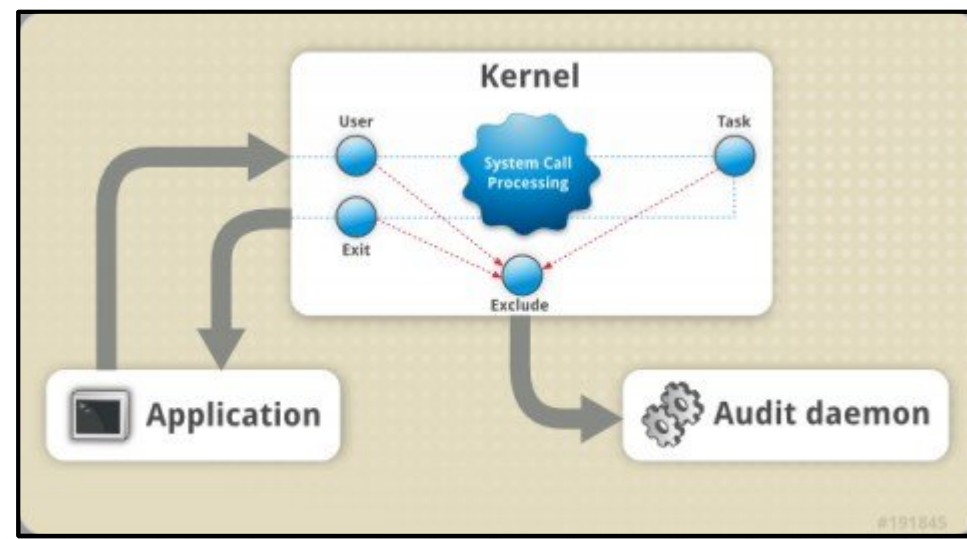


# **Сбор информации, мониторинг и журналирование в РЕД ОС**

# Служба аудита — auditd

Инструмент для мониторинга критичных с точки зрения безопасности системных событий системы, активно разрабатывается специалистами компании Red Hat

- чтение, запись и изменение прав доступа к файлам;
- изменение информации о пользователях и группах;
- попытки неудачной авторизации в системе;
- запуск и завершение работы системы;
- инициация сетевых соединений;
- изменение сетевых настроек;
- запуск и остановка приложений;
- выполнение системных вызовов и др.



# Служба аудита — auditd

Файл конфигурации **/etc/audit/auditd.conf** определяет параметры службы аудита.

**log\_file** — файл, в котором будут храниться логи подсистемы аудита;

**log\_format** — формат, в котором будет сохранены логи;

**max\_log\_file** — максимальный размер файла лога в мегабайтах;

**max\_log\_file\_action** — действие при превышении макс. размера файла лога;  
и др.



# Управление демоном — auditctl

Настройка правил и управления демоном — **auditctl**

При написании правил auditd необходимо учитывать следующее:

- Сначала пишутся фильтры и только потом правила. **Выше** размещать стоит то правило, которое **важнее** учитывать.
- Писать правила лучше от частного к общему.
- Преднастроенные правила иногда конфликтуют с написанными. Поэтому каждое правило лучше тестировать отдельно.
- Определите общее звено и пишите правило для него. Например, для обнаружения запуска шеллов следите за /dev/tty, /dev/pts/.
- Бывает несколько системных вызовов для одного и того же.
- Если есть ложные срабатывания, выберите другой подход.



# Управление демоном — auditctl

## Как тестировать правила

- Проверка тестируемого правила на одном сервере.
- Проверка тестируемого правила в составе имеющихся других правил.
- Проверка нового набора правил на группе серверов.
- Применение правил на всей инфраструктуре.

Если у вас большое количество серверов, то всегда стоит помнить о том, что какой-то участок вашей инфраструктуры может сработать **не так как ожидалось** и породить большое количество событий, поэтому пути решения возможных проблем лучше обдумать сразу.

# Управление демоном — auditctl

Существует три типа правил аудита:

**Правила контроля:** эти правила используются для изменения конфигурации и настроек самой системы аудита.

**Правила системных вызовов:** эти правила используются для мониторинга системных вызовов, выполняемых любым процессом или конкретным пользователем.

**Правила файловой системы:** это просмотр файлов или каталогов. Используя эти правила, мы можем проверять любой доступ к определенным файлам или каталогам.

