МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
 «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)**

Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

**ОТЧЁТ**

**по лабораторной работе №11**

**«**Настройка механизмов организации замкнутой программной среды.

Контроль целостности комплекса средств защиты.**»**

Выполнил: студент гр. ВКБ43

Ковалев Д. П.

Проверил:

доц. Скляров А.В.

**Цель:** изучить принципы и технологии контроля целостности данных (в том числе комплекса средств защиты – КСЗ), реализованных в ОССН. Освоить умения, необходимые для решения задач подсчёта носителей контрольных сумм файлов и оптических носителей, контроля соответствия дистрибутиву, регламентного контроля целостности и создания замкнутой программной среды.

**Задание 1.** Начать работу со входа в ОССН в графическом режиме с учётной привилегированного пользователя пользователя, например: user (уровень доступа — 0, неиерархические категории — нет, уровень целостности — «Высокий») и запустить терминал Fly в «привилегированном» режиме командой sudo fly-term.

Войдем от имени пользователя user, выставив уровни, как сказано по условию задания. Конфигурация представлена на рисунке 1.

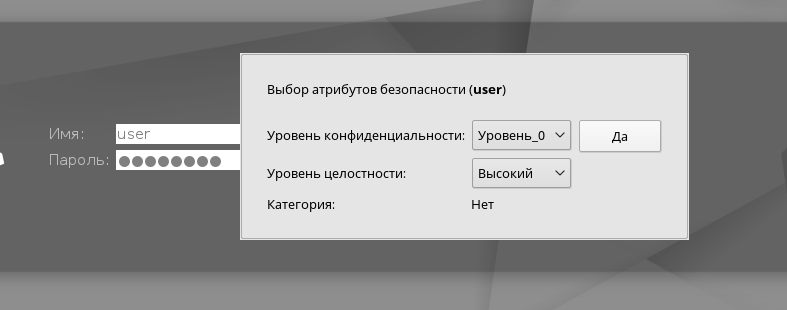


Рисунок 1 – атрибуты безопасности для user

Запустим терминал Fly в «привилегированном» режиме командой sudo flyterm. Результат представлен на рисунке 2.

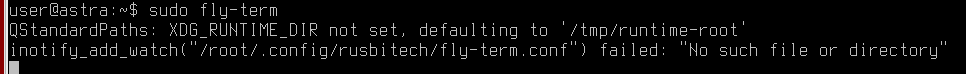


Рисунок 2 – запуск терминала

**Задание 2.** В домашнем каталоге создать подкаталог checksum и скопировать в него все файлы (включая вложенные каталоги) из каталога /etc.

Для начала создадим директорию, для этого будем использовать команду: mkdir /home/user/checksum. Теперь скопируем все содержимое из etc в нашу новую директорию. Для этого будем использовать команду: cp -r /etc/\* home/user/checksum/. Выполнение задания представлено на рисунке 3.

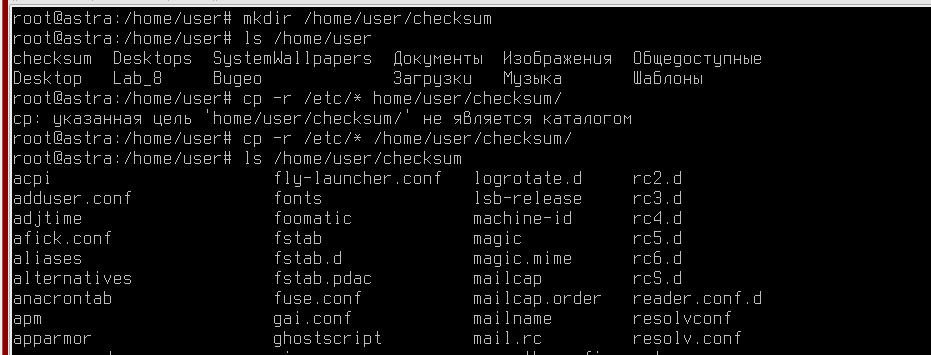


Рисунок 3 – результат выполнения задания

**Задание 3.** Используя алгоритм MD5, вычислить контрольные суммы всех файлов в каталоге /home/user/checksum и перенаправить результат их вычисления в файл /home/user/md5.check, а поток с перечнем ошибок в файл /home/user/error.md5, командой md5sum /home/user/checksum/\* > /home/user/md5check 2> /home/user/error.md5.

Для выполнения задания просто нужно использовать команду, которая предложена по условию задания. Результат представлен на рисунке 4.



Рисунок 4 – выполнение задания

**Задание 4.** Вывести в терминал содержимое файлов /home/user/md5check и /home/user/error.md5 цепочкой команд cat /home/user/md5check; cat /home/user/error.md5 и указать, для каких объектов в каталоге /home/user/checksum контрольные суммы не были созданы.

Для начала воспользуемся командой cat /home/user/md5check, в результате получилось то, что представлено на рисунке 5.

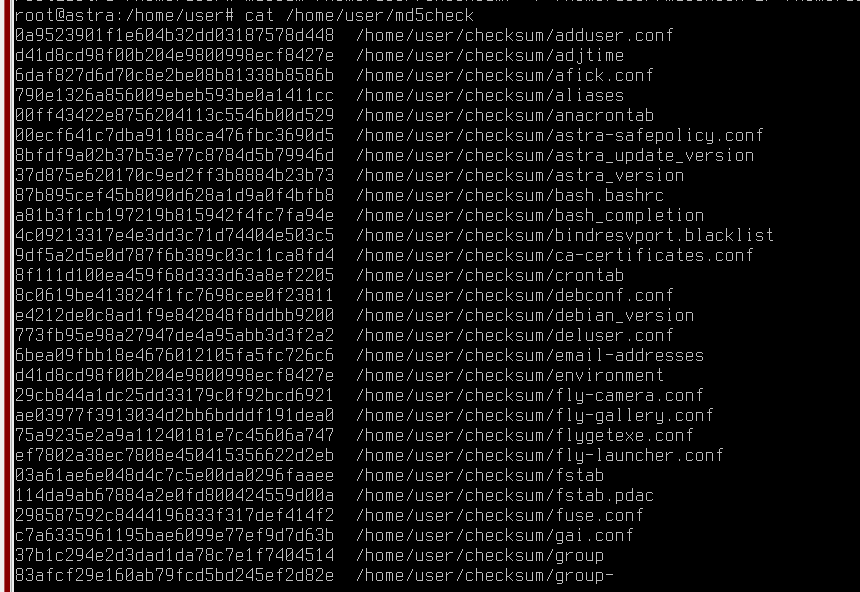


Рисунок 5 – вывод команды cat /home/user/md5check

Теперь воспользуемся командой cat /home/user/error.md5. Результат выполнения команды представлен на рисунке 6. Контрольные суммы не были созданы для каталогов.

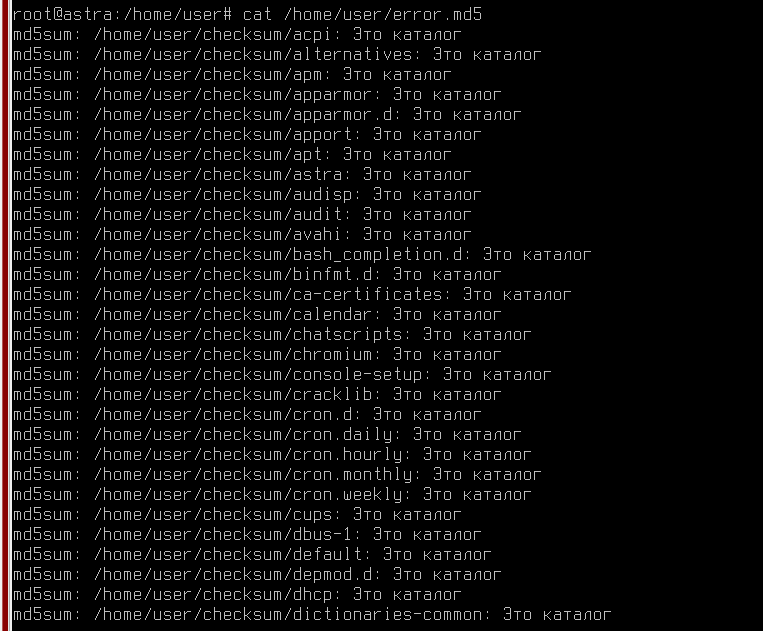


Рисунок 6 – контрольные суммы для каталогов

**Задание 5.** Используя алгоритм SHA-512/256, вычислить контрольные суммы всех файлов в каталоге /home/user/checksum и перенаправить результат вычислений в файл /home/user/sha512256check командой shasum -a 512256 /home/user/checksum/\* > /home/user/sha512256check. Вывести на экран содержимое файла /home/user/sha512256check командой less /home/user/sha512256check.

На рисунке 7 представлен команда для вычисления контрольных сумм и перенаправление в файл.



Рисунок 7 – вычисление контрольных сумм всех файлов

Теперь выведем на экран содержимое файла /home/user/sha512256check

командой less /home/user/sha512256check. Результат представлен на рисунке 8.

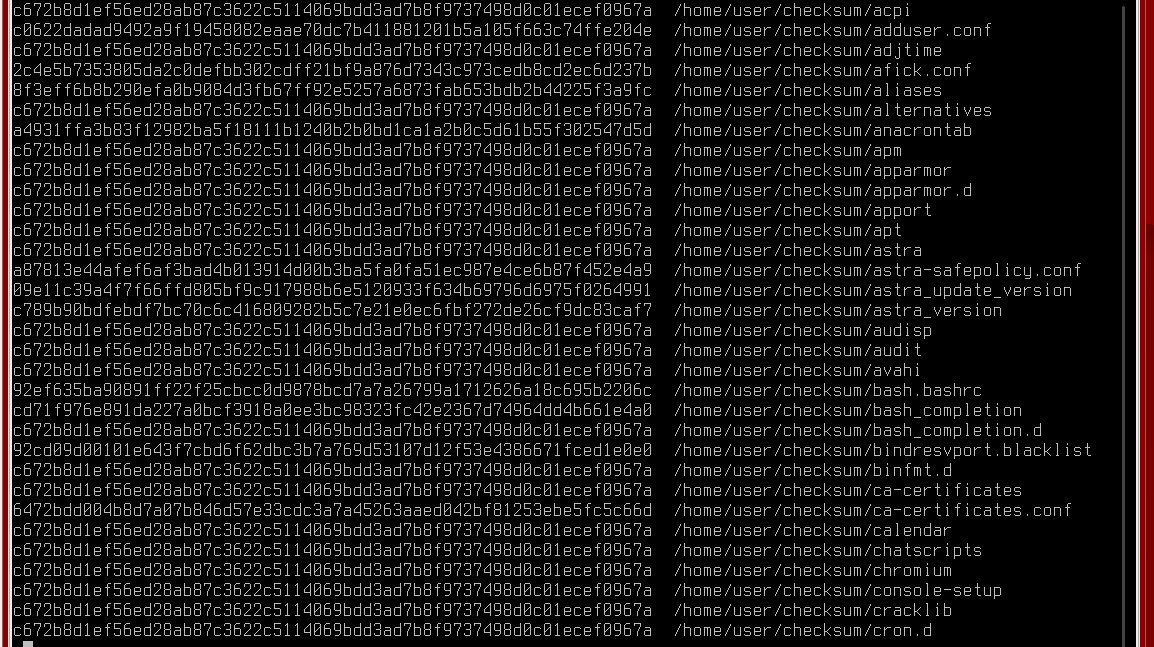


Рисунок 8 – вывод содержимого экрана

**Задание 6.** Используя редактор *vim*, изменить содержимое файла /home/user/checksum/passwd, удалив из него учётную запись суперпользователя (строку root: x: 0:0:root: /root: /bin/bash).

