Отчет по лабораторной работе 6

Даниил Генералов, 1032212280

Содержание

# 1 Цель работы

Развить навыки администрирования ОС Linux. Получить первое прак- тическое знакомство с технологией SELinux. Проверить работу SELinx на практике совместно с веб-сервером Apache.

# 2 Выполнение лабораторной работы

Сначала нужно установить в нашу лабораторную виртуальную машину Apache и отключить пакетный фильтр. После этого мы проверяем, что SELinux работает в режиме enforcing targeted, что Apache запущен (и исправляем это), а также что Apache запущен под своим собственным контекстом httpd\_t, и что есть много SELinux-boolean про httpd fig. 1.

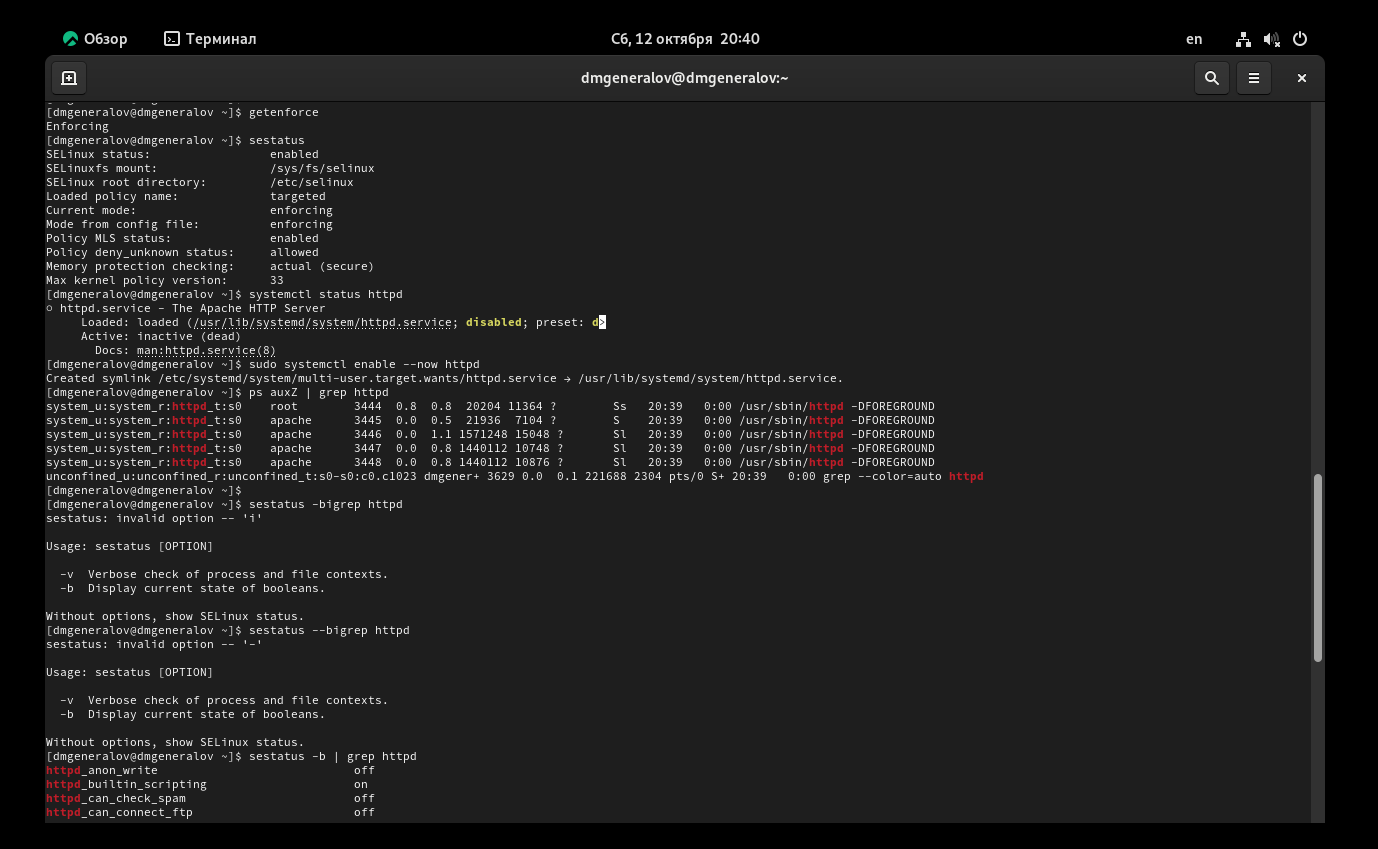


Рис. 1: SELinux

После этого мы видим, что есть 135 классов и 5145 типов в SELinux. Затем мы видим, что в папке /var/www/html и дочерних папках контекст system\_u:object\_r:httpd\_sys\_content\_t, а в /var/www/cgi-bin system\_u:object\_r:httpd\_sys\_script\_exec\_t. Если суперпользователь создаст файл в /var/www/html, то ему будет выдан контекст unconfined\_u:object\_r:httpd\_sys\_content\_t: теперь httpd имеет к нему доступ (рис. fig. 2).