# Управление демоном — **auditctl**

Файл конфигурации правил **/etc/audit/audit.rules**

## Правила контроля

Установлены следующие настройки по умолчанию

-D – Удалить все правила

-b – Размер буфера

-f 1 – Действие в критической ситуации ( 0 - ничего не делать; 1 - отправлять сообщение в dmesg, 2 - отправлять ядро в панику)

Далее могут идти правила действий на события

**auditctl -l** — Просмотр правил. Правила контроля не отображаются.

**auditctl -s** — Статус системы аудита

# Управление демоном — auditctl

Правила системных вызовов.

**auditctl -a <список>, <действие> -S <имя сисвызова> -F <фильтры>**

Всего существует 5 списков событий:

task — создание новых процессов;

entry — вход в системный вызов;

exit — выход из системного вызова;

exclude — исключение событий;

user — параметры пользовательского пространства.

# Управление демоном — auditctl

**auditctl -a <список>, <действие> -S <имя сисвызова> -F <фильтры>**

Действия могут быть:

always – события будут записываться в журнал;

never – не будут записываться в журнал.

# Управление демоном — auditctl

**auditctl -a <список>, <действие> -S <имя сисвызова> -F <фильтры>**

**-F [n=v | n!=v | n<v | n>v | n<=v | n>=v | n&v | n&=v]** - Допустимо использование одного из следующих 8 операторов: равно, не равно, меньше, больше, меньше либо равно, больше либо равно, битовая маска (n&v) и битовая проверка (n&=v).

# Управление демоном — auditctl

Некоторые параметры фильтров

**a0, a1, a2, a3** — аргументы системного вызова соответственно;

**dir** — верхняя директория, за которой необходимо наблюдать;

**euid** — действительный идентификатор пользователя;

**exit** - значение, возвращаемое системным вызовом при выходе;

**key** — поля для фильтра. Облегчает поиск событий в журналах;

**path** - полный путь к файлу, за которым необходимо следить;

**perm** — режим доступа к объекту;

**uid** - идентификатор пользователя;

```
type=SYSCALL msg=audit(1678104656.463:558): arch=c000003e syscall=257 success=no exit=-13 a0=ffffff9c  
a1=55bc38836f30 a2=241 a3=1a4 items=0 ppid=4574 pid=4583 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0  
sgid=0 fsgid=0 tty=pts1 ses=3 comm="useradd" exe="/usr/sbin/useradd"  
subj=unconfined_u:unconfined_r:useradd_t:s0-s0:c0.c1023 key=(null)
```



# Управление демоном — auditctl

Команды будут выполняться только в ходе работы одного сеанса ОС. Правило нужно добавить в файл **/etc/audit/rules.d/audit.rules**

## Правила файловой системы

Следим за /etc/hosts, отслеживаем изменение файла и атрибутов, присваиваем сообщению метку.

**auditctl -w /etc/hosts -p wa -k hosts\_file\_change**

Исключаем из контроля файл

**auditctl -W /etc/passwd**

# autrace

Аудит событий которые вызваны процессами — **autrace**

Получить события по определенному процессу.

**autrace /bin/bash**

Добавляются правила по отбору событий от этого процесса. Теперь можно проконтролировать конкретный процесс и его потомков.

```
[root@redcloud ~]# auditctl -l  
-a always,exit -S all -F pid=1730  
-a always,exit -S all -F ppid=1730
```

# Поиск событий в лог файле — ausearch

Поиска событий по номерам событий:

**sudo ausearch -a номер\_события**

Поиска событий по именам системных вызовов:

**sudo ausearch -sc ptrace -i**

Поиска событий по идентификаторам пользователей:

**sudo ausearch -ui 2010**

Поиска событий по именам исполняемых файлов:

**sudo ausearch -x /usr/bin/nmap**

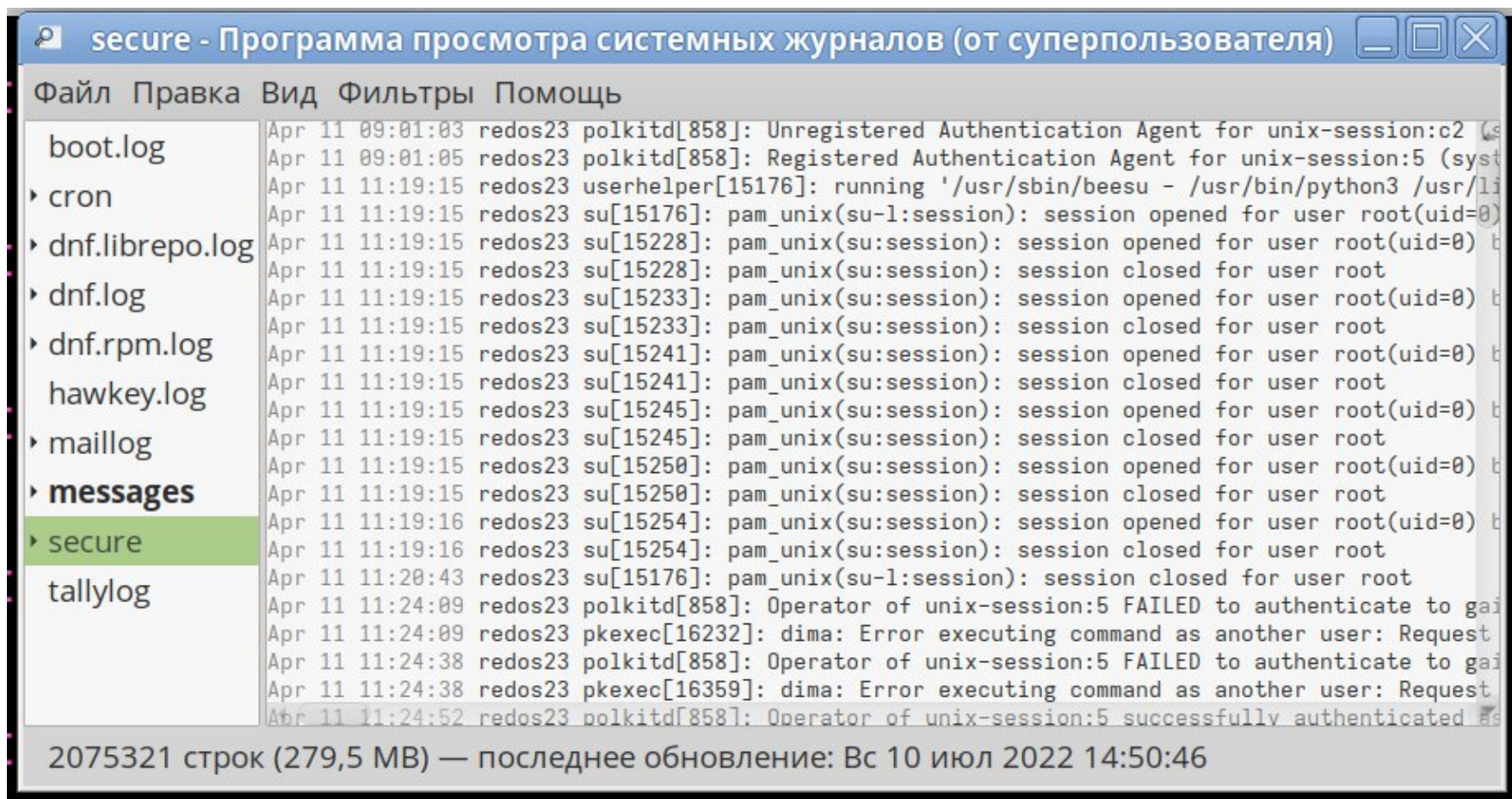
Поиска событий по имени терминала:

**sudo ausearch -tm pts/0**

Поиска событий по именам демонов:

**sudo ausearch -tm cron**

# Графическая программа просмотра логов mate-system-log





# Rsyslog - сервис управления логами

Современная замена службы логгирования syslog.

Основные возможности: многопоточность, поддержка БД, фильтрация, настраиваемый формат вывода.

Использует приоритеты сообщений.

Источники	Приоритеты
Auth, authpriv, cron, daemon, kern, lpr, mail, mark, news, syslo, user, local0 ... local7	emerg (panic)(0), alert(1), crit(2), error(3), warn(3), notice(4), info(5), debug(6).

# Rsyslog - сервис управления логами

Отправить журнал на удаленный сервер:

```
*.info;mail.none;authpriv.none;cron.none @xx.xx.xx.xx:514
```

Здесь 514 — это порт, на котором слушает rsyslog. Настройка rsyslog на прием журнала заключается в запуске сервиса с модулями **imtcp** и **imudp**. Получаемые записи журнала лучше **отфильтровать** и записать **в отдельный файл**.

```
if $fromhost-ip contains '192.168.1.10' then /var/log/192.168.1.10.log
```

(если журнал с хоста содержит '192.168.1.10', то записать его в /var/log/192.168.1.10.log)

# Rsyslog - сервис управления логами

Пример. Сервер с DHCP. Сохраняем записи в отдельный файл. Редактируем **/etc/dhcp/dhcpd.conf**, добавляем **log-facility local6**.