Откроем через vim данный файл. В командой стройке надо просто ввести vim и путь к файлу. В итоге должно получиться так, как представлено на рисунке 9. Чтобы выйти и записать из vim нажимаем “esc”, а потом пишем “:wq”.

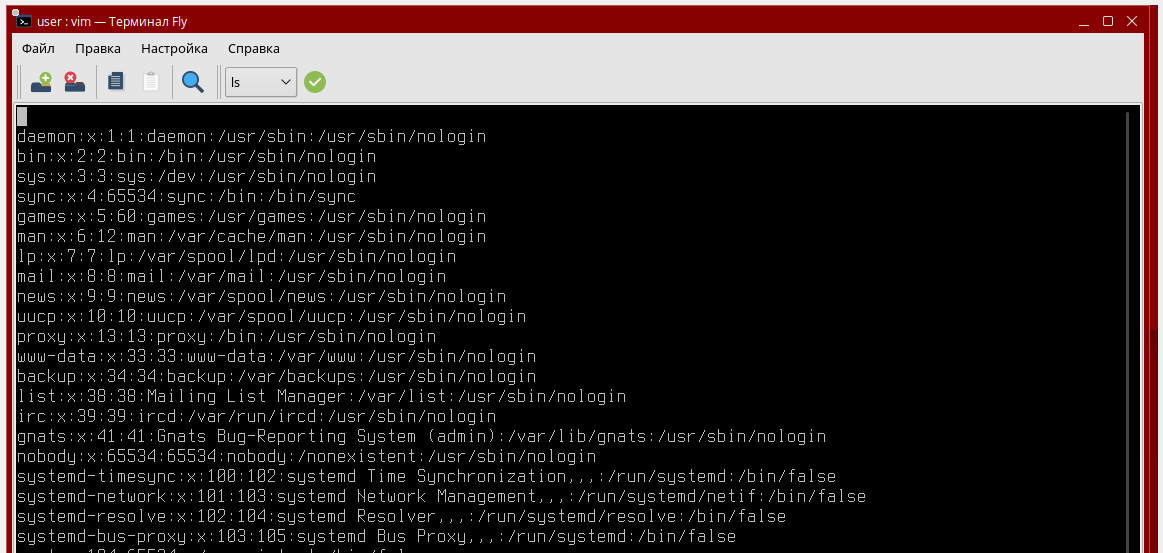


Рисунок 9 – изменение содержимого файла

**Задание 7.** Используя алгоритм MD5, проверить контрольные суммы всех файлов в каталоге /home/user/checksum и перенаправить результат проверки в файл /home/user/fullcheck командой md5sum -с /home/user/md5check > /home/user/fullcheck.

Теперь в терминале выполним команду, которая предложено по условию задания. На рисунке 10 видно результат.



Рисунок 10 – вычисление контрольных сумм

**Задание 8.** Используя алгоритм SHA512/256, проверить контрольные суммы всех файлов в каталоге /home/user/checksum и перенаправить результат проверки (с добавлением) в файл /home/user/full-check командой shasum -a 512256 -c ./home/user/sha512256check >> /home/user/fullcheck.

Выполняем команды, как сказано по условию задания. В результате должно получиться то, что представлено на рисунке 11.

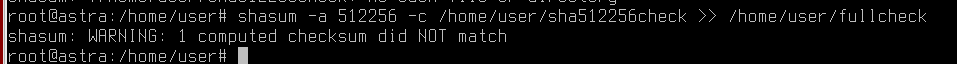


Рисунок 11 – проверка контрольных сумм

**Задание 9.** Найти в файле /home/user/fullcheck строки, указывающие на

файлы с нарушением целостности (содержащие слова ПОВРЕЖДЁН и FAILED), вывести в терминал их содержимое и число цепочкой команд grep 'ПОВРЕЖДЁН' /home/user/fullcheck > /home/user/tmpcheck grep ‘FAILED’ /home/user/fullcheck >> /home/user/tmpcheck wc –l /home/user/tmpcheck; less /home/user/tmpcheck.

Выполняем просто последовательность команд, которые указаны по условию задания. Результаты предложены на рисунках 12 и 13 соответственно.

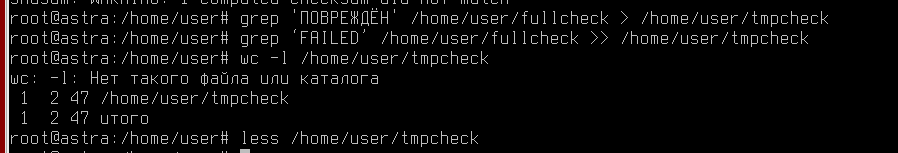


Рисунок 12 – поиск повреждений

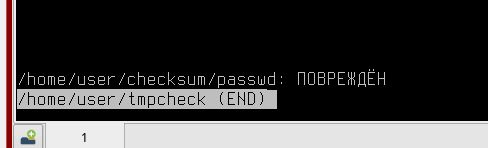
****

Рисунок 13 – поиск повреждений

**Задание 10.** Используя алгоритм ГОСТ P 34. 11-2012 (256 битов), вычислить контрольную сумму файла /home/user/checksum/shadow, перенаправить результат проверки в файл /home/user/gostcheck и вывести в терминал содержимое файла /home/user/gostcheck цепочкой команд gostsum /home/user/checksum/shadow -o /home/user/gostcheck less /home/user/gostcheck.

Выполняем все команды, как предложено по условию задания. В результате должно получиться то, что представлено на рисунках 14 и 15 соответственно.

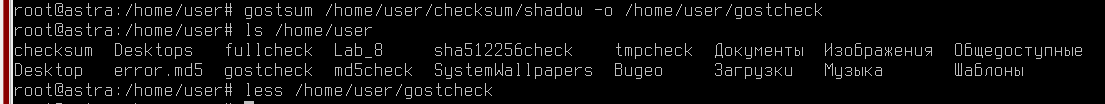


Рисунок 14 – результат выполнения задания

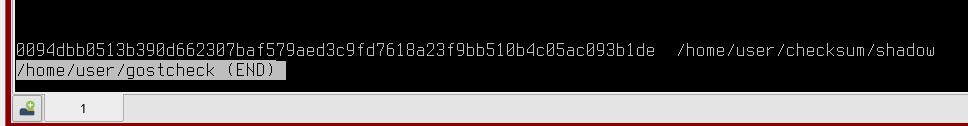


Рисунок 15 – результат выполнения задания

**Задание 11.** Установить оптический диск с дистрибутивом ОССН и, используя алгоритм ГОСТ Р 34. 11-2012 (256 битов), вычислить его контрольную сумму (по умолчанию файл устройства оптического диска /dev/sr0) и перенаправить результат вычисления в файл /home/user/isocheck командой

gostsum –d /dev/sr0 > /home/user/isocheck (выполнение команды занимает длительное время).

В современных версиях Oracle Virtual Box нет поддержки оптических дисков. По крайней мере, автор лабораторной работы не нашел мануалов по этому поводу, которые бы работали на последних версиях, поэтому невозможно выполнить данное задание. Результат представлен на рисунке 16.

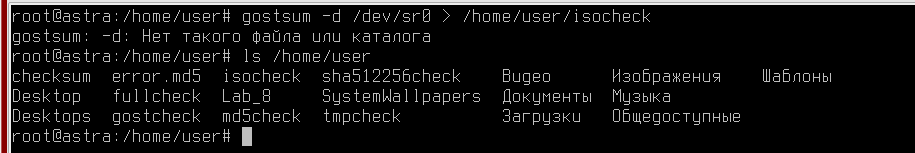


Рисунок 16 – результат задания 11

**Задание 12.** Запустить графическую утилиту fly-admin-int-check и во вкладке «Параметры проверки целостности». Выбрать точку монтирования устройства «Astra Smolensk amd64» (по умолчанию это, чаще всего, каталоги /media/cdrom или /media/cdrom0) и выполнить монтирование. Настроить фильтр проверки целостности в разделе «Принудительно», добавив регулярное выражение, содержащее абсолютный путь ко всем файлам каталога /usr/lib: /usr/lib/\*. Настроить фильтр проверки целостности в разделе «Игнорировать», удалив регулярное выражение, содержащее абсолютный путь к каталогу /tmp. в разделе « Отчёты» задать только текстовый формат файла отчёта, определив путь размещения файла report.txt в каталоге /home/user/report. Изменить содержимое файла /usr/share/doc/libcap2/copyright командой vim /usr/share/doc/libcap2/copyright, удалив в нем две первые строки. Начать проверку и зафиксировать предполагаемое время проверки, перейти во вкладку «Состояние» и проконтролировать статус проверки, после окончания проверки завершить работу графической утилиты. В файле /home/user/report/report.txt найти строки, содержащие текст: «Файлы, целостность которых нарушена», «Контр. сумма» и «/usr/share/doc/libcap2/copyright», сохранить результаты поиска в файл /home/user/report-2 цепочкой команд: grep 'Файлы, целостность которых нарушена' /home/user/report/report.txt -A 4 > /home/user/report-2; grep 'Контр. сумма' /home/user/report/report.txt -A 4 >> /home/user/report-2; grep '/usr/share/doc/libcap2/copyright' /home/user/report/report.txt >> /home/user/report-2.

Выберем точку монтирования, как сказано по условию задания. Выбор точки представлен на рисунке 17.

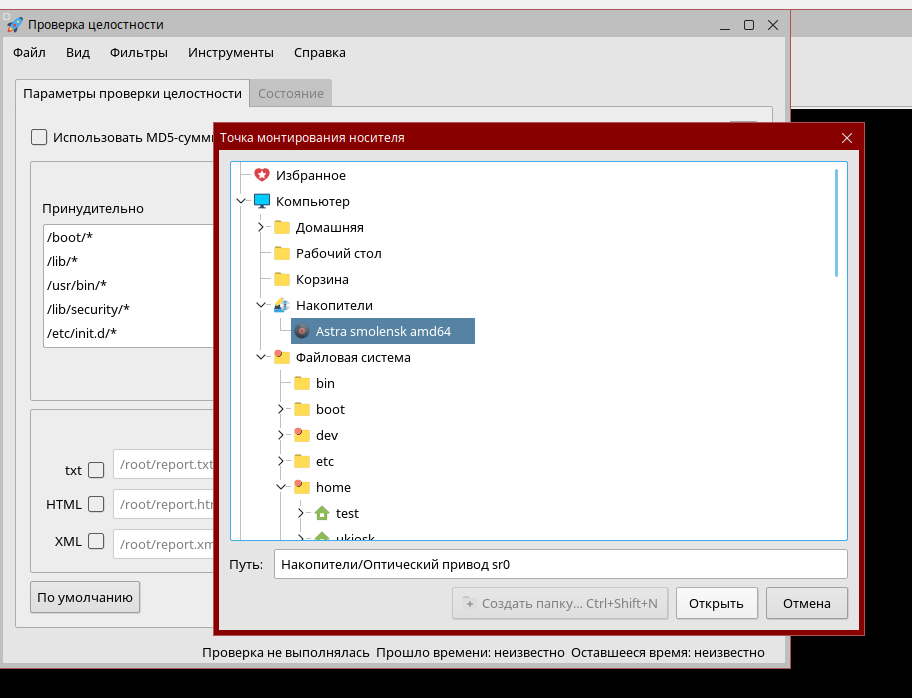


Рисунок 17 – выбор точки монтирования

В нашем случае было был монтирован на /dev/cdrom0, как представлено на рисунке 18.

