Отчёт по лабораторной работе №8

Планировщики событий

Эзиз Хатамов

Содержание

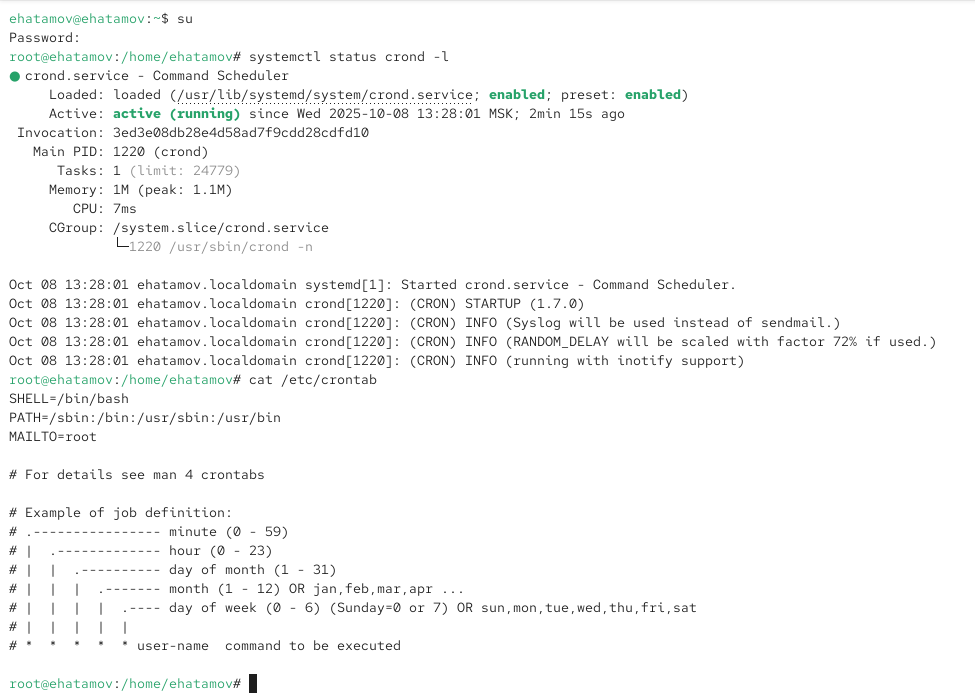
# 1 Цель работы

Получение навыков работы с планировщиками событий cron и at.

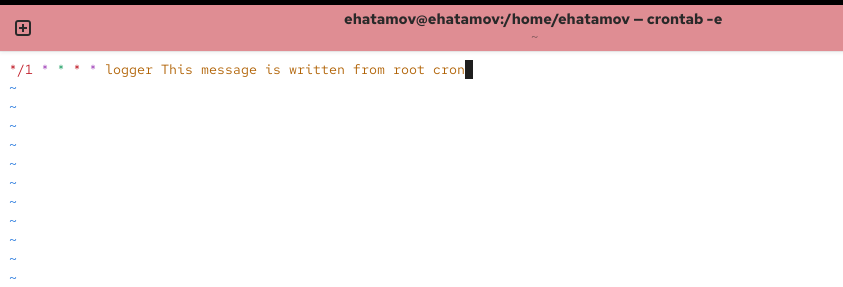
# 2 Отчёт по выполнению работы

## 2.1 Планирование заданий с помощью cron

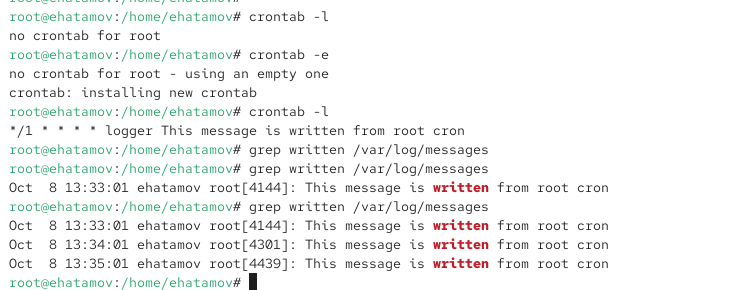
1. В терминале были получены права суперпользователя с помощью команды **su -**.  
   Затем был проверен статус службы планировщика заданий **crond**.  
   Команда **systemctl status crond -l** показала, что служба активна и работает корректно.
2. Далее было просмотрено содержимое конфигурационного файла **/etc/crontab**.  
   Файл содержит переменные среды, комментарии и пример формата задания.

