Лабораторная работа №15

Отчет

Мосолов Александр Денисович

Содержание

Список иллюстраций

1 Цель работы

Получение навыков по работе с журналами системных событий.

2 Задание

1. Настройте сервер сетевого журналирования событий.
2. Настройте клиент для передачи системных сообщений в сетевой журнал на сервере.
3. Просмотрите журналы системных событий с помощью нескольких программ.
4. Напишите скрипты для Vagrant, фиксирующие действия по установке и настройке сетевого сервера журналирования.

3 Выполнение лабораторной работы

Начинаю настройку сервера сетевого журнала. Для этого перехожу в каталог конфигурации rsyslog и создаю файл `netlog-server.conf`, который будет содержать настройки для приема логов по сети (рис. 3.1).

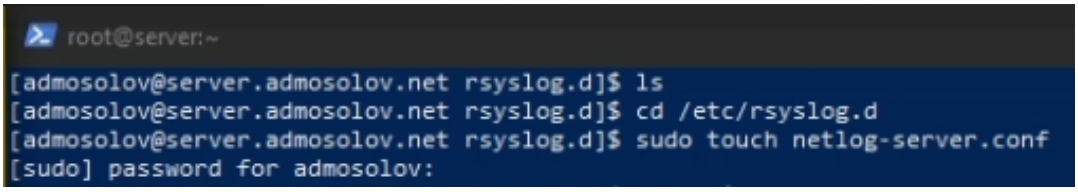


Рис. 3.1: Создание файла конфигурации на сервере

Редактирую созданный файл конфигурации `/etc/rsyslog.d/netlog-server.conf`. Включаю модуль `imtcp` для поддержки протокола TCP и указываю прослушивание порта 514. Это стандартный порт для службы syslog при использовании надежного транспортного протокола (рис. 3.2).

Рис. 3.2: Настройка приема логов по TCP

После внесения изменений в конфигурацию необходимо перезапустить службу `rsyslog`, чтобы новые параметры вступили в силу (рис. 3.3).

Рис. 3.3: Перезапуск службы rsyslog

Проверяю, что служба действительно начала слушать нужный порт. Использую команду `lsof` с фильтрацией по TCP. Видно, что процесс `rsyslogd` прослушивает порт 514 (обозначен как `shell` в выводе `lsof` для этого порта) (рис. 3.4).

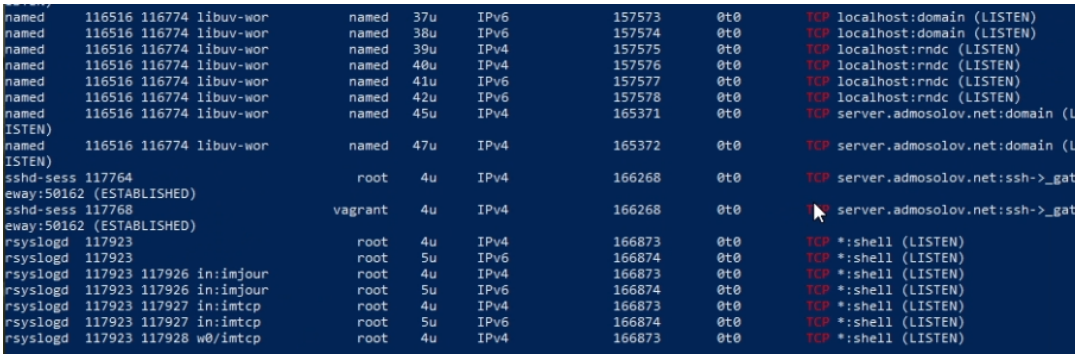


Рис. 3.4: Проверка открытых портов

Чтобы сервер мог принимать соединения извне, настраиваю межсетевой экран (firewall). Добавляю правило, разрешающее входящий трафик на порт 514/tcp, и закрепляю его как постоянное (`--permanent`) (рис. 3.5).