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 18 – монтирование накопителя

Настроим теперь фильтр проверки целостности в разделе “Принудительно”, добавив регулярное выражение, содержащее абсолютный путь ко всем файлам каталога. /usr/lib: /usr/lib/\*. Настройка фильтров целостности была тоже проведена, как представлено на рисунке 19.

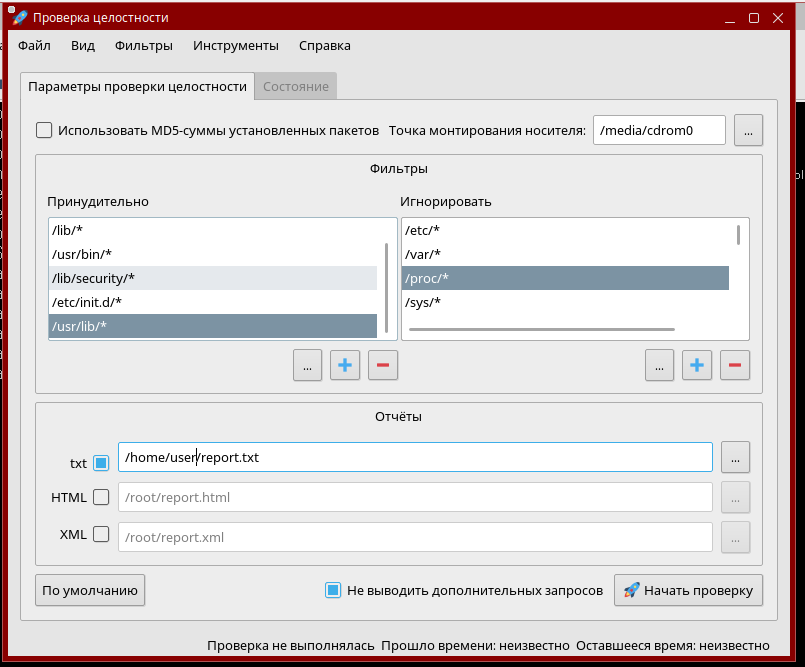


Рисунок 19 – настройка фильтров проверки целостности

Изменим теперь содержимое файла /usr/share/doc/libcap2/copyright

командой vim /usr/share/doc/libcap2/copyright, удалив в нем две

первые строки, как представлено на рисунке 20.

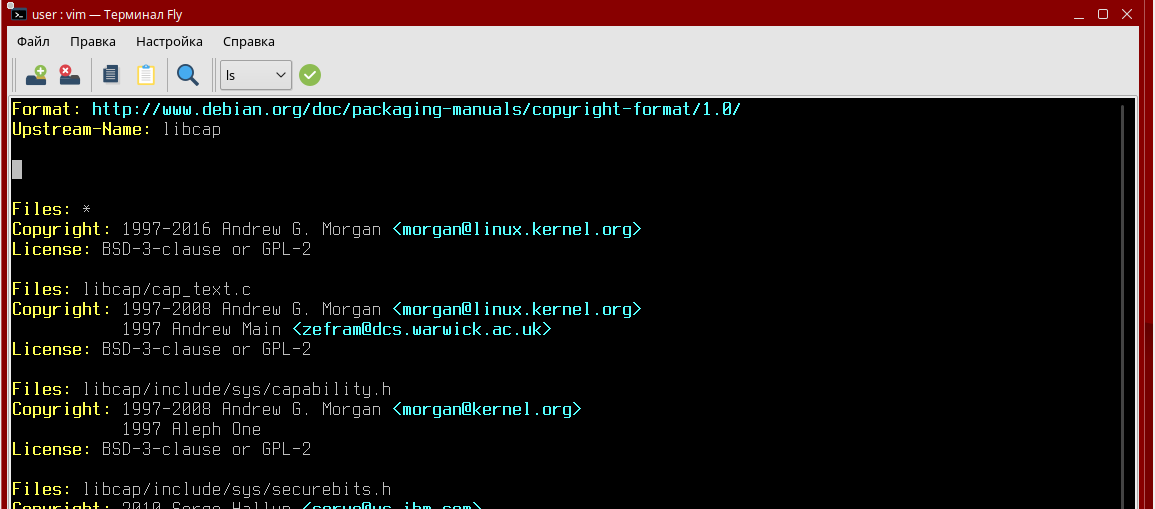


Рисунок 20 – удаление первых двух строк

Начнем проверку целостности. В результате получилось то, что представлено на рисунке 21.

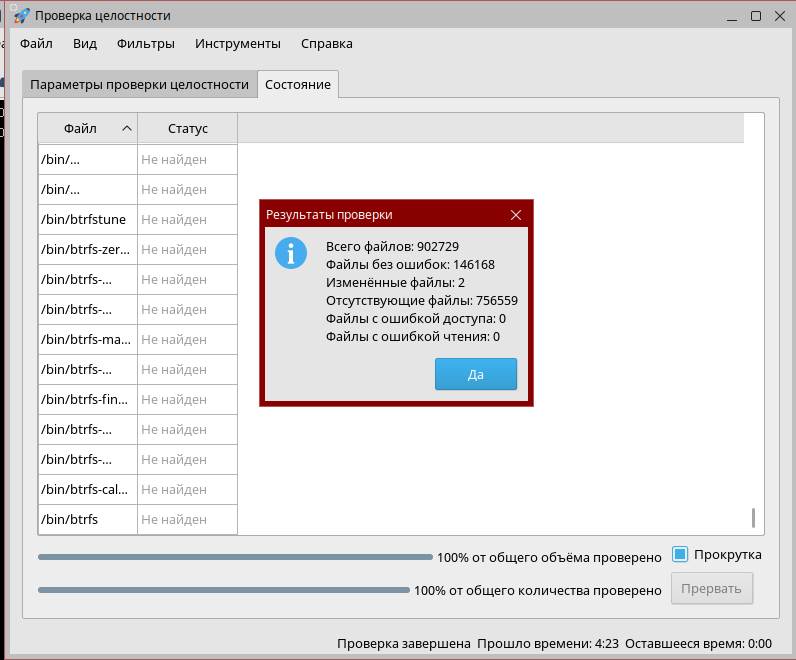


Рисунок 21 – проверка целостности всех файлов

В файле /home/user/report/report.txt нужно найти строки, содержащие текст: «Файлы, целостность которых нарушена», «Контр. сумма» и «/usr/share/doc/libcap2/copyright». Команды для выполнения представлены на рисунке 22.

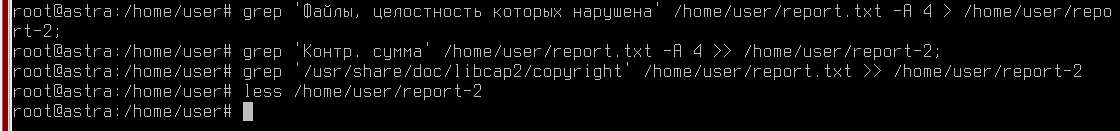


Рисунок 22 – заполнение нарушений целостности

Теперь на рисунке 23 представлено содержимое данного файла. Видно, что целостность была нарушена.

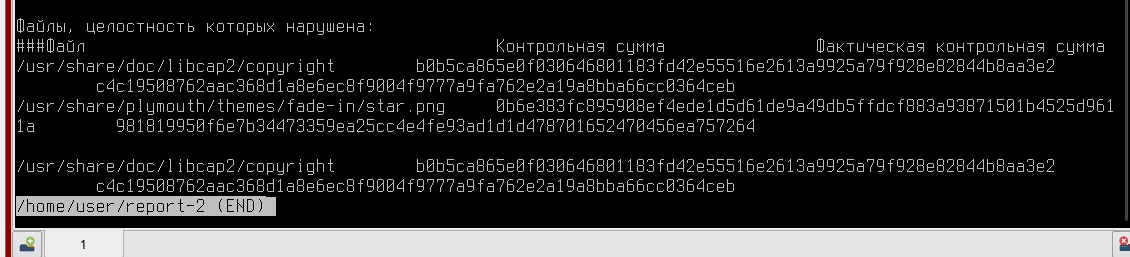


Рисунок 23 – просмотр файла с нарушением целостности

**Задание 13.** Отредактировать секцию directives конфигурационного файла /etc/afick.conf системы AFICK, отменив проверку выполняющихся приложений: исходный вариант секции: directives: running\_files: = yes, отредактированный вариант секции: directives: running\_files: = 0.

Отредактируем секцию, как сказано по условию задания. Результат правок представлен на рисунке 24.

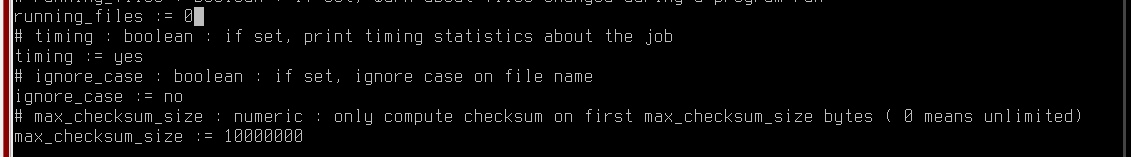


Рисунок 24 – редакция секции directives

**Задание 14.** Отредактировать секцию alias конфигурационного файла

/etc/afick.conf системы AFICK: изменить правило ЕТС, удалив из него проверку размера файловых сущностей и добавив проверку времени их модификации: исходный вариант правила: ETC = р+d+i+u+g+s+md5, отредактированный вариант правила: ЕТС = р+d+i+u+m+c+a+md5. Отредактировать правило MyRule, удалив из него проверку для файловых сущностей количества ссылок на них и добавив проверку контроля целостности мандатных меток безопасности, контроля целостности данных системы аудита безопасности и контроля целостности с использованием криптографического алгоритма ГОСТ Р 34. 11-2012 вместо алгоритма MD5: исходный вариант правила: MyRule = p+d+i+n+u+g+s+b+md5+m, отредактированный вариант правила: MyRule = p+d+i+u+g+s+b+gost+m+е+t.

На рисунке 25 представлено выполнение задания.

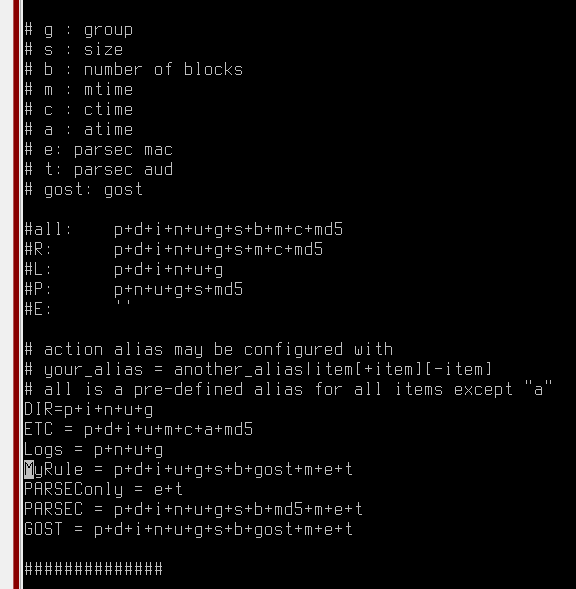


Рисунок 25 – результат выполнения задания

**Задание 15.** Отредактировать секцию file section конфигурационного файла /etc/afick.conf. Заменить для каталога /boot правило проверки GOST на правило проверки PARSEC: **исходный вариант:** /boot GOST, **отредактированный вариант:** /boot PARSEC. Добавить для файловой сущности /etc/fstab правило проверки MyRule: **отредактированный вариант:**

/etc/fstab MyRule, активировать правило проверки по умолчанию для каталога /lib: исходный вариант: #/lib MyRule, отредактированный вариант: /lib MyRule.