* 
* Рис. 1: Содержимое файла /etc/crontab

1. Проверка текущего расписания пользователя **root** командой **crontab -l** показала отсутствие заданий.
2. Через команду **crontab -e** было добавлено задание, выполняющее запись в системный журнал каждую минуту:  
   ***/1*  \* \* \* logger This message is written from root cron**\*.  
   Синтаксис выражения означает:
   * \*/1 — выполнение каждые 1 минуту;
   * \* — каждый час, день, месяц и день недели;
   * команда logger записывает сообщение в системный журнал.

* 
* Рис. 2: Создание задания cron с интервалом в 1 минуту

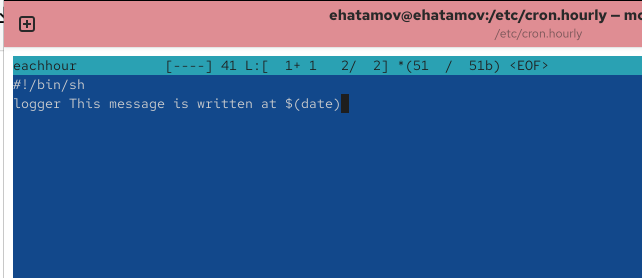
1. После сохранения изменений команда **crontab -l** подтвердила наличие записи в расписании.  
   Через несколько минут проверка системного журнала показала выполнение задачи:  
   **grep written /var/log/messages**.  
   Сообщения с текстом *“This message is written from root cron”* появлялись с интервалом в одну минуту.

* 
* Рис. 3: Проверка выполнения cron-заданий в системном журнале

1. Запись в расписании была изменена на:  
   **0 */1*  \* 1-5 logger This message is written from root cron**.  
   Здесь указано:
   * 0 \*/1 — запуск каждый час в 0-й минуте;
   * 1-5 — выполнение только по рабочим дням (с понедельника по пятницу).

* 
* Рис. 4: Редактирование расписания cron на выполнение по рабочим дням

1. В каталоге **/etc/cron.hourly** был создан файл сценария **eachhour**.  
   В нём был размещён скрипт, выполняющий запись в системный журнал:  
   **logger This message is written at $(date)**.  
   Скрипт делает запись в системный журнал с указанием текущей даты и времени.

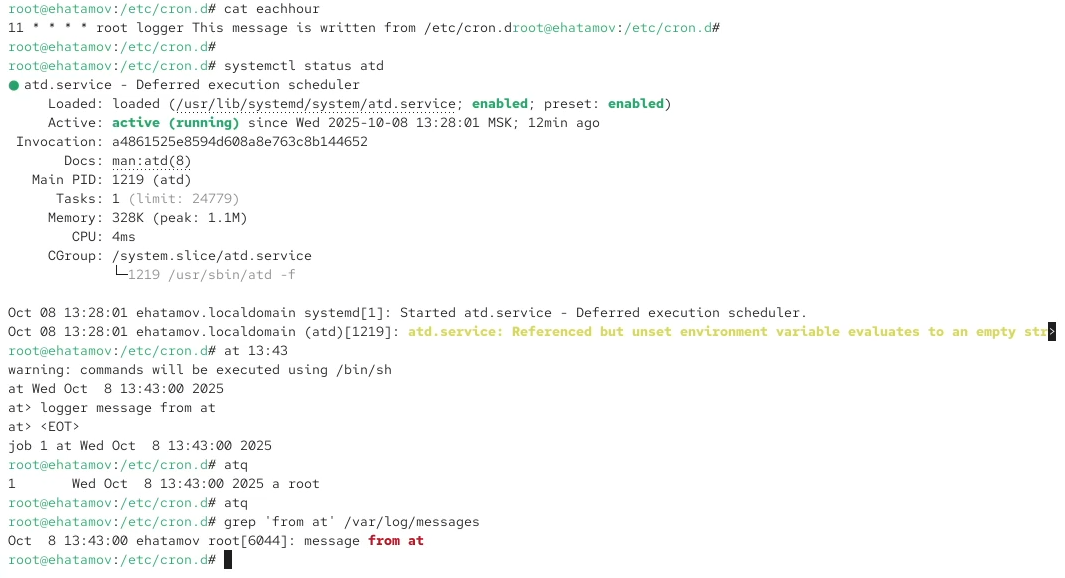
* 
* Рис. 5: Создание сценария eachhour в /etc/cron.hourly

1. Для задания, запускаемого из каталога **/etc/cron.d**, был создан файл **eachhour**, содержащий строку:  
   **11 \* \* \* \* root logger This message is written from /etc/cron.d**.  
   Синтаксис означает:
   * выполнение в **11-й минуте** каждого часа;
   * от имени пользователя **root**;
   * запуск команды **logger**.