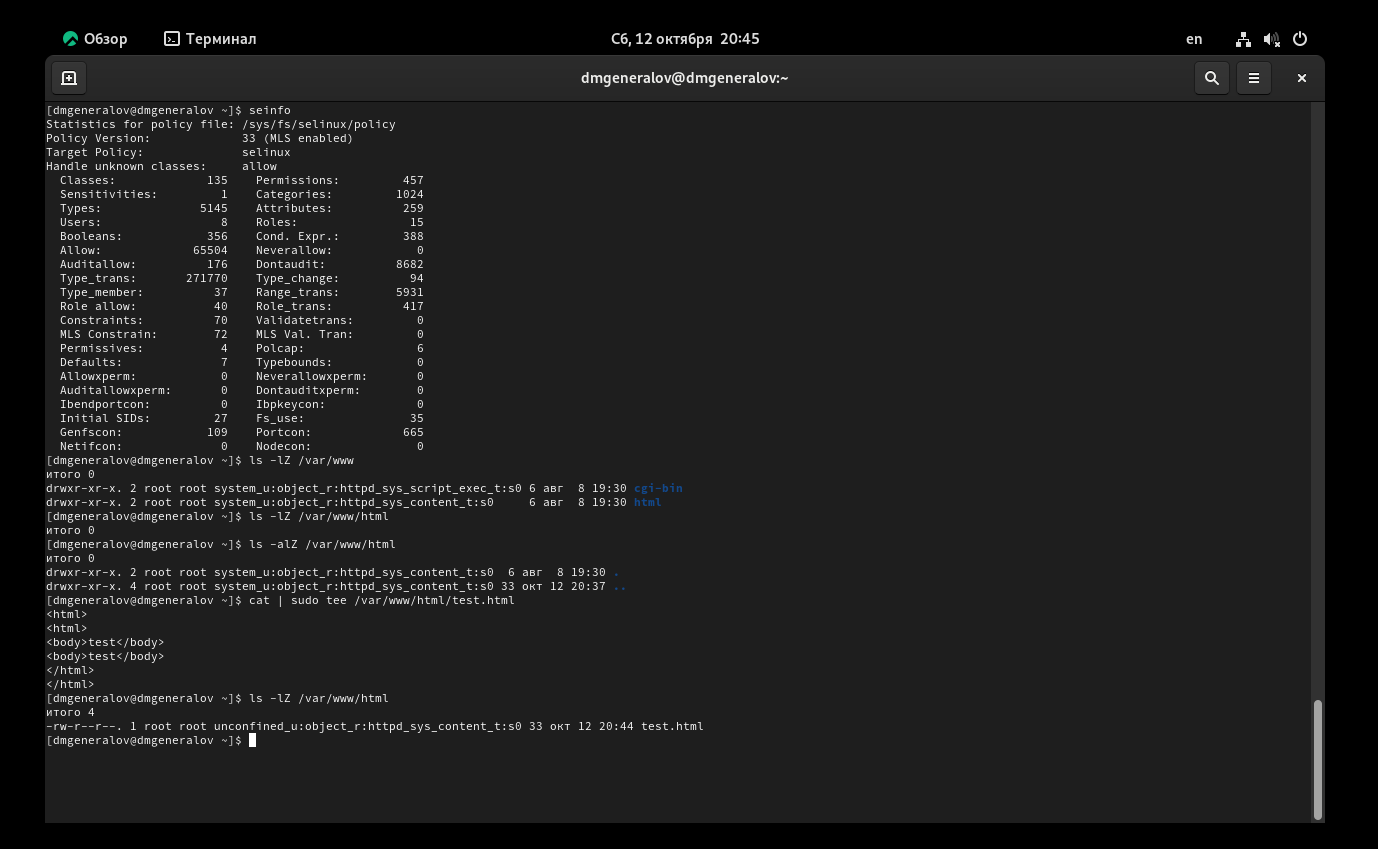


Рис. 2: SELinux

Теперь можно проверить, что сервер имеет к нему доступ, попробовав скачать этот файл с него. После этого мы меняем контекст этого файла, и теперь сервер не имеет к нему доступаю SeTroubleshoot подсказывает, что ошибка доступа произошла именно из-за SELinux, и предлагает, как ее можно исправить (рис. fig. 3).