Отредактируем секцию file section конфигурационного файла. Заменим все, как было предложено по условию задания. В результате должно получиться то, что представлено на рисунках 26, 27.

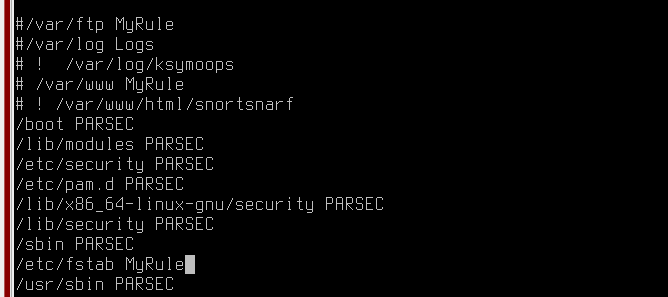


Рисунок 26 – редактирование файла

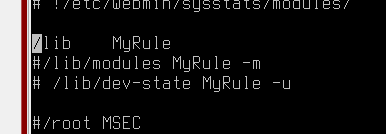


Рисунок 27 – редактирование файла

**Задание 16.** Обновить базу данных системы AFICK с учётом выполненных изменений в секции file section командой afick -u.

В терминал просто вставляем команду из условия. В результате должно получиться то, что представлено на рисунке 28.

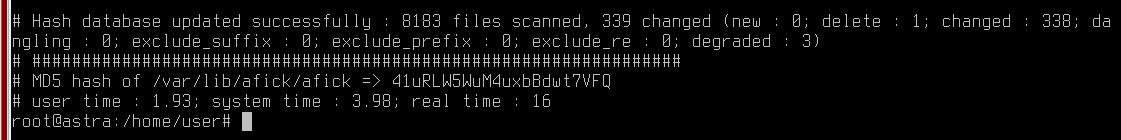


Рисунок 28 – обновление базы данных системы

**Задание 17.** Изменить содержимое файла /etc/fstab, удалив в нем две первые строки.

Откроем через nano /etc/fstab. В результате должно получиться то, что представлено на рисунке 29.



Рисунок 29 – редактирование файла /etc/fstab

**Задание 18.** Запустить графическую утилиту «Контроль целостности файлов» (afick-tk) управления AFICK из меню «Системные» главного пользовательского меню и выполнить принудительную проверку целостности, выбрав действие — сравнение с базой.

Запустим графическую утилиту, после этого выполним сравнение. Результат представлен на рисунке 30.

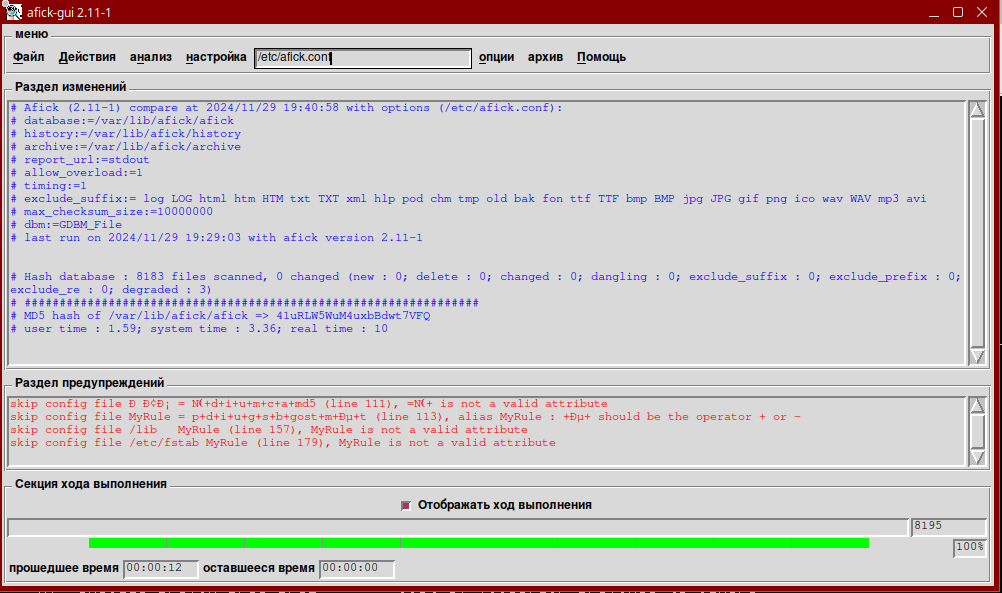


Рисунок 30 – выполнение принудительной проверки целостности

**Задание 19.** После завершения контроля целостности: в меню утилиты afick-tk «Файл — история» определить дату и время последнего принудительного контроля целостности; найти в каталоге /var/lib/afick/archive log-файл, соответствующий выполненной принудительной проверке (значение YYYYMMDDHHMMSS в имени log-файла должно совпадать с найденными в предыдущем пункте датой и временем проверки); просмотреть найденный log-файл с помощью команды less и в его секции #detaled changes найти запись о нарушении целостности файловой сущности /etc/fstab (раздел changed file: /etc/fstab); проанализировать найденную запись о нарушении целостности и определить параметры, соответствующие действиям (action) нарушения целостности, и их текущие значения.

Определим сначала даты, они представлены на рисунке 31, 32.

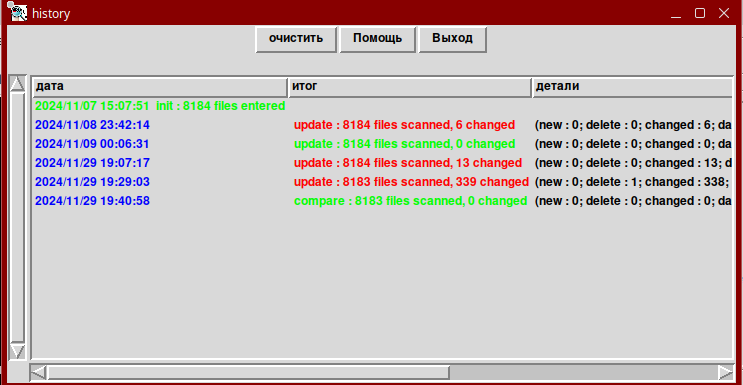


Рисунок 31 – время последнего принудительного контроля целостности

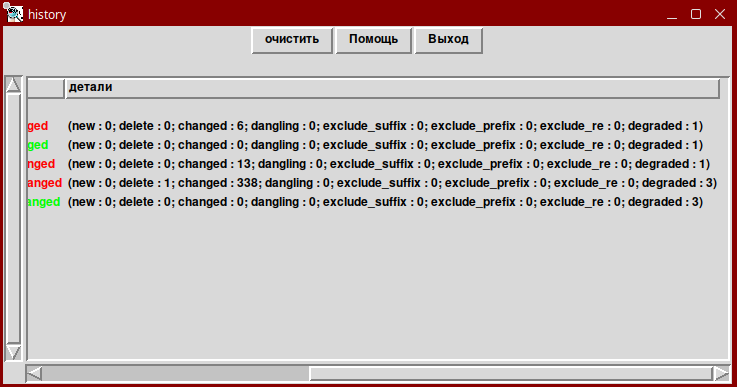


Рисунок 32 – время последнего принудительного контроля целостности

Найдем в каталоге /var/lib/afick/archive log-файл, соответствующий выполненной принудительной проверке (значение YYYYMMDDHHMMSS в имени log-файла должно совпадать с найденными в предыдущем пункте датой и временем проверки). В нашем случае получилось то, что представлено на рисунке 33.

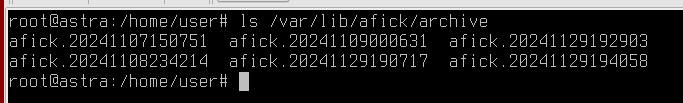


Рисунок 33 – просмотр логов

Просмотрим найденный log-файл с помощью команды less и в его секции #detaled changes найти запись о нарушении целостности файловой сущности /etc/fstab (раздел changed file:/etc/fstab). В результате должно получиться то, что представлено на рисунке 34.

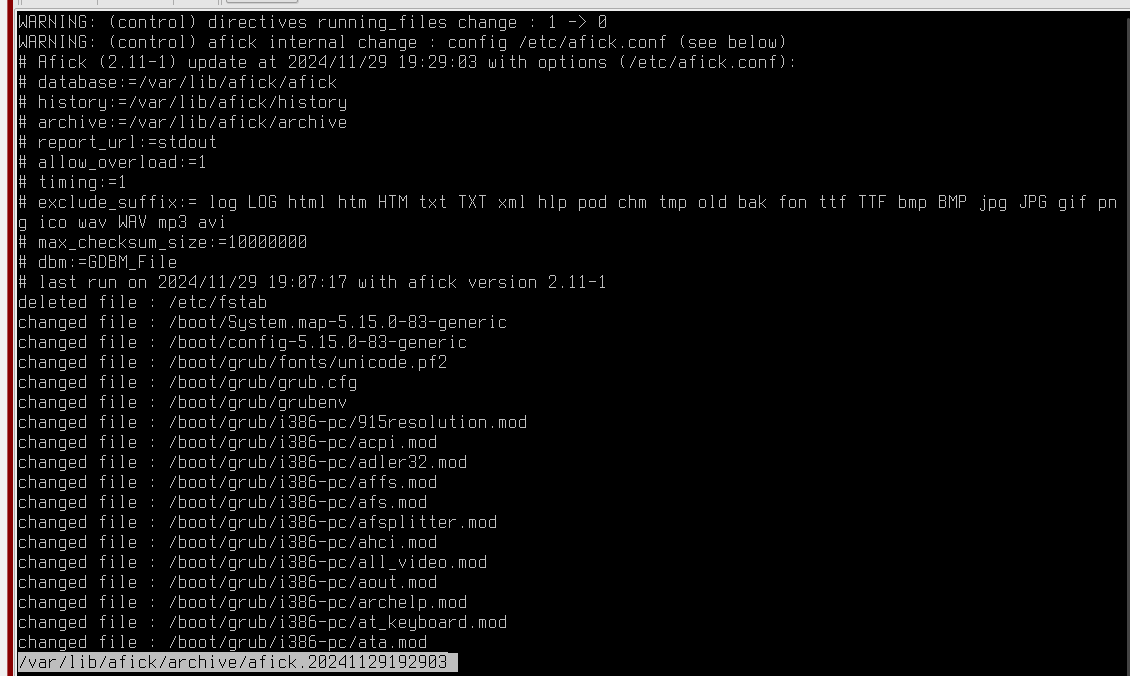


Рисунок 34 – просмотр log файлов

**Задание 20.** Запустить терминал Fly в «привилегированном» режиме командой sudo flyterm.

Запустим терминал, как сказано по условию задания. Результат представлен на рисунке 35.