* 
* Рис. 6: Создание задания в /etc/cron.d

## 2.2 Планирование заданий с помощью at

1. Проверка службы **atd** показала, что она активна и запущена.
2. Было создано однократное задание с помощью команды **at 13:43**.  
   В интерактивной оболочке введена команда **logger message from at**.  
   После закрытия ввода сочетанием **Ctrl + D** задание было добавлено в очередь.
3. Проверка командой **atq** подтвердила наличие задания, запланированного на 13:43.
4. После указанного времени команда **grep ‘from at’ /var/log/messages** показала запись *“message from at”*, подтверждающую успешное выполнение задания.

* 
* Рис. 7: Проверка выполнения задания at в системном журнале

# 3 Контрольные вопросы

1. **Как настроить задание cron, чтобы оно выполнялось раз в 2 недели?**
   * В cron отсутствует прямой параметр для задания интервала в 2 недели.
   * Можно использовать выполнение по дню недели и неделе месяца:  
     0 0 \*/14 \* \* <команда> — запуск каждые 14 дней в полночь.
   * Альтернатива — создать скрипт, который сам проверяет дату и выполняет действия раз в две недели.
2. **Как настроить задание cron, чтобы оно выполнялось 1-го и 15-го числа каждого месяца в 2 часа ночи?**
   * 0 2 1,15 \* \* <команда>
   * Здесь 0 — минуты, 2 — час (2:00), 1,15 — дни месяца, \* \* — каждый месяц и любой день недели.
3. **Как настроить задание cron, чтобы оно выполнялось каждые 2 минуты каждый день?**
   * \*/2 \* \* \* \* <команда>
   * Символ \*/2 в поле минут задаёт выполнение каждые 2 минуты.
4. **Как настроить задание cron, чтобы оно выполнялось 19 сентября ежегодно?**
   * 0 0 19 9 \* <команда>
   * Выполнение 19-го числа 9-го месяца (сентября) в полночь каждого года.
5. **Как настроить задание cron, чтобы оно выполнялось каждый четверг сентября ежегодно?**
   * 0 0 \* 9 4 <команда>
   * 0 0 — запуск в полночь, \* — каждый день месяца, 9 — сентябрь, 4 — четверг (номер дня недели).
6. **Какая команда позволяет вам назначить задание cron для пользователя alice? Приведите подтверждающий пример.**
   * crontab -u alice -e — открыть файл расписания пользователя **alice** для редактирования.
   * Пример: crontab -u alice -e и добавление строки 0 8 \* \* \* echo "Good morning" >> /home/alice/log.txt.
7. **Как указать, что пользователю bob никогда не разрешено назначать задания через cron? Приведите подтверждающий пример.**
   * Для этого используется файл **/etc/cron.deny**.
   * В него добавляется имя пользователя:  
     echo "bob" >> /etc/cron.deny
   * После этого пользователь **bob** не сможет использовать команду crontab.
8. **Вам нужно убедиться, что задание выполняется каждый день, даже если сервер во время выполнения временно недоступен. Как это сделать?**
   * Следует использовать **anacron**, который выполняет пропущенные задания после восстановления системы.
   * Конфигурация находится в файле **/etc/anacrontab**, где можно задать ежедневное выполнение.
9. **Какая команда позволяет узнать, запланированы ли какие-либо задания на выполнение планировщиком atd?**
   * atq — отображает очередь заданий, запланированных с помощью **at**.
   * В выводе будут показаны идентификаторы заданий, время запуска и пользователь, создавший их.

# 4 Заключение

В ходе работы были изучены механизмы планирования заданий в Linux с использованием утилит **cron** и **at**.  
В результате освоены базовые приёмы автоматизации системных процессов и администрирования в среде Linux.