеличится.

### **8.5.14 Для нескольких системных компонентов проверить настройки и убедиться в том, что учетная запись пользователя блокируется не менее чем на 30 минут либо пока администратор не снимет блокировку.**

Здесь, так же, как и в предыдущем пункте, настройка зависит от версии дистрибутива. В RHEL4 модуль `pam_tally` не может сам снимать блокировку, это необходимо делать принудительно, используя команду:

```
pam_tally --user=$USERNAME --reset
```

Для FC12 (RHEL6) файл `/etc/pam.d/{system-auth,sshd}` должен содержать следующую строку:

```
auth required pam_tally2.so onerr=fail deny=6 unlock_time=1800
```

Эта строка указывает на то, что пользователь будет разблокирован спустя 1800 секунд после попытки входа, из-за которой он был заблокирован. Если параметр `unlock_time` не задан,

модуль не будет автоматически разблокировать пользователей, и для снятия блокировки в ручном режиме (что допускается стандартом) потребуется выполнить:

```
pam_tally2 --user=$USERNAME --reset
```

Для RHEL5 допустимо снимать блокировку обоими способами в зависимости от того, какой из них настроен в системе.

**Примечание.** Если пользователь попытается войти в систему до того, как истечет период `unlock_time` (или до того, как *root* разблокирует его в ручном режиме), счетчик неудачных попыток входа будет расти, и время автоматической разблокировки также будет откладываться.

В приведенном ниже примере пользователь блокируется после 3 неудачных попыток входа, время разблокировки — 30 секунд. Если попытаться войти в систему раньше (через 20 секунд), срок разблокировки отодвинется.

```
[root@blacknet pam.d]# grep pam_tally2 /etc/pam.d/sshd  
auth required pam_tally2.so onerr=fail deny=3 unlock_time=30
```



```
[feodor@blacknet .ssh]$ ssh user@localhost
user@localhost's password:
Last login: Wed Jul  7 11:01:23 2010 from localhost.localdomain
[user@blacknet ~]$ sudo date
[sudo] password for user:
Sorry, try again.
[ ... ]
sudo: 3 incorrect password attempts
[user@blacknet ~]$ logout
Connection to localhost closed.
[feodor@blacknet .ssh]$ sleep 20; ssh user@localhost
user@localhost's password:
[ ... ]
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
[feodor@blacknet .ssh]$ sleep 20; ssh user@localhost
user@localhost's password:
Permission denied, please try again.
[ ... ]
Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
[feodor@blacknet .ssh]$ sleep 35; ssh user@localhost
user@localhost's password:
Last login: Wed Jul  7 11:03:03 2010 from localhost.localdomain
[user@blacknet ~]$
```

### **8.5.15 Для нескольких системных компонентов проверить настройки и убедиться в том, что рабочая сессия пользователя блокируется не более чем через 15 минут простоя.**

Время блокировки графической сессии KDE4 задается каждым пользователем отдельно и указывается в файле **\$HOME/.kde/share/config/kcreensaverrc**. За блокировку сессии отвечают следующие параметры:

**Enabled=true**

**Lock=true**

**LockGrace=X**

**Timeout=Y**

Здесь X – время в миллисекундах между запуском заставки и блокировкой экрана, Y – время простоя в секундах, через которое запускается заставка. Согласно требованию,  $Y+(X/1000) < 900$ .

# 11 Требование. Контролировать и отслеживать любой доступ к сетевым ресурсам и данным о держателях карт

---

## Краткое содержание

Технически реализуемые требования этой главы относятся к серверу syslog, подсистеме аудита уровня ядра auditd, настройкам NTP-сервера и системе контроля целостности. Аналогичных пунктов в CIS практически нет.

## 10.2 Методом интервью, изучения журналов протоколирования событий и настроек систем протоколирования осуществить следующие проверки

Требования пунктов 10.2.X относятся к системе auditd, которая должна быть установлена (пакет audit) и запущена (chkconfig auditd on). Правила аудита (определяющие, какую информацию необходимо фиксировать в журнале) должны быть записаны в файл /etc/audit/audit.rules, настройки демона задаются в /etc/audit/auditd.conf. Для обоих этих файлов следует задать разрешения доступа chmod 600. По умолчанию журналы регистрации хранятся в каталоге /var/log/audit (хотя путь может быть переопределен в auditd.conf), который часто располагают на отдельном разделе. Файлы аудита записываются в текстовом формате, однако для просмотра событий лучше воспользоваться утилитами aureport и ausearch. Временные (до перезапуска демона) правила аудита можно задавать с помощью команды auditctl; хотя синтаксис правил это команды не отличается от audit.rules, man auditctl гораздо более информативен.

