ISSN 3033-9359
Выпуск 3 (007), 2025

# НАУЧНЫЙ ЖУРНАЛ ПРОФЕССИОНАЛИТЕТ

Москва

# ПРОФЕССИОНАЛИТЕТ

**№ 3 (007), 2025**

**Научный журнал**

Основан в 2023 году

Зарегистрирован федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций

Номер свидетельства ЭЛ № ФС 77-84640

Дата регистрации 01.02.2023

**Учредитель:**

Уймин А.Г.



**Адрес редакции:**

Адрес редакции: 115432, г. Москва, ул. 5-я Кожуховская, д. 18, корп. 1, пом. 12.

**ПРОФЕССИОНАЛИТЕТ, 2025, № 3 (007)**

# PROFESSIONALITET

**No.3 (007), 2025**

**Scientific Journal**

Founded in 2023

Registered with the Federal Service for Supervision of Communications, Information Technology and Mass Media

Certificate of Registration: EL No. FS 77-84640

Registration Date: 01.02.2023

**Founder:**

Uymin A.G.



**Editorial Office:**

Editorial Office Address: Premises 12, 18, Building 1, 5th Kozhukhovskaya St., Moscow, 115432, Russia.

## СОДЕРЖАНИЕ

## CONTENTS

**Рубрика 1. Информатика и информационные процессы**

***УДК*** *004.451.25*

***Е. А. Еремина*** [1]***, А. Н. Простова*** [1]
[1]*ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация*

# ИЗОЛЯЦИЯ ПРОЦЕССОВ В ОПЕРАЦИОННОЙ СИСТЕМЕ АЛЬТ С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМОВ CGROUPS V2

**Аннотация.** *Проблема управления вычислительными ресурсами в многозадачных операционных системах приобретает особую значимость с распространением сред контейнеризации и необходимостью одновременного выполнения задач с различными требованиями к ресурсам. Традиционные механизмы Linux – nice, cpulimit, ulimit – не обеспечивают комплексной групповой изоляции: nice задаёт лишь рекомендательный приоритет, cpulimit использует сигналы SIGSTOP/CONT, а ulimit ограничивает только сессию пользователя. Проблема интерференции рабочих нагрузок, известная как проблема «шумного соседа», остаётся ключевой в системном администрировании и информационной безопасности. Цель исследования: экспериментальная количественная оценка эффективности механизмов изоляции ресурсов cgroups v2 в дистрибутиве ОС Альт (ядро Linux 6.1.115). Методика. Для проверки гипотез H1a (изоляция CPU) и H1b (изоляция памяти) разработана серия контролируемых экспериментов с применением контроллеров cpuset и memory.max при варьировании нагрузки: 1–2 ядра, лимиты памяти 50–500 МБ, смешанный сценарий с приоритетными и фоновыми процессами. Измеряемые метрики: PSR (принадлежность ядру CPU), nr_switches (переключения контекста), memory.events (счётчики OOM Killer). Результаты. При использовании cpuset значение PSR для 100% наблюдений совпадало с назначенным ядром; nr_switches составил 3–4 переключения без межгрупповой миграции. При превышении лимита memory.max активировался механизм завершения процессов при нехватке памяти OOM Killer (oom_kill=1), не затрагивая другие группы. Выводы. Гипотезы H1a и H1b полностью подтверждены. Cgroups v2 демонстрирует принципиальные преимущества перед традиционными механизмами изоляции. Механизмы cpuset и memory.max могут поддерживать отдельные требования по разделению ресурсов и ограничению их исчерпания, однако сами по себе не доказывают соответствие приказу ФСТЭК России № 118 или PCI DSS 4.0.*

**Ключевые слова:** *cgroups v2; изоляция процессов; cpuset; memory.max; управление ресурсами; контейнеризация; информационная безопасность.*

## Введение

Актуальность проблемы изоляции процессов возрастает с усложнением программного обеспечения, распространением сред контейнеризации и необходимостью одновременного выполнения задач с различными требованиями к ресурсам [1]. Проблема интерференции рабочих нагрузок (известная как проблема «шумного соседа») – когда один процесс дестабилизирует всю систему – остается ключевой в системном администрировании и информационной безопасности. Инструменты Linux широко применяются для моделирования телекоммуникационных сетей [2] и подготовки системных администраторов [1]. Платформы контейнеризации, например, Docker и Podman, занимают в задачах изоляции центральное место, и понимание механизмов ресурсной изоляции приобретает практическую значимость.

Традиционные механизмы управления ресурсами в Linux не обеспечивают комплексной групповой изоляции. Например, команда nice задаёт лишь рекомендательный приоритет [3], cpulimit периодически приостанавливает процесс сигналами SIGSTOP/CONT – такой подход устарел для многоядерных систем [4], ulimit задаёт лимиты на уровне сессии пользователя, но не группы процессов [5]. Всё это стало стимулом к разработке механизма Control Groups.

Объект исследования: механизмы cgroups v2 в ОС Альт.

Предмет исследования: контроллеры cpuset и memory.max и их эффективность в обеспечении изоляции ресурсов.

Цель исследования: экспериментальная количественная оценка эффективности механизмов изоляции ресурсов cgroups v2 в ОС Альт.

Задачи исследования:

1. Провести сравнительный анализ традиционных механизмов управления ресурсами Linux и механизма cgroups v2.
2. Развернуть экспериментальную среду и выполнить контрольные замеры без применения cgroups.

3. Количественно оценить изоляцию CPU с применением контроллера cpuset при изменении числа ядер и типов нагрузки.

4. Количественно оценить изоляцию памяти с применением контроллера memory.max при изменении лимитов.

5. Исследовать сценарий смешанной нагрузки и верифицировать отсутствие взаимного влияния групп.

6. Выработать практические рекомендации по применению cgroups v2 в соответствии с требованиями регуляторов.

**Теоретические основы**

**Архитектура cgroups v2: унифицированная иерархия**

Control Groups (cgroups) – это механизм ядра Linux. Он служит для группировки процессов, а также управления их доступом к системным ресурсам, таким как CPU, память, ввод-вывод. В отличие от cgroups v1, где каждый контроллер монтировался в отдельной иерархии, cgroups v2 предоставляет единую унифицированную иерархию для всех контроллеров, монтируемую в /sys/fs/cgroup [6]. Это устраняет несогласованность политик при одновременном использовании нескольких контроллеров и упрощает конфигурирование.

Иерархия cgroups v2 представляет собой дерево директорий в файловой системе типа cgroup2. Корневая cgroup (/) объединяет все системные процессы. Создание поддиректории в ее иерархии приводит к появлению дочерней группы. Ключевым правилом безопасности является то, что дочерние группы наследуют ограничения родителя, но не могут их превышать [6]. Единая модель иерархии выступает обязательным условием для обеспечения совместного использования cgroups с пространствами имён ядра Linux. Детерминированность изоляции в рамках данного исследования – это свойство механизма гарантировать соблюдение ресурсных ограничений при любых условиях нагрузки, без зависимости от случайных решений планировщика или мгновенного состояния системы.

**Контроллеры cpuset и memory**

Контроллер cpuset закрепляет процессы группы за конкретными ядрами CPU. Он исключает их конкуренцию с процессами из других групп. Это предотвращает атаки по сторонним каналам (cache-timing attacks), связанные с совместным использованием кэшей CPU. Ключевые параметры:

1. cpuset.cpus – список разрешённых ядер; cpuset.mems – список разрешённых NUMA-узлов;
2. cpuset.cpus.effective – фактически доступные ядра с учётом ограничений родителя.

Контроллер memory управляет оперативной памятью группы. Параметр memory.max задаёт жёсткий лимит, то есть при его превышении ядро активирует механизм завершения процессов при нехватке памяти (OOM Killer, Out-Of-Memory Killer). Принципиальным отличием от cgroups v1 является то, что в v2 также поддерживается параметр memory.high. Эта директива устанавливает мягкое ограничение с регулированием скорости выполнения процессов, не приводящее к их немедленному завершению [7]. Для детерминированного срабатывания OOM Killer используется именно memory.max. Параметр memory.swap.max = 0 не дает обойти жесткий лимит через файл подкачки.

**Взаимодействие с пространствами имён (namespaces)**

Механизм cgroups v2 тесно взаимодействует с пространствами имён ядра Linux [8]. В частности, пространство имён cgroup (CLONE_NEWCGROUP) обеспечивает изоляцию представления иерархии cgroups для процесса, который работает внутри контейнера.

Исходя из сказанного выше, процессы внутри контейнера, не имея доступа к родительским группам, видят собственный корень иерархии. Совместное использование cgroups v2 с pid namespace, mount namespace и network namespace дает ту самую изоляцию, которая используется в Docker и Podman. В документации ядра [6] этот подход описан как предпочтительный для контейнерных сред.

**Рубрика 1. Информатика и информационные процессы**

**Литературный обзор**

Значительная часть исследований посвящена сравнению производительности cgroups v1 и v2. В работе Wiedner и др. [9], представленной на IEEE NFV-SDN, экспериментально показано, что cgroups v2 демонстрирует меньшие задержки на высоких перцентилях (P95, P99) по сравнению с v1 при сопоставимой изоляции. Например, реализация v2 выполняет примерно на 2,4% меньше инструкций, а количество миграций процессов сокращается в два раза. В документации systemd по управлению ресурсами [10] параметр MemoryMax= сопоставляется с жёстким пределом памяти cgroups v2; при достижении предела применяется OOM-механизм, что подтверждает применимость контроллера для ограничения потребления памяти сервисами. В документации PostgreSQL [7] отмечается принципиальное различие между мягкими (memory.high) и жёсткими (memory.max) лимитами в cgroups v2 – это различие напрямую учитывается в нашем эксперименте.

Практические аспекты настройки cgroups в ОС Альт представлены в работе Крайнова и Пашиной [11]. Однако данное исследование ограничивается демонстрацией базовых возможностей и не содержит количественной оценки изоляции через метрики PSR, nr_switches и memory.events. Это составляет научную новизну настоящей работы. Таким образом, формализованный эксперимент с количественной оценкой метрик изоляции именно в ОС Альт в открытом доступе отсутствует.

**Методология**

**Гипотезы исследования:**

1. H1a: использование контроллера cpuset в cgroups v2 гарантирует, что процессы ограниченной группы не выполняются на ядрах CPU вне своего cpuset ни при каких условиях нагрузки. Условие принятия гипотезы: для 100% наблюдений PSR находится внутри cpuset.cpus; nr_switches не превышает базовое значение более чем на 20%.
2. H1b: использование контроллера memory.max с отключенным swap гарантирует, что процессы, превышающие установленный лимит памяти, завершаются механизмом OOM Killer до того, как нехватка памяти затронет остальные группы. Условие принятия гипотезы: oom_kill ≥ 1 при превышении лимита или oom_kill = 0 при соблюдении лимита.

Среда исследования: ОС Альт (ALT Linux, ядро Linux версии 6.1.115) [12], cgroups v2 активны по умолчанию, 2 логических ядра CPU (CPU0, CPU1), 4 ГБ ОЗУ.

Инструменты: bash, stress (v1.0.4), htop, vmstat, dmesg.

**Количественные метрики:**

– PSR из команды ps -e -o pid,psr,cmd – номер ядра CPU для верификации изоляции;

– nr_switches из /proc/PID/schedstat – количество переключений контекста (прокси-метрика стабильности);

– memory.events (max, oom, oom_kill) – счётчики событий памяти для оценки OOM Killer;

– dmesg – системный журнал ядра для подтверждения завершения процессов-нарушителей.

Для углубленной оценки латентности (lmbench lat_ctx, perf stat) и пропускной способности требуется дополнительное исследование.

**Процедура эксперимента:**

– контрольный эксперимент без cgroups для получения базовых значений;

– серия тестов cpuset с изменением числа ядер (1, 2) и типов нагрузки;

– серия тестов memory.max с изменением лимитов (50 МБ, 500 МБ, без лимита);

– сценарий смешанной нагрузки (приоритетный + фоновый процессы).

**Экспериментальное исследование**

**Среда и подготовка**

Перед началом экспериментов были выполнены проверка активности cgroups v2 и установка необходимых инструментов:

```
apt-get install stress
stress --version # stress 1.0.4
```

Наличие cgroup2 в /proc/mounts и контроллера cpuset в cgroup.controllers подтвердили активность cgroups v2. Дополнительно установлены утилита stress v1.0.4 и компилятор gcc.

**Контрольный эксперимент (без cgroups)**

Для получения базовых значений метрик два процесса stress запущены без применения cgroups:

```
stress -c 2 &
ps -e -o pid,psr,cmd | grep stress
```

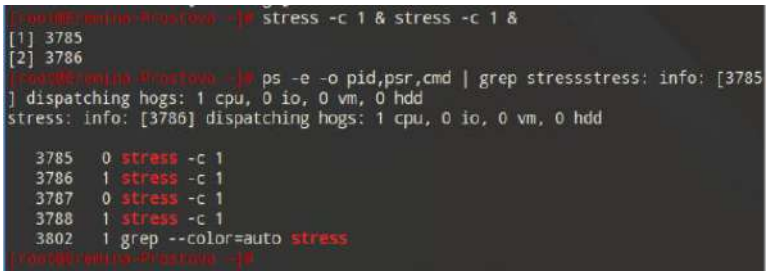


**Рис. 1. Распределение процессов stress по ядрам CPU без использования cgroups**

В контрольном эксперименте планировщик распределил процессы произвольно. Оба ядра загружены, однако администратор не может гарантировать, что критически важный процесс не будет конкурировать с менее приоритетным на одном ядре. Это подтверждает необходимость детерминированных механизмов изоляции.

**Изоляция CPU через cpuset**

Развертывание иерархии для двух изолированных групп group_A (ядро 0) и group_B (ядро 1):

```
# Включение cpuset в корне и создание иерархии
echo "+cpuset" > /sys/fs/cgroup/cgroup.subtree_control
mkdir /sys/fs/cgroup/cpu-isolation
echo "+cpuset" > /sys/fs/cgroup/cpu-isolation/cgroup.subtree_control
mkdir /sys/fs/cgroup/cpu-isolation/group_A
mkdir /sys/fs/cgroup/cpu-isolation/group_B

# Назначение ядер: group_A → 0, group_B → 1
echo "0-1" > /sys/fs/cgroup/cpu-isolation/cpuset.cpus
echo "0"   > /sys/fs/cgroup/cpu-isolation/group_A/cpuset.cpus
echo "1"   > /sys/fs/cgroup/cpu-isolation/group_B/cpuset.cpus

# Запуск процессов и помещение в группы
stress -c 1 & echo $! > /sys/fs/cgroup/cpu-isolation/group_A/cgroup.procs
stress -c 1 & echo $! > /sys/fs/cgroup/cpu-isolation/group_B/cgroup.procs

# Верификация (PSR = номер ядра)
ps -e -o pid,psr,cmd | grep stress
# 5161  0  stress -c 1   <- group_A, ядро 0
# 5163  1  stress -c 1   <- group_B, ядро 1
```


**Рис. 2. Закрепление процессов stress за ядрами CPU через cpuset (вывод ps -e -o pid,psr,cmd)**

Результат: процесс group_A выполнялся исключительно на ядре 0 (загрузка 100%), группа group_B – на ядре 1 (100%). Процессы не конкурировали за процессорное время.

**Количественная оценка стабильности изоляции через nr_switches**

```
STRESS_PID=$(pgrep -n stress)
cat /proc/$STRESS_PID/schedstat
```

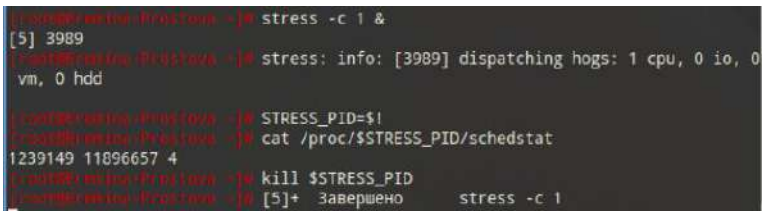


**Рис. 3. Значение nr_switches=4 для процесса stress, изолированного через** cpuset

Низкое значение nr_switches (4 переключения контекста за всё время жизни процесса) свидетельствует об отсутствии миграций между ядрами. Это подтверждает детерминированное выполнение на выделенном ядре.

**Тестирование с изменением числа ядер**

```
# Тест 1: одно ядро (ядро 0)
echo "0"   > /sys/fs/cgroup/cpu-isolation/test/cpuset.cpus
stress -c 1 & echo $! > /sys/fs/cgroup/cpu-isolation/test/cgroup.procs

# Тест 2: два ядра (0 и 1)
echo "0-1" > /sys/fs/cgroup/cpu-isolation/test/cpuset.cpus
stress -c 2 & echo $! > /sys/fs/cgroup/cpu-isolation/test/cgroup.procs
```

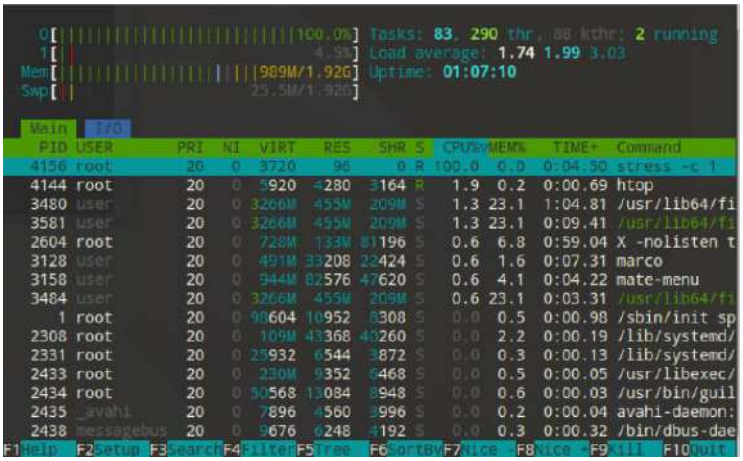


**Рис. 4. htop при cpuset="0": ядро 0 загружено на 100%, ядро 1 свободно**


**Рис. 5. htop при cpuset="0-1": оба ядра загружены равномерно**

**Ограничение: CPU-контроллер (cpu.weight)**

При попытке включить контроллер cpu командой echo "+cpu" > /sys/fs/cgroup/cgroup.subtree_control возникала ошибка «Отказано в доступе». Причиной является наличие realtime-процессов ядра вне корневой cgroup. Результат проверки представлен в таблице 1.

**Таблица 1 – Realtime-процессы, препятствующие включению CPU-контроллера cgroups v2**

| PID | Класс планирования | RT-приоритет | Команда |
|---|---|---|---|
| 15 | SCHED_FIFO (FF) | 99 | [migration/0] |

| | | | |
|---|---|---|---|
| 19 | SCHED_FIFO (FF) | 99 | [migration/1] |
| 41 | SCHED_FIFO (FF) | 50 | [watchdog] |

Согласно документации ядра [6], CPU-контроллер cgroups v2 не может быть включён при наличии RT-процессов (SCHED_FIFO/SCHED_RR) вне корневой cgroup, если ядро собрано с CONFIG_RT_GROUP_SCHED. Процессы migration и watchdog – системные realtime-процессы, которые не могут быть остановлены. Данное ограничение является задокументированным поведением и не является ошибкой.