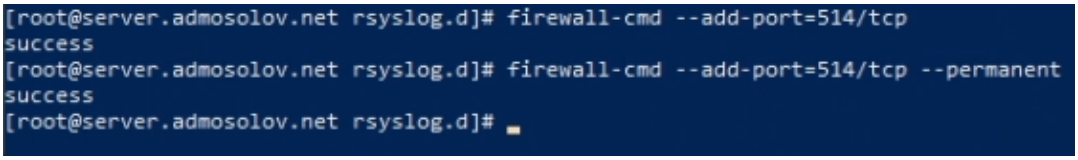


Рис. 3.5: Настройка Firewall на сервере

Перехожу к настройке клиента. На виртуальной машине клиента захожу под суперпользователем, перехожу в каталог конфигурации rsyslog и создаю файл `netlog-client.conf` (рис. 3.6).

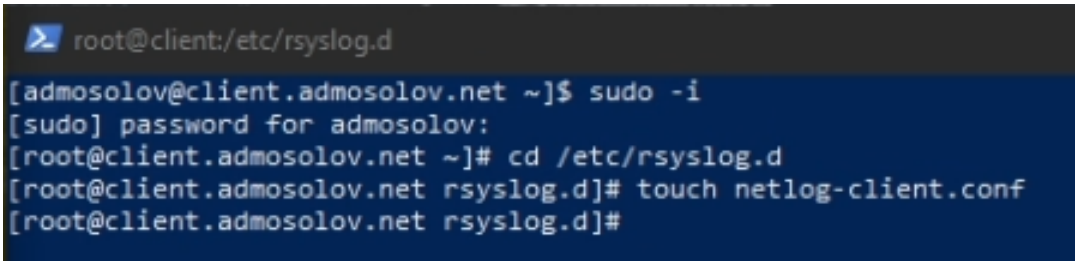


Рис. 3.6: Создание конфигурации на клиенте

В файле конфигурации клиента прописываю правило пересылки всех логов (`*.*`) на удаленный сервер. Использую формат `@@server.admosolov.net:514` (две «собачки» означают использование протокола TCP) (рис. 3.7).

Рис. 3.7: Настройка пересылки логов на сервер

Для применения настроек перезапускаю службу журналирования на клиенте командой `systemctl restart rsyslog` (рис. 3.8).

Рис. 3.8: Перезапуск rsyslog на клиенте

Возвращаюсь на сервер для проверки получения логов. Запускаю команду `tail -f /var/log/messages` для просмотра сообщений в реальном времени. В логе видны записи о перезапуске службы (рис. 3.9).

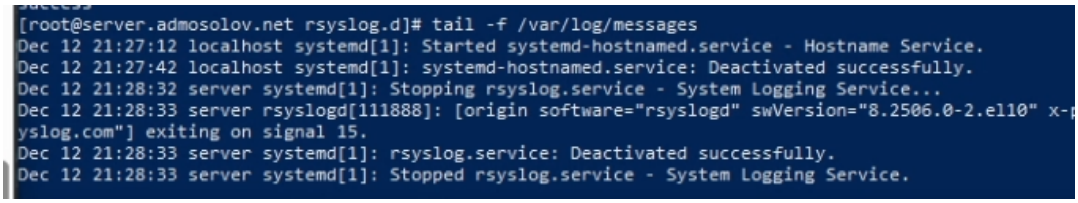


Рис. 3.9: Мониторинг логов на сервере

Чтобы убедиться в работоспособности сетевого журналирования, отправляю тестовое сообщение с клиента с помощью утилиты `logger`. Текст сообщения: “TEST 2: serv CHECK” (рис. 3.10).

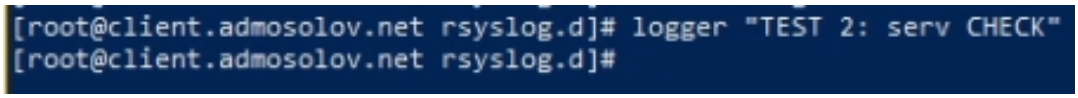


Рис. 3.10: Отправка тестового сообщения

Проверяю журнал на сервере. Вижу, что сообщение, отправленное с хоста `client`, успешно записалось в файл `/var/log/messages` на сервере. Это подтверждает корректность настройки (рис. 3.11).