В первую очередь, в **/etc/audit/audit.rules** следует указать правило фильтрации ненужных сообщений типа cwd (change working directory), которое должно идти первой строкой:

**-a exclude,always -F msgtype=CWD**

Чтобы правила вступили в силу после изменения конфигурационного файла, необходимо выполнить:

**service auditd reload**

Для функционирования системы аудита необходимо также наличие модуля `pam_loginuid`, указанного в строке:

### **session required pam\_loginuid.so**

в файлах `/etc/pam.d/{atd,crond,kdm,kdm-np,login,remote,sshd,xdm}` (список верен для FC12). Кроме того, можно запретить пользователям входить в систему, если не запущен `auditd`, при помощи параметра «`require_auditd`» модуля `pam_loginuid`.

Также требуется включить поддержку аудита во время загрузки ОС, добавив опцию ядра в `/etc/grub.conf`:

**audit=1**

### **10.2.2 Убедиться в том, что любые действия, совершенные с использованием административных полномочий, регистрируются.**

Такое правило может быть записано в `/etc/audit/audit.rules` в виде:

**-a always,exit -F euid=0 -F perm=wxa -k ROOT\_ACTION**

При этом синтаксис правил допускает «перемену мест слагаемых». Так, элемент «`always,exit`» может быть записан как «`exit,always`»; «`euid=0`» может быть записан как «`euid=root`»; в подстроке «`perm=wxa`» блок «`wxa`» может быть записан как «`axw`» и т.п. Детально синтаксис правил описан в `man auditctl`.

В приведенном правиле указано, что всегда (`always`) отражаются API-вызовы (`exit`), совершенные пользователями с EUID = 0 (в том числе теми, кто получил права `root` с помощью `su/sudo`), связанные с выполнением команд (`x`), записью данных (`w`) и изменением атрибутов файлов/каталогов (`a`). Всем таким записям присваивается метка `ROOT_ACTION`, по которой события можно искать и отображать. В приведенном ниже примере видно, что файл `/tmp/audit_me` сначала создается сценарием (записи 1 и 2), после чего открывается в `vi` и перезаписывается (записи 3-14).

```
[root@rf12 audit]# ausearch -ts today -k ROOT_ACTION -f audit_me | aureport -i -f
```

#### File Report

```
=====
# date time file syscall success exe auid event
=====
1. 07/07/10 14:15:50 /tmp/audit_me open yes /bin/bash root 87709
2. 07/07/10 14:15:58 /tmp/audit_me rename yes /bin/sed root 87719
3. 07/07/10 14:51:51 /tmp/.audit_me.swpx open yes /bin/vi root 88098
[ ... ]
13. 07/07/10 14:51:52 /tmp/audit_me chmod yes /bin/vi root 88108
14. 07/07/10 14:51:54 /tmp/.audit_me.swp unlink yes /bin/vi root 88109
```

### 10.2.3 Убедиться в том, что факты доступа к записям о событиях в системе регистрируются.

В файл `/etc/audit/audit.rules` для каждого файла или каталога (как правило, это `/var/log` и все объекты в нем), за которым ведется наблюдение, необходимо внести определенные записи, для которых допускается два вида синтаксиса. Первый из них:

```
-w /path/to/file_or_directory -p rwx
```

Здесь `rwx` – возможные триггеры (чтение/запись/выполнение/доступ) (`man auditctl`). Второй вид синтаксиса (отличается для файлов и каталогов):

```
-a exit,always -S all -F dir=/path/to/dir
```

или

```
-a exit,always -S all -F path=/path/to/file
```

### 10.2.4 Убедиться в том, что неуспешные попытки логического доступа регистрируются.

### 10.2.5 Убедиться в том, что регистрируются факты использования механизмов идентификации и аутентификации.

Такие сообщения обрабатываются демоном `syslog`, а не `auditd`. Все события доступа к системе по умолчанию отражены в `/var/log/secure`, т.к. `/etc/syslog.conf` настроен на добавление сообщений «`authpriv.*`» именно в этот файл журнала.

