# ??????????

### ?????? ???????????

WAF между логгером и фаером.

| | |
|---|---|
| системные | rsyslog, journalctl |
| ИБ | auditd |
| WAF | modsecurity |

### ????????? ???????????

Есть rsyslog и journalctl. journalctl постепенно заменяет rsyslog.

#### Journalctl

| | |
|---|---|
| sudo journalctl | Просмотреть системный журнал |
| sudo journalctl -u ssh | Логи конкретной службы |
| sudo journalctl -f | Последние логи (как tail -f) |
| sudo journalctl -p err | Только ошибки |

Journalctl по умолчанию сохраняет логи в ОП

#### Rsyslog

Rsyslog нужен, если:

- нужны обычные текстовые файлы /var/log/auth.log, /var/log/syslog;
- необходима ротацию логов через logrotate;
- нужно отправлять логи на внешний syslog-сервер (например, в SIEM).

Установка:

```
sudo apt install rsyslog
sudo systemctl enable --now rsyslog
```

Типы логов, настраивается.

| Название журнала | Расположение |
|---|---|

| | |
|---|---|
| Аутентификация | /var/log/auth.log |
| Ядро | /var/log/kern.log |
| Демоны | /var/log/daemon.log |
| Запуск | /var/log/boot.log |
| Обновления | /var/log/apt/history.log<br>/var/log/dpkg.log |
| Cron | /var/log/cron.log |
| Драйверы | /var/log/dmesg |

## Размещение настроечных файлов

| | |
|---|---|
| /etc/rsyslog.conf | Основной файл настроек. Дополнительные файлы настроек подключаются в алфавитном порядке из /etc/rsyslog.d/*.conf |
| /etc/rsyslog.d/50-default.conf | Создается при установке. Базовые правила. Можно комментировать. Например<br>`auth,authpriv.* /var/log/auth.log`<br>`*.*;auth,authpriv.none -/var/log/syslog`<br>`mail.* -/var/log/mail.log`<br>`cron.* -/var/log/cron.log` |
| Рекомендованные файлы | |
| 10-base.conf | Базовые глобальные настройки<br>`$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat` |
| 20-remote.conf | Отправка логов на удалённый сервер<br>`*.* @@syslog.example.com:514` |

| | |
|---|---|
| 30-ssh.conf | Логи от sshd<br>`if ($programname == 'sshd' or $programname == 'sudo') then /var/log/security.log & stop` |
| 40-sudo.conf | Логи sudo<br>`if ($programname == 'sudo') then /var/log/sudo.log` |
| 99-local.conf | Локальные мелкие правки, фильтры, формат вывода |

**Структура логов:** timestamp hostname program identifier log level message content. Можно добавлять хост, с которого было отправлено сообщение.

**Типы событий:**

- kernel: загрузка модулей, аппаратные ошибки,
- authentication: попытки входа, изменения паролей
- service and daemon: работа системных служб и демонов. Запуск служб, ошибки и предупреждения
- network: подключение сетевых интерфейсов, обнаружение сетевых атак, изменения настроек сети
- system: общая системная информация о работе операционной системы. Запуск системы, события загрузки, ошибки файловых систем.
- logging: события журналирования. Создание и ротация журналов, ошибки записи в журналы.

**Конфигурация:** /etc/rsyslog.conf (/etc/syslog.conf). Каждая строка в формате <фильтр> <действие>. Фильтр определяет, какие сообщения должны быть обработаны, а действие определяет, что делать с этими сообщениями (например, записать их в определенный файл журнала). Пример конфигурации syslog:

```
# Маркировать сообщения ядра, которые требуют прямого вывода в файл /dev/console
kern.warning;*.err;authpriv.none;mail.crit       /dev/console

# Записывать все сообщения уровня emergency и выше в файл /var/log/emergency.log
*.emerg                         /var/log/emergency.log

# Записывать все сообщения уровня info и выше в файл /var/log/messages
*.info;mail.none;authpriv.none;cron.none                /var/log/messages

# Записывать все сообщения уровня notice и выше в файл /var/log/messages
```

```
*.notice                        /var/log/messages

# Записывать все сообщения уровня warning и выше в файл /var/log/messages
*.warn                          /var/log/messages

# Записывать все сообщения уровня error и выше в файл /var/log/errors.log
*.err                           /var/log/errors.log

# Записывать все сообщения уровня crit и выше в файл /var/log/critical.log
*.crit                          /var/log/critical.log

# Записывать все сообщения уровня alert и выше в файл /var/log/alert.log
*.alert                         /var/log/alert.log

# Записывать все сообщения уровня debug и выше в файл /var/log/debug.log
*.debug                         /var/log/debug.log

# Записывать все сообщения от локальных7 в файл /var/log/local7.log
local7.*                        /var/log/local7.log
```

Для изменения формата сообщений изменяем файл sudo nano /etc/rsyslog.conf

```
$template RFC3164fmt,"<%PRI%>%TIMESTAMP% %HOSTNAME% %syslogtag%%msg%\n" #добавляем шаблон
auth,authpriv.*                 /var/log/auth.log;RFC3164fmt #привязываем шаблон к лог файлу
```

Приоритет syslog считается по формуле facility * 8 + severity. Facility имеет заданные значения, и определяет к какому типу относится событие. Например, facility #0 относится к событиям ядра, а #4 - к событиям систем безопасности.

Severity отвечает за уровень критичности событий, где 0 это максимум(emergency), а уровень 7 - малоинформативные данные, используемые для отладки(debug).

В нашем примере число 38 получается за счет facility 4 и severity 6 (info).

**Журнал аутентификации**

/var/log/auth.log Содержит информацию об аутентификации пользователей, включая входы в систему, попытки неудачной аутентификации и другие аутентификационные события.

Типы событий в журнале auth.log

**Успешная аутентификация** (Successful Authentication). Это может быть вход через SSH, консоль, графический интерфейс и т. д.

```
Feb 11 14:23:45 server sshd[1234]: Accepted password for user123 from 192.168.1.100 port 12345 ssh2
```

**Неудачная аутентификация** (Failed Authentication). Это может быть вызвано неправильным паролем, недействительным пользователем и т. д.

```
Feb 11 14:30:10 server sshd[1234]: Failed password for invaliduser from 192.168.1.50 port 54321 ssh2
```

**Изменение пароля** (Password Changes). Это позволяет отслеживать изменения паролей и управлять безопасностью.

```
Feb 11 15:45:32 server passwd[5678]: password for user123 has been changed
```

**Изменение параметров аутентификации** (Authentication Configuration Changes).

```
Feb 11 16:12:20 server login[9876]: PAM (login) session opened for user123 by (uid=0)
```

**Запуск или остановка служб аутентификации** (Start/Stop of Authentication Services).

```
Feb 11 17:00:05 server systemd: Starting OpenSSH server daemon...
```

**События, связанные с аутентификацией.** Проблемы с файлами авторизации, недоступность источников аутентификации, ...

```
Feb 11 18:30:15 server su: pam_unix(su:session): session opened for user456 by user123(uid=1000)
```

### Журнал ядра

/var/log/kern.log Содержит сообщения ядра Linux, такие как сообщения об ошибках, предупреждения и другую важную информацию о работе ядра.

Типы событий

**Сообщения ядра** (Kernel Messages). Основные сообщения. Загрузка ядра, проблемах с аппаратным обеспечением, ошибках файловых систем, сетевых событиях, а также другие системные события, связанные с работой ядра.

```
Feb 11 08:30:15 server kernel: [    0.000000] Linux version 5.10.0-14-generic (buildd@lcy01-amd64-017) (gcc (Ubuntu 10.3.0-1ubuntu1) 10.3.0, GNU ld (GNU Binutils for Ubuntu) 2.36.1) #15-Ubuntu SMP Fri Dec 31 14:49:08 UTC 2021 (Ubuntu 5.10.0-14.15-generic 5.10.21)
```

**Сообщения о загрузке и выгрузке модулей ядра** (Kernel Module Loading and Unloading). Полезно для отслеживания загрузки различных драйверов и модулей ядра.

```
Feb 11 08:31:45 server kernel: [    3.141592] Module 'nvidia-drm' is loaded.
```

**Сообщения об ошибках ввода-вывода** (I/O Errors). Если возникают проблемы с вводом-выводом, такие как ошибки чтения или записи на диске, это может быть отражено в kern.log. Помогает отслеживать проблемы с дисками и файловыми системами.

```
Feb 11 09:12:20 server kernel: [  405.837234] ata1: EH complete
```

**Сообщения об ошибках памяти** (Memory Errors). Например ошибка ECC (Error-Correcting Code) или ошибка доступа к памяти.

```
Feb 11 09:55:10 server kernel: [ 1025.556789] EDAC MC0: 1 CE error on CPU0 memory controller (channel 0 /channel 1 = 0x1/0x0): 0x0000 (corrected)
```

**Сообщения о загрузке и выгрузке ядра** (Kernel Boot and Shutdown Messages). Включает сообщения о запуске служб и драйверов. Это полезно для отслеживания процесса загрузки и выявления проблем.

```
Feb 11 10:00:05 server kernel: [    0.000000] Initializing cgroup subsys cpuset
```

**Другие системные сообщения ядра** (Other Kernel System Messages). В kern.log также могут записываться другие системные сообщения, связанные с работой ядра, такие как события управления питанием, управления процессором и т.д.
Feb 11 10:15:30 server kernel: [  765.320123] ACPI: Battery Slot [BAT0] (battery present)

**Журнал демонов**

/var/log/daemon.log Сообщения системных служб и демонов, таких как сервисы SSH, FTP, DHCP и другие.

Типы событий

**Запуск и остановка демонов** (Daemon Start and Stop).

```
Feb 11 08:30:15 server systemd[1]: Started OpenSSH server daemon.
```

**Сообщения об ошибках служб** (Service Error Messages). Могут содержать информацию о проблемах с конфигурацией, недоступности ресурсов и других критических ситуациях.

```
Feb 11 09:45:20 server apache2[1234]: [error] (28)No space left on device: Cannot create SSLMutex
```

**Информационные сообщения** (Informational Messages). Содержат отчеты о выполненных действиях, успешных операциях и других системных событиях.

```
Feb 11 10:00:05 server cron[5678]: (CRON) INFO (Running @reboot jobs)
```

**Сообщения об ошибках в работе демонов** (Daemon Error Messages). Содержат информацию о сбоях, падениях и других аномальных ситуациях, происходящих в процессе выполнения служб.

```
Feb 11 11:15:30 server postfix[9876]: fatal: parameter "smtpd_sender_login_maps": file too large
```

**Сообщения о подключении и отключении к демонам** (Connection and Disconnection Messages). Отслеживают активность пользователя или других систем, пытающихся взаимодействовать с демоном.

```
Feb 11 12:30:45 server sshd[2345]: Accepted publickey for user123 from 192.168.1.100 port 12345 ssh2: RSA SHA256:abc123
```

**Другие системные сообщения** (Other System Messages). Не подпадающие под вышеперечисленные категории, но относящиеся к работе демонов и служб в системе. Это может включать сообщения о перезагрузках, обновлениях и других системных событиях.

```
Feb 11 13:45:10 server systemd[1]: Reloading.
```

**Журнал запуска**

/var/log/boot.log Содержит информацию о процессе загрузки системы, включая загрузку ядра и инициализацию устройств.

Типы событий

**Сообщения ядра** (Kernel Messages). Сообщения, генерируемые ядром Linux во время загрузки. Информация об оборудовании, обнаруженном ядром, настройка ядра, загрузка драйверов и другие системные события, происходящие во время загрузки.

```
[    0.000000] Linux version 5.4.0-91-generic (buildd@lgw01-amd64-035) (gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)) #102-Ubuntu SMP Fri Aug 13 23:50:30 UTC 2021 (Ubuntu 5.4.0-91.102-generic 5.4.143)
```

**Сообщения об инициализации системных служб** (Service Initialization Messages). Запуск системных служб и демонов, которые запускаются во время загрузки операционной системы.

```
[    4.105167] systemd[1]: Started Forward Password Requests to Wall Directory Watch.
```

**Сообщения об ошибке загрузки** (Boot Error Messages).

```
[FAILED] Failed to start Network Time Synchronization.
```

**Информация о загрузочных скриптах и конфигурации** (Boot Scripts and Configuration Information). Выполнение загрузочных скриптов, настройках загрузчика и других параметрах конфигурации, применяемых при загрузке системы.

```
[    0.820671] ACPI: Core revision 20200326
```

**Сообщения об успешном завершении загрузки** (Boot Completion Messages). В конце файла boot.log обычно содержится информация о завершении процесса загрузки. Это может включать сообщения об успешном запуске всех требуемых служб и демонов.

```
[    5.091481] systemd[1]: Started Daily apt download activities.
```

**Другие системные сообщения** (Other System Messages). Например, изменения параметров ядра или настройках безопасности.

```
[    6.220215] random: crng init done
```

**Журнал обновлений**

APT (Advanced Package Tool), основной лог: /var/log/apt/ Файл журнала: /var/log/apt/history.log Старые логи обновления: /var/log/apt/term.log

**Структура записи:**

| | |
|---|---|
| Start-Date | |
| Commandline: | Commandline: apt install some-package<br>Commandline: apt remove some-package<br>Commandline: apt upgrade<br>Commandline: apt-get autoremove |
| Requested-By | Requested-By: user123 (1000) |
| <Type of operation> | Install: some-package:amd64 (version)<br>Remove: some-package:amd64 (version),<br>Upgrade: some-package:amd64 (old-version -> new-version), |
| End-Date | |

Пример установки пакета:

```
Start-Date: 2023-08-15  15:30:45
Commandline: apt install some-package
Requested-By: user123 (1000)
Install: some-package:amd64 (version), another-package:amd64 (version),
End-Date: 2023-08-15  15:31:10
```

DPKG (Debian Package Manager)

Пример dpkg.log:

```
2024-02-10 15:21:05 status installed firefox:amd64 100.0.0-0ubuntu1
2024-02-10 15:21:05 status installed firefox-locale-en:amd64 100.0.0-0ubuntu1
```

Запись в логе содержит статус установки, времени и дате операции, версии и архитектуре установленных пакетов.