**Изоляция памяти через memory.max**

Развертывание инфраструктуры для тестов памяти:

```
echo "+memory" > /sys/fs/cgroup/cgroup.subtree_control
mkdir -p /sys/fs/cgroup/mem-test/limited
```

Для экспериментов разработана тестовая программа на C, выделяющая заданный объём памяти с физическим заполнением (memset) для предотвращения lazy allocation. Заполнено в текстовом редакторе vim:

```
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
   int mb = argc > 1 ? atoi(argv[2]) : 100;
   char *p = malloc(mb * 1024 * 1024);
   if (p) { memset(p, 0, mb * 1024 * 1024); sleep(30); }
   return 0;
}

# Компиляция
gcc /tmp/oom_test.c -o /tmp/oom_test
```

**Эксперимент 1: лимит 50 МБ, запрос 100 МБ (превышение лимита)**

```
# Установка жесткого лимита 50 МБ + отключение swap
echo 52428800 > /sys/fs/cgroup/mem-test/limited/memory.max
echo 0        > /sys/fs/cgroup/mem-test/limited/memory.swap.max

# Запуск и помещение в группу
/tmp/oom_test 100 &
echo $! > /sys/fs/cgroup/mem-test/limited/cgroup.procs
sleep 5

# Анализ результатов
cat /sys/fs/cgroup/mem-test/limited/memory.current


dmesg | tail -5
```

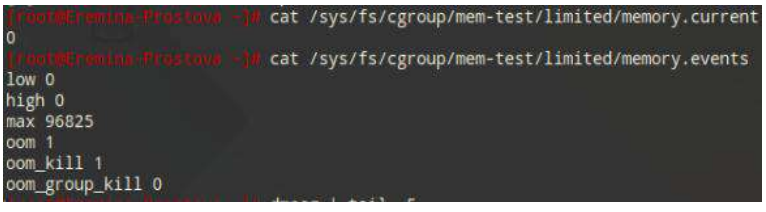


**Рис. 6. memory.events: oom=1, oom_kill=1 – OOM Killer сработал при лимите 50 МБ**

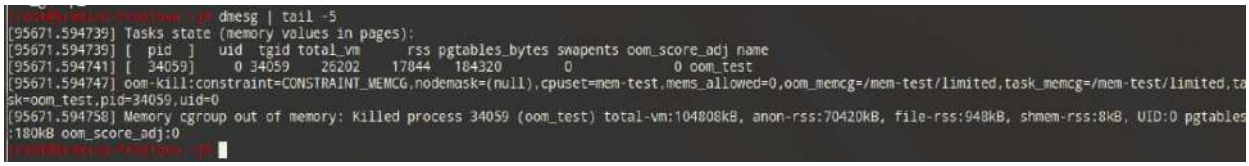


**Рис. 7. dmesg: сообщение ядра о завершении процесса oom_test (pid=34059)**

Аналогично провели еще два эксперимента. Второй – без лимита с штатным завершением, третий – с соблюдением лимита и штатным завершением. Результаты отмечены в таблице 3.

**Сценарий смешанной нагрузки**

```
# Создание изолированных групп для приоритетного и фонового процессов
mkdir /sys/fs/cgroup/cpu-isolation/priority
mkdir /sys/fs/cgroup/cpu-isolation/background
echo "0" > /sys/fs/cgroup/cpu-isolation/priority/cpuset.cpus
echo "1" > /sys/fs/cgroup/cpu-isolation/background/cpuset.cpus

# Приоритетная нагрузка (CPU) → ядро 0
stress -c 1 &
echo $! > /sys/fs/cgroup/cpu-isolation/priority/cgroup.procs

# Фоновая нагрузка (I/O) → ядро 1
dd if=/dev/urandom of=/dev/null &
echo $! > /sys/fs/cgroup/cpu-isolation/background/cgroup.procs
```


**Рис. 8. htop: приоритетный stress на ядре 0 и фоновый dd на ядре 1 – без конкуренции**

## Результаты и обсуждение

### Изоляция CPU (cpuset)

Контроллер cpuset обеспечивает жёсткую детерминированную изоляцию. Процессы группы не используют ядра, назначенные дргуим группам, несмотря на характер и интенсивность нагрузки. Значение PSR всегда соответствовало cpuset.cpus, а низкое значение nr_switches (4 переключения контекста) подтвердило отсутствие миграций. Следует отметить, что данный результат получен на двухъядерной системе. Применимость к многоядерным платформам и средам с NUMA требует отдельного экспериментального обоснования.

В отличие от cpu.weight (пропорциональное распределение при конкуренции за общие ядра), cpuset даёт абсолютную изоляцию. То есть, если ядро 0 простаивает, процесс из group_B не может его использовать, даже при перегрузке ядра 1. Это компромисс между гибкостью использования ресурсов и детерминизмом изоляции. Сводные результаты сценария смешанной нагрузки представлены в таблице 2.

**Таблица 2 – Результаты сценария смешанной нагрузки**

| Процесс | Группа cgroup | Ядро CPU | Загрузка ядра | Конкуренция |
|---|---|---|---|---|
| stress -c 1 | priority | 0 | 100% | Отсутствует |
| dd (I/O) | background | 1 | 100% | Отсутствует |
| stress -c 2 (контроль) | – (без cgroups) | 0 или 1 | ~50% каждый | Присутствует |

Результаты показали, что nr_switches процесса stress в группе priority составил 3 переключения контекста за период измерения, что сопоставимо с 4 при изолированном тесте. Это

подтверждает отсутствие конкуренции процесса stress с фоновым процессом dd за CPU и служит дополнительной верификацией в сценарии смешанной нагрузки.

**Изоляция памяти (memory.max)**

Серия экспериментов подтвердила детерминированность срабатывания OOM Killer при превышении жесткого лимита. Сводные результаты представлены в таблице 3.

**Таблица 3 – Сводные результаты серии экспериментов на memory.max в cgroups v2**

| Эксперимент | Лимит | Запрос | Факт. выделено | Статус процесса | OOM Killer |
|---|---|---|---|---|---|
| Эксп. 1 (превышение) | 50 МБ | 100 МБ | ~69 МБ | Завершён ядром | Активирован (oom_kill=1) |
| Эксп. 2 (без лимита) | max (нет) | 300 МБ | 300 МБ | Штатное завершение | Не активирован |
| Эксп. 3 (соблюдение) | 500 МБ | 400 МБ | ~388 МБ | Штатное завершение | Не активирован (max=0) |

Отключение swap (memory.swap.max = 0) обязательно, так как иначе процесс может превысить жёсткий лимит через файл подкачки, что делает невозможным детерминизм изоляции.

***Подтверждение гипотез H1a и H1b***

Гипотезы H1a и H1b полностью подтверждены. Для гипотезы H1a:

1. PSR всегда совпадал с cpuset.cpus (100% наблюдений);
2. nr_switches – 4 (изолированный тест) и 3 (смешанная нагрузка), что укладывается в критерий ≤ 20% отклонения.

Для гипотезы H1b:

1. при превышении лимита 50 МБ – oom_kill=1 в memory.events, подтверждено в dmesg;
2. процессы других групп не завершались принудительно. В контрольном эксперименте без cgroups процессы распределялись произвольно, что подтверждает необходимость детерминированных механизмов.

**Соответствие требованиям регуляторов**

**Требования ФСТЭК России**

Необходимо отметить, что в данной работе cgroups применяются без полноценной контейнерной изоляции (без namespace). Приказ ФСТЭК № 118 регулирует средства контейнеризации. Применение cgroups как самостоятельного механизма реализует требования о разделении ресурсов, но не заменяет полноценной namespace-изоляции. Полное соответствие требованиям достигается при совместном использовании cgroups v2 и namespaces. Приказ ФСТЭК России № 118 [13] устанавливает обязательное разделение процессов, обрабатывающих данные различных уровней конфиденциальности, с целью предотвращения несанкционированного доступа через разделяемые вычислительные ресурсы. Механизм cpuset реализует это требование. Например, процессы разных уровней доверия закрепляются за непересекающимися наборами ядер CPU. Это обеспечивает контролируемое закрепление процессов за наборами ядер в проверенной конфигурации, но не исключает все классы атак по сторонним каналам.

**Требования PCI DSS 4.0**

В контексте изоляции на уровне ядра релевантен также ГОСТ Р 59006-2020 «Защита информации. Доверенная загрузка» [14], регламентирующий контроль среды выполнения на уровне ядра ОС. Механизмы cgroups v2 могут рассматриваться как элемент обеспечения контролируемой среды выполнения, предусмотренной данным стандартом. Кроме того, управление вычислительными ресурсами регламентируется профилями защиты операционных систем в рамках оценочных уровней доверия (ОУД) по методологии ФСТЭК России, применяемой при сертификации средств защиты информации. Стандарт PCI DSS (версия 4.0, требования 6.3.3 и раздел A1 о многопользовательских средах) [15] требует защиты приложений, обрабатывающих данные платежных карт, от атак «исчерпание ресурсов» (resource exhaustion). Механизм memory.max обеспечивает эту защиту. Этот параметр задаёт жёсткий лимит. Он изолирует последствия утечки памяти или намеренного DoS в одном сервисе: OOM Killer завершает только процесс-нарушитель внутри его cgroup. Счетчик memory.events.max ведёт аудит

попыток исчерпания ресурсов, соответствуя требованиям PCI DSS к логированию инцидентов безопасности.

**Заключение**

Проведённые эксперименты подтверждают работоспособность и практическую пригодность механизмов cgroups v2 в составе ОС Альт. В качестве дальнейших шагов можно выделить измерение латентности с помощью lmbench или perf stat, сравнение поведения cpu.weight и cpuset, а также интеграцию cgroups v2 с пространствами имён для построения полноценной контейнерной изоляции.

Контроллер cpuset обеспечивает жёсткое закрепление процессов за выделенными ядрами CPU. Число переключений контекста nr_switches составило 4, миграции между ядрами отсутствуют. Контроллер memory.max также действует предсказуемо: при превышении лимита активируется OOM Killer (oom_kill=1), причём работа других групп не нарушается.

Гипотезы H1a и H1b полностью подтверждены. В рамках H1a процессы, ограниченные cpuset, ни при одном из приведенных сценариев не выполнялись на ядрах вне своего cpuset. В рамках H1b процессы, превысившие лимит memory.max, завершались OOM Killer до того, как нехватка памяти начинала влиять на остальные группы.

Таким образом, cgroups v2 имеет принципиальные преимущества перед традиционными механизмами (nice, cpulimit, ulimit) в средах с высокими требованиями к предсказуемости. Исследованные механизмы могут использоваться как часть более широкого комплекса мер для поддержки отдельных требований по разделению процессов и защите от исчерпания ресурсов; вывод о нормативном соответствии требует отдельной матрицы требований и процедуры оценки.

**Практические рекомендации:**

- cpuset необходим для жёсткого закрепления критических процессов за выделенными ядрами, особенно в системах реального времени и при обработке конфиденциальных данных;
- memory.max совместно с memory.swap.max=0 используется для детерминированного срабатывания OOM Killer и защиты от resource exhaustion;
- при планировании изоляции учитывать ограничение CONFIG_RT_GROUP_SCHED: CPU-контроллер (cpu.weight) недоступен при наличии RT-процессов ядра.

**Список литературы**

1. Носенко, Д. И. Сетевое и системное администрирование. Подготовка к Демонстрационному экзамену КОД 09.02.06-1-2025 : практикум / Д. И. Носенко, А. П. Золотарёв, А. Г. Уймин [и др.]. – Саратов : Профобразование, 2025. – 216 с. – ISBN 978-5-4488-2908-6. – URL: https://www.iprbookshop.ru/159510.html (дата обращения: 13.05.2025). – Текст : электронный.
2. Уймин, А. Г. Моделирование телекоммуникационной сети средствами сетевых инструментов Linux: инструменты создания цифровых двойников / А. Г. Уймин, О. Р. Никитин // I-methods. – 2023. – Т. 15, № 2. – EDN NFJDVH.
3. Nice and Renice Command in Linux [Электронный ресурс] // GeeksforGeeks. – 2025. – URL: https://www.geeksforgeeks.org/linux-unix/nice-and-renice-command-in-linux-with-examples/ (дата обращения: 14.05.2025). – Текст : электронный.
4. Cpulimit: CPU usage limiter for Linux [Электронный ресурс] // GitHub. – 2025. – URL: https://github.com/opsengine/cpulimit (дата обращения: 14.05.2025). – Текст : электронный.
5. Ulimit(3) – Linux manual page [Электронный ресурс] // man7.org. – 2025. – URL: https://man7.org/linux/man-pages/man3/ulimit.3.html (дата обращения: 14.05.2025). – Текст : электронный.
6. Control Group v2 – The Linux Kernel documentation [Электронный ресурс] // The Linux Kernel. – 2025. – URL: https://docs.kernel.org/admin-guide/cgroup-v2.html (дата обращения: 14.05.2025). – Текст : электронный.

7. Conway, J. Re: Getting out ahead of OOM [Электронный ресурс] / J. Conway // pgsql-admin Mailing List. – 2025. – URL: https://postgrespro.com/list/id/e52f00fc-ffe3-4745-8082-b492f44c6693@joeconway.com (дата обращения: 14.05.2025). – Текст : электронный.
8. Namespaces (7) – Linux manual page [Электронный ресурс] // man7.org. – 2024. – URL: https://man7.org/linux/man-pages/man7/namespaces.7.html (дата обращения: 14.05.2025). – Текст : электронный.
9. Wiedner, F. Control Groups Added Latency in NFVs: An Update Needed? / F. Wiedner, A. Daichendt, J. Andre, G. Carle // 2023 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). – 2023. – P. 40–45. – DOI: 10.1109/NFV-SDN59219.2023.10329612.
10. systemd.resource-control — Resource Control Unit Settings [Электронный ресурс] // freedesktop.org. – 2025. – URL: https://www.freedesktop.org/software/systemd/man/latest/systemd.resource-control.html (дата обращения: 30.09.2025). – Текст : электронный.
11. Крайнов, П. А. Мониторинг системы контрольных групп процессов в ОС Альт / П. А. Крайнов, С. А. Пашина // Всероссийский сборник статей и публикаций института развития образования, повышения квалификации и переподготовки. – 2025. – URL: https://ropkip.ru/publication/415456 (дата обращения: 14.05.2025). – Текст : электронный.
12. ОС Альт – Документация [Электронный ресурс] // BaseALT. – 2025. – URL: https://docs.altlinux.org/ru-RU/index.html (дата обращения: 14.05.2025). – Текст : электронный.
13. Об утверждении Требований по безопасности информации к средствам контейнеризации : приказ ФСТЭК России от 4 июля 2022 г. № 118 [Электронный ресурс] // ГАРАНТ. – 2022. – URL: https://base.garant.ru/405955413/ (дата обращения: 14.05.2025). – Текст : электронный.
14. ГОСТ Р 59006-2020. Защита информации. Доверенная загрузка. Термины и определения [Электронный ресурс]. – М. : Стандартинформ, 2020. – URL: https://docs.cntd.ru/document/1200174521 (дата обращения: 13.05.2025). – Текст : электронный.
15. PCI Data Security Standard (PCI DSS) version 4.0.1 [Электронный ресурс] // PCI Security Standards Council. – 2024. – URL: https://www.pcisecuritystandards.org/document_library/ (дата обращения: 13.05.2025). – Текст : электронный.

**References**

1. Nosenko, D. I., Zolotarev, A. P., & Uymin, A. G. (2025). *Setevoe i sistemnoe administrirovanie. Podgotovka k Demonstratsionnomu ekzamenu KOD 09.02.06-1-2025* [Network and system administration: Practicum]. Profobrazovanie. ISBN 978-5-4488-2908-6. Retrieved May 13, 2025, from https://www.iprbookshop.ru/159510.html (accessed: 13.05.2025).
2. Uymin, A. G., & Nikitin, O. R. (2023). Modeling a telecommunication network using Linux network tools: Digital twin creation tools. *I-Methods*, *15*(2). EDN NFJDVH.
3. Nice and Renice Command in Linux. (2025). *GeeksforGeeks*. Retrieved May 14, 2025, from https://www.geeksforgeeks.org/linux-unix/nice-and-renice-command-in-linux-with-examples/ (accessed: 14.05.2025).
4. Cpulimit: CPU usage limiter for Linux. (2025). *GitHub*. Retrieved May 14, 2025, from https://github.com/opsengine/cpulimit (accessed: 13.05.2025).
5. Ulimit(3) – Linux manual page. (2025). *man7.org*. Retrieved May 14, 2025, from https://man7.org/linux/man-pages/man3/ulimit.3.html (accessed: 14.05.2025).
6. Control Group v2 – The Linux Kernel documentation. (2025). *The Linux Kernel*. Retrieved May 14, 2025, from https://docs.kernel.org/admin-guide/cgroup-v2.html (accessed: 14.05.2025).
7. Conway, J. (2025). Re: Getting out ahead of OOM. *pgsql-admin Mailing List*. Retrieved May 14, 2025, from https://postgrespro.com/list/id/e52f00fc-ffe3-4745-8082-b492f44c6693@joeconway.com (accessed: 14.05.2025).

8. Namespaces (7) – Linux manual page. (2024). *man7.org*. Retrieved May 14, 2025, from https://man7.org/linux/man-pages/man7/namespaces.7.html (accessed: 14.05.2025).
9. Wiedner, F., Daichendt, A., Andre, J., & Carle, G. (2023). Control groups added latency in NFVs: An update needed? In *2023 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)* (pp. 40–45). IEEE. DOI: 10.1109/NFV-SDN59219.2023.10329612.
10. freedesktop.org. (2025). systemd.resource-control — Resource control unit settings. Retrieved September 30, 2025, from https://www.freedesktop.org/software/systemd/man/latest/systemd.resource-control.html (accessed: 30.09.2025).
11. Kraynov, P. A., & Pashina, S. A. (2025). Monitoring the system of control groups of processes in the Alt OS. *All-Russian Collection of Articles and Publications of the Institute for Education Development*. Retrieved May 14, 2025, from https://ropkip.ru/publication/415456 (accessed: 14.05.2025).
12. ALT Linux – Documentation. (2025). *BaseALT*. Retrieved May 14, 2025, from https://docs.altlinux.org/ru-RU/index.html (accessed: 14.05.2025).
13. Federal Service for Technical and Export Control of Russia. (2022). *Ob utverzhdenii Trebovaniy po bezopasnosti informatsii k sredstvam konteynerizatsii* [On approval of Information Security Requirements for Containerization Tools] (Order No. 118). GARANT. Retrieved May 14, 2025, from https://base.garant.ru/405955413/ (accessed: 14.05.2025).
14. GOST R 59006-2020. (2020). *Zashchita informatsii. Doverennaya zagruzka. Terminy i opredeleniya* [Information protection. Trusted boot. Terms and definitions]. Standartinform. Retrieved May 14, 2025, from https://docs.cntd.ru/document/1200174521 (accessed: 13.05.2025).
15. PCI Security Standards Council. (2024). *PCI Data Security Standard (PCI DSS) version 4.0.1*. Retrieved May 14, 2025, from https://www.pcisecuritystandards.org/document_library/ (accessed: 13.05.2025).