Рисунок 35 – запуск терминала

**Задание 21.** Просмотреть загруженные модули ядра ОССН и вывести в терминал данные о невыгружаемом модуле digsig\_verif конвейером команд lsmod | grep «digsig\_verif». Просмотреть информацию о модуле digsig\_verif командой modinfo digsig\_verif. Определить расположение модуля digsig\_verif и информацию о разработчике. Выполнить импорт открытых ключей, используемых для проверки ЭП файлов.

Просмотрим загруженные модуля ядра, использовав команду, которая предложена по условию задания. Результат представлен на рисунке 36. Вывод покажет: имя модуля; используемую память; количество других модулей, которые зависят от него.

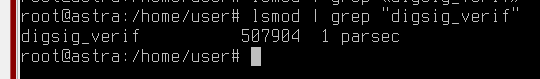


Рисунок 36 – просмотр загруженных модулей

Просмотреть информацию о модуле digsig\_verif командой modinfo digsig\_verif. Определим расположение модуля digsig\_verif и информацию о разработчике. Результате представлен на рисунке 37.

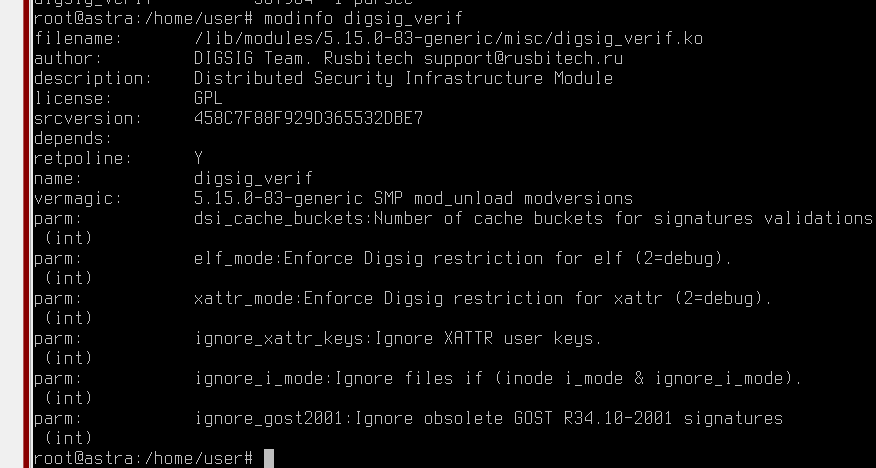


Рисунок 37 – просмотр информации о модуле и разработчике

Выполним импорт открытых ключей, используемых для проверки ЭП файлов. Для этого нам нужно выполнить следующие действия: инициализировать каталог /root/.gnupg при просмотре текущих ключей командой gpg --list-sigs. Результат представлен на рисунке 38.

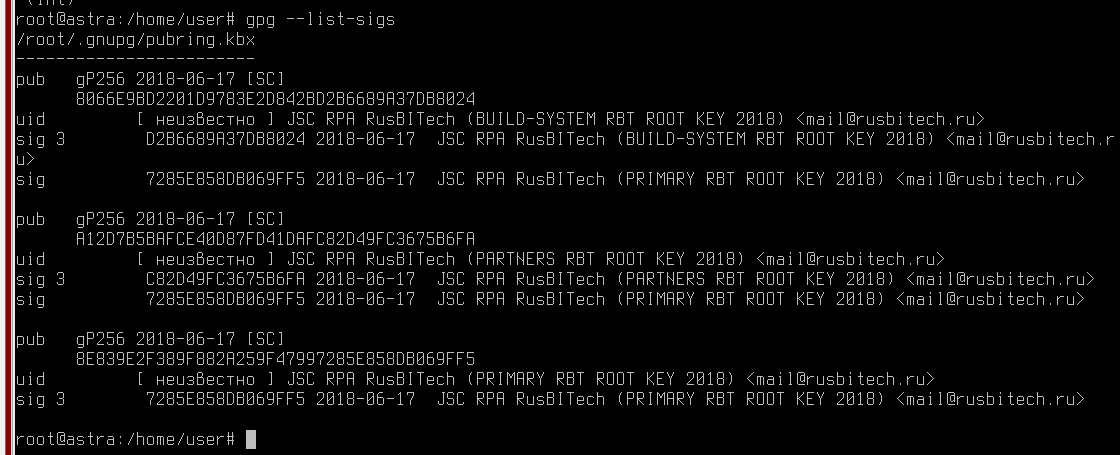


Рисунок 38 – импорт открытых ключей

Теперь импортировуем открытый мастер-ключ «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)» командой gpg --import /etc/digsig/primary\_key\_2018.gpg. Результат представлен на рисунке 39.

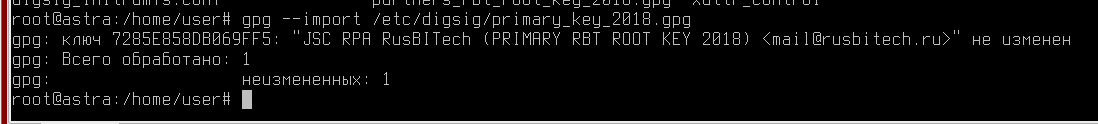


Рисунок 39 – импортирование ключа

Импортировать открытые ключи partners\_rbt\_root\_key\_2018.gpg и build\_system\_rbt\_root\_key\_2018.gpg (данный ключи используется для подписи файлов ОССН), командой gpg --import /etc/digsig/partners\_rbt\_root\_key\_2018.gpg

gpg --import /etc/digsig/build\_system\_rbt\_root\_key\_2018.gpg. Результат выполнения команды представлен на рисунке 40.

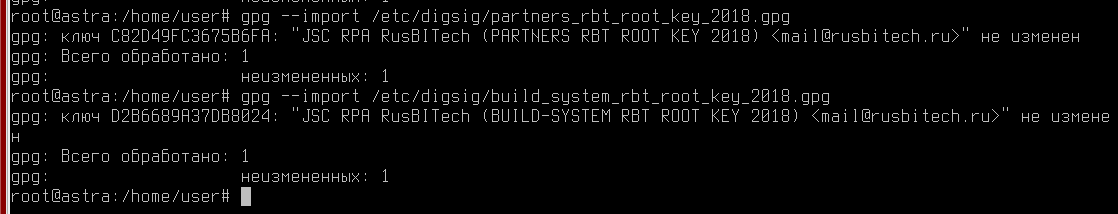


Рисунок 40 – импортирование открытых ключей

Выведем теперь перечень используемых ключей командой gpg --list-keys. Результат представлен на рисунке 41.

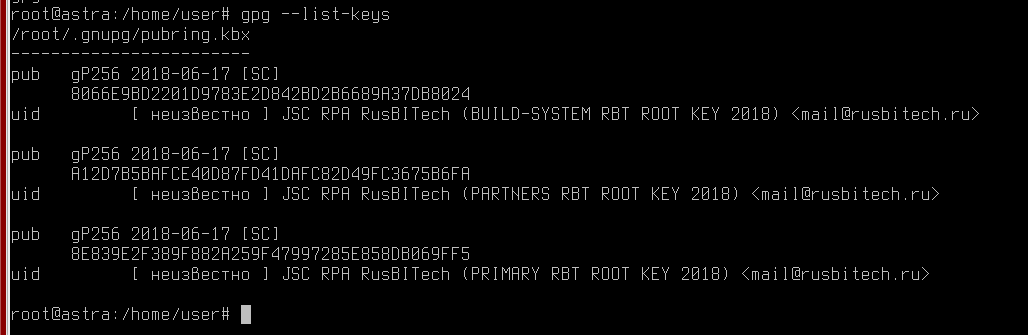


Рисунок 41 – импорт используемых ключей

Теперь нужно вывести текущие ключи командой gpg --list-sigs. Результат представлен на рисунке 42. Данная команда Она отображает список ключей в локальном хранилище, включая их подписи. В выводе мы увидим: основную информацию о ключах (например, идентификатор ключа, владельца, email). Список подписей, которые были сделаны этим ключом для других ключей. сам себя и потом других.

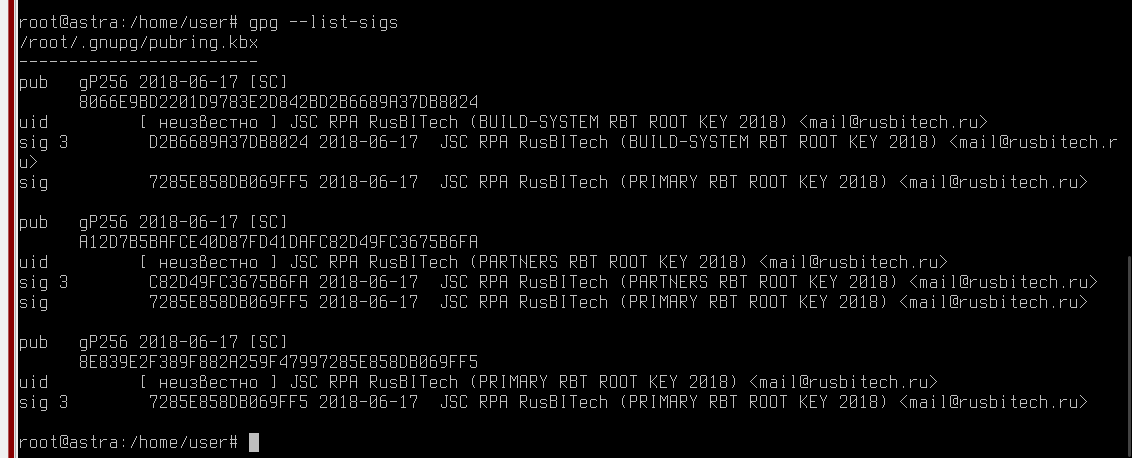


Рисунок 42 – импорт текущих ключей

Проверим корректность ЭП файла /bin/dash командой bsign -w$(which dash). Определим, каким ключом был подписан данный файл по его идентификатору в строке «signer». В нашем случае - JSC RPA RusBITech (BUILD-SYSTEM RBT ROOT KEY 2018) [mail@rusbitech.ru](mailto:mail@rusbitech.ru). В результате должно получиться то, что представлено на рисунке 43.



Рисунок 43 – проверка электронной подписи

Теперь нужно переписать открытый ключ /etc/digsig/build\_system\_rbt-rootkey\_2018.gpg в каталог /etc/digsig/keys командой cp /etc/digsig/build\_system\_rbt\_root\_key\_2018.gpg /etc/digsig/keys. В результате должно получиться то, что представлено на рисунке 44.

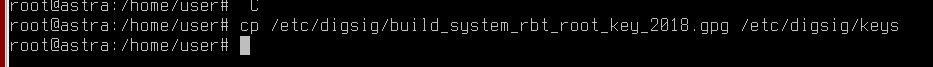


Рисунок 44 – переписывание открытого ключа

Теперь нам нужно перейти в каталог /etc/digsig и изменить файл digsig\_initramfs.conf (значение DIGSIG\_ELF\_MODE установить равным 1). Откроем через nano. Воспользуемся командой: nano /etc/digsig/digsig\_initramfs.conf. В результате должно получиться то, что представлено на рисунке 45.



Рисунок 45 – изменение файла

Проверим корректность установки данного параметра путём открытия настройки «Замкнутой программной среды» в «Панели управления». В нашем случае получилось то, что представлено на рисунке 46.