Журнал установки: /var/log/dpkg.log

Каждая запись в этом файле содержит дату и время операции (например, 2023-08-15 15:38:55), за которыми следует описание операции и подробности о пакете. Например, "installed" указывает на установку пакета, "remove" на удаление, "upgrade" на обновление, "configure" на настройку, а "status" отображает текущее состояние пакета. Далее указывается имя пакета, его архитектура (например, amd64), его версия и новая версия (если применимо). В случае ошибок установки или зависимостей также выводятся соответствующие сообщения об ошибках. Пример ошибки:

```
2023-08-15 15:38:55 configure some-package:amd64 version <none>
dpkg: dependency problems prevent configuration of some-package:
some-package depends on another-package (>= required-version); however:
Package another-package is not installed.
```

YUM (Yellowdog Updater Modified) и DNF (Dandified YUM) (Fedora, CentOS и др.)

Основной лог: /var/log/yum.log

Каждая запись начинается со времени лога - например, Oct 05 18:00:19. После временной метки указывается действие, например, "Installed" для установки пакета, "Erased" для удаления и т. д. Затем идет имя пакета, его версия и, если применимо, старая версия (для обновлений). В случае ошибки указывается сообщение об ошибке.

Pacman, используемый (Arch Linux)

Основной лог: /var/log/pacman.log

В каждой записи [YYYY-MM-DD HH:MM] обозначает дату и время операции. После этого указывается действие, такое как "installed" для установки пакета, "removed" для удаления и т. д. Затем идет имя пакета, его версия и, если применимо, старая версия (для обновлений). В случае ошибок указывается сообщение об ошибке.

### Журнал cron

/var/log/cron.log Запланированные задания, успешные и неудачные выполнения, сообщения или ошибки, связанные с выполнением этих заданий.

```
Feb 10 15:00:01 DEVICE_NAME CRON[1234]: (root) CMD (/usr/bin/php /path/to/script.php)
```

### Журнал драйверов

/var/log/dmesg Сообщения, сгенерированные ядром во время загрузки системы, информация о оборудовании и драйверах.

```
[    0.000000] Linux version 5.4.0-91-generic (buildd@lcy01-amd64-016) (gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)) #102-Ubuntu SMP Fri Nov 5 16:31:28 UTC 2021 (Ubuntu 5.4.0-91.102-generic 5.4.154)
[    0.000000] Command line: BOOT_IMAGE=/boot/vmlinuz-5.4.0-91-generic root=UUID=12345678-1234-1234-1234-1234567890ab ro quiet splash
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   Hygon HygonGenuine
[    0.000000]   Centaur CentaurHauls
[    0.000000]   zhaoxin   Shanghai
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x008: 'MPX bounds registers'
...
[    1.234567] sd 0:0:0:0: [sda] Attached SCSI disk
[    1.345678] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[    1.456789] random: crng init done
[    1.567890] random: 7 urandom warning(s) missed due to ratelimiting
```

Каждая строка начинается с квадратных скобок, в которых указывается время (в секундах и долях секунды) с момента запуска системы. Затем идет версия ядра Linux, аргументы командной строки, поддерживаемые процессоры, подключенные устройства (например, жесткие диски), инициализация файловых систем и другие системные действия.

# ?? ?????

auditd — демон аудита, используется для отслеживания и регистрации действий в системе (запуск процессов, доступ к файлам и директориям, изменения конфигурации). Записывает обычно в */var/log/audit/*. Основные файлы, используемые auditd:

- **audit.log** — основной журнал аудита. Подробная информация о событиях (время, тип, идентификатор процесса, действие, выполненное пользователем, и другие детали).
- **audit.log.[N]** — это архивные файлы аудита, которые содержат старые записи аудита. Например, audit.log.1

Формат логов аудита, создаваемых auditd, обычно структурирован и содержит различные поля, представляющие информацию о событии аудита. Вот типичный формат записи в логе auditd:

```
type=SYSCALL msg=audit(1625525281.877:123): arch=c000003e syscall=2 success=yes exit=0 a0=7ffdf44d1eab a1=0 a2=1 a3=0 items=2 ppid=29942 pid=29943 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=2 comm="ls" exe="/bin/ls" key="audit-wazuh-r"
```

| Поле | Описание |
|---|---|
| type | Тип события аудита, например, SYSCALL, CWD, EXECVE, SOCKET, и т. д. |
| msg | Общее сообщение аудита |
| audit(timestamp:serial) | Временная метка и серийный номер события аудита |
| arch | Архитектура процессора |
| syscall | Имя системного вызова, связанного с событием |
| success | Успешность выполнения системного вызова |
| exit | Код завершения системного вызова |
| a0-a3 | Регистры системного вызова |
| items | Количество элементов в событии аудита. |
| ppid | Идентификатор родительского процесса |
| pid | Идентификатор процесса |
| auid | Идентификатор аудита пользователя. |
| uid | Идентификатор пользователя. |
| gid | Идентификатор группы. |

| Поле | Описание |
| --- | --- |
| euid/suid/fsuid | Эффективный, фактический и файловый идентификаторы пользователя. |
| egid/sgid/fsgid | Эффективный, фактический и файловый идентификаторы группы. |
| tty | Терминал, связанный с процессом |
| ses | Идентификатор сеанса аудита |
| comm | Имя исполняемого файла |
| exe | Путь к исполняемому файлу |
| key | Ключ, связанный с событием аудита |

Это общий формат, и поля могут варьироваться в зависимости от типа события и конфигурации auditd.

### Типы событий auditd

**SYSCALL**: системные вызовы, которые выполняются в системе, такие как открытие файлов, чтение и запись данных, создание процессов и многое другое.

```
type=SYSCALL msg=audit(1625525281.877:123): arch=c000003e syscall=2 success=yes exit=0 a0=7ffdf44d1eab a1=0 a2=1 a3=0 items=2 ppid=29942 pid=29943 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=2 comm="ls" exe="/bin/ls" key="audit-wazuh-r"
```

Лог показывает успешное (success=yes) выполнение системного вызова (syscall=2), который в данном случае может быть open(), поскольку syscall=2 обычно связан с открытием файлов. Процесс с pid=29943, запущенный пользователем с auid=1000, выполняет команду ls (comm="ls" exe="/bin/ls") в текущем терминале (tty=pts0).

**CWD**: изменение текущего рабочего каталога процесса.

```
type=CWD msg=audit(1625525281.877:123):  cwd="/home/user"
```

Лог указывает на текущий рабочий каталог процесса, который в данном случае - /home/user.

**EXECVE**: запуск нового исполняемого файла с помощью системного вызова execve.

```
type=EXECVE msg=audit(1625525281.877:123): argc=3 a0="/bin/ls" a1="-la" a2="/etc"
```

Лог показывает запуск нового процесса с помощью системного вызова execve(). Процесс запускается с аргументами командной строки: /bin/ls -la /etc.

**SOCKET**: сетевые операции, такие как создание сокетов, отправка и получение данных через сокеты и т. д.

```
type=SOCKET msg=audit(1625525281.877:123): saddr=192.168.1.200 sport=12345 daddr=192.168.1.2 dport=22
```

Лог показывает на сетевое событие, где произошло установление соединения между IP-адресами 192.168.1.100 и 192.168.1.2 на портах 12345 и 22 соответственно.

**FILE**: действия с файлами и директориями, такие как открытие, чтение, запись, удаление и изменение атрибутов файлов.

type=SYSCALL msg=audit(1625525281.877:123): arch=c000003e syscall=2 success=yes exit=0 a0=7ffdf44d1eab a1=0 a2=1 a3=0 items=2 ppid=29942 pid=29943 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=2 comm="ls" exe="/bin/ls" key="audit-wazuh-r" name="/etc/passwd"

В этой записи показана успешная попытка доступа к файлу /etc/passwd с помощью системного вызова open(). Процесс с pid=29943, запущенный пользователем с auid=1000, выполняет команду ls, которая пытается получить доступ к файлу /etc/passwd.

**USER_AUTH**: аутентификационные события, такие как входы пользователей в систему, смены паролей и другие операции аутентификации.

type=USER_AUTH msg=audit(1625525281.877:123): pid=12345 uid=0 auid=1000 ses=2 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:authentication acct="user123" exe="/usr/sbin/sshd" hostname=192.168.1.2 addr=192.168.1.3 terminal=ssh res=success'

Лог отображает успешную аутентификацию пользователя. Пользователь с идентификатором uid=0 (обычно root) успешно прошел аутентификацию с помощью SSH.

**USER_ROLE_CHANGE**: изменения роли пользователя.

type=USER_ROLE_CHANGE msg=audit(1625525281.877:123): pid=12345 uid=0 auid=1000 ses=2 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='pam: default-context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 exe="/usr/sbin/sshd" hostname=192.168.1.2 addr=192.168.1.3 terminal=ssh res=success'

Лог указывает на успешное изменение роли пользователя. Пользователь с auid=1000 (идентификатор аудита пользователя) изменил свою роль. В данном случае, изменение роли прошло успешно (res=success).

**LOGIN/LOGOUT**: вход и выход пользователей из системы.

type=USER_LOGIN msg=audit(1625525281.877:123): pid=12345 uid=0 auid=1000 ses=2 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=login id=1000 exe="/usr/sbin/sshd" hostname=192.168.1.2 addr=192.168.1.3 terminal=ssh res=success'

Лог показывает успешный вход пользователя в систему. Пользователь с uid=0 (обычно root) успешно вошел в систему с использованием SSH.

**PROCTITLE**: изменения заголовков процесса.

type=PROCTITLE msg=audit(1625525281.877:123): proctitle="/bin/ls -la /etc"

Запись показывает аргументы командной строки процесса. Процесс с pid=123 запускает команду /bin/ls -la /etc.

**SYSTEM_BOOT/SHUTDOWN**: события загрузки и выключения системы.

type=SYSTEM_BOOT msg=audit(1625525281.877:123): kernel time=1234567890

Эта запись указывает на событие загрузки системы. Время загрузки ядра равно 1234567890, в формате UNIX времени.

### Отличия auditd от журналов в Linux

1. auditd предназначен для аудита событий безопасности, а именно для мониторинга и анализа активности на системе с целью обнаружения и предотвращения угроз безопасности, когда как обычные журналы, такие как `/var/log/syslog`, обычно используются для отслеживания работы системы, включая запуск и остановку служб, сообщения о состоянии аппаратного обеспечения и другие системные события.
2. auditd обычно настраивается администратором системы для мониторинга конкретных событий безопасности в соответствии с требованиями безопасности системы, в то время как обычные журналы логов создаются и используются системой по умолчанию для регистрации различных событий и действий на системе.

### Механизмы защиты auditd

#### Проверка подписи конфигурационных файлов

Обнаруживает изменения в конфигурационных файлах, указывающие на потенциальные нарушения. Настройка проверки подписи конфигурационных файлов в auditd выполняется с использованием инструментов, таких как auditctl и keyctl. Общий подход к настройке можно описать следующим образом:

- Генерация ключей для подписи. Сначала необходимо сгенерировать ключи для подписи конфигурационных файлов. Эти ключи будут использоваться для создания и проверки цифровых подписей файлов. Команда генерирует ключ с именем

auditd_signing_key

```
keyctl add key auditd_signing_key "asymmetric" "trusted:kernel" "new 0" @u
```

- Подписание конфигурационных файлов. Команда подписывает файл /etc/audit/audit.rules с использованием созданного ключа и сохраняет подпись в файл /etc/audit/audit.rules.sig.

```
openssl dgst -sha256 -sign /etc/audit/private/auditd_signing_key -out
/etc/audit/audit.rules.sig /etc/audit/audit.rules
```

- Настройка проверки подписи.

  ```
  auditctl -b 1024 -S verify -k auditd_conf -F subj_type=auditd_t
  ```

Команда настраивает проверку подписи для конфигурационных файлов auditd. Она устанавливает ограничение длины подписи до 1024 байт, определяет тип события verify, связывает событие с ключевой меткой auditd_conf и определяет тип подсубъекта auditd_t для указания на процесс auditd.

**Ограничение доступа к сокетам и файлам**

Предотвращает несанкционированный доступ к данным аудита или их изменение. Настройку доступа можно выполнить с помощью SELinux. SELinux позволяет определить политики безопасности, которые определяют, какие операции разрешены для различных процессов и ресурсов в системе. Вы можете создать или настроить политику SELinux для auditd, чтобы ограничить доступ к его сокетам и файлам. Примеры команд для настройки SELinux для auditd:

```
# Разрешить доступ к сокетам auditd
sudo semanage permissive -a auditd_t
# Разрешить доступ к файлам auditd
sudo semanage fcontext -a -t auditd_exec_t "/путь/к/auditd(/.*)?"
sudo restorecon -R -v /путь/к/auditd
semanage permissive -a auditd_t
```

semanage: утилита для управления SELinux.

permissive: параметр указывает, что типу процесса разрешается выполнение в режиме "поощряемой допускающей политики", что означает, что SELinux регистрирует попытки доступа, но не блокирует их.
-a auditd_t: указывает, что типу процесса auditd_t (тип, используемый для auditd)

разрешается выполнение в режиме "поощряемой допускающей политики".
semanage permissive -a auditd_t позволяет разрешить типу процесса auditd_t выполняться в режиме "поощряемой допускающей политики", что позволяет SELinux регистрировать попытки доступа к ресурсам, связанным с auditd, но не блокирует их.

semanage fcontext -a -t auditd_exec_t "/путь/к/auditd(/.*)?"

fcontext: команда для управления контекстами файлов.
-a: параметр указывает на добавление нового контекста файла.
-t auditd_exec_t: указывает новый контекст файла как auditd_exec_t, что означает, что файл должен быть выполнимым и относиться к типу auditd_t.
"/путь/к/auditd(/.*)?": это регулярное выражение, которое указывает на применение этого контекста ко всем файлам и директориям внутри /путь/к/auditd.
Эта команда добавляет новый контекст файла для всех файлов и директорий в указанном пути (/путь/к/auditd) и его поддиректориях. Это обеспечивает правильный контекст SELinux для файлов, используемых auditd.

restorecon -R -v /путь/к/auditd

restorecon: команда для восстановления контекста SELinux файлов и директорий.
-R: рекурсивно применяет команду ко всем файлам и поддиректориям указанного пути.
-v: подробный вывод, который показывает изменения, внесенные командой.
Эта команда рекурсивно восстанавливает контекст SELinux для всех файлов и директорий в указанном пути (/путь/к/auditd) и его поддиректориях, учитывая новый контекст, установленный ранее с помощью semanage fcontext.