На FC12 вместо `syslog` установлен `rsyslog`, файл его настройки `/etc/rsyslog.conf` допускает тот же синтаксис, что и в `syslog.conf`, позволяя указывать расширенные параметры (рассмотрение которых выходит за рамки данной статьи).

### 10.2.6 Убедиться в том, что регистрируются факты инициализации журналов протоколирования событий.

Настройка `auditd` согласно п. 10.2.3 предполагает регистрацию VCEX системных вызовов, затрагивающих файлы журналов, в том числе связанных с удалением (API-функции `rmdir`, `unlink`) и перезаписью/обнулением (`truncate`, `open` с флагами `O_RDWR`, `O_WRONLY`, `O_TRUNC`).

Второй вариант синтаксиса для этого требования (только для удалений, вызовы `open()` не поддерживаются) может выглядеть следующим образом:

```
[root@rf12 log]# grep LOGS_INIT /etc/audit/audit.rules  
-a exit,always -F dir=/var/log -S truncate -S unlink -S rename -S unlinkat -k  
LOGS_INIT  
[root@rf12 log]# pwd  
/var/log  
[root@rf12 log]# rm -f boot.log-201005*  
[root@rf12 log]# ausearch -ts today -k LOGS_INIT | aureport -i -f
```



## File Report

```
=====
# date time file syscall success exe auid event
=====
1. 07/07/10 19:06:00 boot.log-20100517 unlinkat yes /bin/rm root 89799
2. 07/07/10 19:06:00 boot.log-20100511 unlinkat yes /bin/rm root 89798
3. 07/07/10 19:06:00 boot.log-20100524 unlinkat yes /bin/rm root 89800
4. 07/07/10 19:06:00 boot.log-20100531 unlinkat yes /bin/rm root 89801
[root@rf12 log]#
```

**Примечание.** Когда параметрами команд файловых операций являются относительные пути, утилита `aureport` отображает их тоже как относительные. Чтобы узнать абсолютный путь, следует вызывать не `aureport` (ниже жирным шрифтом выделена команда и подробная информация о действиях):

```
[root@rf12 log]# ausearch -ts today -k LOGS_INIT
time->Wed Jul 7 19:06:00 2010
type=PATH msg=audit(1278515160.180:89798): item=1 name="boot.log-20100511"
inode=11777 dev=fd:01 mode=0100644 ouid=0 ogid=0 rdev=00:00
obj=system_u:object_r:var_log_t:s0
type=PATH msg=audit(1278515160.180:89798): item=0 name="/var/log" inode=7923
dev=fd:01 mode=040775 ouid=0 ogid=501 rdev=00:00 obj=system_u:object_r:var_log_t:s0
type=SYSCALL msg=audit(1278515160.180:89798): arch=40000003 syscall=301 success=yes
exit=0 a0=ffffff9c a1=bfa9a7b5 a2=0 a3=2 items=2 ppid=2604 pid=21257 auid=0 uid=0 gid=0
euuid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=112 comm="rm" exe="/bin/rm"
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key="LOGS_INIT"
[ ... ]
```

В последней строке также отражен AUID – UID, полученный при входе в систему и зарегистрированный модулем `ram_loginud`.

### 10.2.7 Убедиться в том, что регистрируются факты создания и удаления объектов системного уровня.

Ответ на вопрос, что именно считать объектами системного уровня, неоднозначен. Однако можно настроить регистрацию событий, связанных с файлами в каталогах `/etc`, `/bin`, `/sbin`, `/usr/bin`, `/usr/sbin`, `/var/lib`, `/lib`, `/usr/lib` и `/usr/libexec`, а для 64-битных ОС – и в каталогах `/lib64` и `/usr/lib64`. Кроме подсистемы аудита, рекомендуется настроить систему контроля целостности (AIDE, описана в пункте 10.5.5, по умолчанию контролирует все системные файлы).