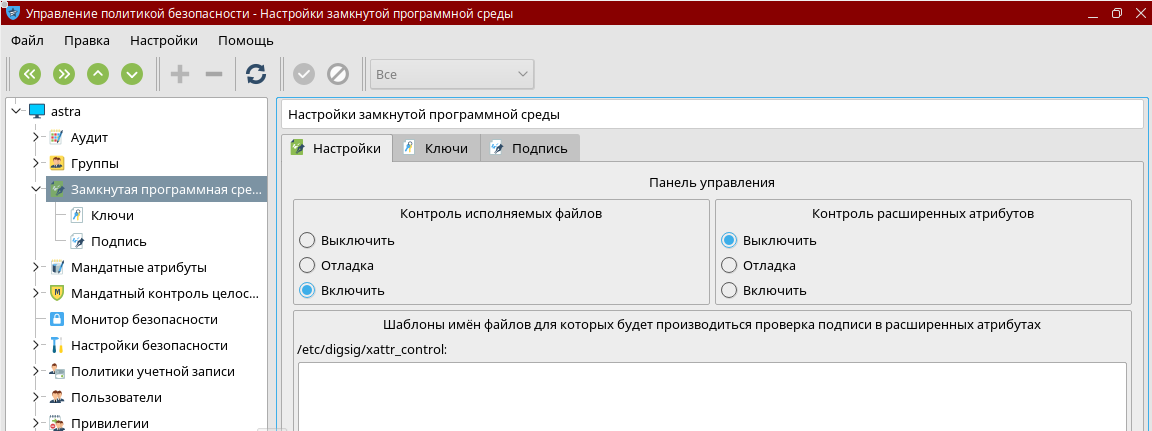


Рисунок 46 – настройки замкнутой программной среды

Теперь нам нужно создать дополнительный ключ ЭП команд. В диалоге команды gpg: выбрать пункт 15 «GOST R 34. 10-2012»; указать длину ключа – 1024; указать неограниченный срок действия дополнительного ключа ЭП, выбр

значение 0 указать параметры: указать полное имя: rootserver, адрес электронной почты: root@server.test и получить User ID: "rootserver [root@server.test](mailto:root@server.test)”. Вывести текущие ключи командой gpg --list-sigs и определить идентификатор ключа "rootserver [root@server.test](mailto:root@server.test)”. Результат представлен на рисунках 47, 48.

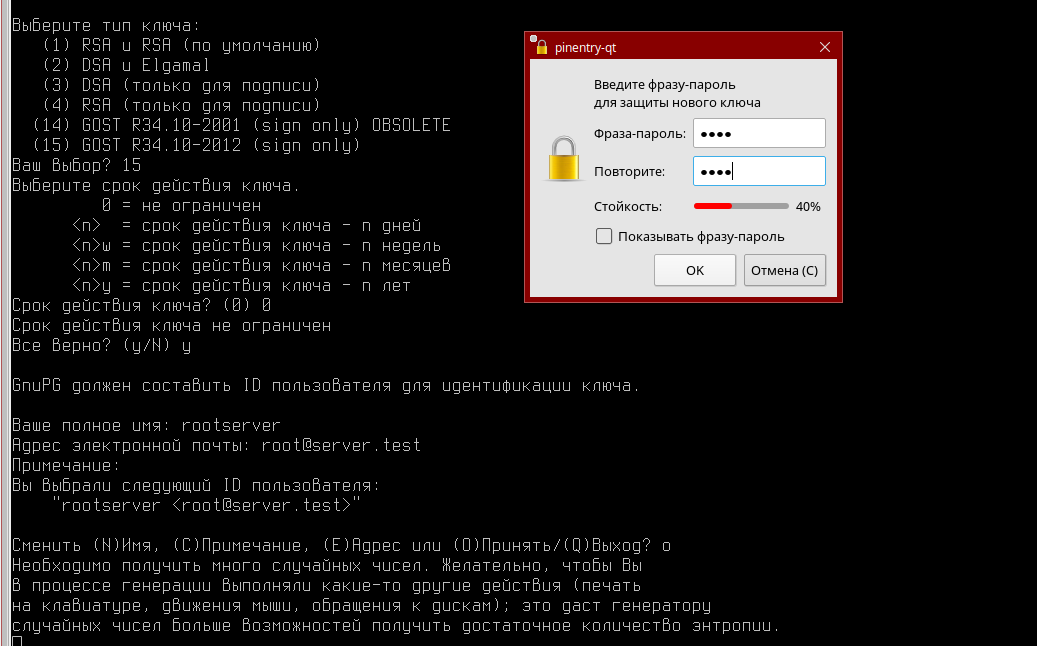


Рисунок 47 – создание дополнительного ключа ЭП

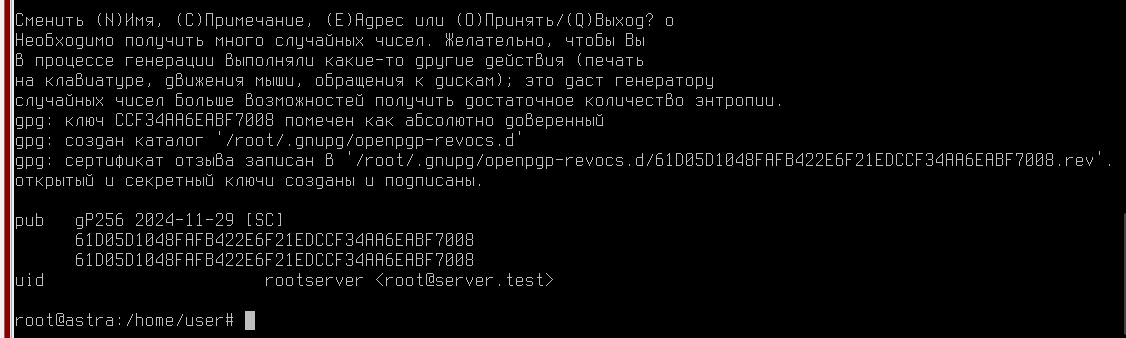


Рисунок 48 – создание дополнительного ключа ЭП

Теперь нам нужно скопировать файл /bin/dash в каталог /root, указав при этом новое имя файла 1.elf. Для этого используем команду: sudo cp /bin/dash /root/1.elf. Сразу подпишем файл новым ключом «rootserver <root@server.test>» командой gpg --default-key CCF34AA6EABF7008 bsign --sign /root/l.elf. В результате должно получиться то, что представлено на рисунке 49.

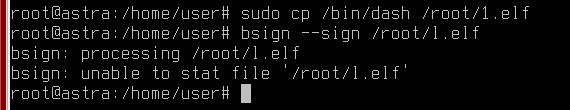


Рисунок 49 – копирование файла и подписывание его же

Выведем теперь новую подпись файла командой bsign -w /root/1.elf и

проверить соответствие идентификатора ключа ЭП в строке «signer. » данным ключа «rootserver <root@server.test>». В результате должно получиться то, что представлено на рисунке 50.

Файл был подписан дважды: один раз в теле файла (в основной подписи) и второй раз в атрибутах файла (xattr). В дополнение к обычной подписи файла, утилита bsign также может добавлять подпись в расширенные атрибуты (xattr) самого файла. Это позволяет подписывать не только сам файл, но и его метаданные, такие как права доступа, дата изменения и другие параметры.

**Обычная подпись**: Первая подпись, которая отображается как signature, — это основная подпись файла, которая создаётся на основе его содержимого.

**Подпись в расширенных атрибутах (xattr)**: Вторая подпись — это подпись, которая добавляется к расширенным атрибутам файла (xattr). Эта подпись используется для проверки целостности файла в его метаданных, в том числе защиты информации о том, как файл был подписан.

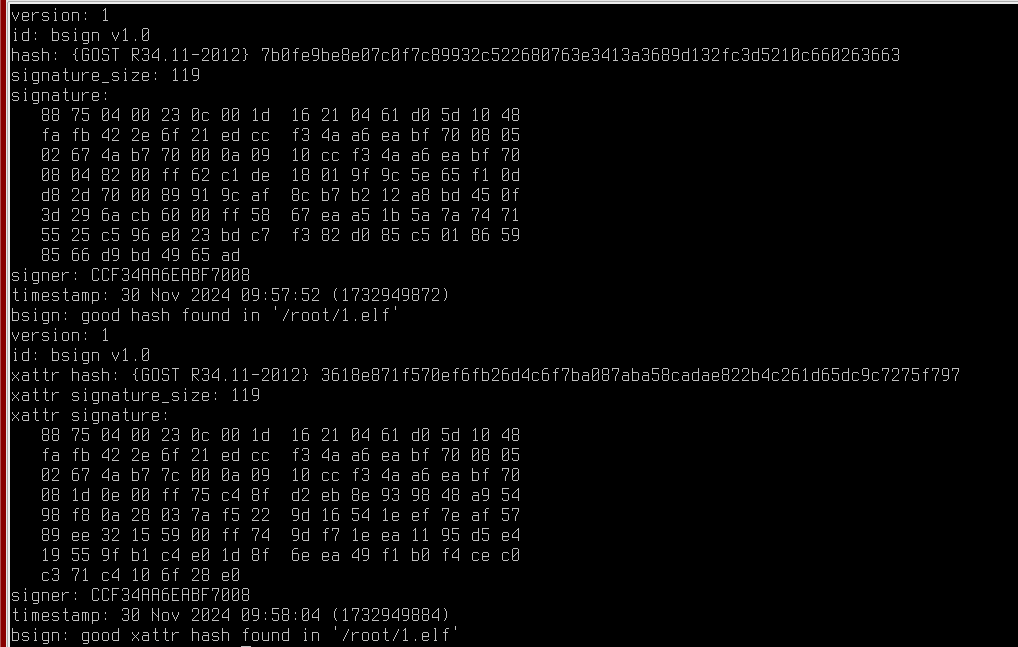


Рисунок 50 – вывод подписи файла

Теперь нам нужно активировать настройки командой sudo update-initramfs -u -k all, а затем выполнить перезагрузку и повторный вход в ОССН. Выполнение команды представлено на рисунке 51.

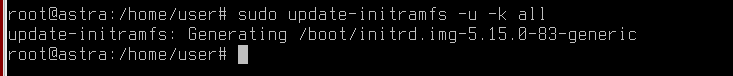


Рисунок 51 – активация настроек

Запустим терминал Fly в «привилегированном» режиме командой sudo

fly-term. Проверить включение штатного режим функционирования модуля

digsig\_verif. В файле /sys/digsig/elf\_mode у нас должно быть установлено значение «1». Для проверки воспользуемся командой cat /sys/digsig/elf\_mode. Результат выполнения представлен на рисунке 52.

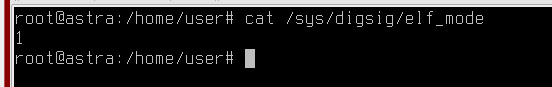


Рисунок 52 – проверка штатного функционирования модуля

Выполним попытку запуска файла /root/1.elf, который был подписан с использованием ключа «rootserver <root@кщserver. test>» (данный ключ не был подписан мастер-ключом «JSC RPA RusBITech (PRIMARY RBT ROOT KEY 2018)»). Результат попытки запуска файла представлен на рисунке 53.