**Отслеживание изменений конфигурации**
auditd может логировать события, связанные с изменениями его собственной конфигурации, что позволяет обнаруживать попытки изменения параметров аудита без разрешения.
Создайте файл правил аудита, который будет отслеживать изменения в конфигурационных файлах auditd. Например, файл с именем auditd-config.rules и запишите в него правило:

-w /etc/audit/audit.rules -p wa -k auditd_config

В этом примере:

-w /etc/audit/audit.rules: указывает путь к конфигурационному файлу audit.rules, который нужно отслеживать.
-p wa: определяет, какие операции нужно отслеживать. В данном случае, w означает запись (write), a - изменение атрибутов (attribute).
-k auditd_config: присваивает ключевую метку (tag) событию для легкого поиска.
Загрузите правила аудита в систему с помощью утилиты auditctl:

sudo auditctl -R /etc/audit/audit.rules

Выполните некоторые изменения в конфигурационном файле audit.rules, чтобы убедиться, что отслеживание работает, например, добавьте или удалите правило аудита. Затем

просмотрите журналы аудита, чтобы убедиться, что изменения были зафиксированы:

sudo ausearch -k auditd_config

Эта команда выведет все события аудита, связанные с ключевой меткой auditd_config, которая была определена в нашем правиле отслеживания изменений конфигурации auditd.

**Правила аудита**

Правила аудита (audit rules) определяют, какие события и действия в операционной системе должны записаны в журнал аудита. Может быть определено:

- Слежение за доступом к файлам и каталогам: запись попыток чтения, записи, выполнения или изменения прав доступа к файлам и каталогам.
- Отслеживание действий пользователей: запись входа и выхода пользователей из системы, смены привилегий, создание или удаление учетных записей и других административных действий.
- Аудит сетевых событий: запись попыток подключения к сетевым ресурсам, отправку или получение сетевого трафика и других сетевых действий.
- Мониторинг системных вызовов: запись системных вызовов, таких как создание процессов, открытие файлов, управление процессами и т. д.

Правила аудита обычно хранятся в файле конфигурации audit.rules.

**Ключи правил**

| | |
|---|---|
| -a | Действие для события, соответствующего правилу. Возможные значения включают:<br>• always: Применять правило всегда.<br>• never: Никогда не применять правило.<br>• exit: Применять правило только при выходе из процесса.<br>• creat,open,openat,truncate,ftruncate,unlink,mkdir,rmdir |

| | |
|---|---|
| -F | Условия, которым должны соответствовать аудитируемые события. Ключ -F может повторяться.<br>`-F path=/etc/passwd: Определяет путь к файлу.`<br>`-F perm=w: Определяет разрешение (например, запись в файл).`<br>`-F uid=0: Определяет идентификатор пользователя.`<br>`-F arch=b64: 64-битные приложения`<br>`-F key=file_modification: Присваивает ключевую метку событию для легкого поиска.` |
| -S | Системные вызовы, которые должны быть аудиторованы.<br>• open: Отслеживать системные вызовы открытия файлов.<br>• execve: Отслеживать системные вызовы выполнения новых программ. |
| -C | Условия по контексту, таким как идентификатор пользователя (UID) или группы (GID).<br>-C uid=0: Определяет аудитируемые события для пользователя с UID 0 (root).<br>-C gid=adm: Определяет аудитируемые события для группы с именем "adm". |
| -k | Присваивает ключевую метку событию, что позволяет идентифицировать события. |
| | |
| | |
| | |

**Примеры правил**

Отслеживание доступа к файлам

```
-a always,exit -F path=/etc/shadow -F perm=r -k sensitive_files_access
```

Отслеживание запуска привилегированных команд

```
-a always,exit -F arch=b64 -S execve -C uid=0 -k privileged_commands
```

Отслеживание изменений файловой системы

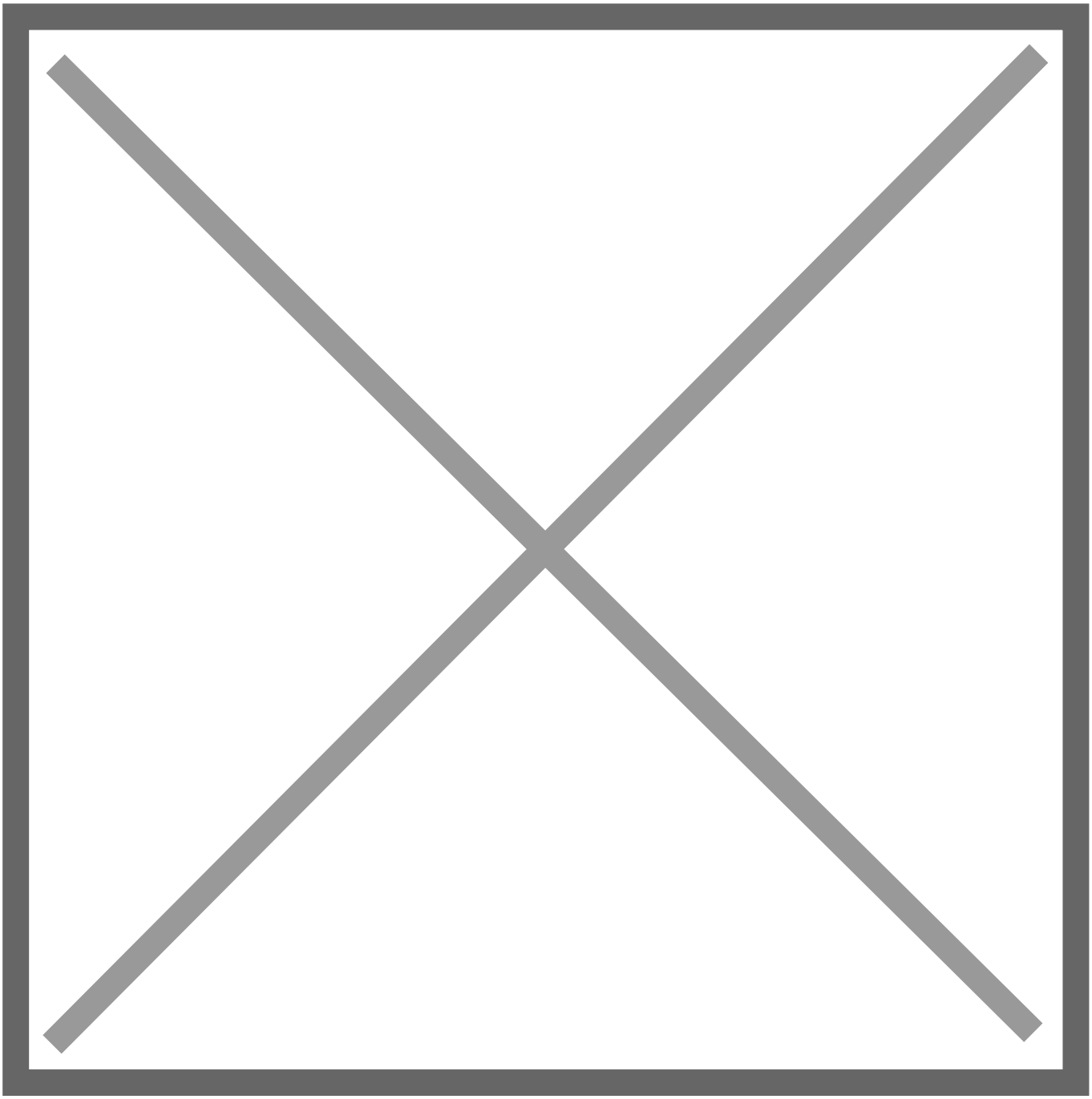
```
-a always,exit -F arch=b64 -S creat,open,openat,truncate,ftruncate,unlink,mkdir,rmdir -F key=file_modification
```

**Пример настройки правил в целом**

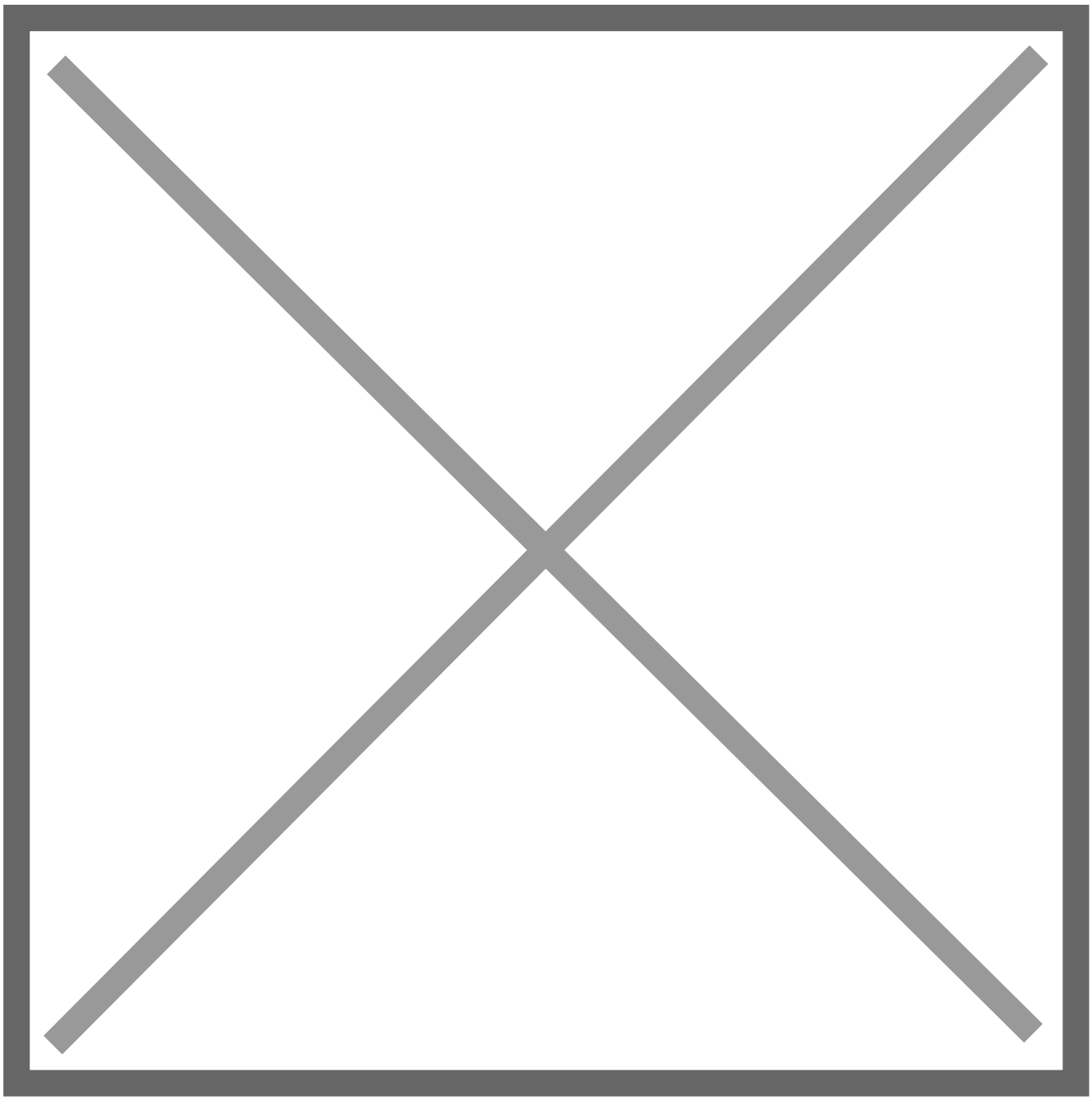
Мы уже рассматривали анализ логов auditd ранее в курсе. Теперь давайте рассмотрим стандартную конфигурацию audtid, где уже собрано большинство полезных правил:

https://github.com/Neo23x0/auditd/blob/master/audit.rules

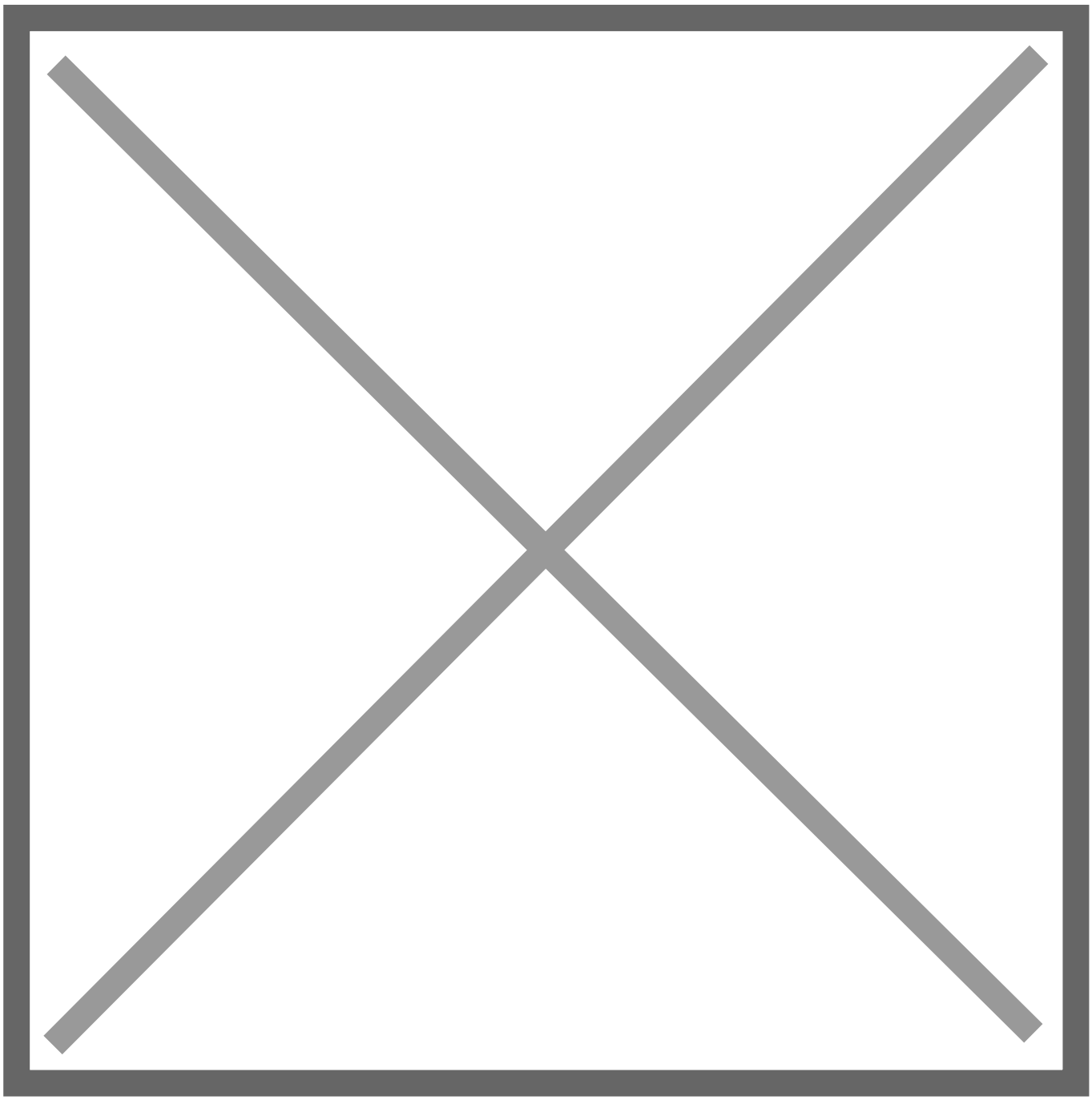
Первым делом мы удаляем все существующие правила командой `-D`:



Затем настраиваем размер буфера в байтах:

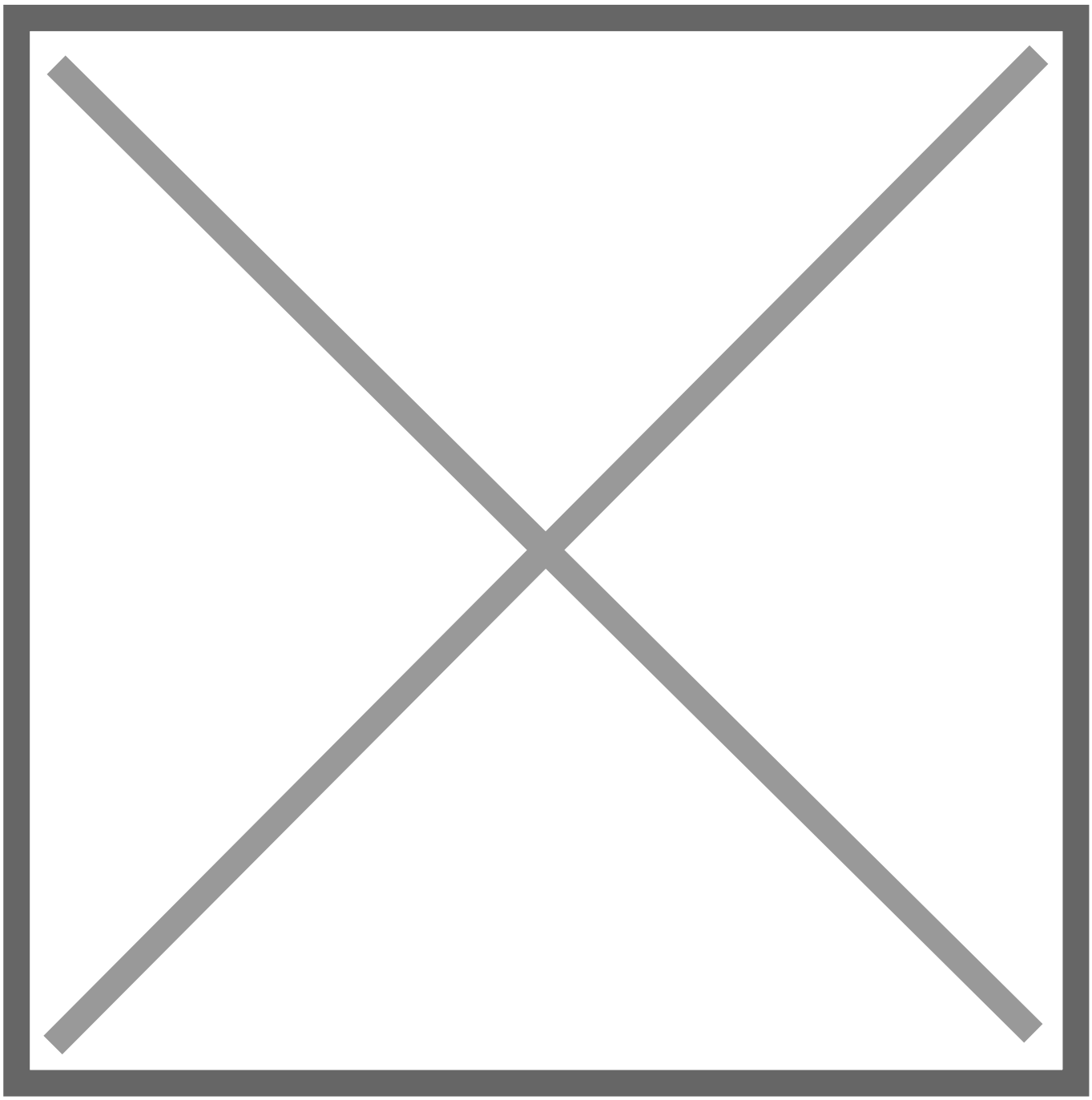


Этот параметр очень важен в сочетании со следующим, -f. Параметр -f (failure) отвечает за поведение системы в случае отказа модуля auditd, в значении 0 при отказе auditd ничего не происходит, в значении 1 выводится сообщение об ошибке, и в значении 2 система прекращает работу.



Параметр `-i` позволяет игнорировать некоторые ошибки, например отсутствие файлов, которые мы указали в правилах мониторинга.

Следующим шагом идет мониторинг доступа к логам и исполняемым файлам auditd. Поскольку в логах могут содержаться очень чувствительные данные (например, пароли, или команды, выполняющиеся в cron от имени администратора) необходимо отслеживать обращения к логам auditd, как успешные, так и неуспешные.



Давайте рассмотрим правила мониторинга файлов на этом примере.

Формат правила для мониторинга файлов и директорий выглядит следующим образом:

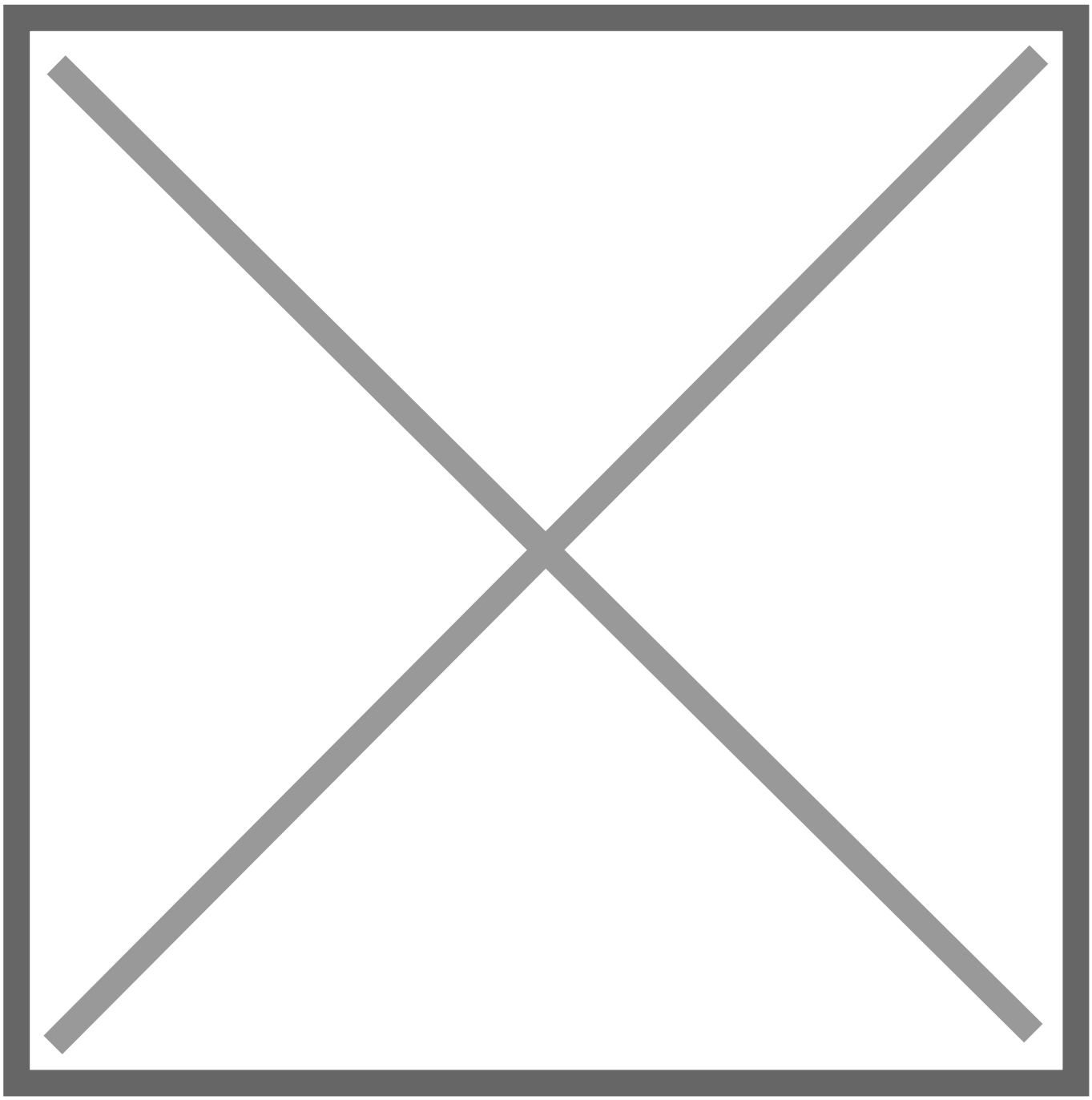
```
auditctl -w путь_к_файлу -p разрешения -k имя_ключа
```

- `путь_к_файлу` — путь к файлу или наблюдаемой директории.
- `-p` (permissions) — фильтр разрешений доступа, которые мы будем мониторить, а именно:
  - `r` — доступ на чтение файлов или директорий;

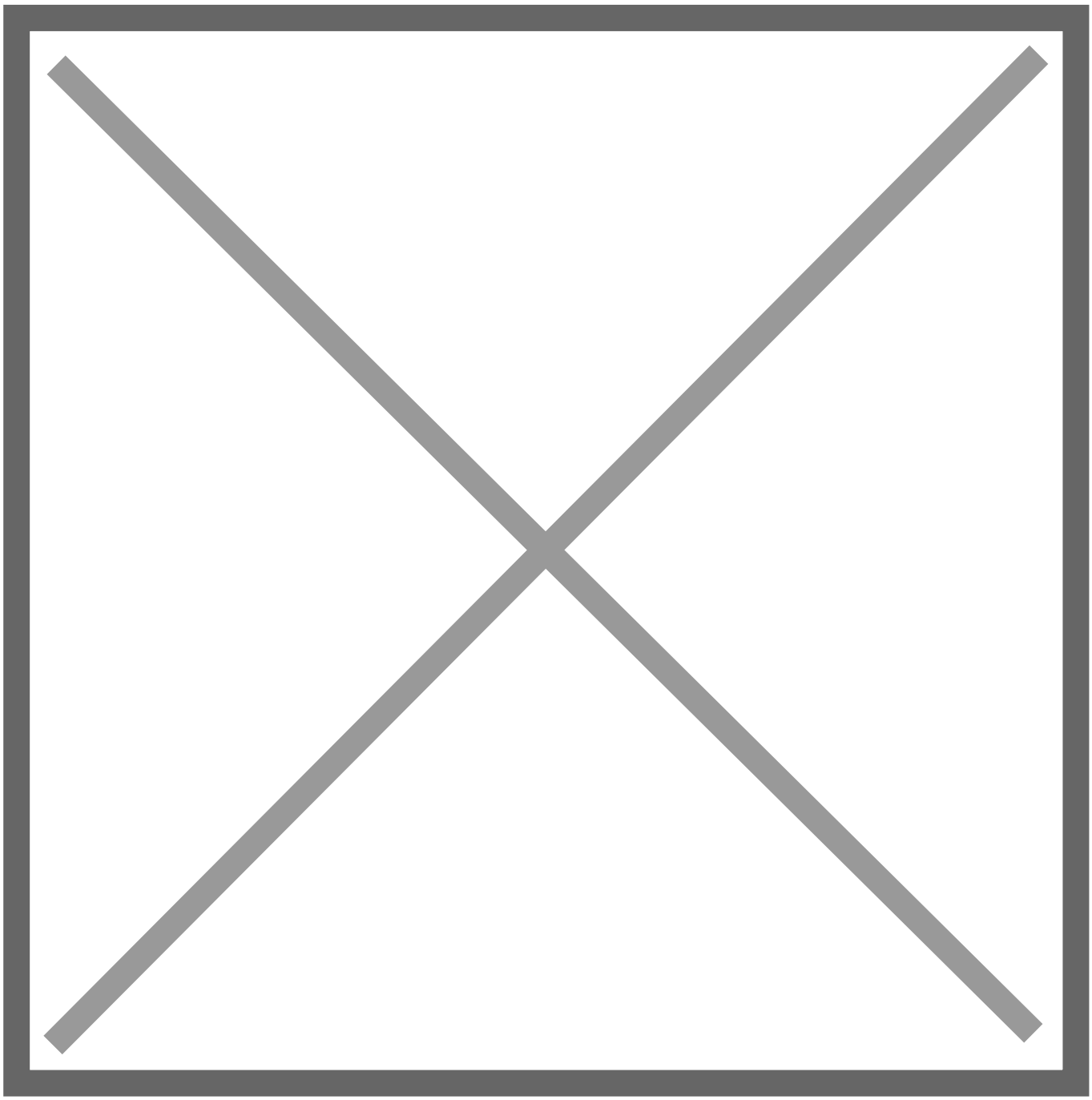
    - `w` — доступ на запись;
    - `x` — выполнение;
    - `a` — изменение аттрибутов.
- `-k` — ключ, с которым auditd запишет это событие в лог. Полезен для быстрого поиска и фильтрации событий.