Настройка аналогична п. 10.2.3 (первый вариант синтаксиса), например:

```
-w /etc -p wa
```

Такая настройка даст следующие записи в журнале аудита:

```
[root@rf12 audit]# touch /etc/auditme2
[root@rf12 audit]# rm -f /etc/auditme2
[root@rf12 audit]# ausearch -ts today -k ETC | aureport -i -f
```

## File Report

```
=====
# date time file syscall success exe auid event
=====
1. 07/07/10 18:58:05 /etc/auditme2 open yes /bin/touch root 89795
2. 07/07/10 18:58:08 /etc/auditme2 unlinkat yes /bin/rm root 89796
```

**10.3 Убедиться, что для каждого протоколируемого события регистрируются следующие параметры:**

### 10.3.1 Идентификатор пользователя.

Поле «auid» (UID при входе в систему) отображается в выходных данных aureport, поле euid (реальный UID, полученный в результате выполнения su/sudo/SUID/SGID) – в выходных данных ausearch.

### 10.3.2 Тип события.

Имя команды и API-вызовы также сохраняются и отображаются.

### 10.3.3 Дата и время.

Регистрируются и отображаются.

### 10.3.4 Успешным или неуспешным было событие.

В выходных данных ausearch присутствует строка «success=»; ausearch позволяет искать события с указанным статусом (удачные/неудачные: -sv yes | -sv no).

### 10.3.5 Источник события.

Указан в поле «tty=» (вывод команды ausearch). Информация о том, откуда и когда данный пользователь зашел на этот терминал, хранятся в **/var/log/secure**.

### 10.3.6 Идентификатор или название данных, системного компонента или ресурса, на которые повлияло событие.

Также отображается в результате выполнения ausearch.

**10.4 Проанализировать процесс получения и распространения точного времени в организации, равно как и связанные с этим конфигурационные параметры для выборки системных компонентов. Убедиться, что выполнены следующие требования:**

**10.4.a Для синхронизации часов используется NTP или схожая технология, удовлетворяющая требованиям 6.1 и 6.2 стандарта PCI DSS.**

Для выполнения этого требования необходимо установить пакет ntp (серверная часть, позволяющая не только указывать время клиентам, но и получать его от вышестоящих узлов) или ntpdate (для клиентов).

Самый простой сценарий синхронизации времени на стороне клиента – это периодический (с помощью cron) запуск ntpdate с указанием удаленных серверов синхронизации. Также можно использовать запуск «ntpd -q». В то же время, на FC12 ntpdate может запускаться из /etc/init.d для синхронизации времени при старте ОС, что больше подходит для клиентских узлов, нежели для серверов.

Для запуска демона NTPD при старте системы необходимо выполнить команду:

**chkconfig ntpd on**

Кроме того, требуется настроить параметры сервера в /etc/ntp.conf (эта настройка будет описана ниже).

**10.4.b Убедиться, что не все внутренние серверы получают информацию о времени из внешних источников (несколько центральных серверов времени в организации должны получать такую информацию и распространять на другие компьютеры в сети).**

Здесь требуется настроить внутренние серверы компании для получения времени от внешних источников (за исключением ситуации, когда компания имеет свой источник точного времени, что актуально, например, для сотовых операторов и удостоверяющих центров), а также указать разрешения на синхронизацию для клиентов. Все эти параметры задаются в файле /etc/ntp.conf.

Адреса внешних серверов указываются с помощью директивы:

**server a.b.c.d**

Здесь адрес сервера может быть представлен как доменным именем, так и IP-адресом.

Чтобы остальные узлы сети могли получить точное время с настраиваемого сервера, требуется разрешить им доступ, используя директиву:

**restrict e.f.g.h mask w.x.y.z nomodify notrap**

и указав адрес сети и ее маску. После выполнения этих операций и перезапуска ntpd клиенты могут проверить синхронизацию времени с помощью команды:

**ntpdate -b \$NTPSERVER**

Здесь \$NTPSERVER — адрес внутреннего NTP-сервера.



**Примечание.** Чтобы обеспечить работу NTP-серверов компании в случае, если канал связи с внешними NTP-серверами пропадет, а также для сетей, не связанных с сетью Интернет и другими источниками точного времени, на центральном NTP-сервере требуется указать:

**server 127.127.1.0**

**fudge 127.127.1.0 stratum 10**

После этого NTP-сервер будет снимать показания с системных часов как с источника точного времени, если остальные источники будут недоступны.

**10.4.с Убедиться, что внешние хосты, с которых сервера времени получают информацию, жестко определены (чтобы предотвратить смену времени злоумышленником). Эта информация может быть дополнительно зашифрована симметричным ключом и списками контроля доступа, определяющими адреса машин, которым разрешено получать обновления времени.**

Чтобы четко определить внешние узлы точного времени, требуется выполнить настройки из предыдущего пункта, а также, чтобы уменьшить риск атаки на DNS-серверы, указать внешние источники в формате IP-адреса, а не FQDN.