### Информация об авторах

Еремина Елизавета Алексеевна – студент ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация, e-mail: elizavetaeremina2@gmail.com

Простова Алиса Никитична – студент ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация, e-mail: prostrova2005@bk.ru

## PROCESS ISOLATION IN THE ALT OPERATING SYSTEM USING CGROUPS V2 MECHANISMS

**Eremina E. A.[1], Prostova A. N.[1]**

[1]National University of Oil and Gas «Gubkin University»

**Abstract.** *The problem of computational resource management in multitasking operating systems is of particular importance in the context of widespread use of containerization environments and the need to simultaneously execute tasks with different resource requirements. Traditional Linux mechanisms – nice, cpulimit, and ulimit – do not provide comprehensive group isolation: nice sets only a recommended priority, cpulimit uses SIGSTOP/CONT signals, and ulimit restricts only the user session. The workload interference problem, known as the 'noisy neighbor' effect, remains a key challenge in system administration and information security. Purpose: experimental quantitative evaluation of the efficiency of cgroups v2 resource isolation mechanisms in the Alt Linux distribution (kernel 6.1.115). Methods. To verify hypotheses H1a (CPU isolation) and H1b (memory isolation), a series of controlled experiments was conducted using cpuset and memory.max controllers with varying workloads: 1–2 cores, memory limits of 50–500 MB, and a mixed scenario with priority and background processes. Measured metrics: PSR (CPU core affinity), nr_switches (context switches), memory.events (OOM Killer counters). Results. With cpuset, PSR matched the assigned core in 100% of observations; nr_switches was 3–4 without cross-group migration. When the memory.max limit was exceeded, the OOM Killer mechanism (oom_kill=1) was activated without affecting other groups. Conclusions. Hypotheses H1a and H1b are fully confirmed. Cgroups v2 demonstrates significant advantages over traditional isolation mechanisms. The mechanisms may support selected resource-separation and exhaustion-control requirements; however, the experiment alone does not demonstrate full compliance with FSTEC Order No. 118 or PCI DSS 4.0.*

**Keywords:** *cgroups v2; process isolation; cpuset; memory.max; resource management; containerization; information security.*

## Information about the authors

Eremina Elizaveta Alekseevna – student Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: elizavetaeremina2@gmail.com

Prostova Alisa Nikitichna – student Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: prostrova2005@bk.ru

**Рубрика 1. Информатика и информационные процессы**

***УДК** 004.056*

***Д. Д. Новикова, С. С. Чурсина***
*ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация*

# СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЭФФЕКТИВНОСТИ СРЕДСТВ МОНИТОРИНГА БЕЗОПАСНОСТИ В РЕАЛЬНОМ ВРЕМЕНИ НА БАЗЕ ПОДСИСТЕМЫ АУДИТА ЯДРА И eBPF-ТЕХНОЛОГИЙ В ОС АЛЬТ

**Аннотация.** *В статье представлены результаты сравнительного экспериментального исследования средств мониторинга безопасности в реальном времени в среде отечественной операционной системы Альт Linux. Рассматриваются два принципиально различных подхода к обнаружению угроз: классическая подсистема аудита ядра auditd и современный eBPF-инструмент Tracee. В ходе работы выявлены практические ограничения совместимости популярных eBPF-решений с виртуализированными средами, а также проведена количественная оценка накладных расходов инструментов на производительность системы и их способности обнаруживать типовые сценарии аномальной активности. Экспериментальная среда развёрнута на базе ОС Альт Рабочая Станция 11.1 с применением гипервизора VirtualBox. Материал будет полезен специалистам по защите информации, системным администраторам и разработчикам защищённых конфигураций на базе отечественных дистрибутивов Linux.*

**Ключевые слова:** *мониторинг безопасности, защита в процессе выполнения, auditd, eBPF, Tracee, ОС Альт Linux, системные вызовы, обнаружение аномалий, производительность ядра, виртуализация.*

## Введение

Обеспечение безопасности операционных систем в условиях непрерывного усложнения ландшафта угроз становится одной из ключевых задач современной информационной безопасности. Традиционные средства защиты, ориентированные на предотвращение вторжений на периметре, всё чаще оказываются недостаточными: злоумышленники, получившие доступ к системе, способны действовать незаметно на протяжении длительного времени. В этом контексте особую значимость приобретают инструменты защиты в процессе выполнения – средства мониторинга поведения системы в режиме реального времени, способные обнаруживать аномальную активность непосредственно в момент её возникновения [1].

Актуальность настоящей работы определяется несколькими факторами. Во-первых, отечественные операционные системы, в частности ОС Альт Linux, получают всё более широкое распространение в государственных и корпоративных информационных системах, однако вопросы применимости современных инструментов мониторинга безопасности в реальном времени в данной среде остаются недостаточно изученными. Во-вторых, существующие сравнительные исследования инструментов этого класса, как правило, ориентированы на контейнеризированные и облачные среды, тогда как поведение данных инструментов на традиционных рабочих станциях под управлением отечественных дистрибутивов Linux практически не исследовано [2]. В-третьих, выбор между инструментами с принципиально различными архитектурами – классическим аудитом на уровне ядра и современными eBPF-решениями – требует эмпирического обоснования, поскольку теоретические характеристики инструментов не всегда соответствуют их поведению в конкретной операционной среде.

Новизна работы заключается в том, что впервые выполнено прямое экспериментальное сравнение классического аудита ядра и eBPF-инструмента в среде отечественной ОС Альт Linux с применением виртуализации, а также эмпирически установлены границы совместимости современных eBPF-решений (Falco, Sysdig) с конфигурацией «VirtualBox + ядро 6.12». В отличие от существующих обзоров, фокусирующихся на контейнерных средах, данное исследование рассматривает поведение инструментов мониторинга на традиционной рабочей станции, что приближает результаты к реальным условиям эксплуатации автоматизированных рабочих мест.

## Теоретические основы мониторинга безопасности в реальном времени

Защита в процессе выполнения (runtime security) – это процесс защиты приложений, рабочих нагрузок и систем от угроз непосредственно во время их активного выполнения. Принципиальное отличие данного подхода от статических методов безопасности состоит в том, что

он направлен не на предотвращение уязвимостей на этапе разработки или развёртывания, а на обнаружение и нейтрализацию угроз, которые проявляются исключительно в процессе работы системы. Сканирование образов и статический анализ кода не способны выявить атаки, эксплуатирующие легитимные системные механизмы уже после запуска приложения – именно этот пробел закрывает мониторинг безопасности в реальном времени.

Все взаимодействия между пользовательскими приложениями и ресурсами операционной системы осуществляются через системные вызовы – интерфейс, предоставляемый ядром Linux. Поскольку практически любое действие в операционной системе сопровождается обращением к ядру через системный вызов, именно этот уровень является оптимальной точкой наблюдения для инструментов мониторинга безопасности: здесь можно отследить открытие файлов, запуск процессов, сетевые соединения и изменения прав доступа. Современные инструменты используют два основных механизма перехвата: подсистему аудита ядра, встроенную в Linux, и технологию eBPF.

Подсистема аудита (auditd) присутствует в ядре Linux начиная с версии 2.6. Архитектурно она состоит из модуля в пространстве ядра, перехватывающего системные вызовы, и демона в пространстве пользователя, который получает события через сетевой интерфейс ядра и записывает их в журнал [4]. Правила аудита задаются с помощью утилиты auditctl и определяют, какие именно события подлежат регистрации. Важной характеристикой auditd является детерминированность: инструмент регистрирует ровно те события, для которых явно определены правила, что обеспечивает предсказуемость и минимальный уровень ложных срабатываний.

Технология eBPF (Extended Berkeley Packet Filter) встроена в ядро Linux и позволяет безопасно исполнять пользовательские программы непосредственно в ядре без изменения его исходного кода и без загрузки дополнительных модулей [3]. Программа, написанная на подмножестве языка C, компилируется в eBPF-байткод, проходит строгую верификацию и после динамической компиляции исполняется с производительностью, близкой к нативному коду ядра. Ключевым преимуществом eBPF перед традиционными модулями ядра является безопасность: верификатор полностью исключает некорректное поведение программы до её запуска, тогда как ошибочный модуль ядра способен вызвать критический сбой системы [5].

Для присоединения eBPF-программ к точкам перехвата используется механизм kprobes (Kernel Probes), появившийся в ядре версии 2.6.9. При регистрации kprobe сохраняется копия целевой инструкции, а её первый байт заменяется инструкцией программного прерывания. Когда процессор достигает этой точки, управление передаётся обработчику eBPF-программы. Такая схема позволяет динамически устанавливать точки перехвата практически в любом месте кода ядра без его перекомпиляции и перезагрузки системы.

Важным дополнением к eBPF является механизм CO-RE (Compile Once – Run Everywhere), реализованный через BTF (BPF Type Format). CO-RE позволяет скомпилировать eBPF-программу один раз и запускать её на любом совместимом ядре без повторной компиляции, что существенно упрощает распространение инструментов.

Среди инструментов мониторинга безопасности, использующих eBPF, можно выделить Falco и Tracee. Falco – инструмент обнаружения угроз, разработанный компанией Sysdig и переданный в CNCF (Cloud Native Computing Foundation) [6]. Tracee – инструмент безопасности в процессе выполнения от компании Aqua Security, использующий сигнатурный подход к обнаружению аномалий [7]. Сигнатуры Tracee описывают поведенческие паттерны, характерные для известных техник атак, и организованы в соответствии с базой знаний MITRE ATT&CK.

Рассмотренные инструменты реализуют два принципиально различных подхода к обнаружению аномалий. Первый подход – явный мониторинг на основе правил – представлен auditd. Его эффективность прямо зависит от полноты и корректности настроенных правил. Второй подход – поведенческое обнаружение на основе сигнатур – представлен Tracee. Он не требует ручной настройки правил для каждого типа угроз, однако его действенность определяется актуальностью сигнатурной базы.

**Методология эксперимента**

Для проведения эксперимента была развёрнута среда на базе гипервизора VirtualBox. Тестовая среда представляла собой виртуальную машину под управлением ОС Альт Рабочая Станция 11.1 со следующими характеристиками: оперативная память – 4096 МБ, количество виртуальных процессоров – 4, объём жёсткого диска – 30 ГБ. Использование виртуальной машины обусловлено тем, что контрольные точки VirtualBox позволяют гарантированно восстанавливать систему в исходное состояние между тестами разных инструментов, обеспечивая воспроизводимость условий эксперимента [10].

На подготовительном этапе на виртуальную машину были установлены инструменты нагрузочного тестирования: stress-ng, fio и sysbench. Для автоматизации сбора метрик производительности и фиксации результатов обнаружения были разработаны два сценария командной оболочки: benchmark.sh обеспечивает последовательный запуск нагрузочных тестов и запись результатов в CSV-файл; attack_scenarios.sh выполняет набор сценариев имитации атак и фиксирует факт срабатывания инструмента мониторинга.

Эксперимент состоял из двух независимых блоков. Блок 1 – оценка производительности: измерение накладных расходов инструмента мониторинга на работу системы в штатном режиме без выполнения атак. Измерения проводились по четырём метрикам: загрузка процессора под нагрузкой, скорость случайной записи на диск (IOPS), скорость случайного чтения с диска (IOPS) и задержка запуска процессов. Блок 2 – оценка точности обнаружения: последовательное выполнение десяти сценариев, имитирующих типовые техники атак, с фиксацией факта и времени срабатывания инструмента.

Порядок выполнения экспериментальных прогонов предусматривал последовательный откат к контрольной точке, установку исследуемого инструмента, запуск сценария сбора метрик в трёх режимах (baseline, auditd, tracee), затем запуск сценария attack_scenarios.sh для соответствующего инструмента. Каждый тест повторялся трижды для минимизации случайных отклонений.

**Исследование ограничений совместимости eBPF-стека**

Первоначально в качестве исследуемых инструментов защиты в процессе выполнения были выбраны Falco и Sysdig – наиболее широко применяемые решения данного класса с открытым исходным кодом. Однако в ходе развёртывания экспериментальной среды были выявлены ограничения, обусловленные совместным использованием гипервизора VirtualBox и ОС Альт Linux 11.1 с ядром версии 6.12. Данный этап работы является самостоятельным практически значимым результатом: он позволяет установить границы применимости современных eBPF-решений в виртуализированных средах на базе отечественных дистрибутивов Linux.

Falco версий 0.38.2 и 0.39.2 не запустился ни в одном из доступных режимов работы. Режим modern eBPF завершался с ошибкой инициализации: в гостевой ОС под управлением VirtualBox тип eBPF-программ BPF_PROG_TYPE_TRACING, необходимый для работы данного режима, оказался недоступен. Аналогичная проблема зафиксирована в баг-трекере проекта применительно к различным дистрибутивам Linux [8]. Компиляция legacy eBPF-пробы оказалась невозможной из-за отсутствия пакета kernel-devel для ядра 6.12 в репозитории ОС Альт Linux. Sysdig, доступный в репозитории ALT Linux, также несовместим с ядром 6.12: загрузка драйвера завершается ошибкой, сборка модуля через DKMS невозможна [6].

В связи с изложенным итоговый эксперимент представляет собой сравнение двух инструментов с принципиально различной архитектурой: auditd, встроенного в ядро и не требующего внешних зависимостей, и Tracee – eBPF-инструмента, развёртываемого в виде контейнера и использующего механизм CO-RE. Наличие BTF-информации в директории /sys/kernel/btf/vmlinux на тестовой машине было подтверждено, что сделало возможным запуск Tracee в условиях ограниченной поддержки eBPF со стороны гипервизора.

**Сценарии имитации атак**

Для оценки способности инструментов обнаруживать подозрительную активность был сформирован набор из десяти сценариев, охватывающих основные классы аномального поведения на уровне операционной системы. При формировании набора авторы опирались на

матрицу MITRE ATT&CK, представляющую собой общепризнанную базу знаний о тактиках и техниках злоумышленников [9]. Перечень сценариев представлен в таблице 1.

**Таблица 1 – Перечень сценариев имитации атак**

| № | Идентификатор | Описание |
|---|---|---|
| 1 | write_etc_passwd | Запись в файл /etc/passwd |
| 2 | read_etc_shadow | Чтение файла /etc/shadow |
| 3 | suid_bit_set | Установка SUID-бита на исполняемый файл |
| 4 | write_dev_shm | Создание файла в разделяемой памяти /dev/shm |
| 5 | netcat_listen | Открытие сетевого порта через netcat |
| 6 | process_spawn | Массовый запуск дочерних процессов |
| 7 | chmod_system_file | Изменение прав доступа к системному файлу |
| 8 | crontab_modification | Добавление записи в crontab |
| 9 | network_download | Загрузка файла из сети через curl |
| 10 | read_ssh_key | Попытка чтения приватного SSH-ключа |

Каждый сценарий выполнялся в интерактивном режиме: после выполнения команды предусматривалась пауза в 10 секунд для наблюдения за реакцией инструмента, после чего результат фиксировался вручную в файл detection_results.csv.

**Результаты тестирования производительности**

Эксперимент по оценке производительности проводился поэтапно: сначала на чистой системе без инструментов мониторинга (режим baseline), затем последовательно с запущенным auditd и Tracee. В качестве итоговых значений использовалось среднее арифметическое трёх прогонов.

Первый прогон выполнялся на машине в исходном состоянии. Вывод терминала в процессе сбора метрик представлен на рисунке 1.

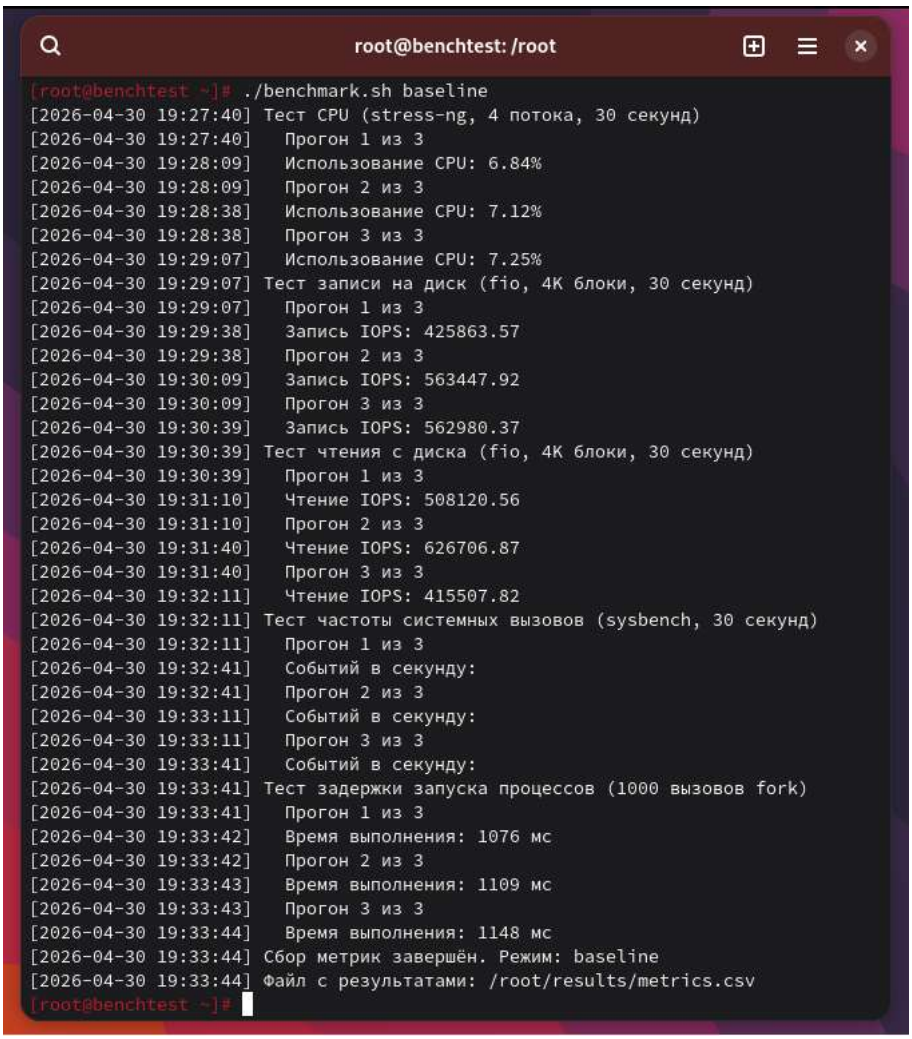


**Рис. 1. Вывод терминала при сборе метрик на чистой системе (baseline)**

Полученные значения зафиксированы в файле metrics.csv и составили: средняя загрузка процессора – 7,07%, средняя скорость записи на диск – 517 431 IOPS, средняя скорость чтения – 516 778 IOPS, суммарное время запуска 1000 дочерних процессов – 1111 мс. Данные значения служат контрольной точкой для расчёта накладных расходов.

Перед началом второго прогона была выполнена проверка состояния службы auditd; результат подтвердил, что служба активна и функционирует штатно (рисунок 2).

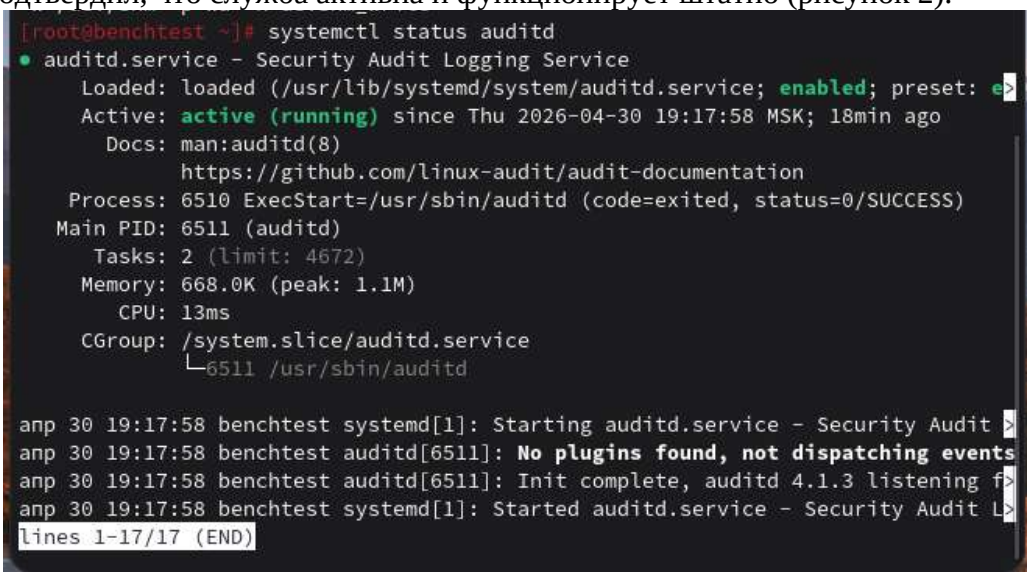


**Рис.2. Статус службы auditd (active/running)**

Процесс сбора метрик в режиме auditd представлен на рисунке 3.