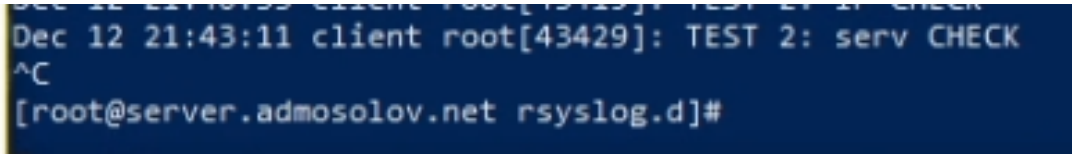
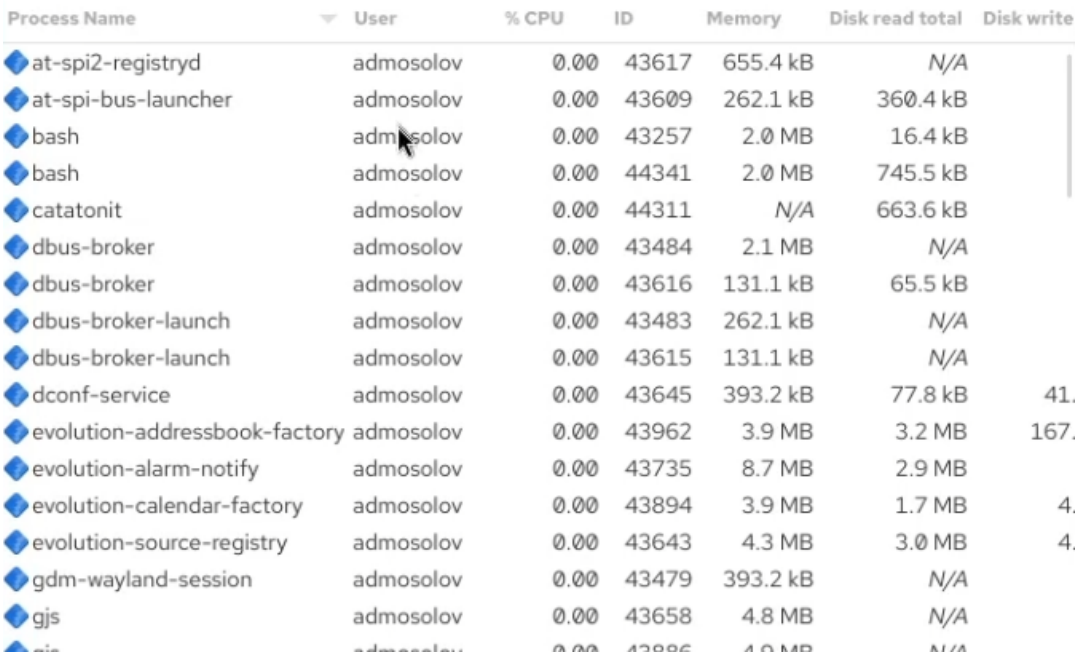


Рис. 3.11: Получение тестового сообщения на сервере

Перехожу к заданию по использованию графических утилит. На клиенте запускаю `gnome-system-monitor` (Системный монитор) для визуального наблюдения за процессами (рис. 3.12).

Рис. 3.12: Запуск gnome-system-monitor

В открывшемся окне системного монитора просматриваю список запущенных процессов, потребление ресурсов (CPU, память). Это позволяет отслеживать активность системы в графическом режиме (рис. 3.13).



Process Name	User	% CPU	ID	Memory	Disk read total	Disk write
at-spi2-registryd	admosolov	0.00	43617	655.4 kB	N/A	
at-spi-bus-launcher	admosolov	0.00	43609	262.1 kB	360.4 kB	
bash	admosolov	0.00	43257	2.0 MB	16.4 kB	
bash	admosolov	0.00	44341	2.0 MB	745.5 kB	
catatonit	admosolov	0.00	44311	N/A	663.6 kB	
dbus-broker	admosolov	0.00	43484	2.1 MB	N/A	
dbus-broker	admosolov	0.00	43616	131.1 kB	65.5 kB	
dbus-broker-launch	admosolov	0.00	43483	262.1 kB	N/A	
dbus-broker-launch	admosolov	0.00	43615	131.1 kB	N/A	
dconf-service	admosolov	0.00	43645	393.2 kB	77.8 kB	41.
evolution-addressbook-factory	admosolov	0.00	43962	3.9 MB	3.2 MB	167.
evolution-alarm-notify	admosolov	0.00	43735	8.7 MB	2.9 MB	
evolution-calendar-factory	admosolov	0.00	43894	3.9 MB	1.7 MB	4.
evolution-source-registry	admosolov	0.00	43643	4.3 MB	3.0 MB	4.
gdm-wayland-session	admosolov	0.00	43479	393.2 kB	N/A	
gjs	admosolov	0.00	43658	4.8 MB	N/A	