Списки контроля доступа указываются с помощью директивы `restrict`, как было показано в п. 10.4.b.

**10.5.1 Убедиться в том, что доступом к журналам протоколирования событий обладают только те сотрудники, которым такой доступ необходим в соответствии с их должностными обязанностями.**

Это требование выполняется аналогично пункту 7.1.1.I. Необходимо следить как за правами на чтение файлов и каталогов, так и за их владельцами и группами, а также за списками ACL.

**10.5.2 Убедиться в том, что актуальные журналы протоколирования событий защищены от неавторизованного изменения при помощи механизмов контроля доступа, физической и/или сетевой сегментации.**

Требование аналогично п. 10.5.1, однако здесь контролю подлежат права на запись.

**10.5.3 Убедиться в том, что резервные копии журналов протоколирования событий оперативно сохраняются на централизованный сервер протоколирования или отдельный носитель, где их изменение было бы затруднено.**

Журналы регистрации событий ведутся двумя демонами — `syslogd (rsyslogd)` и `auditd`.

Чтобы `syslogd` отправлял события на удаленный сервер в реальном времени, требуется на клиенте в файле `/etc/syslog.conf` (или `/etc/rsyslog.conf`) для каждой категории событий указать строку следующего вида:

**facility.priority @remote.host.address**



На сервере возможность принимать сообщения из сети обеспечивается с помощью опции запуска «r» (в файле **/etc/sysconfig/syslog**); для rsyslogd необходимо в файле **/etc/rsyslog.conf** вынести из комментариев следующие строки:

```
$ModLoad imudp.so  
$UDPServerRun 514
```

Удаленную пересылку журналов auditd настроить сложнее. Для этого потребуется установить дополнительный пакет audispd-plugins и его зависимости:

```
yum install audispd-plugins
```

После этого для **базовой** настройки пересылки журналов на стороне клиента (имя узла **fc12**) требуется внести изменения в следующие конфигурационные файлы:

```
/etc/audit/auditd.conf:  
name_format = none  
/etc/audisp/audispd.conf:  
name_format = HOSTNAME
```

В противном случае при удаленной отправке имя узла будет дублироваться. Затем:

```
/etc/audisp/audisp-remote.conf:  
remote_server = 10.111.113.27  
port = 64667  
/etc/audisp/plugins.d/au-remote.conf:  
active = yes
```

На стороне сервера (имя узла **sles11**, адрес 10.111.113.27) потребуется изменить всего один файл. Порт выбран произвольно, главное, чтобы он был свободен, и клиент был настроен так же.

```
/etc/audit/auditd.conf:  
tcp_listen_port = 64667
```

После перезапуска auditd на клиенте и сервере можно будет увидеть следующее:

```
[root@rf12 audit]# touch /etc/auditme.tmp  
sles11:/var/log/audit # tail -f audit.log  
[ ... ]
```

```
node=rf12 type=SYSCALL msg=audit(1278592521.992:90379): arch=40000003 syscall=5  
success=yes exit=3 a0=bfc647e8 a1=8941 a2=1b6 a3=ffffff items=2 ppid=2604 pid=3475  
aid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts3 ses=112  
comm="touch" exe="/bin/touch" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
key="ETC"
```



```
node=rf12      type=PATH      msg=audit(1278592521.992:90379):  item=0      name="/etc/"
inode=32642    dev=fd:04      mode=040755      ouid=0      ogid=0      rdev=00:00
obj=system_u:object_r:etc_t:s0
```

```
node=rf12      type=PATH      msg=audit(1278592521.992:90379):  item=1
name="/etc/auditme.tmp"  inode=5729 dev=fd:04 mode=0100644 ouid=0 ogid=0 rdev=00:00
obj=unconfined_u:object_r:etc_t:s0
[ ... ]
```

Для авторизации и аутентификации клиента может применяться Kerberos. Расширенная поддержка удаленной отправки и обработка журналов регистрации описана в [3].

### 10.5.5 Убедиться в наличии систем контроля целостности файлов для защиты журналов регистрации событий от несанкционированных изменений.

Основная система контроля целостности для Linux – это AIDE (пакет aide). Файл настройки системы – /etc/aide.conf; дополнительная информация доступна по man aide.conf.