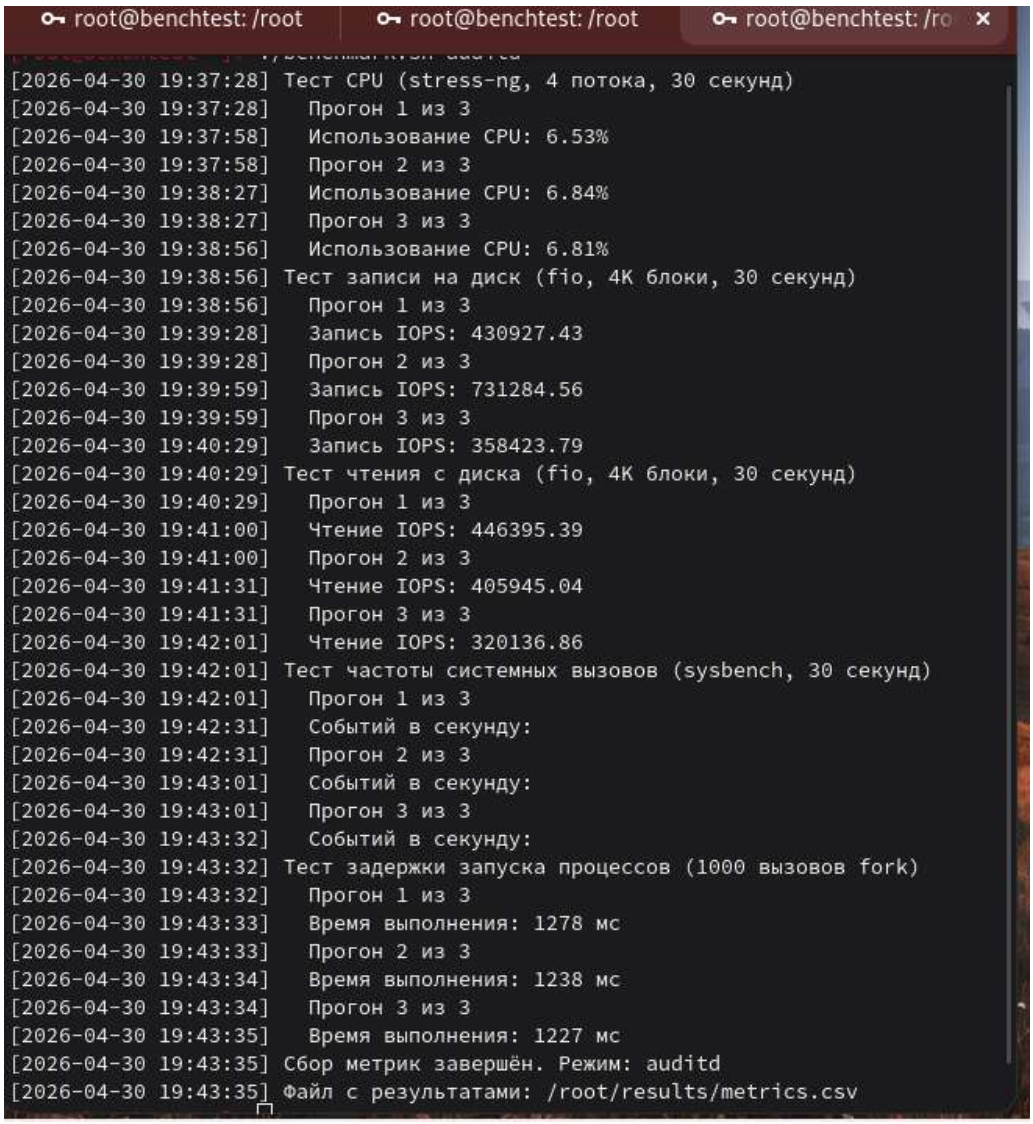


**Рис. 3. Вывод терминала при сборе метрик в режиме auditd**

Средние значения по трём прогонам для auditd составили: загрузка процессора – 6,73%, запись на диск – 506 878 IOPS, чтение с диска – 390 826 IOPS, задержка запуска процессов – 1248 мс. Прослеживается умеренный рост задержки при запуске процессов относительно baseline при практически неизменной загрузке процессора.

Tracee развёртывался как контейнер, факт его работы подтверждался командой docker ps (рисунок 4).


**Рис. 4. Список запущенных контейнеров: Tracee активен**

Для наблюдения за потоком событий в реальном времени использовалась команда docker logs -f tracee. Пример вывода с зафиксированными событиями представлен на рисунке 5.


**Рис. 5. Вывод событий Tracee в режиме реального времени**

Процесс сбора метрик в режиме Tracee представлен на рисунке 6.

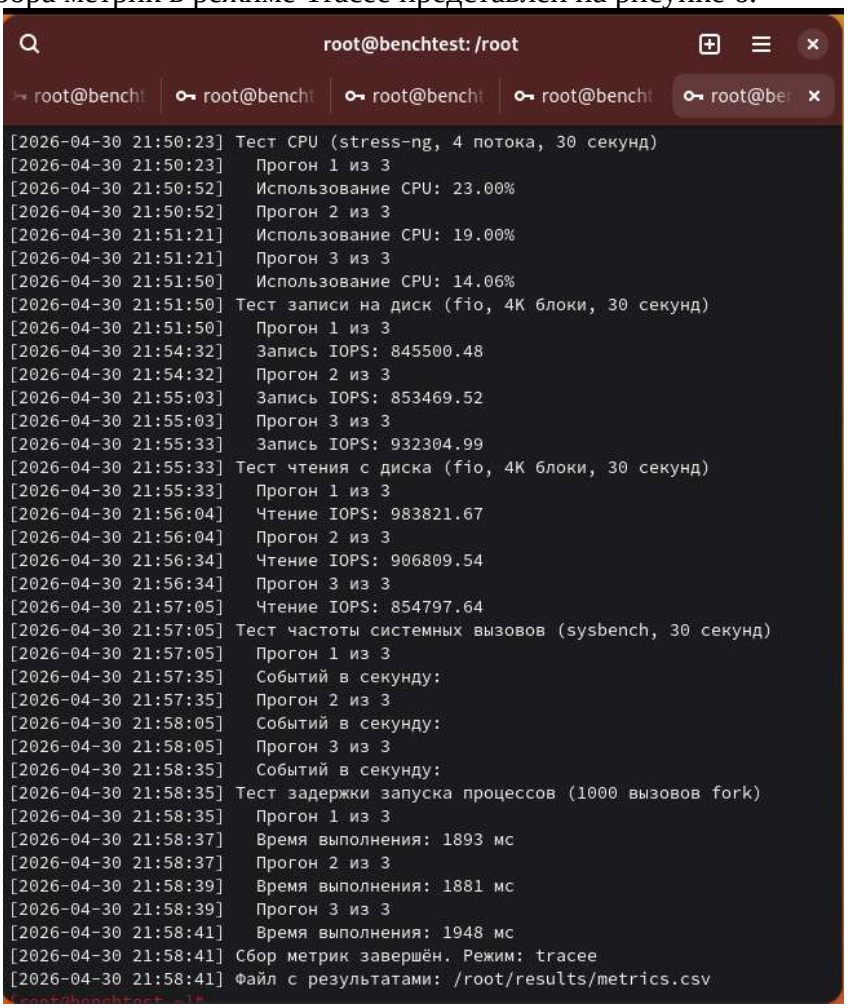


**Рис. 6. Вывод терминала при сборе метрик в режиме Tracee**

Средние значения для Tracee составили: загрузка процессора – 18,69%, запись на диск – 877 092 IOPS, чтение с диска – 915 143 IOPS, задержка запуска процессов – 1907 мс. Значение загрузки процессора существенно превышает как baseline, так и показатели auditd, что объясняется контейнерной архитектурой Tracee: инструмент обрабатывает поток системных вызовов в пространстве пользователя параллельно с основными рабочими процессами.

Сводное сравнение всех трёх режимов по ключевым метрикам представлено на рисунке 7. Накладные расходы каждого инструмента в процентном выражении относительно baseline – на рисунке 8.

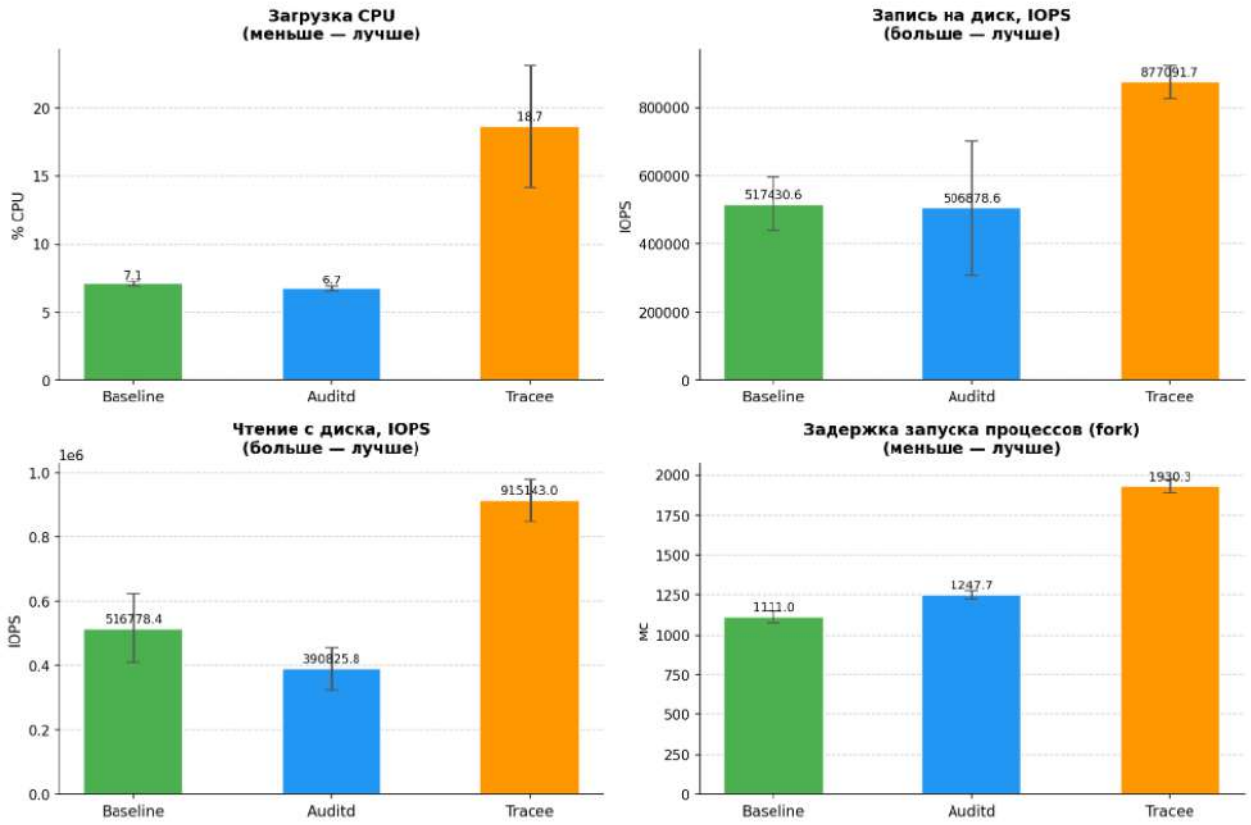


**Рисунок 7 – Сравнение производительности: Baseline, Auditd, Tracee**

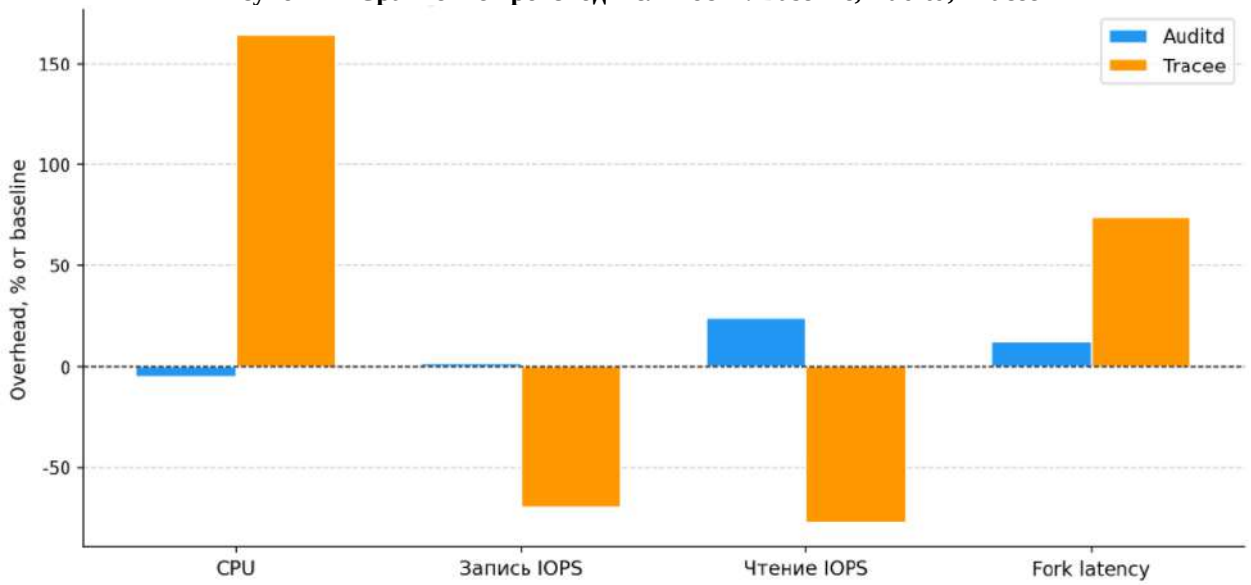


**Рис. 8. Накладные расходы Auditd и Tracee относительно baseline**

Анализ полученных данных позволяет сделать следующие выводы. Auditd практически не оказывает измеримого влияния на загрузку процессора: значение 6,73% незначительно отличается от базового показателя 7,07%, что укладывается в пределы статистического разброса. Tracee, напротив, демонстрирует загрузку процессора на уровне 18,69% – увеличение на 164% относительно baseline, что обусловлено дополнительными накладными расходами на межпроцессное взаимодействие.

Задержка запуска процессов наглядно демонстрирует различие в архитектурах. Baseline: 1111 мс на 1000 процессов. Auditd: 1248 мс (+12,3%). Tracee: 1907 мс (+71,6%). Рост задержки при Tracee объясняется тем, что каждый запуск процесса порождает событие, которое должно быть обработано eBPF-программой и передано в пользовательское пространство контейнера.

Показатели IOPS демонстрируют значительный разброс между прогонами, что является характерной особенностью виртуализированной среды: планировщик ввода-вывода хостовой системы вносит существенную нестабильность в результаты дисковых тестов. По этой причине метрики IOPS в рамках данного эксперимента следует интерпретировать с осторожностью.

**Результаты тестирования обнаружения аномалий**

В ходе тестирования для auditd наблюдение производилось за журналом /var/log/audit/audit.log посредством утилиты ausearch. Результаты представлены на рисунке 9.

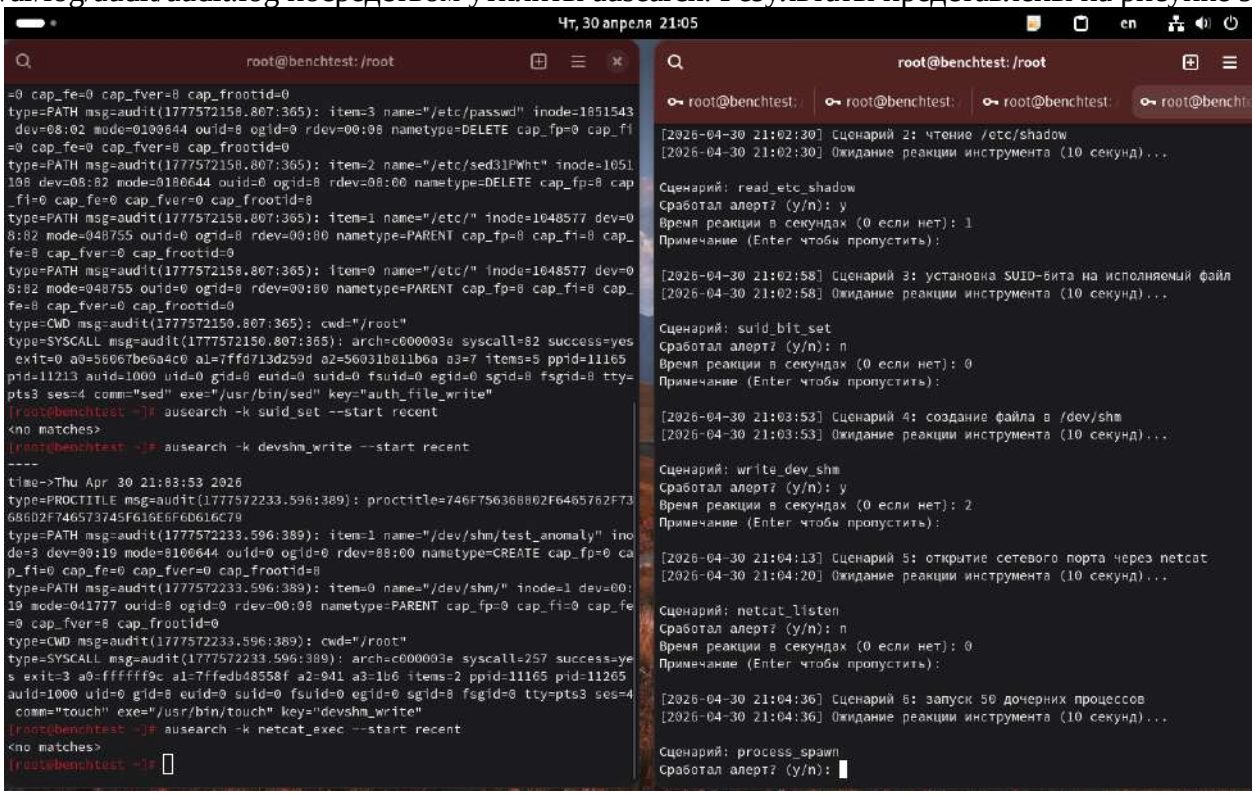


**Рис. 9. Срабатывания auditd при моделировании атак**

Auditd зафиксировал четыре из десяти сценариев: запись в /etc/passwd, чтение /etc/shadow, создание файла в /dev/shm и модификацию crontab. Время реакции во всех случаях составило не более двух секунд. Шесть сценариев не были обнаружены: правило для установки SUID-бита сформулировано некорректно из-за особенностей интерпретации числовых значений утилитой auditctl; сценарии с сетевым взаимодействием не сопроводились срабатыванием из-за отсутствия соответствующих правил; сценарий с чтением SSH-ключа не выполнился, поскольку файл /root/.ssh/id_rsa отсутствовал на тестовой машине.

Для Tracee наблюдение осуществлялось за выводом контейнера командой docker logs -f tracee. Результаты представлены на рисунке 10.

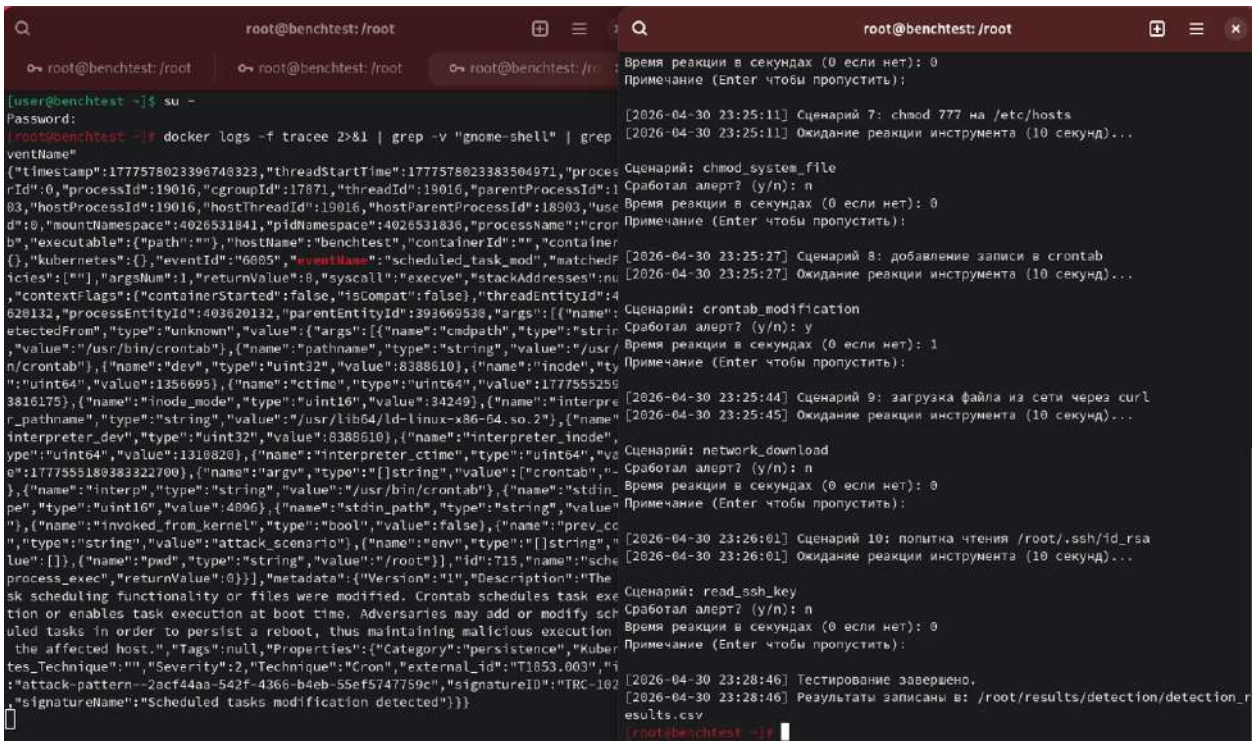


**Рис. 10. Срабатывания Tracee при моделировании атак**

Tracee обнаружил один из десяти сценариев – модификацию crontab. Инструмент идентифицировал событие как scheduled_task_mod с привязкой к технике MITRE ATT&CK T1053.003 и зафиксировал его в течение одной секунды. Низкий показатель объясняется ориентацией инструмента на контейнеризированные рабочие нагрузки: библиотека сигнатур оптимизирована под угрозы, актуальные для контейнерных сред [7]. Модификация системных файлов непосредственно на хосте без контейнерного контекста не попадает в приоритетные сценарии обнаружения по умолчанию.