Теперь записи DHCP можно перенаправить в отдельный файл. Добавляем в файл настроек **rsyslog** строчку:

```
local6.*    /var/log/dhcp.log
```

Перезагружаем обе задействованные службы **dhcpd** и **rsyslog**

В результате сервис rsyslog будет отправлять все записи службы dhcp **не** в общий файл **/var/log/messages**, а отдельный **/etc/dhcp/dhcpd.conf**



# logrotate

Утилита **Logrotate** автоматически обрабатывает журналы в зависимости от определенных условий и правил соответствия. Все основные настройки программы находятся в файле **/etc/logrotate.conf**.

Папка для дополнительных настроек **/etc/logrotate.d**.

```
[dima@REDOS00 ~] cat /etc/logrotate.d/bootlog
/var/log/boot.log
{
    missingok 1
    daily 2
    copytruncate 3
    rotate 7 4
    notifempty 5
}
```

# logrotate

Некоторые параметры управления и обработки журналов.

**rotate** — сколько старых логов нужно хранить;

**create** — создать пустой лог файл после перемещения старого;

**dateext** — добавляет дату ротации перед заголовком старого лога;

**delaycompress** — не сжимать последний и предпоследний журнал;

**maxage** — выполнять ротацию, если журналы старше, чем указано;

**missingok** — не выдавать ошибки, если лог файла не существует;

**olddir** — перемещать старые логи в отдельную папку;

**postrotate/endscript** — выполнить команды после ротации;

**size** — размер лога, когда он будет перемещен;

# logrotate

## Проверка конфигурации

Проверяйте корректность файлов настройки. Проверка выведет все, что планируется сделать, но не будет изменять файлы на диске.

**logrotate -d /etc/logrotate.d/bootlog**

Если нет сообщений об ошибках, то можно считать, что настройка завершена.

# systemd-journald

**systemd-journald** — демон (служба), который собирает записи журнала из всей системы.

Файл настройки **/etc/systemd/journald.conf**. После редактирования файла необходимо перезапустить службу:

```
# systemctl restart systemd-journald
```

Для просмотра журналов используют утилиту `journalctl`. Основной формат вывода имеет вид

**время события** **хост** **источник события** **само сообщение**

Вид вывода можно изменить на **short, verbose, json, json-pretty, cat**  
**sudo journalctl -o json-pretty**

# journalctl

Фильтрация по уровням важности сообщений.

**journalctl -p 5**

Просмотр логов с момента последней загрузки:

**journalctl --boot**

Фильтровать по дате и времени

**Journalctl --since "2012-01-20 14:05:50" --until "2012-01-20 15:05:50"**

Просмотр логов службы nginx:

**journalctl --unit nginx**

Просмотр логов процесса с PID 1234:

**journalctl \_PID=1234**

Просмотр логов процессов, запущенных от пользователя с UID 1010

**journalctl \_UID=1010**

# journalctl

Просмотр логов процесса по пути /usr/bin/docker:

**journalctl /usr/bin/docker**

Просмотр логов ядра:

**journalctl -k**

Фильтрация по тексту сообщения

**journalctl --grep timer**

Вывести общий размер лог файлов на диске

**journalctl --disk-usage**

Очистить логи, давностью больше указанного периода

**journalctl --vacuum-time=1years**

Очистить логи, уменьшить размер до указанного

**journalctl --vacuum-size=2G**

# dmesg — сообщения ядра Линукс

Поддерживаемые категории журналирования **-f**:

**kern** (ядро), **user** (пространство пользователя), **mail** (сервис почты), **daemon** (системные службы), **auth** (сообщения безопасности), **syslog** (от Syslogd), **lpr** (от служб печати).

Доступные уровни журналирования **-l**:

**emerg (0), alert (1), crit (2), err (3), warn (4), notine (5), info (6), debug (7)**



# dmesg — сообщения ядра Линукс

Во время загрузки логи собираются другим приложением. Можно просто прочитать файл `/var/log/dmesg`

Основные ключи

- C, --clear** — очистить буфер сообщений ядра;
- H, --human** — включить вывод, удобный для человека;
- k, --kernel** — отображать только сообщения ядра;
- u, --userspace** — сообщения от программ пространства пользователя;
- w, --follow** — после вывода всех сообщений ждать новых;
- x, --decode** — выводить категорию и уровень журналирования.



**Спасибо за внимание!**

**[www.red-soft.ru](http://www.red-soft.ru)  
[redos@red-soft.ru](mailto:redos@red-soft.ru)**