Предполагается, что AIDE будет запускаться по cron (с ключом --check или --compare), периодичность запуска зависит от особенностей узла.

Настройки AIDE «из коробки» включают в себя отслеживание объектов в /var/log. Особенность этих файлов заключается в том, что они могут быть удалены или переименованы во время системной авторотации журналов (демон logrotate). Чтобы каждое выполнение logrotate не приводило к необходимости расследовать ряд инцидентов, перед настройкой AIDE следует ознакомиться с параметрами logrotate (файл /etc/logrotate.conf, каталог/etc/logrotate.d/).

**Внимание.** Подсчет контрольных сумм может занимать несколько минут полной загрузки одного ядра процессора. Например, на виртуальной машине (VMWare, одно ядро Intel Core2, 3.06 ГГц) подсчет контрольных сумм занимает порядка 7 минут. Так происходит потому, что, во-первых, AIDE не поддерживает многопоточность, во-вторых, для многих объектов хэши вычисляются не одним, а двумя или более алгоритмами, и, в-третьих, по умолчанию настроена проверка всех системных объектов (/etc, /boot, /bin, /usr, /sbin, /lib, /root). Так, нам удалось сократить время выполнения в 1.5 раза, указав в файле настройки пропуск каталога /usr/share.

### 10.7.в Убедиться, что журналы протоколирования событий доступны в течение одного года и находятся в оперативном доступе не менее трех месяцев.

Для выполнения этого требования представляется разумным настроить долговременное хранение информации на центральном сервере сбора журналов регистрации (т.к. согласно п. 10.5.5 должна быть настроена пересылка журналов на удаленный узел), причем при анализе данных удобнее использовать СУБД, а не текстовые файлы.

С примерами таких инсталляций можно ознакомиться в [1] и [2]. Основные преимущества СУБД для хранения журналов регистрации перед традиционным подходом – расширенные

возможности архивации и репликации, а также веб-ориентированное отображение с поддержкой фильтрации данных.

**Список источников:**

1. [http://wiki.rsyslog.com/index.php/Here comes the first story](http://wiki.rsyslog.com/index.php/Here_comes_the_first_story)
2. <http://rexser.kharkov.ru/2010/01/06/easyids-rsyslog/>
3. <http://people.redhat.com/sgrubb/audit/prelude.txt>

## **12** Требование. Поставщики услуг с общей средой должны защищать среду данных платежных карт

---

### Краткое содержание

Самый очевидный способ выполнить требования раздела А.1 – выделить для каждого клиента виртуальный сервер (или набор серверов), к которому должны будут предъявляться требования вышеописанных глав.

Аналогов для требований А.1 в CIS нет.

**А.1.1 Если хостинг-провайдер позволяет клиентам запускать приложения (например, скрипты), следует убедиться, что эти приложения запущены под уникальным идентификатором. Например:**

- Ни одно приложение и ни один пользователь не может использовать имени пользователя, от которого работает разделяемый веб-сервер.
- Все CGI-скрипты, используемые клиентом, должны быть созданы и запущены от имени идентификатора клиента.

Разграничение полномочий пользователей подробно описано в п. 7.2.1, большая часть настроек здесь может быть выполнена и проверена аналогичным образом (в том числе с помощью su/sudo и членства в группах). Отдельно следует остановиться на второй рекомендации данного пункта — в случае различных пользователей и общего веб-сервера можно установить флаг SUID на CGI-сценарии.

**А.1.2.а Убедиться, что ни один из клиентов не обладает правами администратора/суперпользователя.**

Здесь применяются те же рекомендации, что и в пункте 7.2.1.

**А.1.2.б Убедиться, что каждый клиент имеет права чтения, записи и выполнения только своих утилит и данных. Для этого ограничения могут применяться средства типа chroot, jail и т.п. ВАЖНО: файлы клиента не должны быть доступны группе пользователей.**

Основные рекомендации изложены в п. 7.1.1.

Для удобства управления пользователями их данные зачастую помещаются в пределах одного каталога верхнего уровня (по умолчанию /home), где легко проверить права доступа

к их домашним каталогам. В самом простом случае достаточно установить разрешения доступа 700 на каждый домашний каталог, а также убедиться в отсутствии у этого каталога списка ACL.