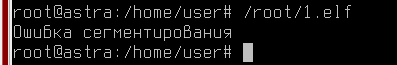


Рисунок 53 – проверка попытки запуска файла

Установим значение ключа DIGSIG\_ELF\_MODE=0 в конфигурационном файле /etc/digsig/digsig\_initramfs.conf, активировать настройки командой sudo update-initramfs -u -k all. После этого в «привилегированном» режиме терминала нам нужно выполнить команду /root/1.elf. Результат представлен на рисунке 54.

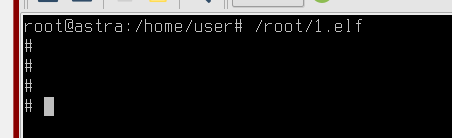


Рисунок 54 – попытка запуска файла после изменения параметра

Теперь нам нужно выйти из запущенного интерпретатора «dash» (файл 1.elf) командой exit. Сразу же активируем настройки командой sudo update-initramfs -u -k all, затем выполним перезагрузку и повторный вход в ОССН. Выполнение команд представлено на рисунке 55.



Рисунок 55 – выход из интерпретатора dash

Создадим ключи и выполнить подпись файла конфигурации. Для этого в начале нам нужно запустить терминал Fly от имени учётной записи привилегированного пользователя командой fly-term. После этого скопировать файлы /etc/passwd и /bin/dash в каталог ~ и сменить владельца на user1. Результат выполнения представлен на рисунке 56.

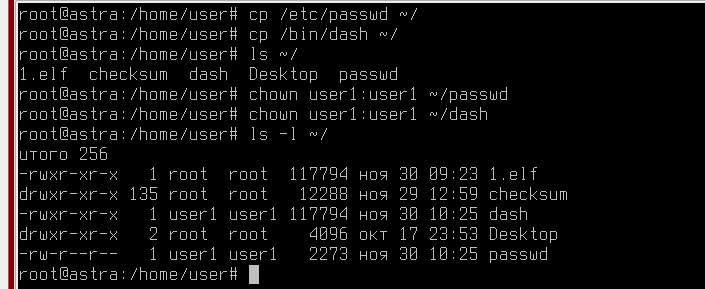


Рисунок 56 – создание ключа и подпись файла конфигурации

Выполним команду генерации мастер-ключа для подписи в xattr командой gpg --full-generate-key. Нам нужно выбрать алгоритм (15) и установить имя: xattr-key. Результат выполнения представлен на рисунке 57.



Рисунок 57 – генерация мастер-ключа

Выполним теперь команду генерации ключа для подписи в xattr командой gpg --full-generate-key, выбрать алгоритм (15) и установить имя: xattr-key-sign. В результате должно получиться то, что представлено на рисунке 58.

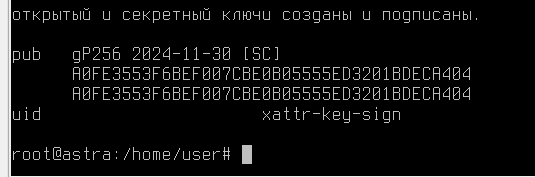


Рисунок 58 – генерация ключа для подписи

Выполним подпись ключа «xattr-key-sign» командой gpg --sign-key "xattr-key-sign" > xattr-key-sign.gpg. Результат выполнения данной команды представлен на рисунке 59.

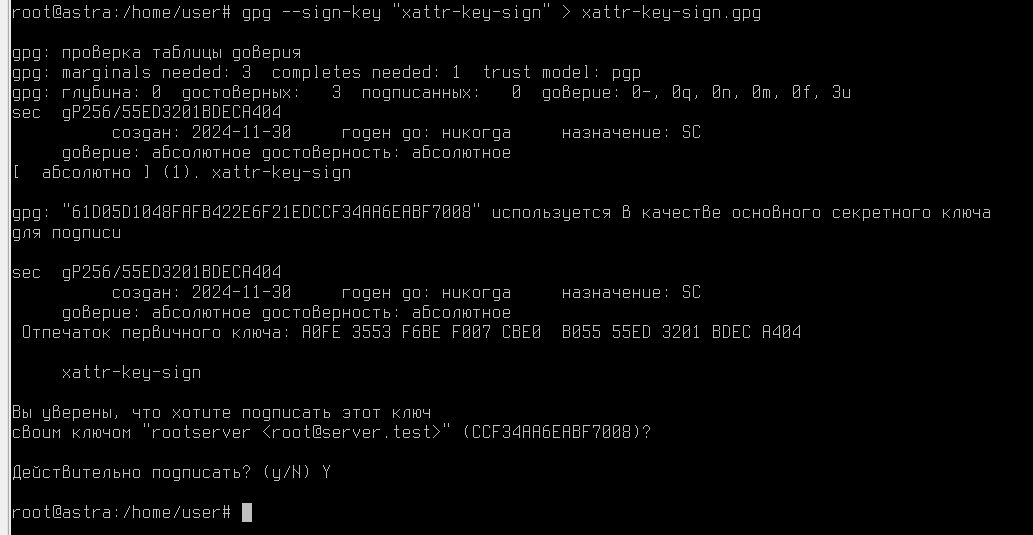


Рисунок 59 – генерация подписи ключа

Экспортируем ключ «xattr-кеу» командой gpg --export "xattr-key" xattr-key.gpg. Результат выполнения команды представлен на рисунке 60.

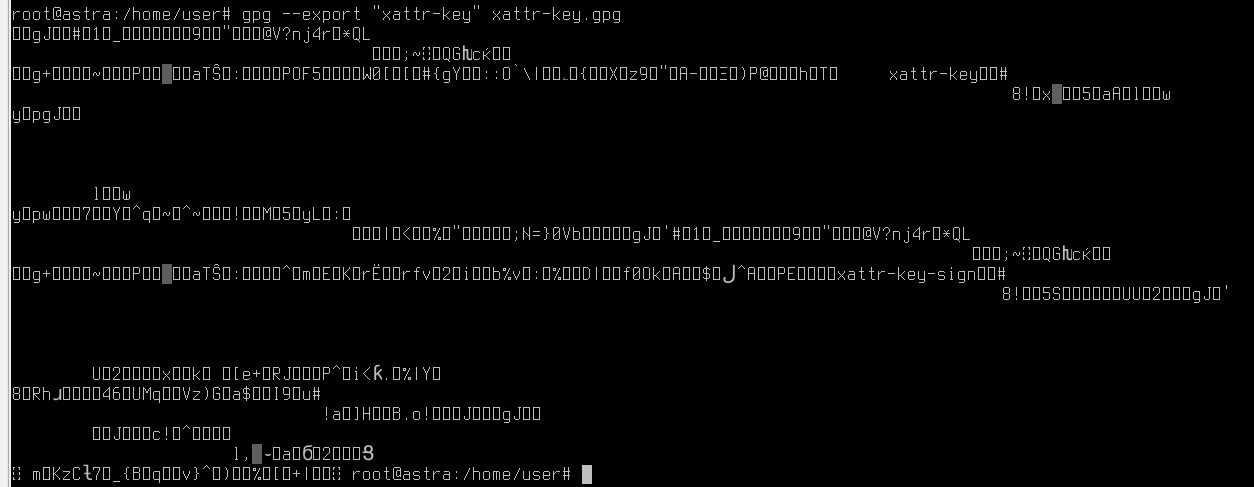


Рисунок 60 – экспорт ключа

Проверим наличие подписанного ключа «xattr-key-sign» командой gpg --list-sigs. Ключ «xattr-key-sign» должен быть подписан ключом «xattr-key». Результат выполнения команды представлен на рисунке 70.

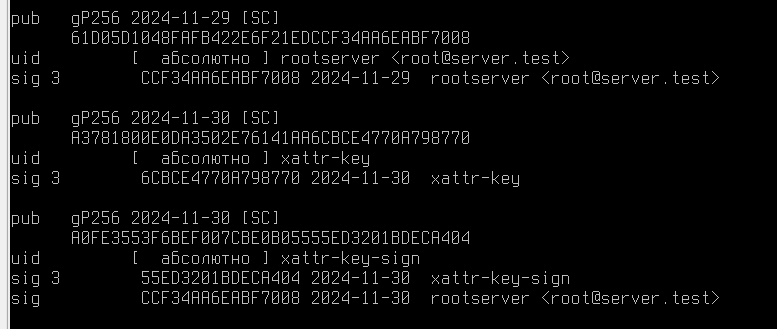


Рисунок 70 – проверка наличия подписанного ключа

Создадим хэш файл ~/passwd и запишем его в расширенные атрибуты

командой sudo bsign --hash ~/passwd. Команда представлена на рисунке 71.

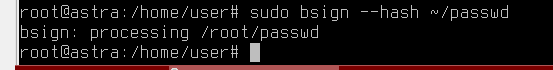


Рисунок 71 – запись в расширенные атрибуты у файла

Создадим файл ~/.gnupg/gpg.conf с содержимым: default-key идентификатор\_ключа\_xattr-key-sign. В нашем случае должно получиться то, что представлено на рисунке 72.

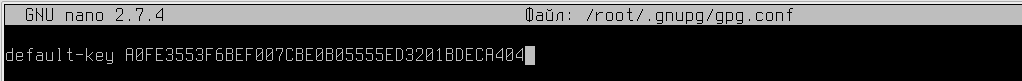


Рисунок 72 – создание файла с ключом

Выполним подпись файла passwd командой bsign --sign ~/passwd. В результате должно получиться то, что представлено на рисунке 73.

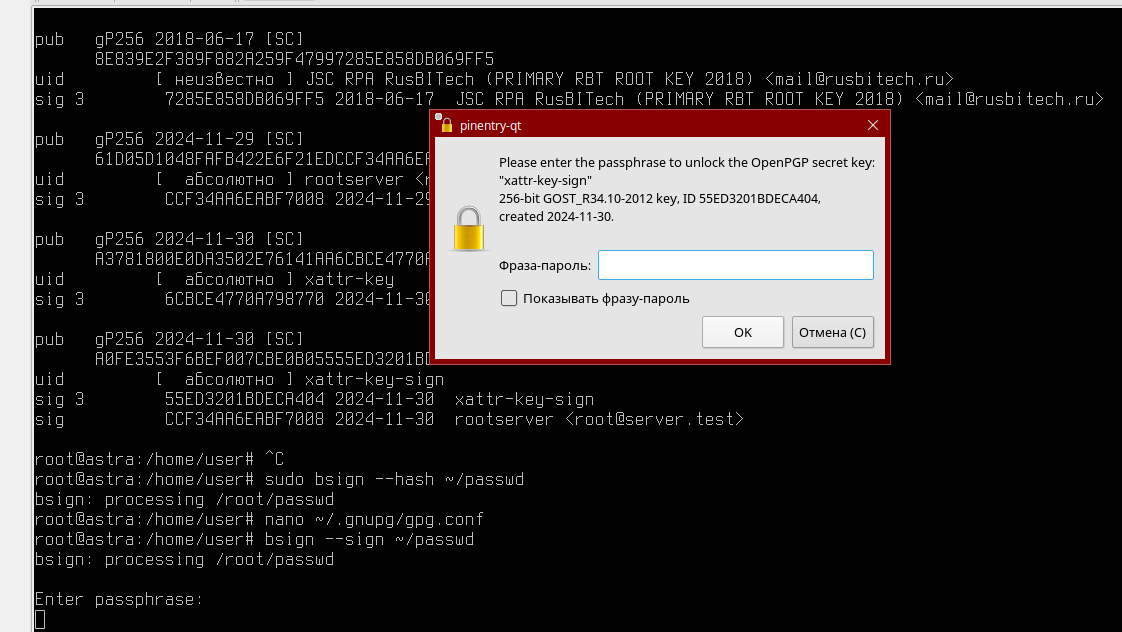


Рисунок 73 – подпись файла

Выполнить проверку подписи файла passwd командой bsign -w ~/passwd. Команда и результат выполнения представлены на рисунке 74.