Рис. 3.13: Интерфейс системного монитора

Также просматриваю журналы событий на сервере с помощью утилиты `journalctl`. Использую флаг `-f` для отслеживания новых записей. В логе отображаются действия менеджера пакетов PackageKit (рис. 3.14).

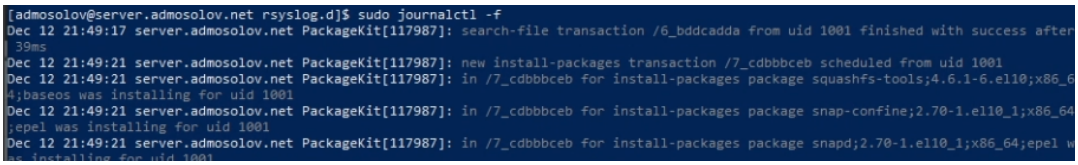


Рис. 3.14: Просмотр логов через journalctl

Приступаю к автоматизации процесса настройки через Vagrant. На сервере создаю структуру каталогов внутри `/vagrant/provision/server/` для хранения конфигурационных файлов и копирую туда текущий конфиг `netlog-server.conf` (рис. 3.15).

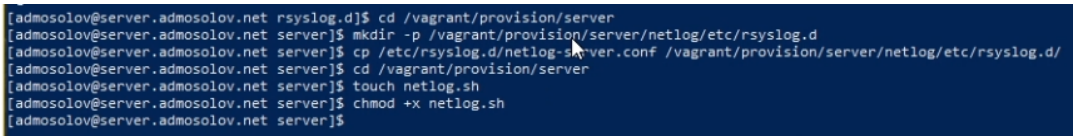


Рис. 3.15: Подготовка каталогов для провижининга сервера

Создаю скрипт `netlog.sh` для сервера. В нем прописываю команды копирования конфигурации в `/etc/`, восстановления контекста безопасности SELinux, настройки firewall и перезапуска службы rsyslog (рис. 3.16).