Для обеспечения дополнительного ограничения доступа следует указать umask 0077 в файле настройки соответствующей прикладной службы (/etc/bashrc, /etc/vsftpd/vsftpd.conf и т.п.).

Если структура пользовательских каталогов более сложна, можно воспользоваться собственными, созданными под конкретную задачу сценариями на основе команды find.

#### **A.1.2.c Убедиться, что у клиента отсутствует право записи в разделяемые системные библиотеки и исполняемые файлы.**

Задача, поставленная в данном пункте, решается с помощью сценария, использующего список пользователей и команду find, который будет рекурсивно просматривать каталоги, заданные регулярным выражением

```
(/usr(/local)?)?/(s?bin | libexec | lib(64)?)
```

Указанный путь может быть дополнен в зависимости от ПО, установленного в системе (коммерческое ПО часто устанавливается в /opt), а также на основе значения переменной \$PATH для каждого из пользователей.

#### **A.1.2.d Убедиться, что просмотр журналов протоколирования доступен только владельцу.**

Проверка тривиальна: права доступа + ACL; все остальные аспекты (включая настройки пользователей и административных привилегий) уже были рассмотрены в соответствующих разделах.

#### **A.1.2.e Чтобы убедиться, что ни один клиент не может использовать все ресурсы сервера для эксплуатации уязвимостей, убедиться, что для каждого клиента установлены системные лимиты на:**

- **Использование дискового пространства**
- **Использование канала**
- **Использование памяти**
- **Использование ресурсов ЦПУ**

Для ограничения использования ресурсов могут применяться как средства ОС (если приложения пользователей запущены в рамках одной ОС), так и средства виртуализации (KVM, Xen, OpenVZ/Virtuozzo, VMWare ESX, VMWare Server, Microsoft Hyper-V и т.п.).

Рассмотрим стандартные средства Linux, т.к. каждое средство виртуализации имеет свои внутренние механизмы ограничения ресурсов, описание которых выходит за рамки данной статьи.

I. Чтобы ограничить использование клиентами **дискового пространства**, необходимо выполнить следующие действия.

Во-первых, следует убедиться в том, что все каталоги, доступные для записи пользователям, вынесены на отдельные дисковые разделы. К таким каталогам относятся, по крайней мере, /home и /tmp. Кроме того, отдельные разделы необходимо выделить для /var/log и (желательно) /var/log/audit.

Во-вторых, нужно настроить пользовательские квоты. При использовании файловых систем EXT3/EXT4 (XFS не рассматриваем из-за ограниченной поддержки этой ФС в RHEL) следует выполнить следующее:

1. Для разделов, доступных пользователям для записи, в опциях монтирования /etc/fstab указать:

**usrquota,grpquota**

После этого перемонтировать требуемую ФС:

**mount \$FS -o remount**

2. В корне каждого раздела (пусть это будут /tmp и /home) создать два двоичных файла квот

```
for fs in /tmp /home; do  
    cd "$fs"  
touch aquota.user aquota.group  
chmod 600 aquota.user aquota.group  
cd -  
done
```

3. На смонтированных ФС квоты активируются путем выполнения следующих команд:

```
quotacheck -aRvugm  
quotaon -avug
```

4. Для редактирования квот пользователей и групп, а также «льготного периода» существует команда edquota. При этом на каждую ФС пользователю и группе можно задать отдельную квоту.

5. Просмотреть использование квот для пользователей и файловых систем можно с помощью команд:

**quota [-u user] [-g group]** – каждый пользователь может получить информацию о своих квотах и их использовании;



**repquota [-a] [-v]** – выводит данные об использовании квот для каждой ФС (более наглядно, чем quota).

6. С дополнительной информацией по дисковым квотам можно ознакомиться на страницах руководства man: **mount, quotacheck, quotaon, quotaoff, edquota, quota, repquota**.

**Примечание.** При использовании виртуальных машин для выполнения клиентского ПО дисковое пространство обычно ограничивается размером виртуального диска, выделяемого виртуальной машине.

II. Для **ограничения канала** в основном применяется сетевое оборудование.

Ограничить с помощью средств ОС входящий и исходящий трафик (всех видов и протоколов) для некоторого пользователя на разделяемом сервере, где работает множество пользователей, – нетривиальная задача, решение которой не описывается в документации производителя (ни RedHat, ни Fedora официально об этом не сообщают).