Итого, мы отслеживаем запись(w), чтение(r) и изменение аттрибутов(a) для каталогов `/var/log/audit` и `/var/audit`.

Продолжим:

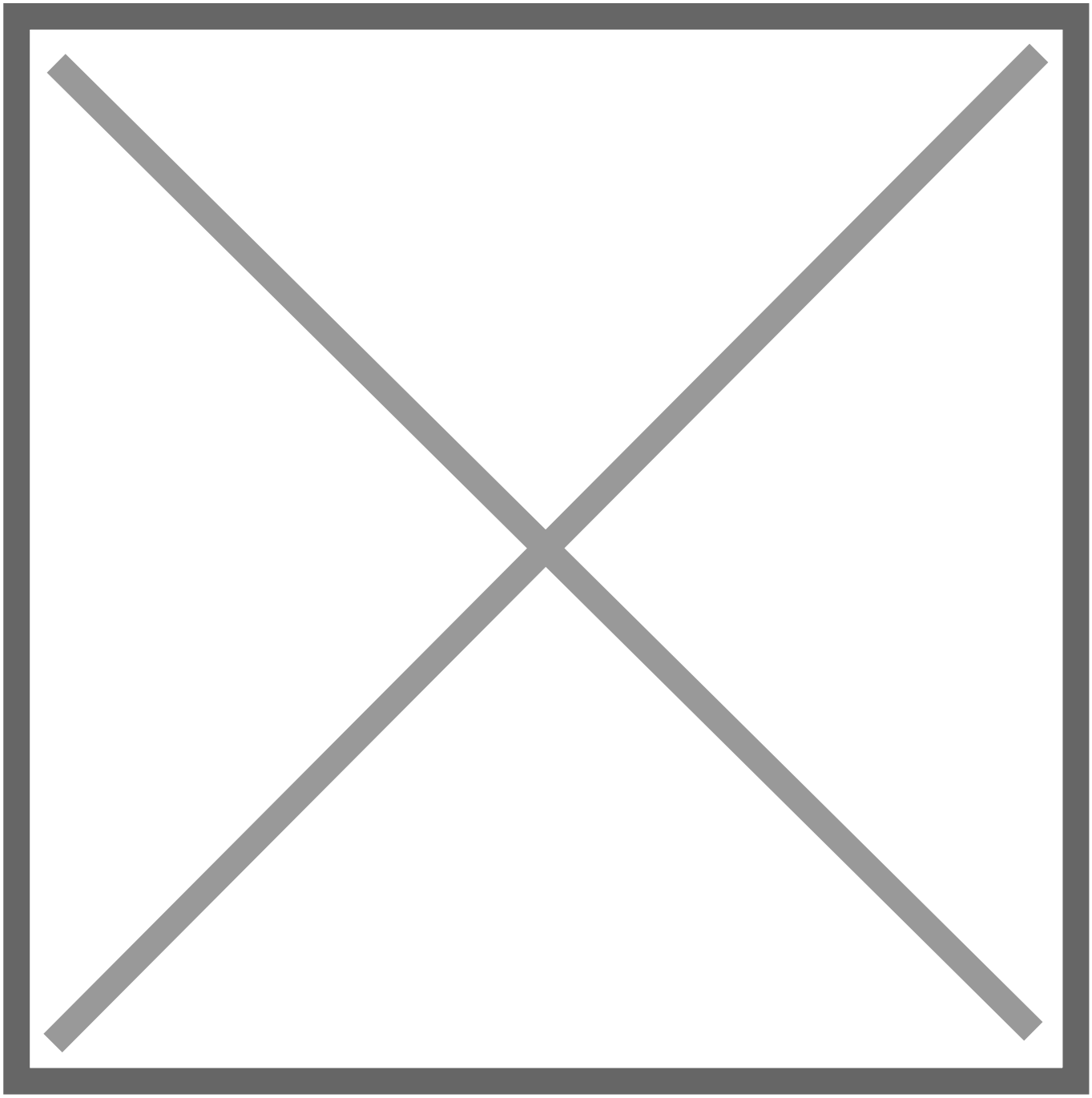


Здесь мы отслеживаем изменение конфигурации auditd.



А здесь выполнение (ключ `-x`, execute) команд, которые могут повлиять на конфигурацию auditd: auditctl, auditd, augenrules.

Продолжим изучать файл конфигурации auditd:



Рассмотрим правило отслеживания системных вызовов на этом примере. В общем случае правило выглядит так:

```
auditctl  -a действие,фильтр [ -F arch=архитектура_системы -S имя_системного_вызова] -F
поле=значение  -k имя_ключа
```

Где:

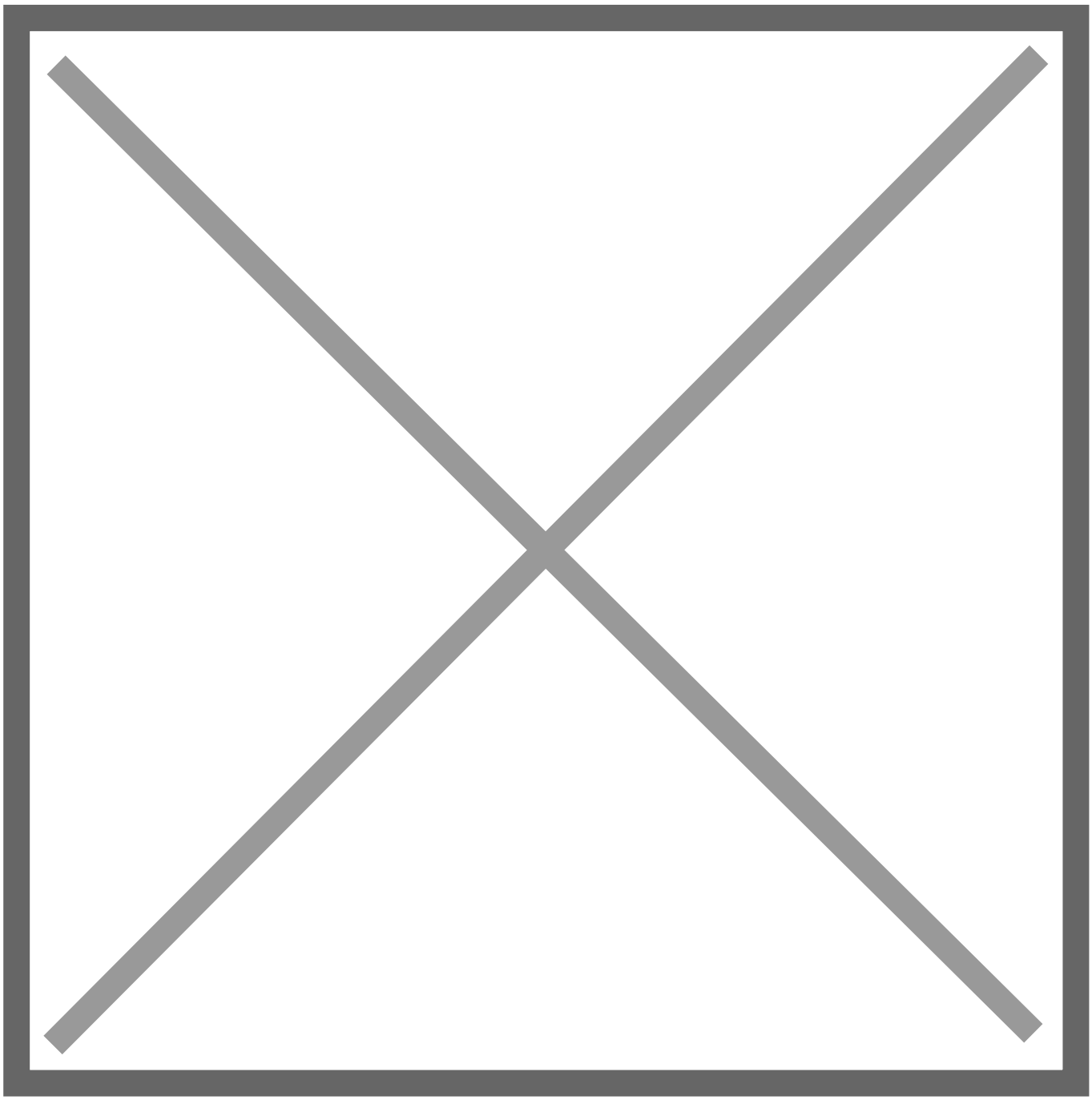
- ***Действие и фильтр*** *определяют, какое событие будет регистрироваться*. *Действие(action) может быть либо* `always` , либо `never`. Параметр *filter* определяет

какой фильтр событий ядра будет применяться к данному правилу. Возможные значения фильтра: `task`, `exit`, `user`, and `exclude`. В большинстве случаев используется фильтр `exit`, когда событие записывается при завершении системного вызова.
- ***system_call****(имя_системного_вызова)* позволяет применить фильтр на основании системного вызова. Мы можем указать несколько системных вызовов используя ключ -S.
- ***-F*** *— фильтр, позволяющий нам отслеживать определенные события, которые нам интересны*. Фильтров может быть несколько, они указываются в формате "ключ"[!=><]"значение", например, path=/usr/bin/ls или auid>=1000.
- **-k имя_ключа** *— опциональный, но очень полезный параметр, позволяющий добавить метку для отслеживаемого события*.

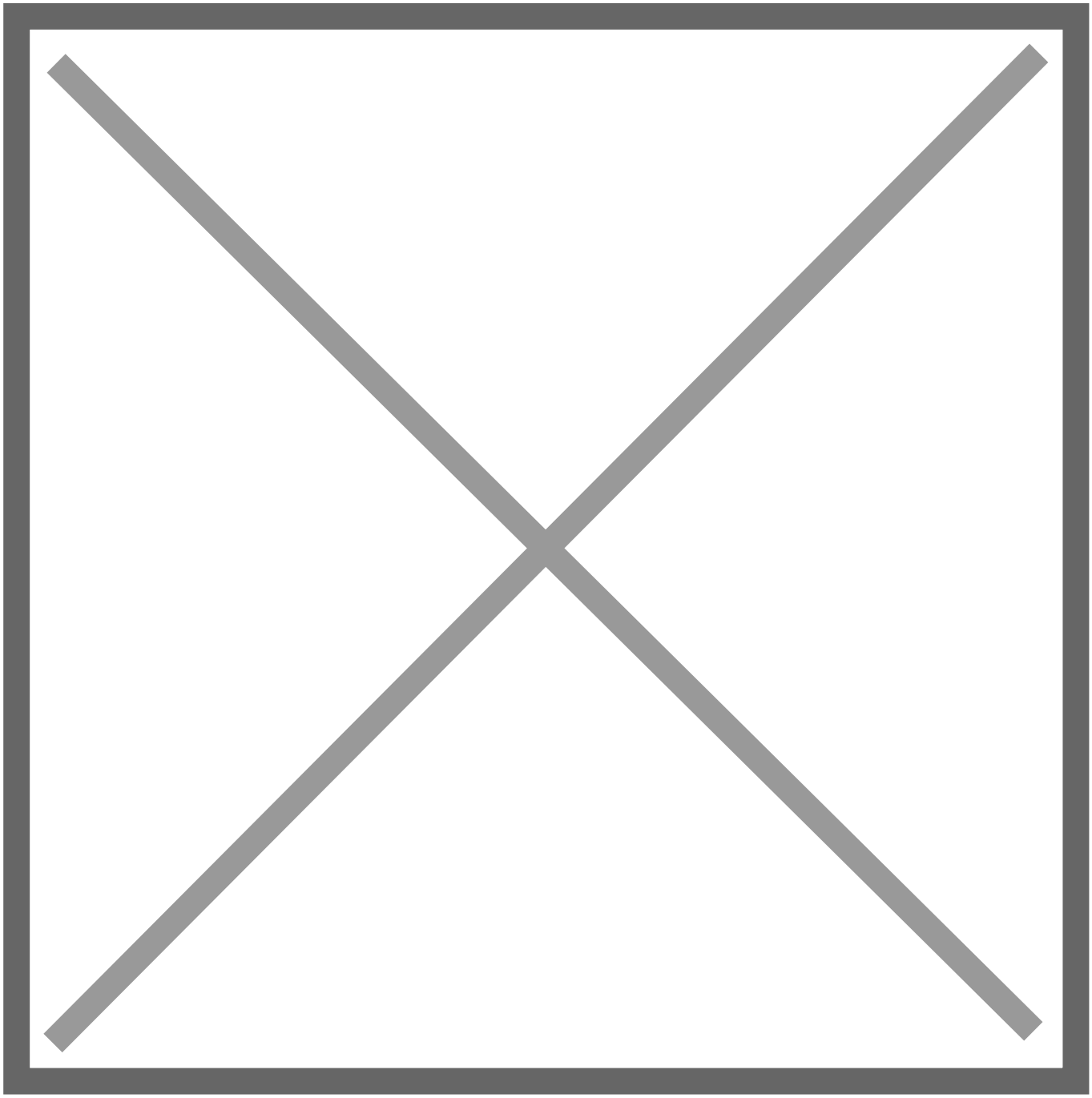
Итак, на примере выше мы мониторим события запуска (-F perm=x) команд auditd (ausearch, aureport, и т.д)  после их выполнения (exit) и помечаем эти события меткой audittools.