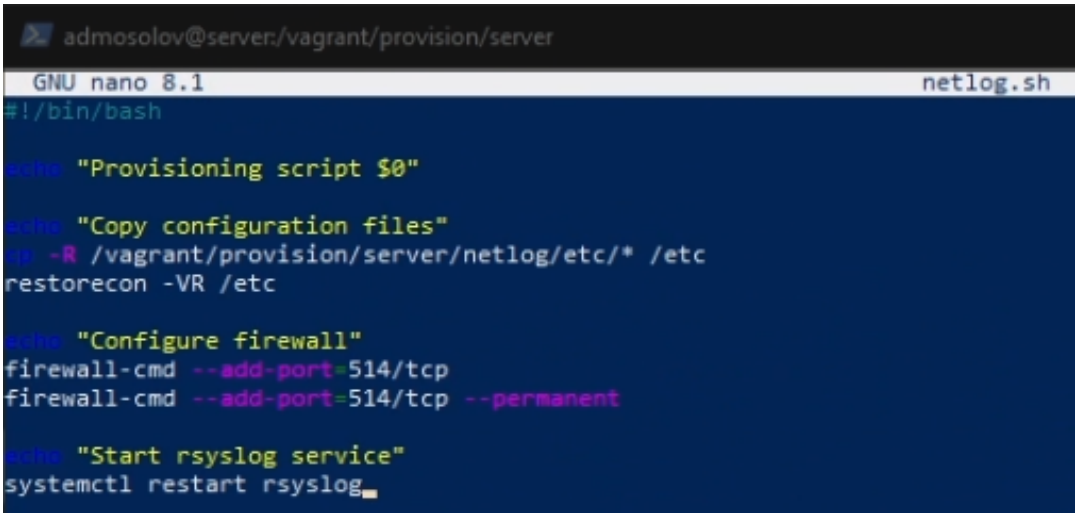


Рис. 3.16: Скрипт автоматизации для сервера

Аналогичные действия выполняю для клиента. Создаю структуру папок в `/vagrant/provision/client/` и копирую туда настроенный файл `netlog-client.conf` (рис. 3.17).

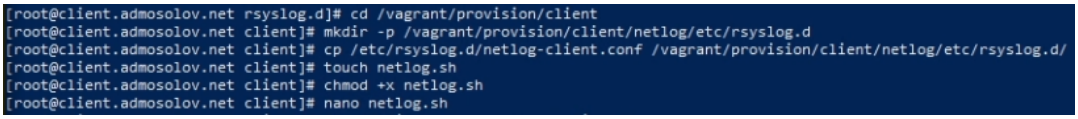


Рис. 3.17: Подготовка каталогов для провижининга клиента

Создаю скрипт `netlog.sh` для клиента. Он включает установку утилиты `lnav` (для удобного просмотра логов), копирование конфигурации, настройку SELinux и перезапуск службы (рис. 3.18).

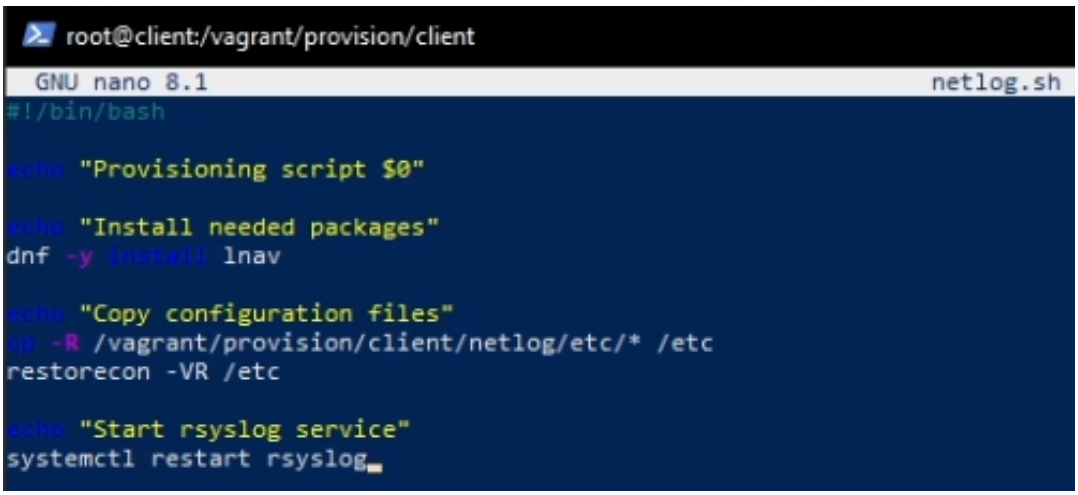


Рис. 3.18: Скрипт автоматизации для клиента

В завершение, вношу изменения в основной файл `Vagrantfile` для виртуальной машины `server`. Добавляю секцию provision типа `shell`, указывающую на созданный скрипт `provision/server/netlog.sh` (рис. 3.19).

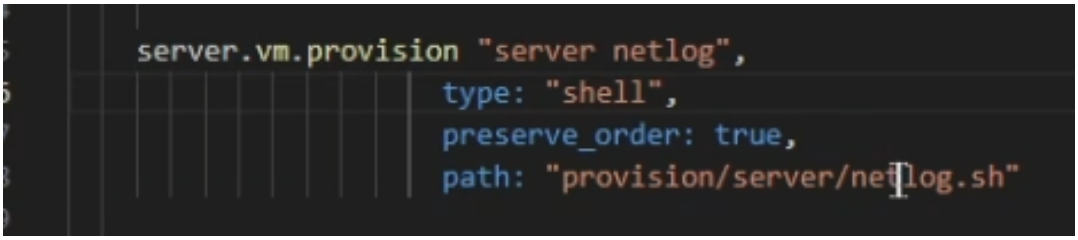


Рис. 3.19: Настройка Vagrantfile для сервера

Также добавляю секцию provision в `Vagrantfile` для виртуальной машины `client`, указывая путь к скрипту `provision/client/netlog.sh`. Теперь настройка журналирования будет выполняться автоматически при развертывании машин (рис. 3.20).