Сводная матрица обнаружения по всем сценариям и обоим инструментам представлена на рисунке 11.

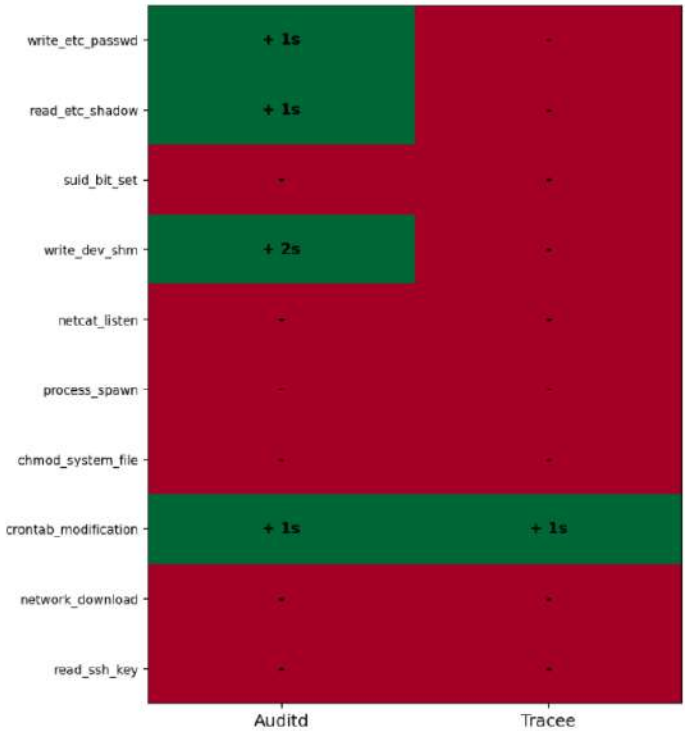


**Рис. 11. Матрица обнаружения атак: Auditd и Tracee**

Обобщённые результаты сравнительного анализа характеристик инструментов сведены в таблицу 2.

**Таблица 2 – Сравнительный анализ характеристик инструментов**

| Характеристика | Auditd | Tracee |
|---|---|---|
| Архитектура перехвата | Подсистема аудита ядра | eBPF (kprobes + tracepoints) |
| Способ развёртывания | Системный пакет (в составе ОС) | Контейнер |
| Настройка обнаружения | Ручная (правила auditctl) | Автоматическая (сигнатуры) |
| Загрузка процессора (отн. baseline) | –4,8% (в пределах погрешности) | +164,4% |
| Задержка процессов (отн. baseline) | +12,3% | +71,6% |
| Обнаружено сценариев (из 10) | 4 | 1 |
| Зависимость от гипервизора | Отсутствует | Частичная (требуется BTF) |

**Выводы**

Проведённое экспериментальное исследование позволило сформулировать ряд обобщающих выводов.

Во-первых, установлено, что современные eBPF-инструменты (Falco, Sysdig) несовместимы с рядом виртуализированных конфигураций на базе отечественных дистрибутивов Linux. Проблема обусловлена ограниченной поддержкой типов eBPF-программ со стороны гипервизора VirtualBox в сочетании с отсутствием необходимых заголовочных файлов ядра новых версий в репозиториях ОС Альт. Данный результат имеет самостоятельную практическую значимость для проектирования защищённых сред.

Во-вторых, сравнительное тестирование auditd и Tracee показало, что классическая подсистема аудита ядра обеспечивает минимальное влияние на производительность системы (прирост задержки процессов около 12%) и высокую предсказуемость обнаружения при условии корректной настройки правил. Современный eBPF-инструмент Tracee демонстрирует значительные накладные расходы (прирост загрузки процессора на 164%) и ограниченную применимость вне контейнеризированных сред.

В-третьих, ни один из инструментов не является универсальным решением, одинаково пригодным для всех сценариев применения. Выбор между auditd и eBPF-инструментами должен определяться условиями эксплуатации: для статических рабочих станций с чётко определённой моделью угроз предпочтителен auditd; для динамических контейнеризированных сред с широким спектром угроз целесообразно применять eBPF-инструменты при условии обеспечения совместимости программно-аппаратного стека.

Таким образом, защита рабочих станций на базе отечественных операционных систем требует комбинирования классических механизмов аудита ядра с дополнительными средствами контроля целостности, а внедрение eBPF-инструментов должно предваряться оценкой совместимости конкретных версий гипервизора, ядра и прикладного инструмента.

**Библиографический список**

1. Шаньгин В. Ф. Информационная безопасность компьютерных систем и сетей: учебное пособие. М.: ИД «ФОРУМ»: ИНФРА-М, 2019. 416 с.
2. Таненбаум А. С., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2019. 1120 с.
3. What is eBPF? // eBPF.io. URL: https://ebpf.io/what-is-ebpf/ (дата обращения: 11.05.2025).
4. Linux Audit Documentation // The Linux Kernel Archives. URL: https://docs.kernel.org/audit/ (дата обращения: 11.05.2025).
5. Райс Л. Изучаем eBPF: программирование ядра Linux для наблюдаемости и безопасности. М.: ДМК Пресс, 2023. 350 с.

6. What is Falco? Open source runtime threat detection // Sysdig. URL: https://www.sysdig.com/learn-cloud-native/what-is-falco (дата обращения: 11.05.2025).
7. The Story of Tracee: The Path to Runtime Security Tool // Aqua Security. URL: https://www.aquasec.com/blog/open-source-container-runtime-security/ (дата обращения: 11.05.2025).
8. [Modern eBPF] libpman: tracing program type is not supported // GitHub – falcosecurity/falco, issue #2792. URL: https://github.com/falcosecurity/falco/issues/2792 (дата обращения: 11.05.2025).
9. MITRE ATT&CK Framework // MITRE Corporation. URL: https://attack.mitre.org/ (дата обращения: 11.05.2025).
10. Уймин А. Г. Сетевое и системное администрирование. Демонстрационный экзамен КОД 1.1: учебно-методическое пособие. СПб.: Лань, 2020. 480 с.

**References**

1. Shangin V. F. Information Security of Computer Systems and Networks: A Training Manual. M.: ID «FORUM»: INFRA-M, 2019. 416 p. (In Russ).
2. Tanenbaum A. S., Bos H. Modern Operating Systems. 4th ed. SPb.: Piter, 2019. 1120 p. (In Russ).
3. What is eBPF? Available from: https://ebpf.io/what-is-ebpf/ (accessed: 11.05.2025).
4. Linux Audit Documentation. Available from: https://docs.kernel.org/audit/ (accessed: 11.05.2025).
5. Rice L. Learning eBPF: Programming the Linux Kernel for Enhanced Observability and Security. M.: DMK Press, 2023. 350 p. (In Russ).
6. What is Falco? Open-source runtime threat detection. Available from: https://www.sysdig.com/learn-cloud-native/what-is-falco (accessed: 11.05.2025).
7. The Story of Tracee: The Path to Runtime Security Tool. Available from: https://www.aquasec.com/blog/open-source-container-runtime-security/ (accessed: 11.05.2025).
8. [Modern eBPF] libpman: tracing program type is not supported. Available from: https://github.com/falcosecurity/falco/issues/2792 (accessed: 11.05.2025).
9. MITRE ATT&CK Framework. Available from: https://attack.mitre.org/ (accessed: 11.05.2025).
10. Uimin A. G. Network and System Administration. Demonstration Exam COD 1.1: A Training Manual. SPb.: Lan, 2020. 480 p. (In Russ).

## Информация об авторах

Чурсина Серафима Сергеевна – студент кафедры «Математических методов обеспечения безопасности систем», ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, e-mail: chursinass@mail.ru.

Новикова Дарья Дмитриевна – студент кафедры «Математических методов обеспечения безопасности систем», ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, e-mail: novikova.d@mail.ru


# COMPARATIVE ANALYSIS OF REAL-TIME SECURITY MONITORING TOOLS BASED ON KERNEL AUDIT SUBSYSTEM AND eBPF TECHNOLOGIES IN ALT LINUX


***D. D. Novikova[1], S. S. Chursina[1]***

[1]National University of Oil and Gas «Gubkin University», Moscow, Russian Federation



**Abstract.** *This paper presents the results of a comparative experimental study of real-time security monitoring tools in the Alt Linux operating system environment. Two fundamentally different approaches to threat detection are examined: the classical kernel audit subsystem (auditd) and a modern eBPF-based tool (Tracee). The study identifies practical compatibility limitations of popular eBPF solutions with virtualized environments and provides a quantitative assessment of the performance overhead and anomaly detection capabilities of the tested tools. The experimental environment is deployed on Alt Linux Workstation 11.1 using the VirtualBox hypervisor. The findings contribute to the informed selection of security monitoring tools for domestic Linux distributions.*



**Keywords:** *security monitoring, runtime protection, auditd, eBPF, Tracee, Alt Linux, system calls, anomaly detection, kernel performance, virtualization.*



## Information about the authors

Chursina Serafima Sergeevna – student of the Department of Mathematical Methods of System Safety Assurance, Gubkin Russian State University of Oil and Gas (National University), Moscow, e-mail: chursinass@mail.ru.

Novikova Daria Dmitrievna – student of the Department of Mathematical Methods of System Safety Assurance, Gubkin Russian State University of Oil and Gas (National University), Moscow, e-mail: novikova.d@mail.ru

**ПРОФЕССИОНАЛИТЕТ, 2025, № 3 (007)**

***УДК** 004.451:004.7*

***Э. В. Тищенко[1], В. Ю. Корякин[1]***

*[1]ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация*

# ИССЛЕДОВАНИЕ ФУНКЦИОНАЛА ПО УПРАВЛЕНИЮ СЕТЬЮ ПОСРЕДСТВОМ ПОДСИСТЕМЫ D-BUS В ОС АЛЬТ

**Аннотация.** В статье исследуются возможности управления сетевыми подключениями в операционной системе Альт с использованием универсальной системной шины D-Bus и сервиса NetworkManager. Актуальность работы обусловлена необходимостью построения гибких средств автоматизации сетевой подсистемы, которые способны не только выполнять заранее заданные команды, но и реагировать на изменения состояния интерфейсов в режиме реального времени. Рассмотрены теоретические основы межпроцессного взаимодействия D-Bus, принципы адресации объектов, использование сервисов, методов и сигналов, а также архитектура NetworkManager как основного механизма управления проводными и беспроводными подключениями в современных Linux-дистрибутивах. Отдельное внимание уделено различию между управлением через статические конфигурационные файлы и программным обращением к API системного демона. Практическая часть включает два эксперимента: сравнение опросного Polling-подхода с событийно-ориентированным переключением на резервный канал через D-Bus API и разработку Python-агента, изменяющего правила firewall в зависимости от активного сетевого подключения. Полученные результаты показывают, что событийная модель D-Bus обеспечивает более быструю реакцию на отказ основного интерфейса, снижает необходимость постоянного опроса системы и позволяет создавать автоматизированные средства управления безопасностью. Показано, что такой подход может применяться в учебных стендах, при администрировании рабочих станций и при разработке сервисов мониторинга. Сделан вывод о применимости D-Bus API NetworkManager для построения надежных и расширяемых решений, интегрированных в инфраструктуру ОС Альт.

**Ключевые слова:** D-Bus, ОС Альт, сетевые интерфейсы, сигналы, NetworkManager, API, автоматизация.

## Введение и теоретические основы

Сетевая подсистема является фундаментальным компонентом любой современной операционной системы. Хотя методы управления посредством вызова утилит командной строки надежны, они не позволяют достичь нужной гибкости и оперативности для построения сложных систем автоматизации. В операционных системах семейства Linux, включая ОС Альт, де-факто стандартом для межпроцессного взаимодействия (IPC) является шина сообщений D-Bus. Этот механизм обеспечивает универсальную и масштабируемую связь между системными службами и приложениями [1, с. 325].

В современных дистрибутивах Linux, включая BaseALT, управление сетью осуществляется преимущественно с помощью демона NetworkManager. Он был выбран для исследования, поскольку стал де-факто стандартом и полностью основан на взаимодействии с шиной D-Bus. NetworkManager обеспечивает автоматическое обнаружение сетевых устройств, унифицированное управление проводными и беспроводными подключениями, а его интеграция с D-Bus делает его идеальным инструментом для изучения программного управления сетью [2]. Выбирать etcnet, к примеру, нельзя, так как etcnet является способом управления сетью через статические файлы, а D-Bus используется для динамического управления интерфейсами и событиями через API, а не через прямое редактирование конфигурационных файлов [4]. Далее в эксперименте № 1 будет проведен сравнительный анализ производительности двух автоматизированных методов для подтверждения тезиса. Изучение принципов работы D-Bus API NetworkManager даст понимание того, как реализовывать программное управление сетью, способное реагировать на изменения в системе, а не просто выполнять предопределенные команды [6].

Цель данной статьи – исследование и демонстрация на практике возможности управления сетью в ОС Альт через D-Bus API сервиса NetworkManager.

## Подсистема межпроцессного взаимодействия D-Bus

D-Bus (Desktop Bus) — это система межпроцессного взаимодействия (IPC), являющаяся системной шиной для обмена сообщениями между приложениями. Работа шины зависит от следующих принципов [7]:

– шина: центральный демон, который отвечает за пересылку сообщений между процессами. Существует 2 шины – системная, которая отвечает за коммуникацию системных серверов, и шина сеанса, с помощью которой связываются между собой пользовательские приложения;
– сервисы: сервисами называют приложения, которые регистрируют свое имя при подключении к шине для обеспечения доступа других приложений к ним [5, с. 203];
– объекты и пути: сервисы предоставляют свой функционал и доступ к своим данных по средствам объектов, адресация к которым осуществляется благодаря иерархическим путям;
– интерфейсы: интерфейсы логически группируют методы и сигналы, формируются объектами;
– методы: методами являются функции, которые вызываются клиентами для выполнения какого-либо действия;
– сигналы: сообщения, которые сервис рассылает всем подписчикам для уведомления о событиях [3].

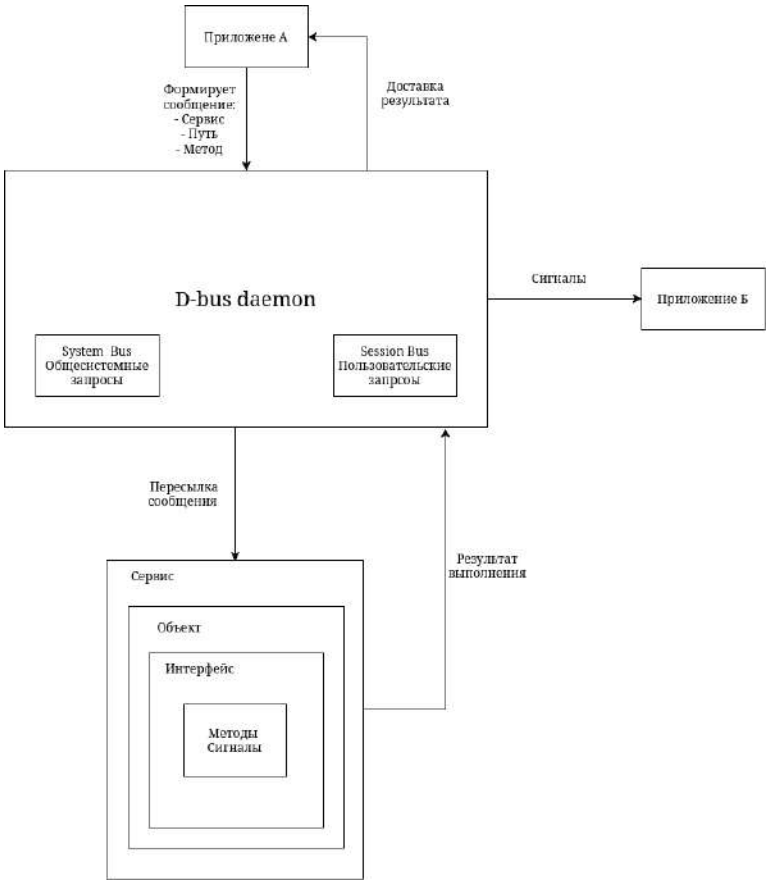


**Рис. 1. Структурная схема IPC**

**Сервис NetworkManager и его интеграция с D-Bus**

NetworkManager – это системный демон, его основной задачей является стандартизация и упрощение процессов настройки и поддержания сетевых соединений. Он автоматически обнаруживает сетевые устройства, управляет профилями подключений и выбирает оптимальное соединение [2].

Полностью раскрыть свой потенциал NetworkManager может через D-Bus. Он регистрируется на системной шине под именем org.freedesktop.NetworkManager и предоставляет набор объектов для управления. Интерфейс D-Bus предоставляет возможность:

– получать список сетевых интерфейсов и их состояние;
– создавать, изменять и удалять настройки/конфигурации подключения к сети;
– активировать и деактивировать соединения;

– подписываться на сигналы об изменении состояния сети, устройствах или точках доступа Wi-Fi.

Иными словами, D-Bus – это не способ обмена информацией, а фундаментальный слой, на котором построена вся современная архитектура управления сетью в ОС Альт.

**Методология исследования**

Объект исследования: подсистема управления сетью операционной системы Альт.

Предмет исследования: функциональные возможности по управлению сетью, предоставляемые через D-Bus API сервиса NetworkManager.

Среда проведения: исследование проводилось на операционной системе BaseALT. В качестве основных инструментов для взаимодействия с D-Bus использовались утилиты командной строки gdbus и busctl, а также язык программирования Python с библиотекой python-dbus для разработки программного агента.

Экспериментальная часть была спланирована таким образом, чтобы последовательно исследовать все аспекты взаимодействия: от изучения структуры API до создания автоматизированного решения. Общий план экспериментов представлен в таблице 1.

**Таблица 1 – Информация о проведенных экспериментах**

| | Цель эксперимента | Средства | Ожидаемый результат |
|---|---|---|---|
| | Автоматическое переключение на резервный канал | NM, nmcli, D-Bus API, Python | Сравнение двух методов переключения, которые реализованы посредством кода |
| | Динамическое изменение контекста безопасности | nmcli, Python, D-Bus API, iptables | Python-агент, меняющий политики безопасности firewall в зависимости от подключения |

**Эксперимент 1. Автоматическое переключение на резервный канал**

Цель данного эксперимента заключается в том, чтобы исследовать и сравнить эффективность двух разных методов реализации: Polling-метода (опросный) и событийно-ориентированного метода (D-Bus) на базе NetworkManager.

Polling-метод.