> *Использование ключа  -a never позволяет исключать события из мониторинга. Необходимо учитывать, что правила читаются сверху вниз, поэтому правила-исключения лучше помещать в начало файла конфигурации auditd.*

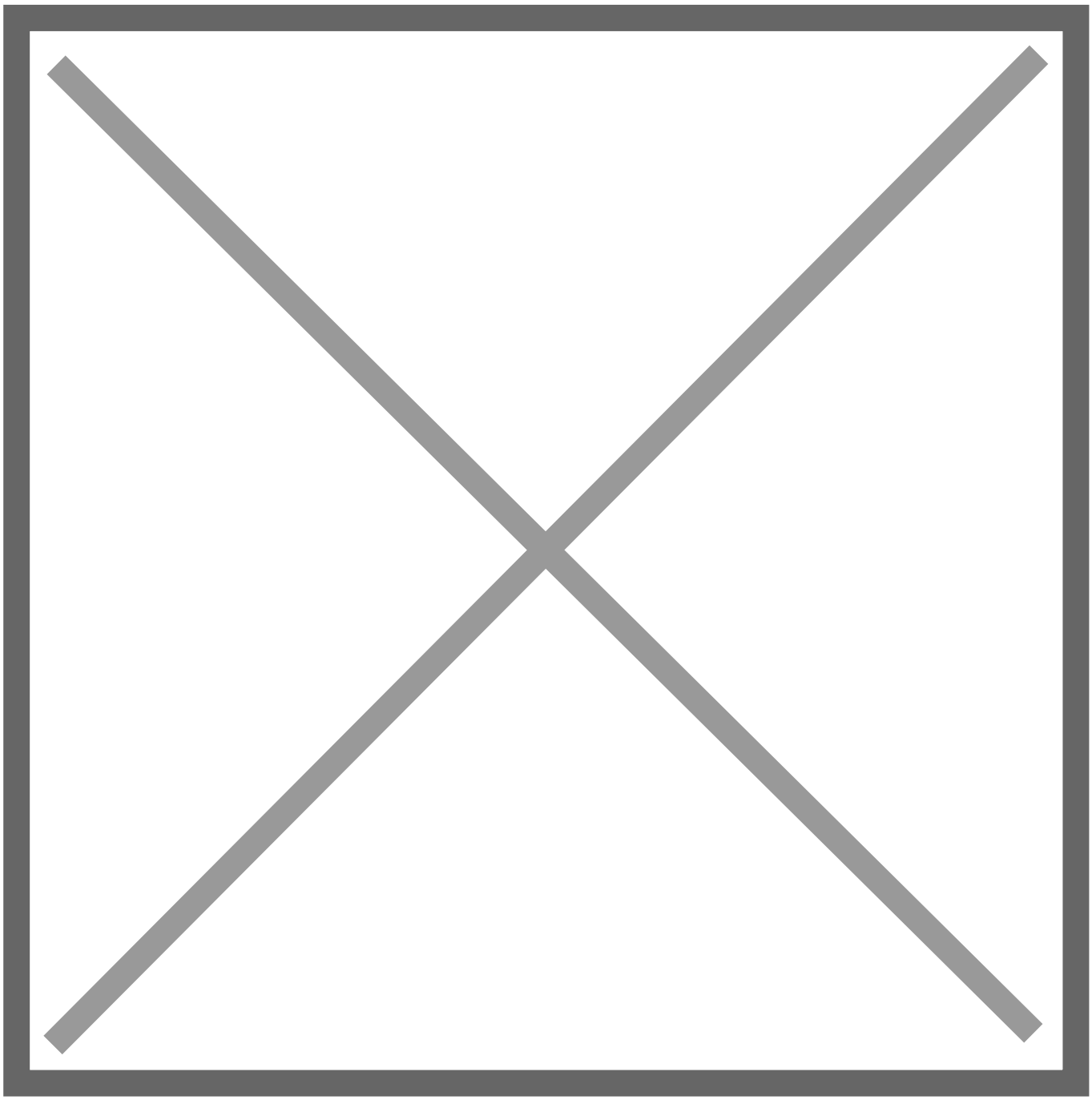
Далее идут исключения:



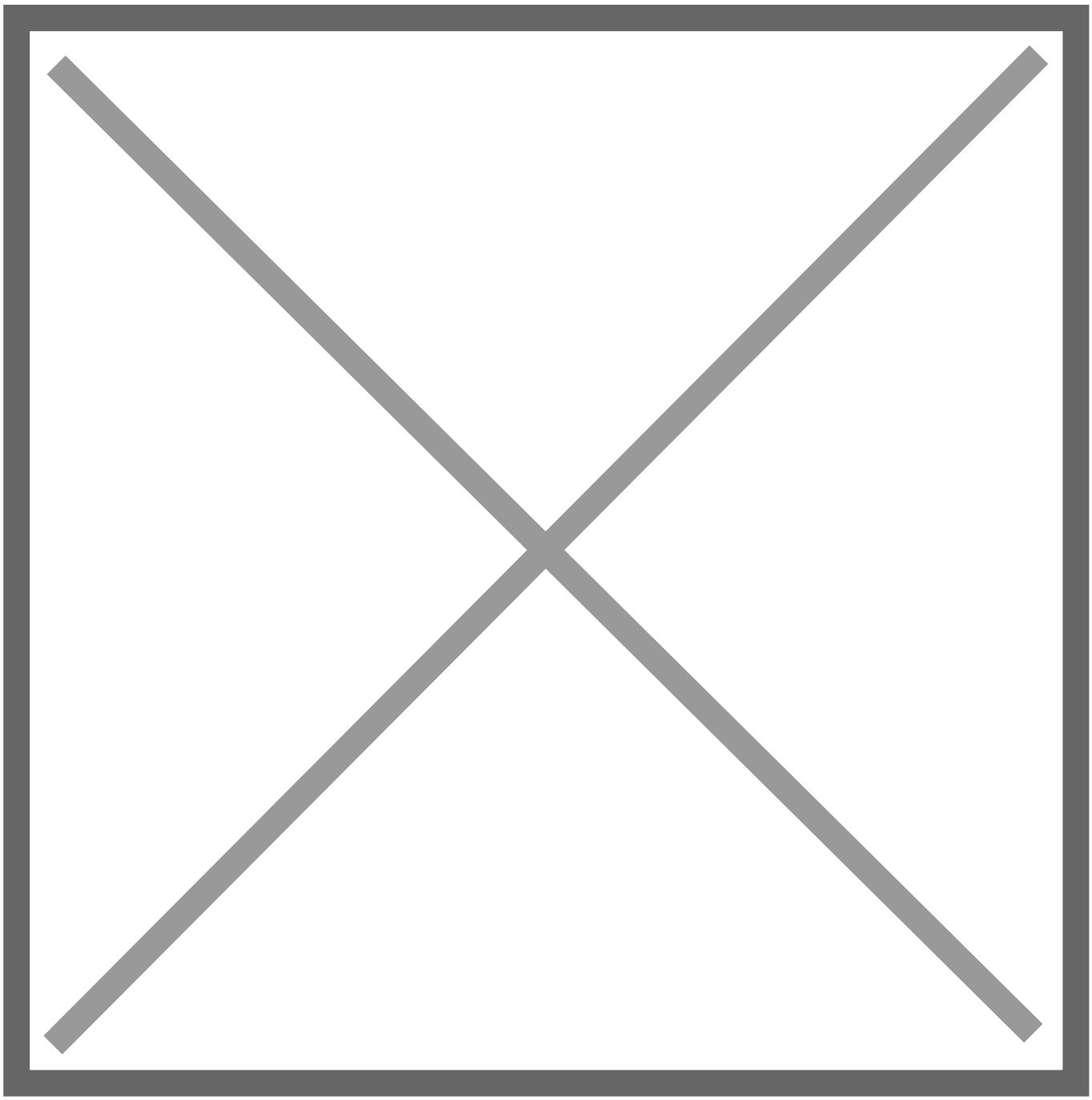
Мы видим два типа исключений  -a never и -a always,exclude.



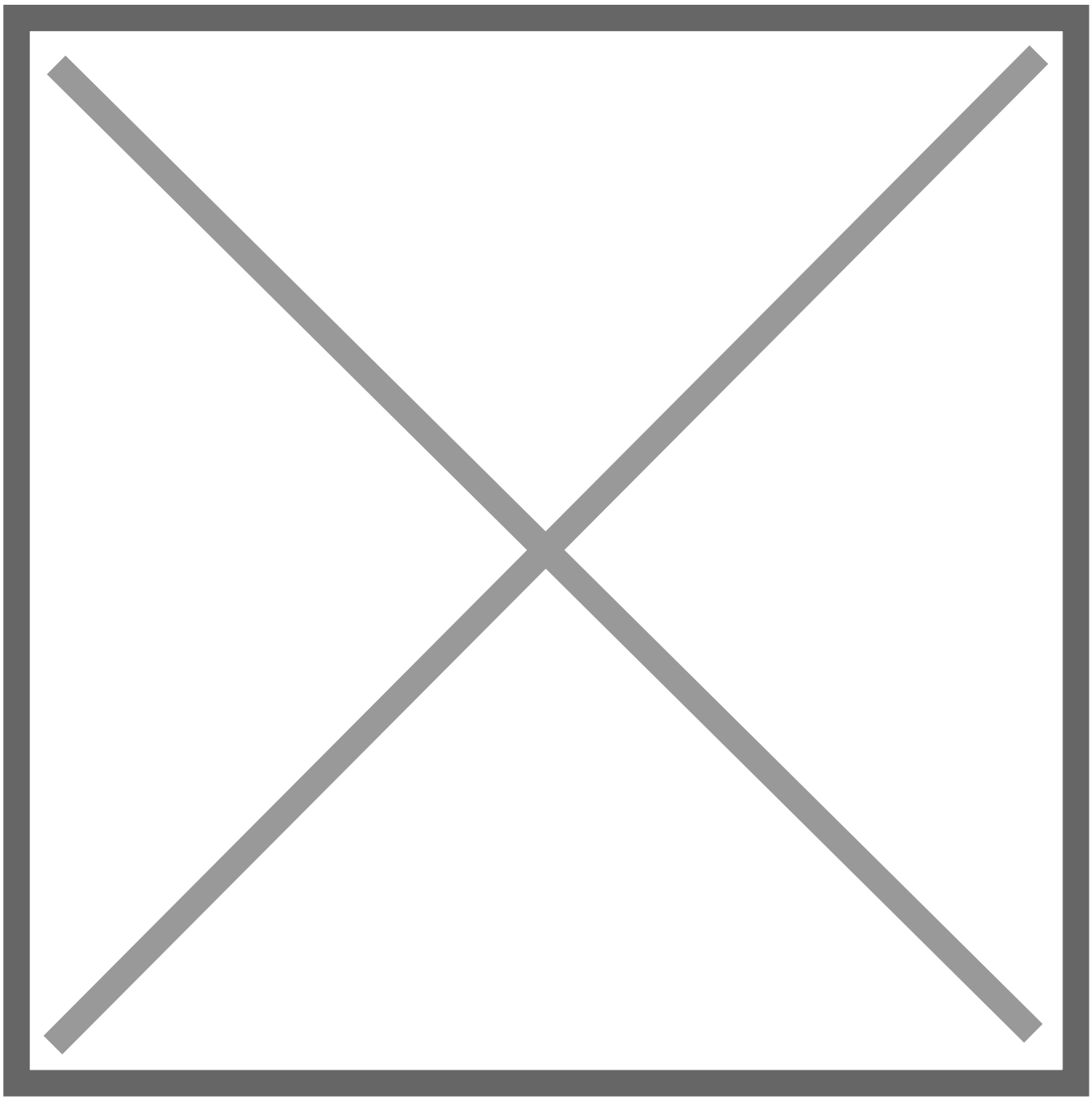
Поскольку при работе с инцидентом мы всегда опираемся на метки времени, очень важно следить, чтобы на всех устройствах время было синхронизировано. Атакующие могут попытаться изменить текущее время на время в прошлом, чтобы затруднить поиск событий. Данное правило мониторит изменение времени с помощью системных вызовов, а также запись и изменение атрибутов файла /etc/localtime.



Мониторинг доступа к базам данных пользователей системы и паролям, а также к файлу sudoers.



Отслеживание выполнения команды passwd.