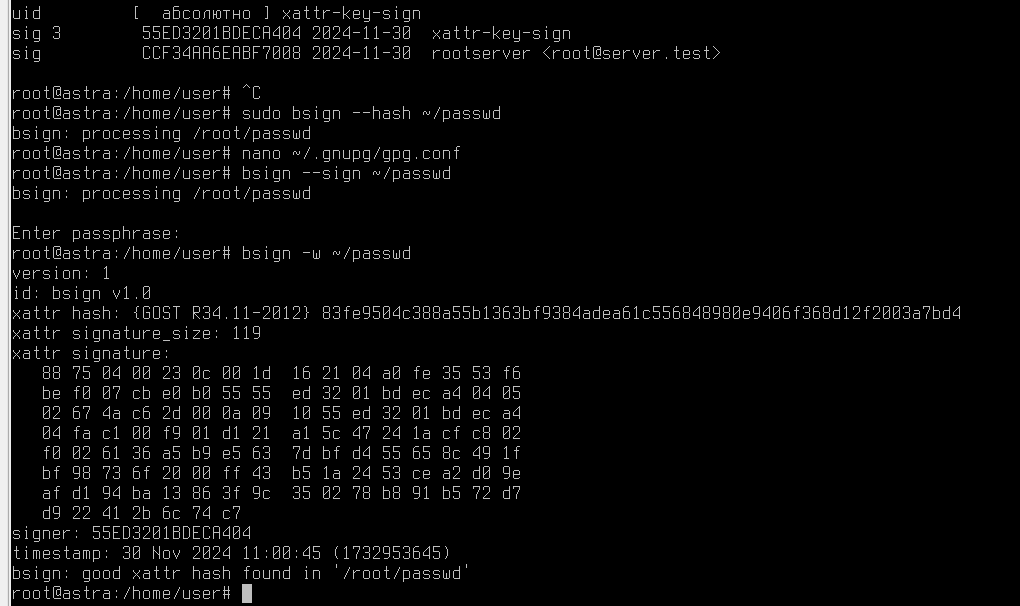


Рисунок 74 – выполнение проверки подписи файла

Теперь скопируем ключи (xattr-key-sign.gpg и xattr-key.gpg) для работы

с подписями файлов в каталог /etc/digsig/xattr\_keys. Команда и результат выполнения представлены на рисунке 75.



Рисунок 75 – копирование ключей с подписями файлов

В графическом файловом менеджере fly-fm перейти в каталог «Домашний» и открыть в контекстном меню «Свойства», «Подпись» файла passwd. Результат представлен на рисунке 76.

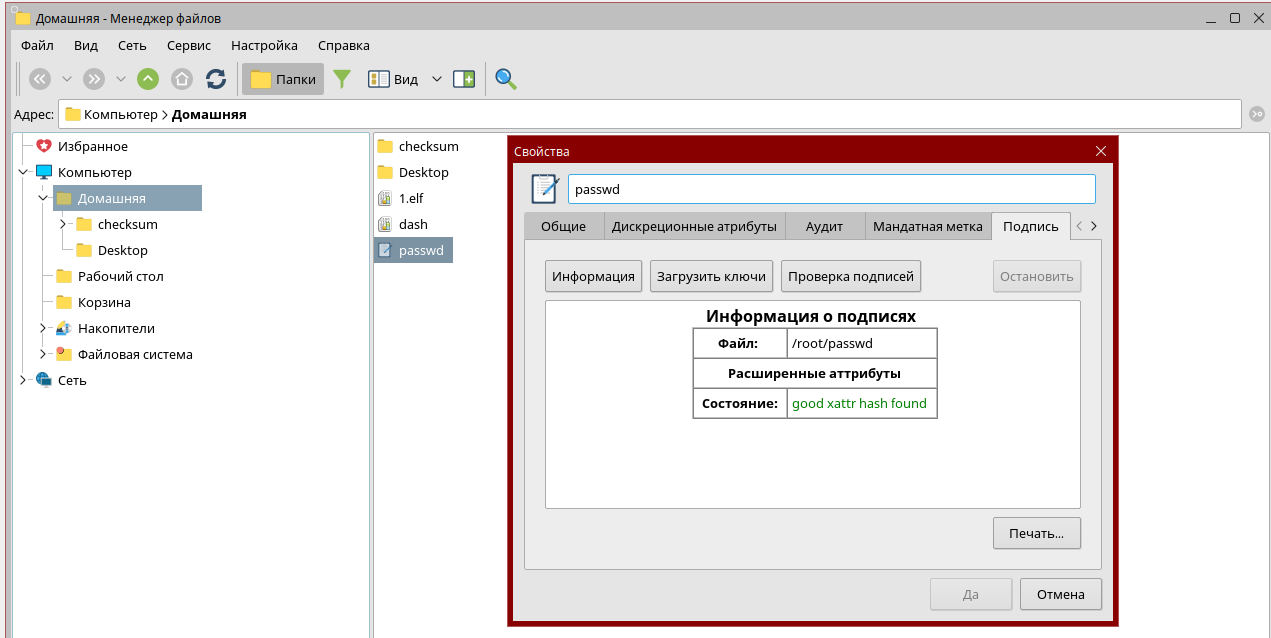


Рисунок 76 – свойства файла

Теперь нам нужно нажать на две кнопки - «Загрузить ключи» и «Информация». В результате должно получиться то, что представлено на рисунке 77.

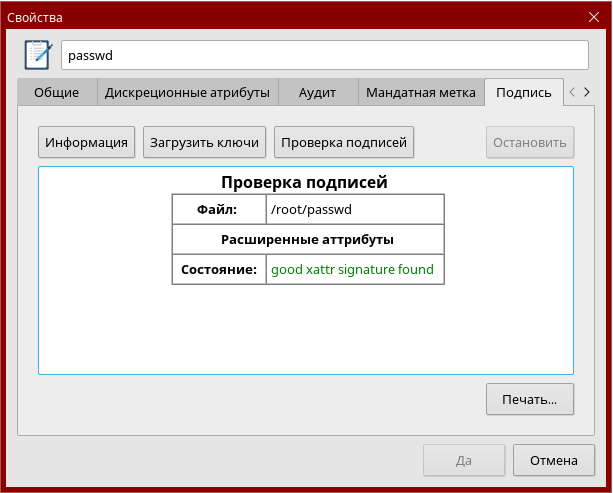


Рисунок 77 – просмотр свойства файла

**Контрольные вопросы**

1. В чем заключается отличие команд md5sum, shasum и gostsum с точки зрения вычисления контрольной суммы файлов?

Отличие заключается в используемом **алгоритме хеширования**:

* **md5sum** — использует алгоритм MD5 (128-битный хеш). Считается криптографически нестойким и уязвимым к коллизиям. Не рекомендуется для использования в security-целях.
* **shasum** — по умолчанию использует алгоритм SHA-1 (160-битный хеш). Также более не считается надёжным. Важно: утилита shasum поддерживает и другие алгоритмы семейства SHA (см. вопрос 3).
* **gostsum** — использует российский криптографический стандарт **ГОСТ Р 34.11-2012** (алгоритм "Стрибог"), который может выдавать хеши длиной 256 или 512 бит. Является современным и стойким алгоритмом, рекомендованным к использованию в РФ.

**Ключевое отличие:** разная криптографическая стойкость и принадлежность к стандартам (международные vs. российский).

2. В каком формате организован вывод команд md5sum, shasum и gostsum при вычислении контрольной суммы файлов?

Все три команды по умолчанию используют **одинаковый формат вывода**, совместимый с форматом, который ожидает утилита checksum для проверки:

<хеш> <путь\_к\_файлу>

* **<хеш>** — это строка шестнадцатеричных цифр, представляющая рассчитанную контрольную сумму.
* **<путь\_к\_файлу>** — это относительный или абсолютный путь к файлу, для которого вычислялась сумма.

3. В какой из команд md5sum, shasum или gostsum возможно изменение алгоритма хэширования?

Изменение алгоритма хэширования возможно в утилите **shasum**.

Она поддерживает несколько алгоритмов семейства SHA через ключ -a (algorithm):

* -a 1 — SHA-1 (по умолчанию)
* -a 224 — SHA-224
* -a 256 — SHA-256 (**наиболее распространённый вариант**)
* -a 384 — SHA-384
* -a 512 — SHA-512

4. Какие правила в конфигурационном файле системы регламентного контроля целостности AFICK сформированы по умолчанию, а какие являются специфическими для подсистемы безопасности PARSEC?

Система контроля целостности **AFICK** (Another File Integrity Checker) использует конфигурационный файл afick.conf.

* **Правила по умолчанию** (регламентируют общие настройки сканирования):
  + scan — директории для полного сканирования.
  + ignore — маски файлов и директорий, которые следует игнорировать при сканировании (например, /tmp/\*, \*.log).
  + alert — маски файлов, для которых следует генерировать предупреждения при изменении.
  + safe — маски файлов, изменения в которых считаются безопасными и не вызывают тревог (например, файлы в /var/log/).
  + log — файл журнала AFICK.
* **Специфические правила для PARSEC**:  
  Подсистема безопасности **PARSEC** добавляет в afick.conf правила, связанные с объектами контроля доступа (ОКД) и её собственной конфигурацией. Это могут быть правила для отслеживания целостности:
  + ignore \*psc\_db\* — игнорирование файлов БД политик безопасности.
  + Специальные правила для файлов в директориях /etc/parsec/, /var/parsec/.
  + Правила для ключевых системных файлов, связанных с мандатным контролем доступа (МКД).

5. Как инициализировать базу данных системы регламентного контроля целостности AFICK после внесения изменений в её конфигурационный файл?

После внесения изменений в afick.conf необходимо **переинициализировать базу данных**, так как старая БД не соответствует новому набору правил (появились новые игнорируемые пути и т.д.).

6. Каким видом модулей ядра ОССН является модуль digsig\_verif?

После внесения изменений в afick.conf необходимо **переинициализировать базу данных**, так как старая БД не соответствует новому набору правил (появились новые игнорируемые пути и т.д.).

7. Какой формат файлов ключей ЭП СПО использует модуль digsig. verif?

Модуль digsig\_verif использует формат **X.509** для файлов открытых ключей. Это стандартный формат для сертификатов, широко используемый в инфраструктуре открытых ключей (PKI).

Ключи, как правило, хранятся в файлах с расширениями .crt или .pem.

8. Какой файл сценария командного интерпретатора bash применяется при добавлении дополнительных ключей ЭП для модуля digsig\_verif?

Для добавления дополнительных ключей (сертификатов) в доверенное хранилище модуля digsig\_verif используется файл-сценарий:  
**/etc/initramfs-tools/scripts/init-top/digsig\_verif**

Этот скрипт выполняется на самой ранней стадии процесса загрузки системы (в рамках initramfs) и отвечает за загрузку доверенных ключей в модуль ядра. Для добавления нового ключа его необходимо добавить в контейнер certs.db, на который ссылается данный скрипт, или модифицировать сам скрипт.