В основе данного метода лежит периодический опрос состояния сетевого интерфейса посредством утилиты nmcli. Для реализации был написан Bash-скрипт, который через заданный интервал проверяет состояние основного интерфейса и при обнаружении сбоя/отказа инициирует переключение на резервное подключение. Ниже приведен листинг скрипта:

```
#!/bin/bash

CHECK_INTERVAL=1

MAIN_DEV="enp0s3"
MAIN_CON="main-nat"
BACKUP_CON="second-nat"

echo "Запуск мониторинга (Polling). Интервал: ${CHECK_INTERVAL}c"
echo "Нажмите Ctrl+C для остановки"

while true; do
  STATE=$(nmcli -g GENERAL.STATE device show $MAIN_DEV | cut -c 1)
  #echo $STATE

  if [ "$STATE" == '3' ]; then
    echo "[FAIL] Обнаружен отказ $MAIN_DEV в $(date +%H:%M:%S.%N)"
    echo "Переключение на $BACKUP_CON..."

    # Замер времени начала переключения
    start_time=$(date +%s%N)

    nmcli connection up "$BACKUP_CON"
```

```
    end_time=$(date +%s%N)
    echo "Переключение завершено за $(( ($end_time - $start_time) / 1000000 )) мс"

    # Выходим, чтобы не зациклиться, если второй тоже упадет
    break
  fi

  sleep $CHECK_INTERVAL
done
```

Для тестирования необходимо выключить резервный интерфейс. После запускаем скрипт ./polling_test.sh и отключаем основное устройство при выключенном втором nmcli dev disconnect enpos3. Ниже представлен листинг работы скрипта при обнаружении неисправности работы основного интерфейса:

```
[root@rtr test_area]# ./polling_test.sh
Запуск мониторинга (Polling). Интервал: 1с
Нажмите Ctrl+C для остановки
[FAIL] Обнаружен отказ enp0s3 в 16:42:30.635068158
Переключение на second-nat...
Подключение успешно активировано (активный путь D-Bus:
/org/freedesktop/NetworkManager/ActiveConnection/86)
Переключение завершено за 716 мс
```

Из листинга выше видно, что процесс переключения на резервное подключение занял 716 мс.

Событийно-ориентированный метод.

В данном методе используется прямое взаимодействие с NetworkManager через D-Bus. Для проверки работоспособности был написан Python-скрипт, который подписывается на сигнал PropertiesChanged сетевого устройства и реагирует на изменение его состояния. При получении сигнала о сбое/отключении основного устройства, Python-агент немедленно вызывает метод ActivateConnection посредством D-Bus, тем самым активируя резервное подключение. Листинг программы представлен ниже:

```
import dbus
import dbus.mainloop.glib
from gi.repository import GLib
import time

MAIN_IFACE = "enp0s3"      # Интерфейс, который мониторим
BACKUP_CON_NAME = "second-nat" # Имя подключения для переключения
BACKUP_IFACE = "enp0s8"    # Физический интерфейс для бэкапа

NM_DEVICE_STATE_ACTIVATED = 100
NM_DEVICE_STATE_DISCONNECTED = 30
NM_DEVICE_STATE_FAILED = 120

def get_device_path(interface_name):
  """Находит объектный путь устройства по имени (например, enp0s3)"""
  bus = dbus.SystemBus()
  nm = bus.get_object('org.freedesktop.NetworkManager', '/org/freedesktop/NetworkManager')
  devices = nm.GetDevices(dbus_interface='org.freedesktop.NetworkManager')

  for dev_path in devices:
    dev_obj = bus.get_object('org.freedesktop.NetworkManager', dev_path)
    iface = dev_obj.Get('org.freedesktop.NetworkManager.Device', 'Interface',
                dbus_interface='org.freedesktop.DBus.Properties')
    if iface == interface_name:
      return dev_path
  return None
```

```
def get_connection_path(connection_name):
    """
    Находит объектный путь подключения (connection path) по его имени.
    Это обязательно, так как D-Bus работает с путями, а не с именами.
    """
    bus = dbus.SystemBus()
    settings_proxy = bus.get_object('org.freedesktop.NetworkManager',
                                    '/org/freedesktop/NetworkManager/Settings')
    settings_iface = dbus.Interface(settings_proxy, 'org.freedesktop.NetworkManager.Settings')

    connections = settings_iface.ListConnections()

    for conn_path in connections:
        conn_proxy = bus.get_object('org.freedesktop.NetworkManager', conn_path)
        conn_settings = dbus.Interface(conn_proxy, 'org.freedesktop.NetworkManager.Settings.Connection')


        config = conn_settings.GetSettings()

        # config['connection']['id'] - это то самое имя, которое мы видим в nmcli
        if config['connection']['id'] == connection_name:
            return conn_path

    print(f"[ОШИБКА] Подключение '{connection_name}' не найдено в настройках!")
    return None

def properties_changed(interface_name, changed_properties, invalidated_properties):
    """Обработчик сигнала об изменении свойств основного устройства"""
    if 'State' in changed_properties:
        new_state = changed_properties['State']

        if new_state == NM_DEVICE_STATE_DISCONNECTED or new_state == NM_DEVICE_STATE_FAILED:
            print(f"[SIGNAL] Фиксация отказа основного канала (State: {new_state}) в {time.time()}")

            start_time = time.time()

            try:

                nm_proxy = bus.get_object('org.freedesktop.NetworkManager', '/org/freedesktop/NetworkManager')
                nm_iface = dbus.Interface(nm_proxy, 'org.freedesktop.NetworkManager')

                print(f"-> Активация {BACKUP_CON_NAME} на {BACKUP_IFACE}...")

                nm_iface.ActivateConnection(
                    backup_conn_path,
                    backup_dev_path,
                    "/"
                )

                end_time = time.time()
                elapsed_ms = int((end_time - start_time) * 1000)

                print(f"[УСПЕХ] Переключение выполнено за {elapsed_ms} мс (Pure D-Bus)")

            except dbus.exceptions.DBusException as e:
                print(f"[ОШИБКА] Не удалось переключиться: {e}")


            loop.quit()
```

```
if __name__ == "__main__":
    dbus.mainloop.glib.DBusGMainLoop(set_as_default=True)
    bus = dbus.SystemBus()

    print("--- Инициализация Pure D-Bus Failover ---")

    backup_dev_path = get_device_path(BACKUP_IFACE)
    if not backup_dev_path:
        exit(1)
    print(f"Устройство бэкапа найдено: {backup_dev_path}")

    backup_conn_path = get_connection_path(BACKUP_CON_NAME)
    if not backup_conn_path:
        exit(1)
    print(f"Подключение найдено: {backup_conn_path}")

    main_dev_path = get_device_path(MAIN_IFACE)
    if not main_dev_path:
        exit(1)
    print(f"Мониторинг устройства: {main_dev_path}")
    print("Ожидание событий...\n")

    bus.add_signal_receiver(
        properties_changed,
        bus_name='org.freedesktop.NetworkManager',
        path=main_dev_path,
        dbus_interface='org.freedesktop.DBus.Properties',
        signal_name='PropertiesChanged'
    )

    loop = GLib.MainLoop()
    loop.run()
```

Далее необходимо предварительно выключить резервный интерфейс, запустить выполнение скрипта и эмулировать сбой основного интерфейса. В результате получаем такой вывод скрипта:

```
[SIGNAL] Фиксация отказа основного канала (State: 30) в 1770039928.9765048
-> Активация second-nat на enp0s8...
[УСПЕХ] Переключение выполнено за 29 мс (Pure D-Bus)
```

Из вывода программы можно заметить скорость выполнения переключения на резервное подключение – 29 мс, что значительно превосходит по скорости реакции традиционный Polling-подход. Использование D-Bus не создаёт дополнительной нагрузки на CPU, поскольку мониторинг состояния сетевого интерфейса осуществляется событийно – за счёт обработки сигналов NetworkManager, а не посредством постоянного опроса состояния системы. Сравнительный анализ представлен в виде таблицы ниже:

**Таблица 2 – Сравнительный анализ подходов**

| Замеры | Время, мс | Процессор, % | Память, % |
|---|---|---|---|
| Polling | 741.6 | 0.0-0.7 | 0.2 |
| D-Bus | 22.8 | 0.0 | 1.0 |

Таким образом, можно сделать вывод о том, что использование D-Bus в подобных вопросах является более выгодным решением с точки зрения нагрузки на систему.

**Эксперимент 2. Динамическое изменение контекста безопасности.**

Целью данного эксперимента является исследование возможности динамического изменения сетевой безопасности на основе получаемых сигналов от NetworkManager.

Суть эксперимента лежит в разработке микросервиса, который ловит сигнал переключения подключения и на его основе меняет настройки firewall.

Предварительно было создано новое подключение:

```
# nmcli con add type ethernet con-name local-private-net ifname enp0s9 ip4 77.77.0.121/24
```

Python-агент подписывается на сигнал StateChanged от NetworkManager. При активации подключения, оно идентифицируется, после чего применяется соответствующая политика безопасности. Для приватной сети SSH разрешен, а для обычной – SSH блокируется. Далее представлен листинг агента:

```
import dbus
import dbus.mainloop.glib
from gi.repository import GLib
import subprocess

TRUSTED_UUID = "96e6f92e-8dc7-4b5f-b89e-c12748e84c8f"
UNTRUSTED_UUID = "6761581e-4c8e-49ec-a7aa-a9d98d115ca7"

BLOCK_RULE = "INPUT -p tcp --dport 22 -j DROP"

def get_uuid_by_device_path(dev_path):
    """Получает UUID активного подключения по пути устройства"""
    try:
        bus = dbus.SystemBus()
        dev_proxy = bus.get_object('org.freedesktop.NetworkManager', dev_path)


        active_conn_path = dev_proxy.Get('org.freedesktop.NetworkManager.Device', 'ActiveConnection',
                                         dbus_interface='org.freedesktop.DBus.Properties')

        if not active_conn_path or active_conn_path == "/":
            return None

        ac_proxy = bus.get_object('org.freedesktop.NetworkManager', active_conn_path)
        uuid = ac_proxy.Get('org.freedesktop.NetworkManager.Connection.Active', 'Uuid',
                            dbus_interface='org.freedesktop.DBus.Properties')
        return uuid
    except Exception as e:
        print(f"Ошибка при получении UUID: {e}")
        return None

def apply_security_policy(uuid):
    """Логика безопасности"""
    if not uuid:
        return

    print(f"[*] Анализ безопасности для UUID: {uuid}")

    if uuid == UNTRUSTED_UUID:
        print(f"[ALERT] Недоверенная сеть. Блокируем SSH...")
        try:
            check = subprocess.run(["iptables", "-C", *BLOCK_RULE.split()], capture_output=True)
            if check.returncode != 0:
                subprocess.run(["iptables", "-A", *BLOCK_RULE.split(), "-m", "comment", "--comment", "DBUS_SEC_AGENT"],
check=True)
                print("   -> Правило добавлено (SSH закрыт).")
            else:
                print("   -> Правило уже активно.")
        except subprocess.CalledProcessError:
            pass # Ошибки игнорируем (правило уже есть)

    elif uuid == TRUSTED_UUID:
```

```
        print(f"[INFO] Доверенная сеть. Открываем SSH...")
        try:
            while True:
                res = subprocess.run(["iptables", "-D", *BLOCK_RULE.split(), "-m", "comment", "--comment",
"DBUS_SEC_AGENT"],
                                     capture_output=True)
                if res.returncode != 0:
                    break
            print("   -> Правило удалено (SSH открыт).")
        except Exception:
            pass

def device_state_changed(new_state, old_state, reason, sender_path):
    """
    Аргументы сигнала StateChanged NM:
    new_state (uint32)
    old_state (uint32)
    reason (uint32)
    sender_path (str) - это мы просим добавить через path_keyword
    """
    NM_DEVICE_STATE_ACTIVATED = 100

    if new_state == NM_DEVICE_STATE_ACTIVATED:
        print(f"\n[EVENT] Интерфейс {sender_path} стал АКТИВНЫМ")
        uuid = get_uuid_by_device_path(sender_path)
        if uuid:
            apply_security_policy(uuid)
        else:
            print("Не удалось определить UUID (возможно, интерфейс без IP).")

if __name__ == "__main__":
    dbus.mainloop.glib.DBusGMainLoop(set_as_default=True)
    bus = dbus.SystemBus()

    print("--- Агент безопасности запущен ---")
    print(f"Trusted: {TRUSTED_UUID}")
    print(f"Untrusted: {UNTRUSTED_UUID}")


    bus.add_signal_receiver(
        device_state_changed,
        bus_name='org.freedesktop.NetworkManager',
        dbus_interface='org.freedesktop.NetworkManager.Device',
        signal_name='StateChanged',
        path_keyword='sender_path'
    )

    loop = GLib.MainLoop()
    loop.run()
```

Для проверки работы, в одном терминале запускается скрипт, а в другом меняются подключения через nmcli con up.