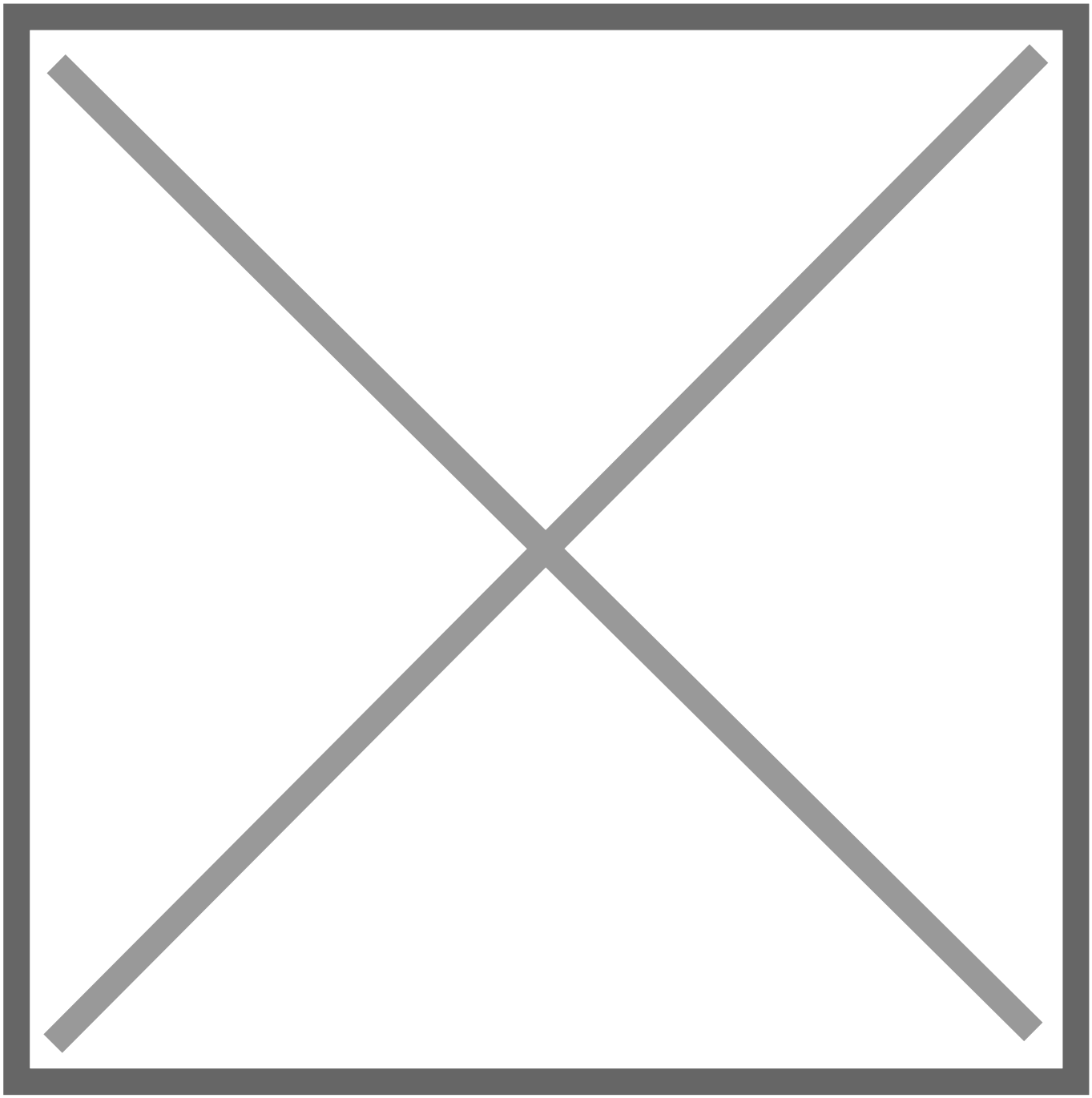
С помощью мониторинга системного вызова *connect* мы можем отслеживать сетевые соединения, и даже ловить remote shell.

В примере выше мы получим запись в лог, если процесс bash успешно откроет сетевое подключение.

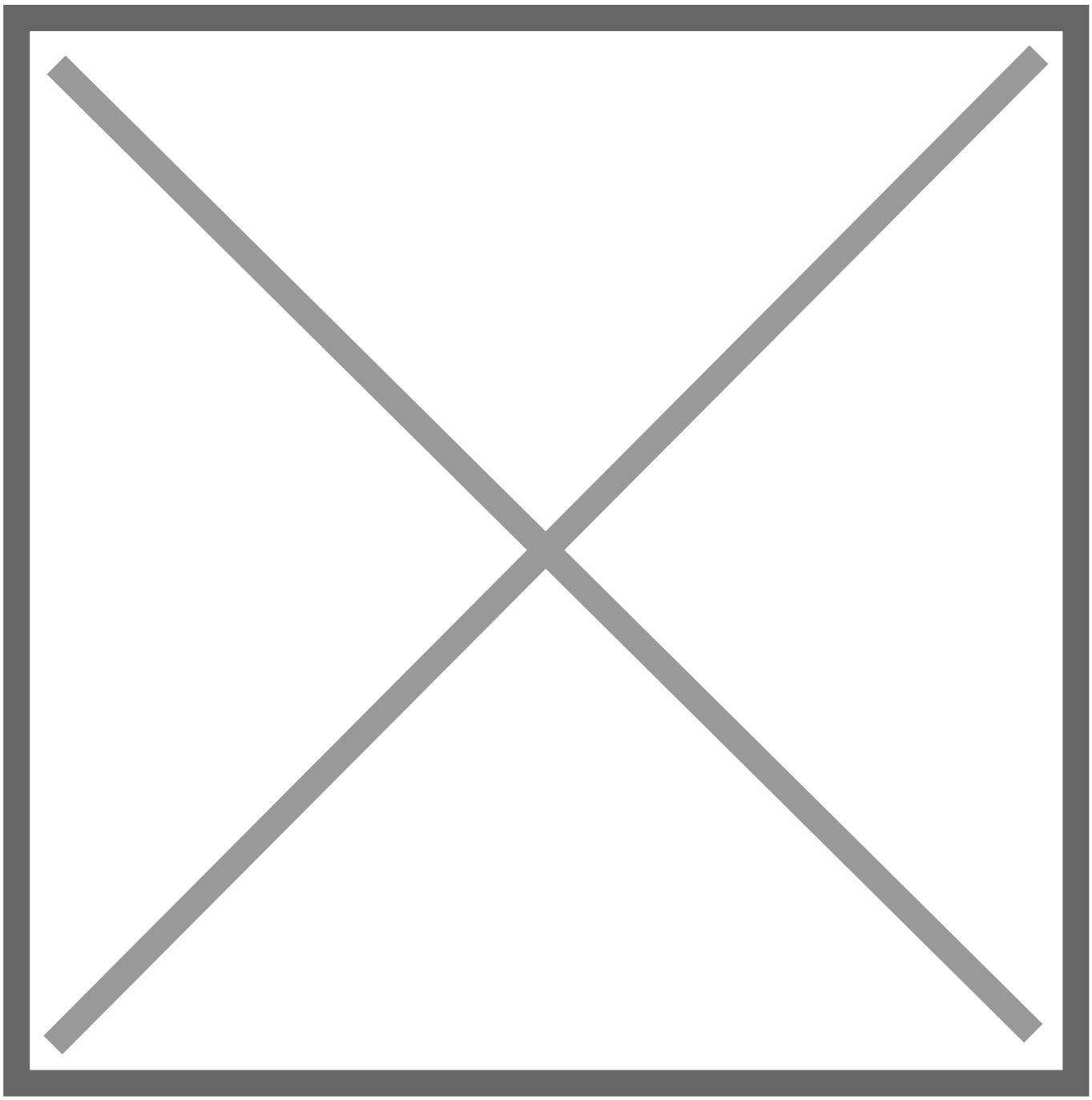
> Мониторинг всех сетевых подключений может генерировать огромное количество событий, особенно на серверах, которые предоставляют публичные сервисы, например, веб-сервера. Такие правила необходимо использовать с осторожностью, особенно, если у нас включен параметр -

> f=2 который вешает систему при переполнении очереди событий auditd.

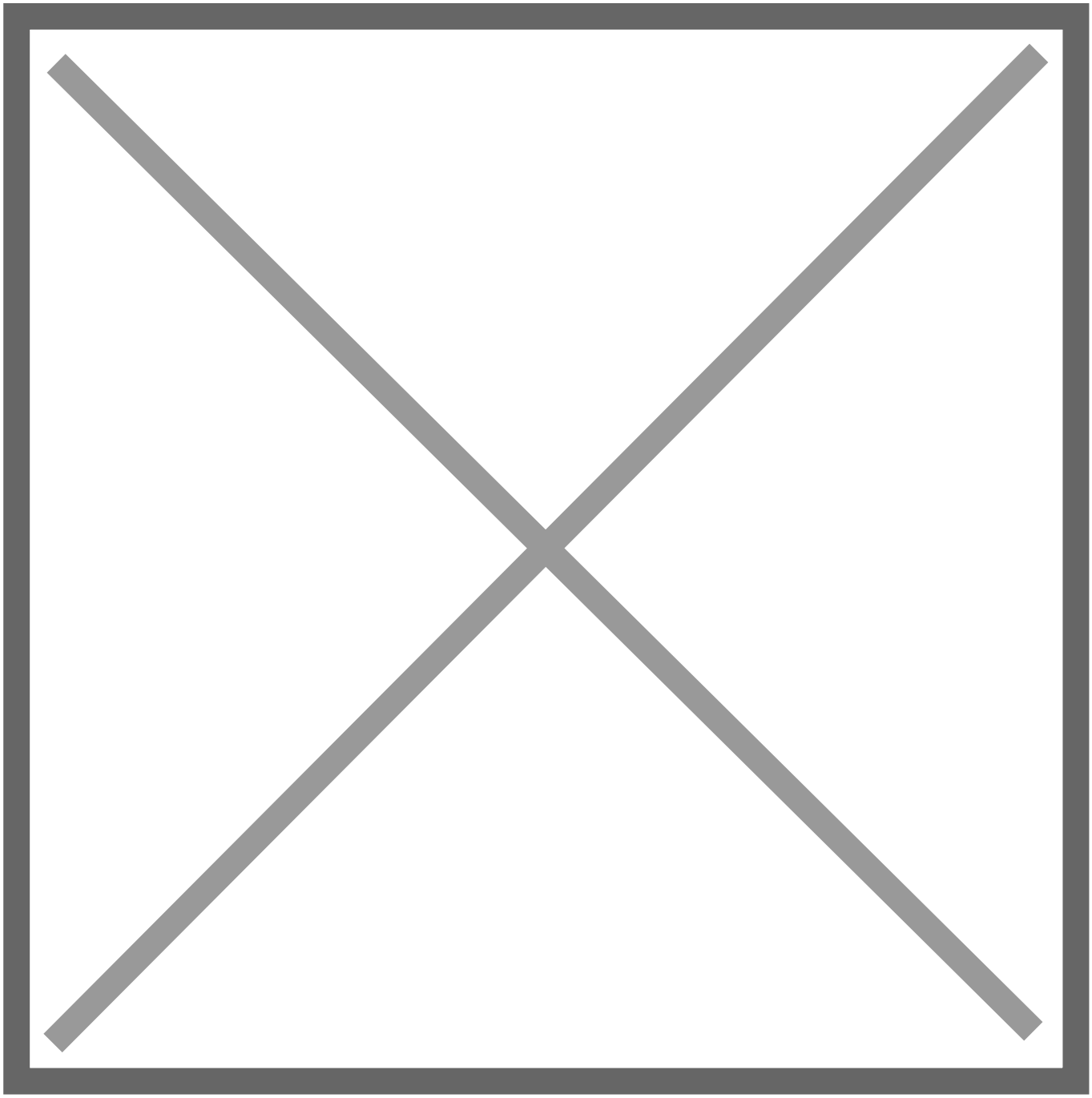
Идем дальше.



Тут мы отслеживаем неуспешные(succes=0) попытки доступа к системным директориям.

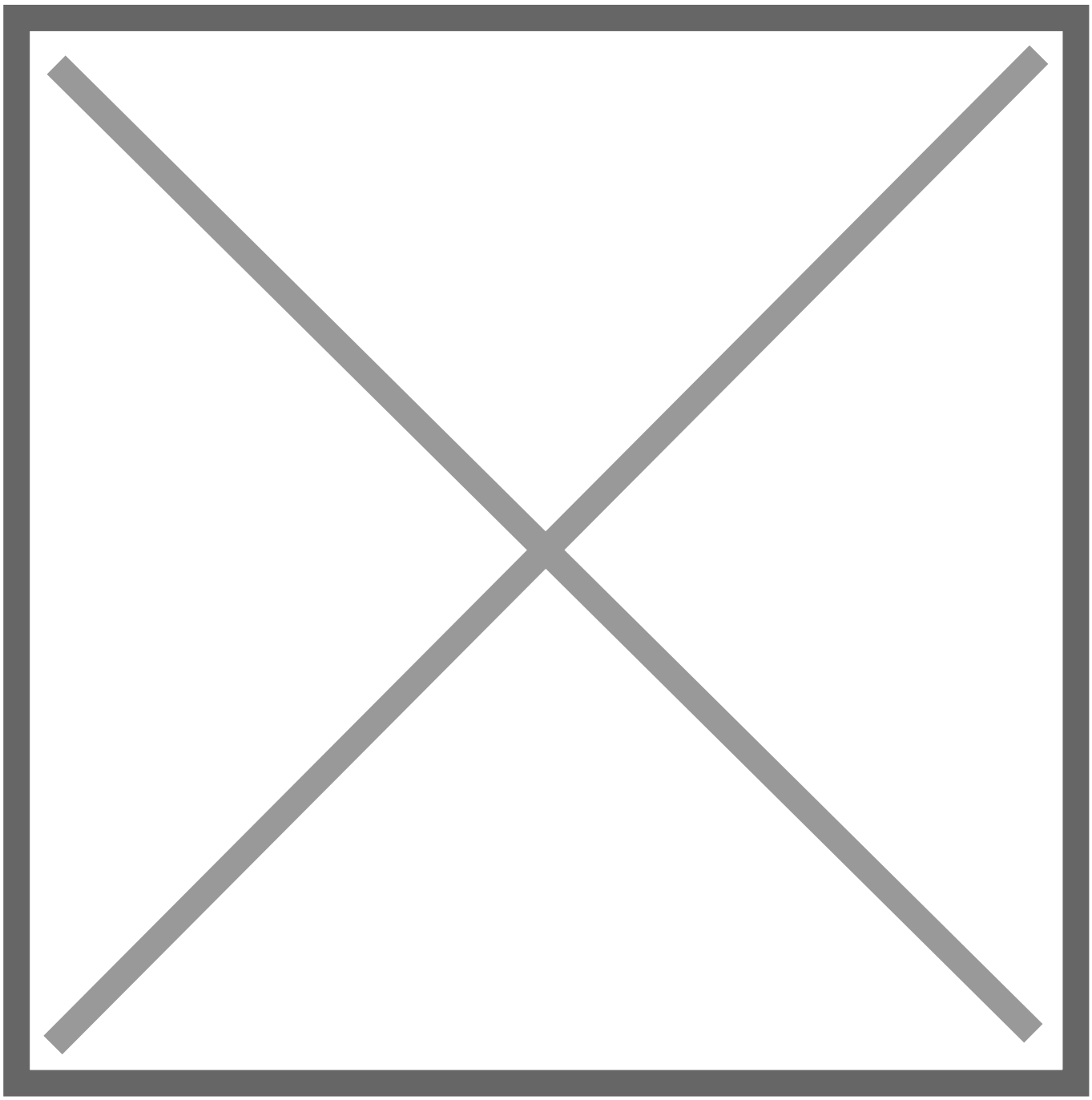


Мониторинг выполнения команд, часто используемых атакующими. Необходимо помнить, что в зависимости от окружения эти правила могут вызвать множество ложных срабатываний, поскольку эти команды могут использоваться как системными администраторами, так и скриптами.