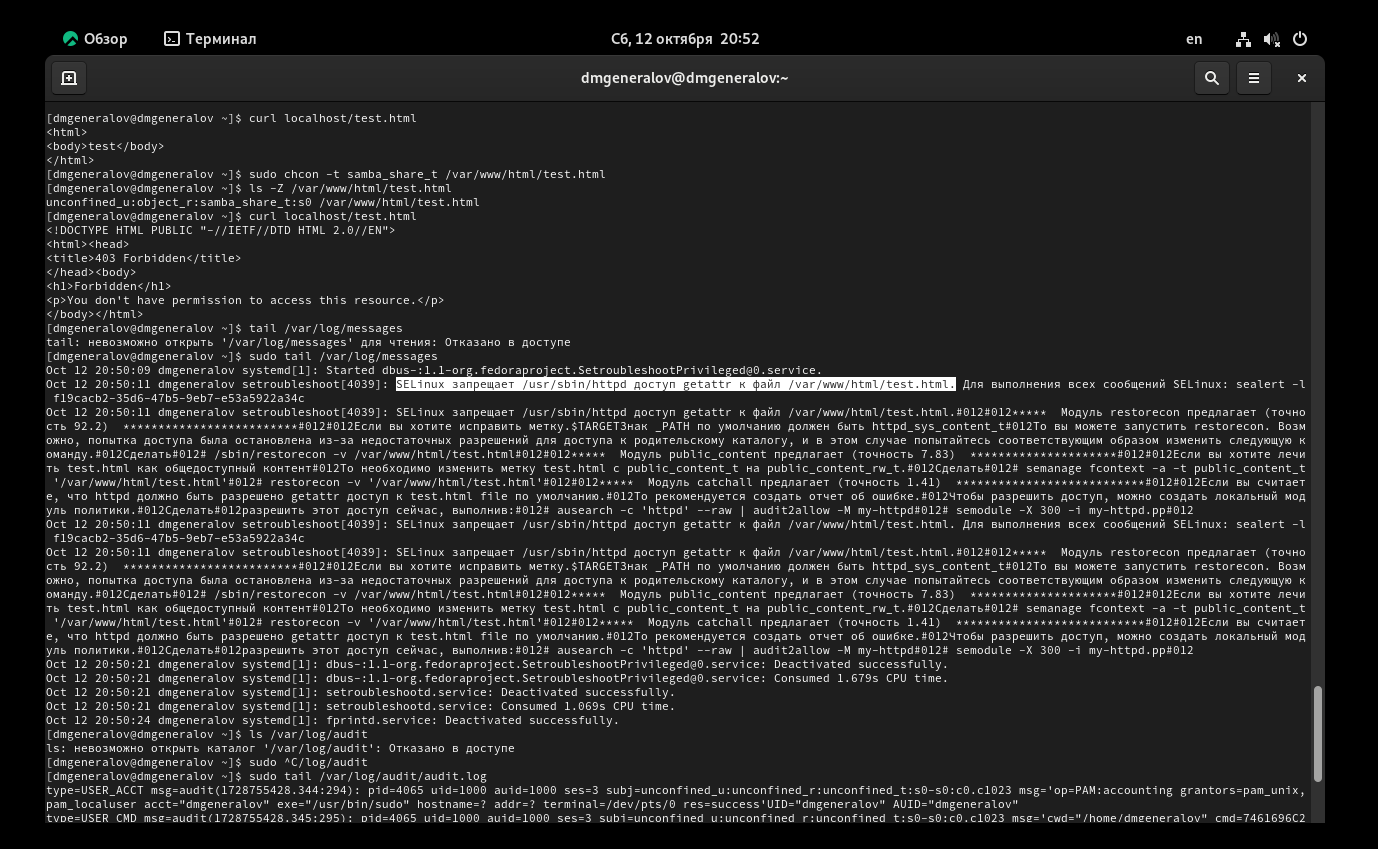


Рис. 3: SELinux

Если мы настроим, чтобы Apache пытался слушать порт 81, а не порт 80, то он смог запуститься, потому что оказывается в настройках по умолчанию порт 81 уже разрешен для типа http\_port\_t. Чтобы продемонстрировать ситуацию, когда это не разрешено, я вместо этого выбрал порт 31337: теперь запуск httpd блокируется. Чтобы это было разрешено, нужно добавить этот порт в список разрешенных для типа http\_port\_t: после этого запуск будет успешен, и, вернув исходный контекст файлу, можно будет скачать его (рис. fig. 4).

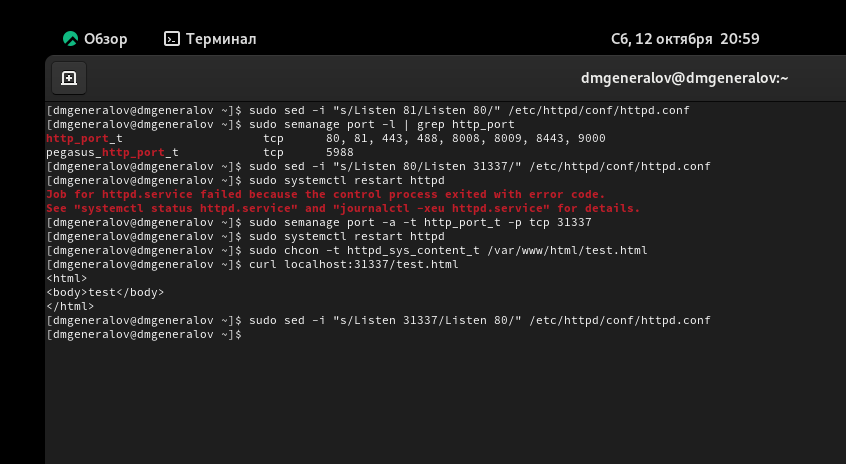


Рис. 4: SELinux

# 3 Выводы

В этой лабораторной работе мы рассмотрели, как использовать SELinux для ограничения возможностей процессов, на примере Apache.