При проверке политик для разных подключений можно заметить корректность работы агента:

```
# приватное подключение
INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt:ssh /* DBUS_SEC_AGENT */
```

```
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

#обычное подключение
[root@rtr test_area]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Данный эксперимент показывает, что с помощью D-Bus можно создавать сложные системы для гибкого обеспечения безопасности сети.

**Заключение**

В ходе исследования было подробно изучено управление сетью в BaseALT с помощью NetworkManager и шины D-Bus. При проведении экспериментов были наглядно показаны возможности взаимодействия с сетью через API, цель была достигнута.

Было выяснено, что D-Bus – это не просто набор команд, а целая система управления сетевой инфраструктурой. Эксперименты подтвердили возможность построения систем, способных в реальном времени реагировать на изменения состояния сети. Благодаря D-Bus API появляется возможность создавать более надежные, гибкие и быстрые решения, которые будут интегрированы в ядро ОС.

С практической точки зрения ценность данного исследования в демонстрации силы D-Bus как инструмента автоматизации.

**Библиографический список**


1. Шевченко, Д. А. Применение D-Bus для управления KDE в ОС АЛЬТ / Д. А. Шевченко, И. С. Боготов // Вестник науки. – 2025. – № 6. – С. 324–333.
2. Part III. D-Bus API Reference // NetworkManager Developer Documentation. – URL: https://networkmanager.dev/docs/api/latest/spec.html (дата обращения: 06.12.2025).
3. D-Bus // freedesktop.org Documentation. – URL: https://www.freedesktop.org/wiki/Software/dbus/ (дата обращения: 30.09.2025).
4. Уймин, А. Г. Нормирование показателей при организации киберполигона по сетевым технологиям / А. Г. Уймин // Нефть и газ – 2024 : тезисы докладов 78-й Международной молодежной научной конференции, Москва, 2024. – Москва, 2024. – С. 1253–1254.
5. Резник, В. Г. Операционные системы. Часть 2 : учебное пособие / В. Г. Резник. – Томск : ТУСУР, 2016. – 216 с.
6. Wheeler, D. Вводное руководство D-Bus / D. Wheeler, J. Palmieri, C. Walters // freedesktop.org. – 2020. – URL: https://dbus.freedesktop.org/doc/dbus-tutorial.html (дата обращения: 30.09.2025).
7. Потапов, А. А. Работа сетевой подсистемы в отечественных ОС / А. А. Потапов, Е. В. Чулисов // Мировая наука. – 2025. – № 1(94). – DOI: 10.5281/zenodo.14889548.


**References**


1. Shevchenko, D. A., & Bogotov, I. S. (2025). Using D-Bus to control KDE in ALT OS. Vestnik Nauki, (6), 324–333.
2. NetworkManager Project. (n.d.). Part III. D-Bus API Reference. NetworkManager Developer Documentation. Retrieved December 6, 2025, from https://networkmanager.dev/docs/api/latest/spec.html

3. freedesktop.org. (n.d.). D-Bus. Retrieved September 30, 2025, from https://www.freedesktop.org/wiki/Software/dbus/
4. Uymin, A. G. (2024). Normalization of indicators in the organization of a cyber polygon on network technologies. In Oil and gas – 2024: Abstracts of the 78th International Youth Scientific Conference (pp. 1253–1254).
5. Reznik, V. G. (2016). Operating systems. Part 2. Tomsk State University of Control Systems and Radioelectronics.
6. Wheeler, D., Palmieri, J., & Walters, C. (2020). D-Bus tutorial. freedesktop.org. Retrieved September 30, 2025, from https://dbus.freedesktop.org/doc/dbus-tutorial.html
7. Potapov, A. A., & Chulisov, E. V. (2025). The work of the network subsystem in domestic operating systems. Mirovaya Nauka, 1(94). https://doi.org/10.5281/zenodo.14889548

## Информация об авторах

Тищенко Эльдар Валерьевич – студент ФКБ ТЭК, ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, Российская Федерация, e-mail: eldar.tishchenko@gmail.com.

Корякин Владислав Юрьевич – студент ФКБ ТЭК, ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, Российская Федерация, e-mail: v.koryakin4@yandex.ru.

# RESEARCH OF NETWORK MANAGEMENT FUNCTIONALITY USING THE D-BUS SUBSYSTEM IN THE ALT OS

**E. V. Tishchenko[1], V. Y. Koryakin[1]**

[1]National University of Oil and Gas «Gubkin University»

**Abstract.** The article examines the possibilities of managing network connections in the ALT operating system by using the universal D-Bus system bus and the NetworkManager service. The relevance of the study is determined by the need to develop flexible tools for automating the network subsystem, which are able not only to execute predefined commands but also to respond to changes in interface states in real time. The paper considers the theoretical foundations of D-Bus interprocess communication, object addressing principles, the use of services, methods and signals, as well as the architecture of NetworkManager as a key mechanism for managing wired and wireless connections in modern Linux distributions. The practical part includes two experiments: a comparison of the polling approach with event-oriented failover switching through the D-Bus API and the development of a Python agent that changes firewall rules depending on the active network connection. The results show that the D-Bus event model provides a faster response to a failure of the primary interface, reduces the need for constant system polling and makes it possible to create automated security management tools. It is concluded that the NetworkManager D-Bus API can be used to build reliable and extensible solutions integrated into the ALT OS infrastructure.

**Keywords:** D-Bus, ALT OS, network interfaces, signals, NetworkManager, API, automation.

## Information about the authors

Eldar Valeryevich Tishchenko is a student of the Federal Design Bureau of Fuel and Energy Complex, Gubkin Russian State University of Oil and Gas (NRU), Moscow, Russian Federation, e-mail: eldar.tishchenko@gmail.com.

Vladislav Yuryevich Koryakin is a student of the Federal Design Bureau of Fuel and Energy Complex, Gubkin Russian State University of Oil and Gas (NRU), Moscow, Russian Federation, e-mail: v.koryakin4@yandex.ru.

**ПРОФЕССИОНАЛИТЕТ, 2025, № 3 (007)**

***УДК*** *004.056*

***А. В. Фоменко*** [1]***, Я. Ю. Зеленов*** [1]
[1]*ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, Российская Федерация*

# ВОПРОСЫ ОБЕСПЕЧЕНИЯ ЗАЩИТЫ ОТ АТАКИ CAM TABLE OVERFLOW НА L2 УСТРОЙСТВАХ

**Аннотация.** В современных локальных сетях атака CAM Table Overflow представляет критическую угрозу безопасности уровня L2, приводящую к переполнению таблицы MAC-адресов коммутатора потоком фейковых адресов. Успешная реализация атаки переводит устройство в режим широковещательной рассылки (flooding), что позволяет злоумышленнику перехватывать конфиденциальный трафик и вызывает критическую нагрузку на ресурсы оборудования (CPU до 85%, RAM до 75%), нарушая доступность сетевых услуг. Целью работы стал сравнительный анализ методов нейтрализации данной угрозы: Port Security, Storm Control и DHCP Snooping. Методология исследования включала моделирование атаки с помощью утилиты macof и оценку эффективности защитных механизмов на оборудовании вендоров Cisco, Eltex и MikroTik. Критериями выбора служили скорость реакции, точность блокировки и влияние на трафик. Результаты экспериментов подтвердили превосходство технологии Port Security. При активации данного механизма коммутатор мгновенно (за 2–5 секунд) блокирует порт нарушителя в режиме secure-shutdown, ограничивая таблицу двумя записями вместо 8124. Нагрузка на процессор снижается до 10%, а оперативная память разгружается до 35%, при этом работа легитимных узлов не прерывается. В отличие от аналогов, Port Security обеспечивает прямую защиту от причины атаки, а не только смягчает её последствия. Сделан вывод, что внедрение Port Security является наиболее универсальным и надежным решением для защиты коммутаторов L2+ от переполнения CAM-таблиц. Рекомендовано использовать данный механизм как базовый

**Ключевые слова:** CAM таблица, MAC адрес, L2 устройства, защита коммутаторов, Port Security, перехват трафика.

## Введение

В современных сетях передачи данных коммутаторы L2 являются критически важным элементом инфраструктуры сети. Атака CAM table overflow — переполнение таблицы MAC-адресов коммутатора — представляет серьезную угрозу безопасности, позволяя злоумышленнику перехватывать трафик и нарушать функционирование сети. Несмотря на известность атаки, многие сетевые администраторы не уделяют достаточного внимания настройке механизмов защиты.

Цель работы: провести сравнительную оценку защиты от атаки CAM table overflow и экспериментально подтвердить эффективность выбранного метода на оборудовании различных вендоров.

Задачи исследования:

- проанализировать механизм атаки CAM table overflow;
- исследовать существующие методы защиты и выбрать наиболее эффективный;
- реализовать конфигурации выбранного метода защиты на коммутаторах Cisco, Eltex, MikroTik;
- провести анализ влияния атаки и метода защиты на ресурсы устройства и сетевую задержку
- экспериментально доказать эффективность защиты;

## Литературный обзор

С точки зрения IEEE(стандарт спецификации), CAM-Table overflow — это атака, эксплуатирующее стандартное поведение коммутатора, описываемое IEEE 802.3. RFC 2863 хоть и не определяет MAC-адреса напрямую, описывает стандартизацию управления интерфейсами, их обслуживание и свойства.

Современные коммутаторы работает следующим образом. получает данные от обращающихся к нему устройств и постепенно заполняет CAM-таблицу (таблицу коммутации) их

MAC-адресами. При последующих обращениях коммутатор считывает адрес устройства-отправителя, анализирует таблицу коммутации и определяет по ней, на какое устройство нужно переслать данные [1]. Прочие компьютеры при этом не «знают» о факте передачи информации, поскольку она не имеет к ним отношения.

В статье [2] описан смысл атаки MAC-flooding, так как обе атаки используют схожие механизмы переполнения таблицы MAC-адресов в коммутаторах. Она помогает глубже понять методы реализации атаки, а также существующие подходы к защите и обнаружению, что важно для критического анализа в публикации.

Статья [7] помогла подробно понять механизм атаки CAM Table Overflow и принципы её реализации в локальных сетях. Полученные сведения использованы при описании теоретической части работы и объяснении уязвимости коммутаторов второго уровня.

**Анализ темы**

Как именно происходит атаки с переполнением CAM-таблицы? Злоумышленник использует специализированное программное обеспечение, которое генерирует и передает в сеть непрерывный поток Ethernet-кадров, содержащих случайно сгенерированные MAC-адреса. Данные кадры поступают на один из портов целевого коммутатора, который в соответствии со стандартной логикой работы начинает добавлять эти фиктивные адреса в свою CAM-таблицу [2].

По мере заполнения таблицы до ее максимальной емкости коммутатор либо прекращает добавление новых записей, либо начинает вытеснять ранее зарегистрированные легитимные записи. Это нарушает нормальный процесс коммутации - при поступлении кадров на адреса, отсутствующие в переполненной таблице, коммутатор переходит в режим широковещательной рассылки, дублируя трафик на все порты, принадлежащие соответствующей VLAN [2].

Возникающая уязвимость позволяет злоумышленнику, находящемуся в том же широковещательном домене, осуществлять перехват всего сетевого трафика, включая передачу конфиденциальной информации, учетных данных пользователей и других критически важных данных.

Для предотвращения атаки CAM-table overflow существует 3 основных метода защиты:

1. Port Security

*Port-Security* – механизм обеспечения безопасности и контроля доступа, основанный на контроле изучаемых MAC-адресов. Может использоваться как дополнение существующей аутентификации 802.1x и аутентификации MAC. Port-security контролирует доступ неавторизованных устройств к сети, проверяя MAC-адрес источника принятого кадра и доступ к неавторизованным устройствам, проверяя MAC-адрес назначения отправленного кадра [3].

2. Storm-control

*Storm control (контроль шторма)* – это сетевая функция, которая предотвращает перегрузку сети из-за чрезмерного широковещательного, многоадресного или неизвестного одноадресного трафика. Функция storm-control задаёт максимальное количество передаваемых пакетов в секунду, тем самым отбрасывая ненужные пакеты при превышении данного значения [6].

3. DHCP Snooping

*DHCP Snooping* – это механизм безопасности на коммутаторах, который защищает от атак через подмену DHCP-сервера и косвенно помогает предотвращать CAM table overflow. Принцип работы основан на разделении портов на доверенные (для подключения легитимных DHCP-серверов) и недоверенные (для пользовательских устройств). На недоверенных портах блокируются все DHCP-ответы, что предотвращает подключение фальшивых серверов. Одновременно механизм автоматически создает и поддерживает базу данных соответствий IP-MAC-адресов, полученных через легитимные DHCP-транзакции [4].

Для корректного сравнения методов необходимо разграничить уровни их воздействия на архитектуру коммутатора. Атака CAM Table Overflow эксплуатирует уязвимость плоскости данных (Data Plane), перегружая таблицу коммутации.

Port Security является мерой защиты именно плоскости данных (Data Plane protection),

так как контролирует входящий трафик на уровне порта и предотвращает запись нелегитимных MAC-адресов в аппаратную таблицу. Это прямой метод нейтрализации вектора атаки.

Storm Control также относится к защите плоскости данных, но работает реактивно: он не предотвращает заполнение таблицы, а лишь ограничивает интенсивность широковещательного трафика (последствие атаки), когда таблица уже переполнена и коммутатор перешел в режим flooding.

DHCP Snooping функционирует преимущественно в плоскости управления (Control Plane), фильтруя служебные протоколы. Сам по себе он не защищает от прямого заполнения CAM-таблицы фейковыми MAC-адресами. Его эффективность против данной атаки раскрывается только в комплексе с технологиями Dynamic ARP Inspection (DAI) и IP Source Guard (IPSG). В такой связке DHCP Snooping формирует доверенную базу соответствий (binding table IP-MAC-порт), которую используют DAI и IPSG для блокировки пакетов с подмененными адресами на уровне портов, тем самым косвенно предотвращая переполнение. Сравним данные методы для выяснения самого эффективного в контексте защиты от атаки переполнения CAM таблицы:

**Таблица 1. Сравнительная характеристика методов защиты**

| Критерий | Port Security | Storm Control | DHCP Snooping (в связке с DAI/IPSG) |
|---|---|---|---|
| Основная функция | Защита от переполнения CAM-таблицы | Борьба с широковещательными штормами | Защита от подмены DHCP-сервера |
| Эффективность против CAM table overflow | Прямая и полная защита | Косвенная защита (борьба с последствиями) | Косвенная защита (предотвращение атак через DHCP) |
| Уровень защиты | Data Plane (прямая защита порта) | Data Plane (ограничение трафика) | Control Plane + Data Plane (комплексная фильтрация) |
| Принцип работы | Ограничение MAC-адресов на порту, блокировка при нарушении | Ограничение трафика по типам (broadcast/unicast/multicast) | Фильтрация DHCP-сообщений, создание базы доверенных клиентов |
| Реакция на атаку | Мгновенная блокировка порта | Подавление превышающего трафика | Блокировка неавторизованных DHCP-серверов |
| Влияние на работу сети | Локальное (только на атакованном порту) | Общее (защищает всю подсеть) | Общее (защищает всю подсеть) |
| Простота настройки | Простая конфигурация | Требует тонкой настройки порогов | Сложная настройка, требует интеграции с другими механизмами |
| Преимущества | Наиболее эффективен против целевой атаки, простая настройка, минимальные ложные срабатывания | Защита от последствий атаки, контроль широковещательного трафика | Дополнительная защита сети, создание базы доверенных клиентов |
| Недостатки | Требует точной настройки лимитов MAC-адресов | Не предотвращает атаку, только смягчает последствия | Не защищает напрямую от CAM table overflow, сложная настройка |

Таким образом, Port Security является наиболее эффективным самодостаточным методом защиты непосредственно от вектора атаки CAM Table Overflow, обеспечивая превентивную блокировку на уровне порта (Data Plane). В то время как Storm Control лишь смягчает последствия атаки, а DHCP Snooping требует развертывания дополнительного комплекса мер (DAI, IPSG) для достижения сопоставимого уровня защиты. Учитывая критерии простоты внедрения и прямой направленности действия, для целей данного исследования выбран метод Port Security.

**Материалы и методы (ход исследования)**

Перейдем к практической реализации на оборудованиях различных вендоров:

```
enable
configure terminal
switchport port-security
interface fastethernet 0/1
```

```
switchport port-security
switchport port-security maximum 2
switchport port-security violation shutdown
switchport port-security mac-addres sticky
end
```

```
Mac Entries for Vlan 30:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0

Total Mac Address Space Available: 8043

Switch#
Switch#
Switch#enable
Switch#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#switchport port-security
Switch(config)#interface fastethernet 0/1
Switch(config-if)#switchport port-security
Switch(config-if)#switchport port-security maximum 2
Switch(config-if)#switchport port-security violation shutdown
Switch(config-if)#switchport port-security mac-addres sticky
Switch(config-if)#end
Switch#
```

**Рис. 1. Конфигурация Port-Security на Cisco**

```
configure terminal
interface fastEthernet0/1
switchport port-security mode max-addresses
switchport port-security mac-limit 3
switchport port-security action shutdown
switchport port-security enable
exit
exit
copy running-config startup-config
```

```
console#configure terminal
console(config)#interface Fastethernet0/1
console(config-if)#switchport port-security mode max-addresses
console(config-if)#switchport port-security mac-limit 3
console(config-if)#switchport port-security action shutdown
console(config-if)#switchport port-security enable
console(config-if)#exit
console(config)#exit
console#copy running-config startup-config
Building configuration ...
[OK]
console#
```

**Рис.2. Конфигурация Port-Security на Eltex**

```
/interface ethernet switch rule
add ports=ether1 learn-limit=2 learning=disabled action=drop disabled=no
```

```
switch:
[admin@MikroTik] /interface ethernet switch rule> /
[admin@MikroTik] > interface ethernet switch rule
[admin@MikroTik] /interface ethernet switch rule> add ports=ether1 learn-limit=2 learning=disabled
  \ action=drop disabled=no
[admin@MikroTik] /interface ethernet switch rule> /
```

**Рис. 3. Конфигурация Port-Security на MikroTik**

Далее проведем эксперимент, где проведем атаку CAM-table overflow на коммутатор с базовыми настройками, настроим на коммутаторе port-security и убедимся в эффективности работы port-security.

Топология сети:


**Рис. 4. Топология экспериментальной сети**

Подключаем компьютер пользователя и злоумышленника к коммутатору, и проводим базовую настройку коммутатора, следующими командами:

```
Switch> enable
Switch# configure terminal
Switch(config)# hostname CAM-TEST-SWITCH
Switch(config)# no ip domain-lookup
Switch(config)# interface range fastethernet 0/1-2
Switch(config-if-range)# switchport mode access
Switch(config-if-range)# switchport access vlan 1
Switch(config-if-range)# no shutdown
Switch(config-if-range)# exit
```

Мы установили компьютеры в один VLAN, чтобы эксперимент не вывел из строя коммутатор. Далее установили компьютерам статические ip адреса и проверили CAM-таблицу

```
Switch#show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count  : 2
Static  Address Count  : 0
Total Mac Addresses    : 2

Mac Entries for Vlan 3:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0

Mac Entries for Vlan 10:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0
```

**Рис.5. CAM-таблица до атаки**

На компьютере злоумышленника используя команду

sudo macof -i eth0 -n 15000

проводим атаку на коммутатор без включенной защиты port-security. Смотрим количество mac адресов и убеждаемся в переполнении

```
Switch#show mac address-table dynamic | count
% Incomplete command.

Switch#show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count   : 8124
Static  Address Count   : 0
Total Mac Addresses     : 8124

Mac Entries for Vlan 3:
---------------------------
Dynamic Address Count   : 0
Static  Address Count   : 0
Total Mac Addresses     : 0

Mac Entries for Vlan 10:
---------------------------
Dynamic Address Count   : 0
Static  Address Count   : 0
Total Mac Addresses     : 0
```

**Рис. 6. CAM-таблица после атаки**

```
4    102c.2206.60ab    DYNAMIC     Fa1/0/29
4    102e.b058.eda4    DYNAMIC     Fa1/0/29
4    102e.ea16.d70b    DYNAMIC     Fa1/0/29
4    1032.4973.92fa    DYNAMIC     Fa1/0/29
4    1035.ad03.ad4e    DYNAMIC     Fa1/0/29
4    1037.8c62.1536    DYNAMIC     Fa1/0/29
4    1037.d535.8552    DYNAMIC     Fa1/0/29
4    1038.0031.5554    DYNAMIC     Fa1/0/29
4    103c.3b7c.76e7    DYNAMIC     Fa1/0/29
4    1045.d26c.25d7    DYNAMIC     Fa1/0/29
4    104a.425a.e135    DYNAMIC     Fa1/0/29
4    104f.9e4e.b160    DYNAMIC     Fa1/0/29
4    1054.4474.9961    DYNAMIC     Fa1/0/29
4    1055.c418.a4b4    DYNAMIC     Fa1/0/29
4    1056.c35c.01d5    DYNAMIC     Fa1/0/29
4    105f.3614.2f5b    DYNAMIC     Fa1/0/29
4    1062.6336.daa4    DYNAMIC     Fa1/0/29
4    1062.af62.b3df    DYNAMIC     Fa1/0/29
4    106d.a55b.eaef    DYNAMIC     Fa1/0/29
4    1071.6823.e21a    DYNAMIC     Fa1/0/29
4    1074.c256.3d48    DYNAMIC     Fa1/0/29
4    1076.ca35.61ad    DYNAMIC     Fa1/0/29
4    107d.ec7b.5235    DYNAMIC     Fa1/0/29
4    107e.342f.31
```

**Рис. 7. Содержимое CAM-таблица после атаки**

## Обсуждение

Приведем результаты мониторинга в таблицу:

**Таблица 2. Результаты мониторинга**

| Время | Количество адресов | Состояние | CPU (%) | RAM (%) |
|---|---|---|---|---|
| 0 сек | 2 | Начало атаки | 5 | 30 |
| 10 сек | 2458 | Активное заполнение | 45 | 55 |
| 20 сек | 8124 | Таблица переполнена | 85 | 75 |
| 30 сек | 8124 | Режим flooding | 85-90 | 75 |

Как можно заметить, трафик между коммутатором и пользователем становится доступен

для перехвата. Переходим к настройке Port-Security. На коммутаторе вводим команды

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/1
Switch(config-if)# switchport port-security
Switch(config-if)# switchport port-security maximum 2
Switch(config-if)# switchport port-security violation shutdown
Switch(config-if)# switchport port-security mac-address sticky
Switch(config-if)# end
```

Таким образом проведена конфигурация port-security (на других коммутаторах аналогично вводятся свои команды, приведенные выше). Проводим атаку уже известной командой.

sudo macof -i eth0 -n 15000

Убеждаемся в работе метода защиты:

```
Mac Entries for Vlan 30:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0

Total Mac Address Space Available: 8043

Switch#show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count  : 2
Static  Address Count  : 0
Total Mac Addresses    : 2

Mac Entries for Vlan 3:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0

Mac Entries for Vlan 10:
```

**Рис. 8. CAM-таблица после атаки**

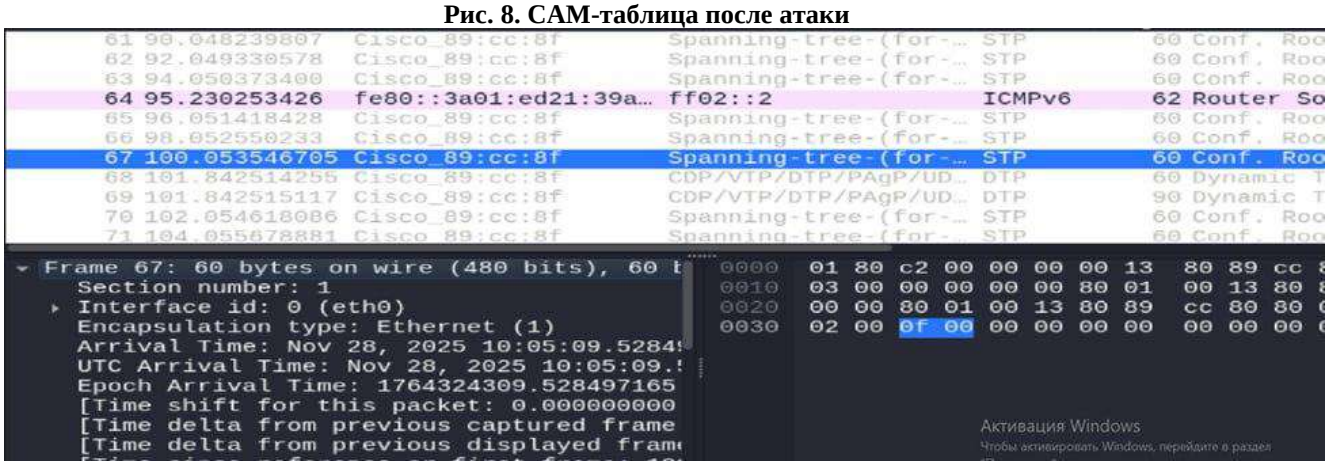


**Рис. 9. Wireshark блокировка компьютера атакующего**

Подведем анализ эффективности защиты в виде таблицы

**Таблица 3. Эффективности защиты**

| Параметр | Без защиты | С Port Security |
|---|---|---|
| Заполнение CAM- таблицы | 8124 записей | 2 записи |
| Время реакции | 20-30 секунд | 2-5 секунд |
| Состояние порта | Active (flooding) | Secure-shutdown |
| Возможность перехвата | Да | Нет |

| Влияние на легитный трафик | Прерывается | Не прерывается |
|---|---|---|
| Средняя задержка (Latency) | 145 мс (потери 40%) | 1.2 мс (стабильно) |
| Нагрузка на CPU | 85-95% | 5-10% |
| Нагрузка на RAM | 75% | 30-35% |

Эксперимент моделировал условия предельной нагрузки (stress-test), при которых утилизация CPU достигала 90%, а таблица MAC-адресов была переполнена на 100%. В таких нестандартных условиях коммутатор переходил в аварийный режим работы (flooding), что привело к критической деградации сервиса. Применение Port Security позволило исключить работу устройства в стрессовом режиме, поддерживая утилизацию ресурсов в пределах штатных значений (<10%) даже при интенсивной внешней атаке

Дополнительно была проведена оценка влияния механизмов защиты на сетевую задержку (latency). В режиме атаки без защиты средняя задержка возросла с 1 мс до 150 мс с потерей 40% пакетов. При активации Port Security после кратковременного прерывания (2 сек) задержка стабилизировалась на уровне 1.2 мс, что подтверждает минимальное влияние механизма защиты на производительность легитимного трафика.

**Заключение**

В ходе нашего исследования мы разобрали существующие методы защиты от атаки CAM-table overflow, а также на практике убедились в эффективности метода port-security. Было установлено, что данный метод защиты является универсальным решением от атаки, который не только предотвращает утечку трафика, но и не даёт злоумышленнику нагрузить устройство ненужными данными, в следствие чего идет большая нагрузка на оперативную память и процессор, а также вызывает задержку передачи трафика вплоть до 145 мс и потери пакетов (40%).

Эксперимент подтвердил, что Port Security является надежным и эффективным механизмом защиты коммутаторов L2 от атак переполнения CAM-таблицы. Коммутаторы, настроенные приведенным методом, будут защищены от возможных атак, направленных на переполнение таблицы адресов, и будут продолжать выполнение своих функций.