Самое простое решение – ограничить полосу пропускания для пользователей, под задачи которых выделены виртуальные серверы с собственными IP-адресами. В этом случае можно на сервере виртуализации настроить ограничения с помощью утилиты **tc** [1], либо на сетевом оборудовании создать правила ограничения трафика по IP-адресам.

III. Ограничения **использования оперативной памяти**

Объем оперативной памяти процессов пользователя может быть ограничен в файле **/etc/security/limits.conf** (связанном с модулем pam\_limits). Так, для пользователя alex ограничение записывается в виде:

```
alex - as N
```

Здесь N – максимальный объем памяти (в Кб), отведенный процессам данного пользователя, «as» (address space) указывает на тип ограничения, «-» сразу устанавливает оба вида лимитов – «мягкий» и «жесткий».

Экспериментальным путем установлено, что на один экземпляр bash требуется, как минимум, 10 Мб памяти, при этом инициализация сессии bash произойдет с ошибками, и смогут запуститься далеко не все стандартные утилиты командной строки (соответствующие сообщения выделены в тексте):

```
[root@blacknet security]# su - alex
id: cannot find name for user ID 501
-bash: warning: setlocale: LC_CTYPE: cannot change locale (ru_RU.UTF-8): No such file or
directory
[ ... ]
```

```
id: cannot find name for user ID 501
[I have no name!@blacknet ~]$ date
Mon Jul 12 10:55:03 MSD 2010
```



```
[I have no name!@blacknet ~]$ dd if=/dev/zero bs=1M count=100 | bzip2 -9c > /dev/null  
bzip2: couldn't allocate enough memory  
Input file = (stdin), output file = (stdout)
```

**Примечание.** Может существовать каталог `/etc/security/limits.d/`, файлы которого также отвечают за различные ограничения пользователей.

Средства виртуализации, упомянутые выше, позволяют ограничить объем оперативной памяти, выделяемой каждому виртуальному серверу.

#### IV. Ограничения *использования CPU*

Традиционные методы ограничения использования CPU не позволяют задать лимиты в виде процентных долей, несмотря на то, что этот способ является самым удобным. Сторонними разработчиками предпринимаются попытки изменить ситуацию – например, существует утилита `cpulimit` [2], однако она работает на довольно примитивном уровне (отправляя процессам сигналы `SIGSTOP` и `SIGCONT`), не позволяет ограничить доступ для пользователей и групп (только для процессов и их потомков) и имеет массу других недостатков.

Среди открытых средств виртуализации OpenVZ является наиболее гибким инструментом для настройки ограничения использования CPU. Однако, в отличие от официально поддерживаемых KVM и Xen, при его использовании в соответствии с [3] рекомендуется отключить SELinux.

#### **A.1.3.a Убедиться, что протоколирование событий удовлетворяет следующим критериям:**

- **Протоколирование настроено для всех типичных используемых на сервере приложений сторонних производителей**
- **Протоколирование включено по умолчанию**
- **Журналы доступны для просмотра администратору и клиенту, для которого выполняется протоколирование**
- **Журналы расположены в каталогах, доступных клиенту**

Учитывая, что клиентское ПО может быть практически любым, общих рекомендаций по ведению журналов регистрации на разделяемом сервере дать нельзя. Единственное, что можно посоветовать в такой ситуации – использование виртуальных серверов для каждого из клиентов. С кратким сравнением технологий виртуализации можно ознакомиться по ссылке [4].

#### **Список источников:**

1. <http://lartc.org/lartc.html>
2. <http://cpulimit.sourceforge.net/>
3. [http://wiki.openvz.org/Quick\\_installation](http://wiki.openvz.org/Quick_installation)
4. [http://en.wikipedia.org/wiki/Comparison\\_of\\_platform\\_virtual\\_machines](http://en.wikipedia.org/wiki/Comparison_of_platform_virtual_machines)

## 13 Исследовательский центр Positive Research

---

Positive Research – один из крупнейших в Европе исследовательских центров в области информационной безопасности. Он является инновационным подразделением компании Positive Technologies, ключевого эксперта в сегменте практических аспектов защиты информации.

С 2004 года при содействии Positive Research лидеры ИТ-отрасли, среди которых Microsoft, Cisco, Google, Avaya, Citrix, VmWare, Trend Micro, устранили несколько сотен уязвимостей и недочетов систем безопасности.