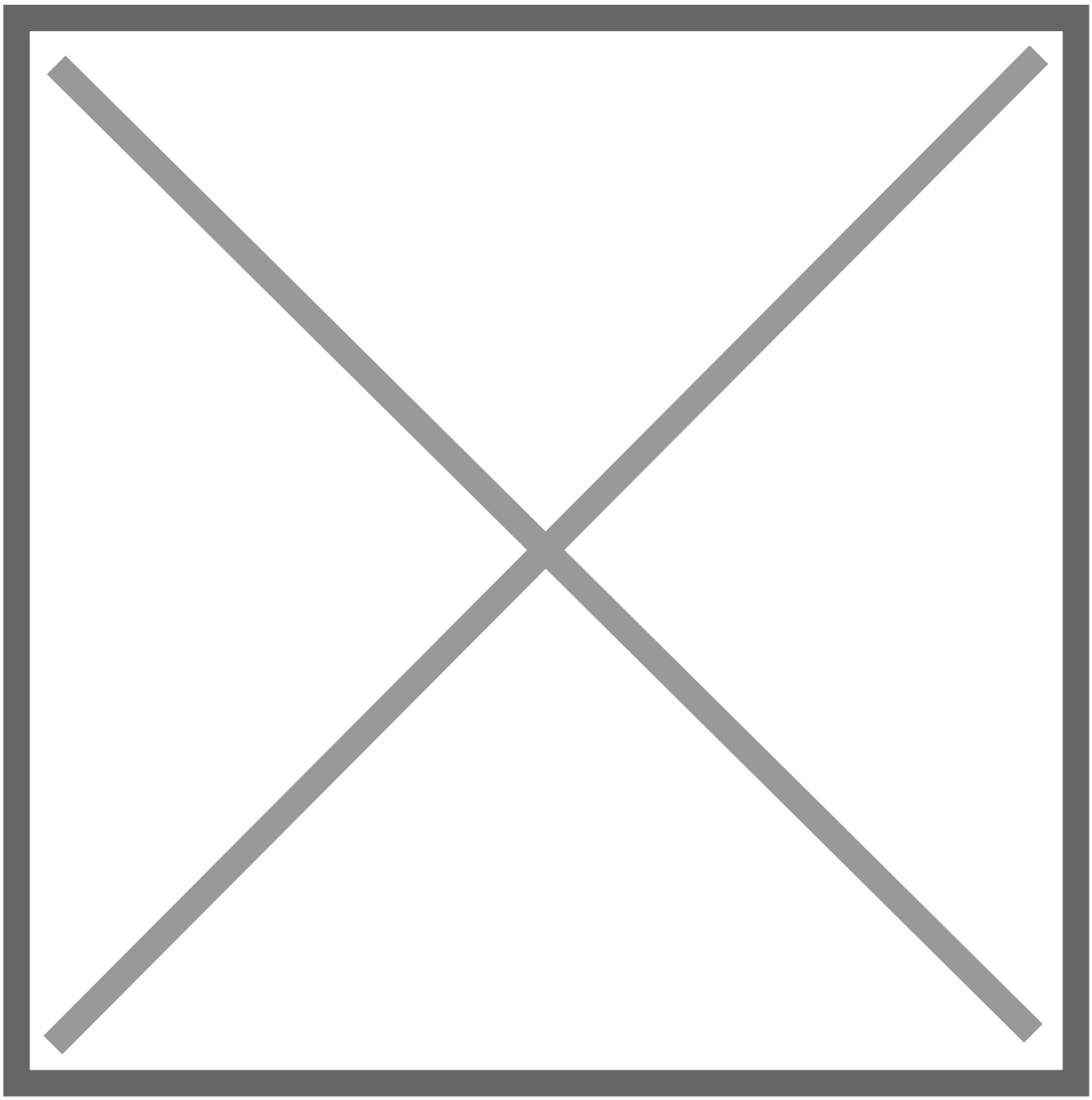


Запуск программ sssd пользователями.

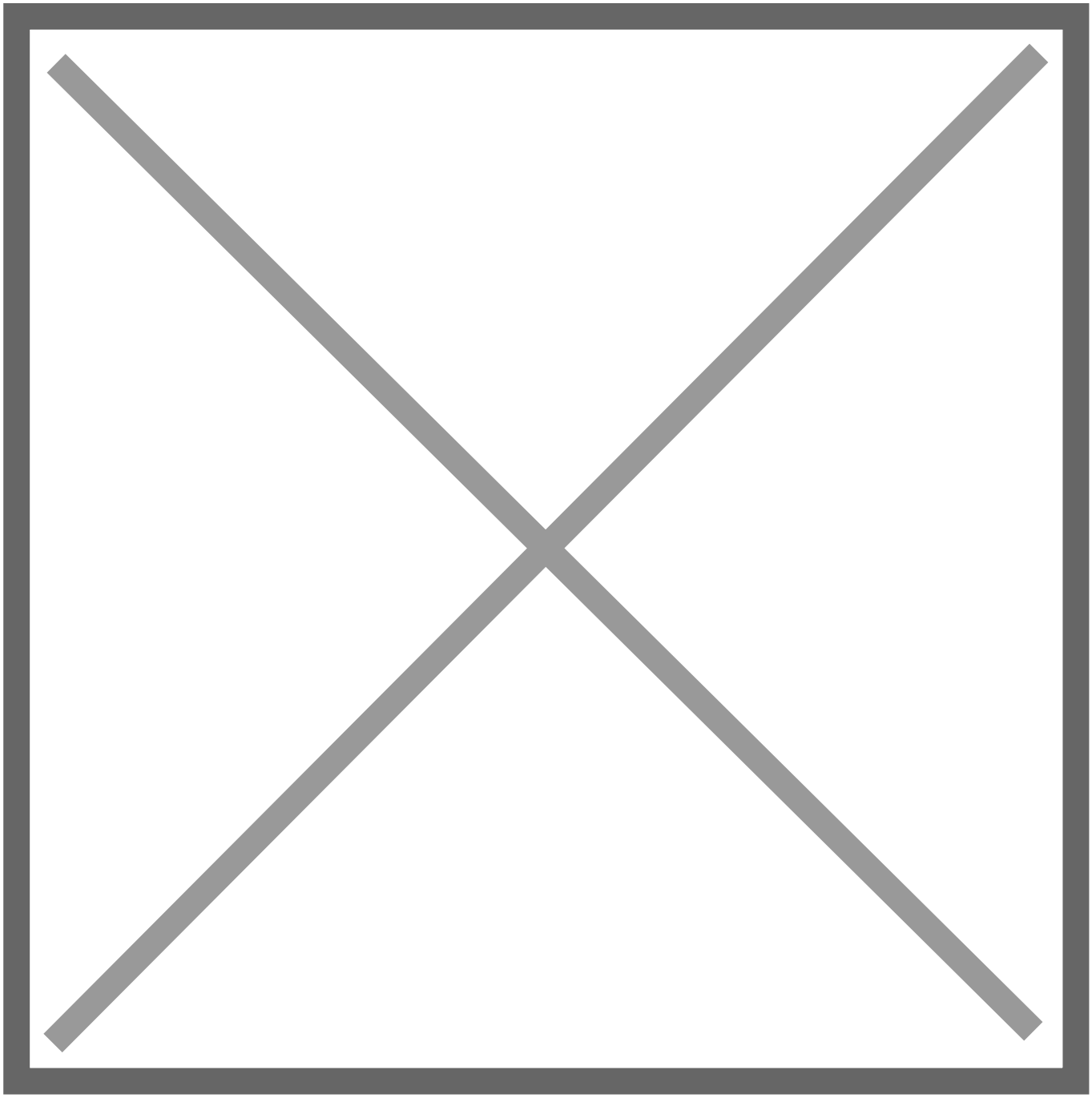
> В описании правил можно часто встретить auid!=4294967295. Это число транслируется системой в -1, и означает, что UID еще не установлен системой. Вместо 4294967295 можно использовать число "-1" или значение "unset".



Отслеживание запуска команд от пользователя www-data. Применимо к веб-серверам, euid(effective uid) может отличаться. Правило может помочь обнаружить взлом веб-сервера(например, пользователь www-data запускает bash),  но также может генерировать большое количество ложно-положительных срабатываний, если от имени веб-сервера запускаются какие-либо скрипты.

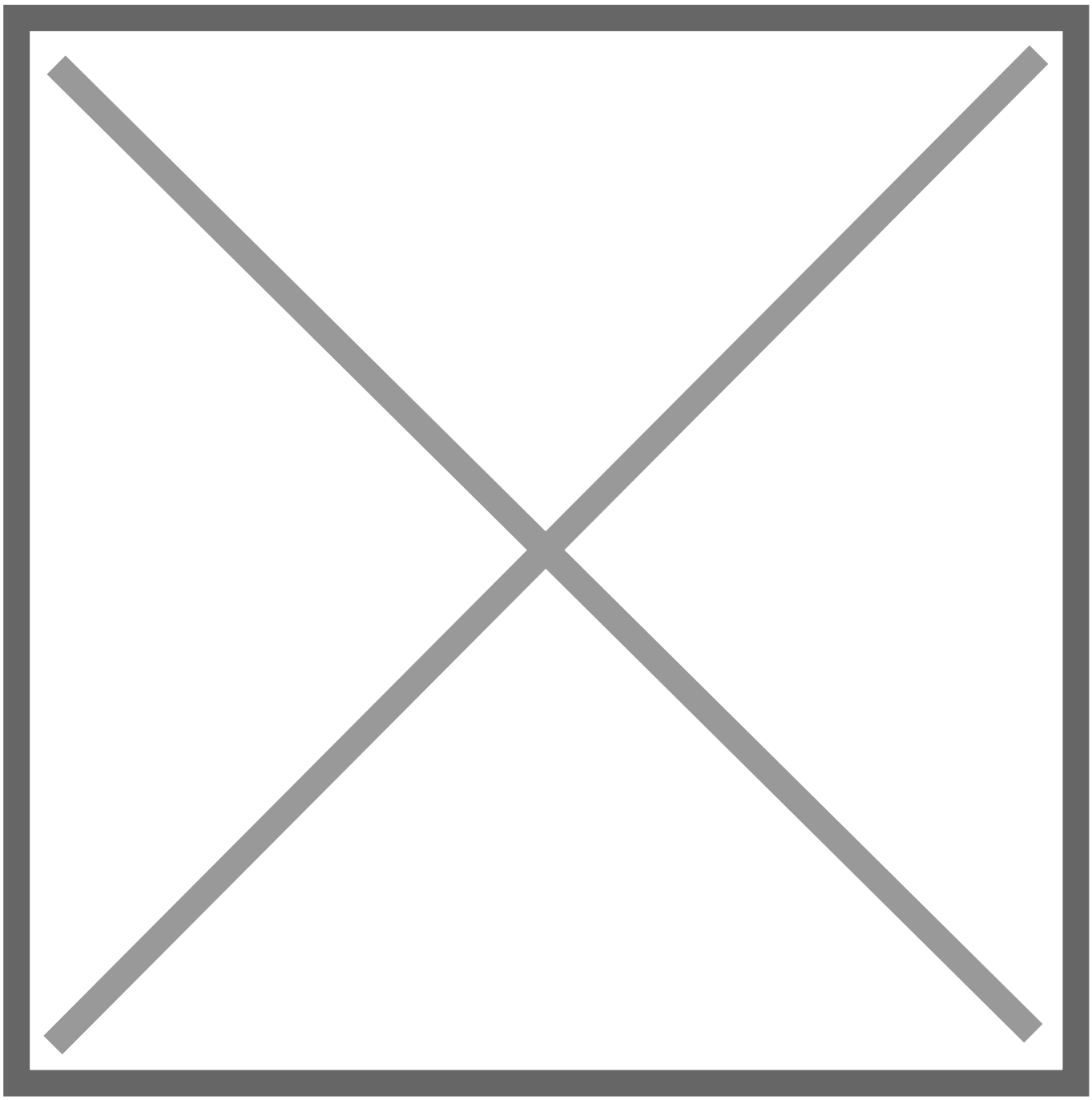


Отслеживание выполнения команд от пользователя root(euid=0). Это правило может генерировать очень много событий, следует использовать с осторожностью.



Удаление файлов пользователями системы(auid>=1000).

И, последней строчкой мы можем добавить правило, запрещающее дальнейшее изменение праил auditd. За это отвечает ключ -e (enable). Значение 0 позволяет выключить мониторинг, значение 1 - включить, а значение 2 делает конфигурацию неизменяемой.



Revision #10
Created 18 October 2025 16:50:18 by Admin
Updated 30 October 2025 14:18:10 by Admin