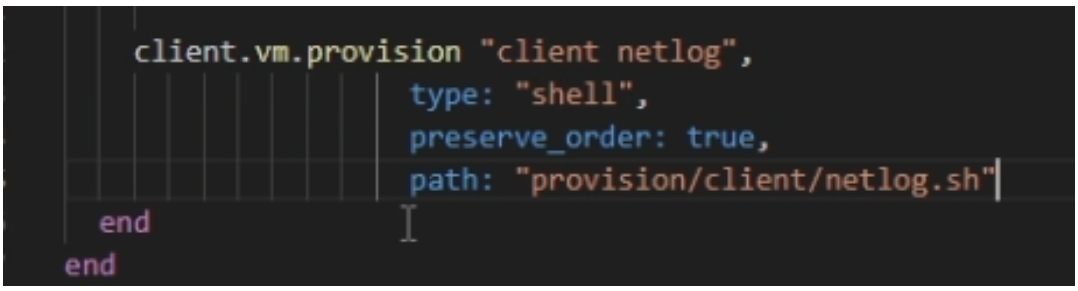


Рис. 3.20: Настройка Vagrantfile для клиента

4 Выводы

В ходе выполнения лабораторной работы я освоил принципы централизованного сбора логов в Linux. Я настроил сервер rsyslog для приема сообщений по протоколу TCP и клиент для их отправки. Также я научился анализировать логи с помощью `tail`, `journalctl` и графических утилит, а также написал скрипты автоматизации настройки окружения для Vagrant.

5 Ответы на контрольные вопросы

1. **Какой модуль rsyslog вы должны использовать для приёма сообщений от journald?** Для приема сообщений от системы journald в rsyslog используется модуль `imjournal`.

2. **Как называется устаревший модуль, который можно использовать для включения приёма сообщений журнала в rsyslog?** Устаревший модуль, который использовался для чтения системного лога через сокет `/dev/log`, называется `imuxsock`.

3. **Чтобы убедиться, что устаревший метод приёма сообщений из journald в rsyslog не используется, какой дополнительный параметр следует использовать?** Чтобы отключить локальное логирование через сокет (устаревший метод), используется параметр `$OmitLocalLogging on`.

4. **В каком конфигурационном файле содержатся настройки, которые позволяют вам настраивать работу журнала?** Основной файл конфигурации — `/etc/rsyslog.conf`. Дополнительные настройки часто выносятся в файлы в каталоге `/etc/rsyslog.d/`.

5. **Каким параметром управляется пересылка сообщений из journald в rsyslog?** Пересылка управляется параметром `ForwardToSyslog=yes` в файле конфигурации `/etc/systemd/journald.conf`.

6. **Какой модуль rsyslog вы можете использовать для включения сообщений из файла журнала, не созданного rsyslog?** Для отслеживания

текстовых файлов логов, созданных другими приложениями, используется модуль `imfile`.

7. **Какой модуль rsyslog вам нужно использовать для пересылки сообщений в базу данных MariaDB?** Для записи логов в базу данных MySQL/MariaDB используется модуль `ommysql`.

8. **Какие две строки вам нужно включить в rsyslog.conf, чтобы позволить текущему журнальному серверу получать сообщения через TCP?** Необходимо загрузить модуль и запустить прослушивание порта: `bash $ModLoad imtcp $InputTCPServerRun 514`

9. **Как настроить локальный брандмауэр, чтобы разрешить приём сообщений журнала через порт TCP 514?** Необходимо выполнить команды: `bash firewall-cmd --add-port=514/tcp firewall-cmd --add-port=514/tcp --permanent`