**Библиографический список**

1. Олифер, В. Г. Компьютерные сети. Принципы, технологии, протоколы : учебник для вузов / В. Г. Олифер, Н. А. Олифер. — 5-е изд., юбил. – Санкт-Петербург : Питер, 2021. – 1008 с. : ил. – (Серия «Учебник для вузов»). – ISBN 978-5-4461-1584-0.
2. АТАКА CAM-TABLE OVERFLOW. МЕТОДЫ ЗАЩИТЫ [Электронный ресурс] // Security.Lab – URL: https://cyberleninka.ru/article/n/ataka-cam-table-overflow-metody-zaschity (дата обращения: 11.11.2025).
3. Настройка Port Security на коммутаторах SNR / [Электронный ресурс] // Документация NAG: [сайт]. – URL: https://nag.wiki/pages/viewpage.action?pageId=25107728(дата обращения 11.11.2025)
4. Уймин, А. Г. Компьютерные сети. L2-технологии: Практикум / А. Г. Уймин. – Москва: Ай Пи Ар Медиа, 2024. - 191 с. – ISBN 978-5-4497-2539-4. EDN AXDYGY
5. Cisco Catalyst 2960 Series Switches Configuration Guide [Электронный ресурс] // Cisco Systems. – 2023. URL: https://www.cisco.com/c/en/us/support/switches/catalyst-2960-series switches/products-installation-and-configuration-guides-list.html.
6. Основы сетевой безопасности: защита коммутаторов L2 [Электронный ресурс] // Habr: статья. – 2020. – URL: https://habr.com/ru/articles/502480/ (дата обращения: 11.01.2025).
7. Трещев И.А. «О реализации атак типа MAC-FLOOD» / статья (eLIBRARY / научная публикация). // 2022. URL: https://elibrary.ru/item.asp?id=49992198 (дата обращения 11.11.2025).
8. Настройка Port Security на коммутаторах Eltex MES1400/MES2400 // База знаний Eltex [Электронный ресурс] // Документация. – URL: https://eltexcm.ru/baza-znanij/ethernet-kommutatory-mes/mes14xx24xx3400-xx37xx/interfejsy-vlan/mes-nastrojka-port-security-mes1400-mes2400.html

**References**

1. Olifer, V. G., & Olifer, N. A. (2021). *Computer networks: Principles, technologies, and protocols* (5th anniversary ed.). Piter.
2. *CAM-table overflow attack: Protection methods*. (n.d.). Security.Lab. Retrieved November 11, 2025, from https://cyberleninka.ru/article/n/ataka-cam-table-overflow-metody-zaschity
3. NAG. (n.d.). *Configuring Port Security on SNR switches*. NAG Documentation. Retrieved November 11, 2025, from https://nag.wiki/pages/viewpage.action?pageId=25107728
4. Uymin, A. G. (2024). *Computer networks: L2 technologies. A practical course*. IPR Media.
5. Cisco Systems. (2023). *Cisco Catalyst 2960 Series Switches Configuration Guide*. https://www.cisco.com/c/en/us/support/switches/catalyst-2960-series-switches/products-installation-and-configuration-guides-list.html
6. *Fundamentals of network security: Protecting L2 switches*. (2020). Habr. Retrieved January 11, 2025, from https://habr.com/ru/articles/502480/
7. Treshchev, I. A. (2022). On the implementation of MAC-flood attacks. eLIBRARY.RU. Retrieved November 11, 2025, from https://elibrary.ru/item.asp?id=49992198
8. Eltex. (n.d.). *Configuring Port Security on Eltex MES1400/MES2400 switches*. Eltex Knowledge Base. https://eltexcm.ru/baza-znanij/ethernet-kommutatory-mes/mes14xx24xx3400-xx37xx/interfejsy-vlan/mes-nastrojka-port-security-mes1400-mes2400.html

## Информация об авторах

Фоменко Артём Владимирович – студент 3-его курса, факультет КБ ТЭК, группы КИ-23-01 ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, e-mail: 1studentnow1@gmail.com

Зеленов Ярослав Юрьевич – студент 3-его курса, факультет КБ ТЭК группы КИ-23-01 ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И.М. Губкина», г. Москва, e-mail: mobzila567@gmail.com

***A. V. Fomenko [1], Ya. Yu. Zelenov [1]***

[1]National University of Oil and Gas «Gubkin University»

## PROTECTION AGAINST CAM TABLE OVERFLOW ATTACKS ON L2+ DEVICES

**Abstract.** In modern local area networks, the CAM Table Overflow attack poses a critical L2 security threat, causing the switch's MAC address table to become flooded with a flood of fake addresses. A successful attack puts the device into flooding mode, allowing the attacker to intercept sensitive traffic and places a critical load on hardware resources (CPU up to 85%, RAM up to 75%), disrupting network service availability. The objective of this study was to compare mitigation methods for this threat: Port Security, Storm Control, and DHCP Snooping. The research methodology included attack simulation using the macof utility and an evaluation of the effectiveness of defense mechanisms on Cisco, Eltex, and MikroTik equipment. The selection criteria included response speed, blocking accuracy, and impact on traffic. The experimental results confirmed the superiority of Port Security technology. When this mechanism is activated, the switch instantly (within 2-5 seconds) blocks the intruder's port in secure-shutdown mode, limiting the table to two entries instead of 8124. CPU load is reduced to 10%, and RAM is unloaded to 35%, while legitimate nodes continue to operate. Unlike similar mechanisms, Port Security provides direct protection against the root cause of the attack, rather than merely mitigating its consequences. It is concluded that implementing Port Security is the most universal and reliable solution for protecting L2+ switches from CAM table overflows. It is recommended to use this mechanism as a basic one.

**Keywords:** CAM table, MAC address, L2+ device, switch protection, Port Security, traffic interception

## Information about the authors

Fomenko Artem Vladimirovich - 3rd year student, Faculty of Design Bureau of Fuel and Energy Complex, group KI-23-01, Federal State Autonomous Educational Institution of Higher Education "Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: 1studentnow1@gmail.com

Zelenov Yaroslav Yurievich - 3rd year student, Faculty of Design Bureau of Fuel and Energy Complex group KI-23-01 of the Federal State Autonomous Educational Institution of Higher Education "RSU of Oil and Gas (NRU) named after I.M. Gubkin", Moscow, e-mail: mobzila567@gmail.com

**Рубрика 2. Методы и системы защиты информации, информационная безопасность**

УДК 004.056.5:004.7

**Д. Г. Галустян [1], С. А. Соколов [1]**
[1]ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, Российская Федерация

# ВОПРОСЫ ОБЕСПЕЧЕНИЯ ЗАЩИТЫ RIPv2 НА СЕТЕВЫХ УСТРОЙСТВАХ

**Аннотация.** Статья посвящена исследованию уязвимостей протокола маршрутизации RIPv2 и способов обеспечения его защищенной эксплуатации на сетевых устройствах. Актуальность работы обусловлена тем, что RIPv2 продолжает применяться в учебных, корпоративных и специализированных сетях, а его дистанционно-векторная логика делает протокол чувствительным к подмене маршрутной метрики. Рассматривается сценарий атаки, при котором злоумышленник, имеющий доступ к общему L2-сегменту, формирует с помощью Scapy поддельные RIPv2-пакеты с заниженным количеством переходов и тем самым воздействует на выбор маршрута. В качестве методологической основы использованы анализ спецификаций RIP/RIPv2, документации производителей сетевого оборудования и лабораторное моделирование на маршрутизаторах Cisco, MikroTik и Eltex. Экспериментальная часть включает проверку работы протокола без аутентификации, с текстовым паролем и с криптографической MD5-аутентификацией по RFC 2082. Дополнительно рассматриваются ограничения применяемых механизмов защиты, особенности конфигурирования устройств разных производителей и влияние ложных обновлений на доступность пользовательских подсетей. Показано, что отсутствие защиты и применение открытого пароля не препятствуют внедрению ложных маршрутов, тогда как MD5-аутентификация отклоняет неподтвержденные обновления и снижает риск нарушения связности. Сформулированы практические рекомендации по безопасной настройке RIPv2, включающие применение ключевых цепочек, фильтрацию маршрутов, ограничение RIP-сегментов и регулярный аудит таблиц маршрутизации. Выработанные меры ориентированы на практическое применение в учебных стендах, сегментах малого предприятия и инфраструктурах, где отказ от RIPv2 пока невозможен.

**Ключевые слова:** RIPv2, маршрутизация, аутентификация, MD5, Scapy, сетевая безопасность, подмена метрики.

## Введение

Несмотря на широкое распространение более современных протоколов маршрутизации (OSPF, EIGRP), протокол RIPv2 сохраняет практическую значимость в ряде специфических случаев. Во-первых, он продолжает использоваться в legacy-инфраструктурах, где переход на более современные протоколы нецелесообразен или невозможен. Во-вторых, RIPv2 часто встречается в промышленных сетях и специализированном оборудовании, где поддержка OSPF может отсутствовать.

RIPv2 расширяет классический RIP за счёт передачи масок подсетей, маршрутов по умолчанию, тегов маршрутов и поля аутентификации, сохраняя при этом дистанционно-векторный алгоритм и ограничение по числу переходов (метрика – от 1 до 15 hop’ов). RIPv2 работает поверх UDP-транспорта (порт 520), отправляя периодические multicast-обновления на устройства, не обеспечивая шифрования содержимого сообщений. Спецификация предусматривает три варианта аутентификации обновлений маршрутов: отсутствие аутентификации, пароль в открытом виде и криптографическая аутентификация на основе MD5. Документация крупных производителей сетевых устройств подчёркивает, что по умолчанию аутентификация RIPv2 отключена. В других случаях используется текстовая строка, при этом рекомендуется переход на MD5 или аналогичные криптографические методы, так как пароль в открытом виде легко перехватывается и впоследствии используется для формирования поддельных пакетов маршрутизации. Развитие средств анализа и генерации сетевого трафика, таких как Scapy, существенно упростило задачу злоумышленнику, таким образом, имея доступ к L2-сегменту, можно не только отправлять произвольные RIPv2-пакеты, но и целенаправленно подменять значения количества hop’ов, таким образом изменяя таблицу маршрутизации атакуемого устройства.

Следовательно, вопросы корректной настройки и защиты RIPv2, особенно в части противодействия атакам с подменой метрики маршрутов, остаются актуальными для значительного числа реальных сетей.

Объект исследования: безопасность сетевых устройств сетевого уровня (L3), использующих протокол маршрутизации RIPv2.

Предмет исследования: механизмы защиты RIPv2 на сетевом оборудовании от атак на основе подмены количества hop’ов в RIPv2-пакетах с помощью инструментов пакетной генерации, методы их реализации и эффективность в реальной инфраструктуре.

Цель исследования: Анализ уязвимостей RIPv2, связанных с подменой метрики, и существующих механизмов его защиты; разработка и экспериментальная проверка комплекса конфигурационных мер, повышающих устойчивость сетевой инфраструктуры к таким атакам.

**Обзор и методология исследования**

RIPv2 является развитием классического RIP с поддержкой CIDR, меток маршрута, multicast-доставки обновлений и поля для аутентификации. Протокол реализует дистанционно-векторный алгоритм, периодически рассылая таблицу маршрутов, где для каждого префикса указываются сеть, маска и метрика – численное значение, интерпретируемое как количество переходов (hop count) до назначения. Разновидности атак на RIPv2 представлены в таблице 1. Оценка угроз производилась в соответствии с открытым стандартом для расчёта количественных оценок уязвимостей – CVSS (Common Vulnerability Scoring System). Метрики для оценки: вектор атаки, сложность атаки, требуемые привилегии.

**Таблица 1 – Разновидности атак на RIPv2**

| № | Название атаки | Описание | Оценка угрозы по CVSS |
|---|---|---|---|
| 1 | Hop-count route attack (занижение метрики) | Захват трафика, отправка маршрутов с меньшей метрикой | 8.2 |
| 2 | Route Injection (вставка ложных маршрутов) | Перегрузка таблицы маршрутизации, добавление несуществующих сетей в RIP-обновления | 7.5 |
| 3 | Route Poisoning spoof | Отправка обновлений с метрикой 16 от имени соседа, удаление маршрута из таблицы | 6.5 |
| 4 | Neighbor Spoofing | Отправка RIP-пакета с поддельным IP соседа, полный контроль над маршрутизацией | 9.1 |

В данной работе рассматривается сценарий атакующего воздействия атаки hop-count spoofing, так как она является наиболее популярной и простой, но в то же время оказывает критическое воздействие на сетевые устройства. Суть атаки заключается в формировании поддельных RIPv2-сообщений, где для интересующих подсетей указываются намеренно заниженные метрики (1–2 хопа). Поскольку классический RIP всегда выбирает маршрут с минимальной метрикой, такие записи воспринимаются как более предпочтительные, что позволяет злоумышленнику:

– перехватывать трафик к целевым подсетям;

– перенаправлять трафик в пустоту, не маршрутизируя его далее;

– провоцировать нестабильность маршрутизации за счёт частой смены значения метрик.

Для генерации подобных пакетов можно использовать Scapy (библиотеку на Python для конструирования, модификации и отправки произвольных сетевых пакетов). Инструмент Scapy предоставляет реализацию протокола RIP/RIPv2 как отдельного слоя, что позволяет программно задавать любые значения метрики и других полей.

**Основные гипотезы исследования:**

– в сетях, где RIPv2 работает без аутентификации или использует текстовый пароль, злоумышленник, имеющий доступ к L2-сегменту и инструментам рассылки пользовательских пакетов, может успешно подменять количество hop’ов в RIPv2-пакетах, внедряя ложные маршруты и нарушая доступность устройств;

1. Включение MD5-аутентификации на всех соседствах RIPv2 существенно усложняет или делает невозможной подмену количества hop’ов с неподтверждённых устройств, однако не исключает риск атак при компрометации ключа.

– комплекс описанных мер защиты существенно снижает вероятность успешной атаки и уменьшает её последствия для маршрутизации.

**Рубрика 2. Методы и системы защиты информации, информационная безопасность**

**Методы исследования**

Тип исследования: анализ безопасности протокола с элементами экспериментального тестирования в лабораторной среде.

Характеристика среды исследования: лабораторная сетевая инфраструктура, включающая физические коммутаторы и маршрутизаторы производителей Cisco, MikroTik, Eltex, а также два PC. На PC установлена операционная система Linux.

**Методы сбора данных**

– захват и анализ RIPv2-трафика с помощью Wireshark;

– логирование маршрутизаторов (команды просмотра таблиц маршрутизации, режимы debug для RIPv2, журналы аутентификации и безопасности);

– измерение доступности и числа изменений маршрутов при различных сценариях атаки.

**Процедура проведения исследования**

1. Настройка базовой топологии с использованием RIPv2 без аутентификации.

2. Формирование атакующих RIPv2-пакетов в Scapy с заниженным количеством hop’ов для маршрута и их рассылка.

3. Анализ влияния подмены количества hop’ов на таблицы маршрутизации и доступность сетевых ресурсов.

4. Включение простой текстовой аутентификации, повторение атак с предварительным перехватом пароля и подменой количества hop’ов в аутентифицированных RIPv2-пакетах.

5. Включение MD5-аутентификации на всех линках RIPv2, повторение атак с поддельными пакетами.

6. Сравнительный анализ поведения оборудования разных производителей и эффективности различных комбинаций защитных механизмов.

**Методы обработки данных:**

Количественный анализ успешности атаки (по критериям изменения маршрутов и потери связности); оценка времени сходимости при штатной работе и в условиях атак; качественный анализ журналов и диагностических сообщений устройств при корректной и некорректной аутентификации RIPv2-пакетов.

**Основные этапы экспериментального исследования**

**Лабораторная среда эксперимента**

Целью эксперимента является воспроизведение небольшой сети, где RIPv2 используется для обмена маршрутной информацией между несколькими маршрутизаторами, а атакующий узел с Scapy имеет доступ к одному из общих сегментов.

**Топология сети**

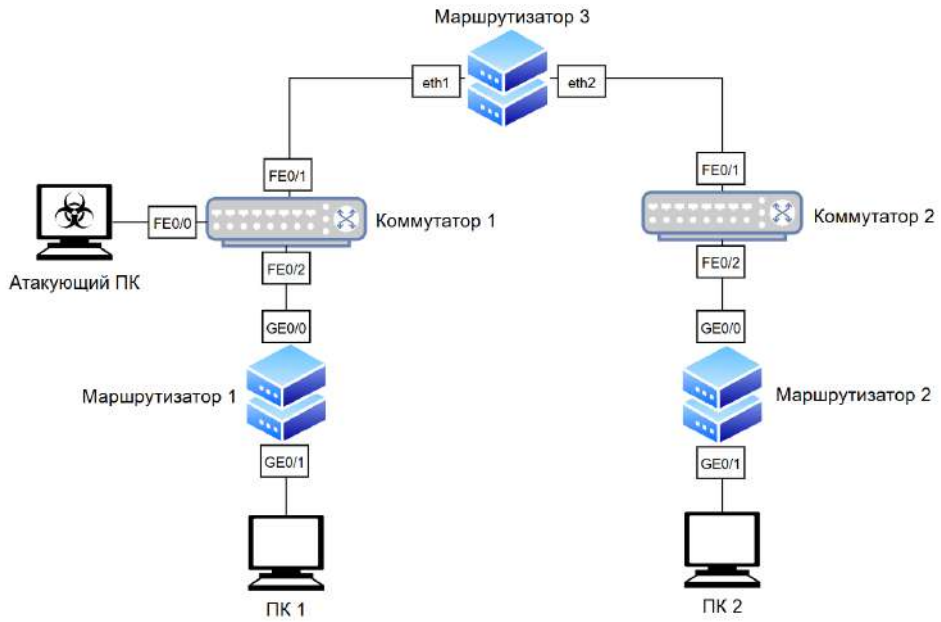


**Рис. 1. Топология сети для проведения эксперимента**

**Конфигурация машин**

Для проведения работ необходимо подготовить следующее оборудование:

Сетевое оборудование: коммутатор и маршрутизаторы (Cisco, MikroTik, Eltex).

Рабочие станции:

Атакующий ПК – компьютер на базе ОС Linux.

ПК1 и ПК2 – компьютер с любой операционной системой и сетевым интерфейсом.

Соединительные компоненты: Ethernet-кабели для подключения рабочих станций к коммутаторам.

Проведём настройку оборудования:

Подключаем атакующий ПК к коммутатору как показано в топологии. Атакующий ПК должен находиться в одной подсети с маршрутизатором, чтобы он воспринимал получаемые пакеты как исходящие от соседних устройств. Команды, используемые для настройки маршрутизатора каждого производителя представлены в таблице 2.

**Таблица 2 – Команды первичной настройки маршрутизаторов**

| **Блок команд** | **Cisco (Маршрутизатор 1)** | **Eltex (Маршрутизатор 2)** | **MikroTik (Маршрутизатор 3)** |
|---|---|---|---|
| Настройка первого интерфейса | enable<br>configure terminal<br>interface GigabitEthernet0/0<br>ip address 192.168.100.1 255.255.255.252<br>no shutdown<br>exit | enable<br>configure terminal<br>interface gigabitethernet 0/0<br>ip address 192.168.200.1 255.255.255.252<br>no shutdown<br>exit | interface ethernet set ether1 disabled=no<br>ip address add address=192.168.100.2/30 interface=ether1 |
| Настройка второго интерфейса | interface GigabitEthernet0/1<br>ip address 10.10.10.1 255.255.255.0<br>no shutdown<br>exit | interface gigabitethernet 0/1<br>ip address 20.20.20.1 255.255.255.0<br>no shutdown<br>exit | interface ethernet set ether2 disabled=no<br>ip address add address=192.168.200.2/30 interface=ether2 |
| Настройка RIPv2 | router rip<br>version 2<br>no auto-summary<br>network 192.168.100.0<br>network 10.0.0.0<br>exit<br>write memory | router rip<br>version 2<br>no auto-summary<br>network 192.168.200.0<br>network 20.20.20.0<br>exit<br>write | routing rip set enabled=yes<br>routing rip interface add interface=ether1<br>routing rip network add network=192.168.100.0/30<br>routing rip network add network=192.168.200.0/30 |

**Моделирование атаки**

Атакующий ПК посылает серию RIPv2-пакетов, в которых метрика имеющихся в таблице адресов предоставляется как более оптимальная. Это приводит к пересчёту таблиц маршрутизации для выбора более предпочтительного маршрута. Далее приведен листинг скрипта авторского написания на Scapy, генерирующий ложный RIPv2 пакет в лабораторной среде. В реальном скрипте параметры должны передаваться как аргументы, чтобы обеспечить универсальность работы инструмента.

```
from scapy.all import *
# IP адрес атакуемого устройства
target_ip = "192.168.100.2"
# Параметры ложного пакета: метрика и сеть
fake_metric = 1
network_addr = "10.10.10.10"
network_mask = "255.255.255.0"
# Формирование записи RIP для пакета
rip_entry = RIPEntry(
  AF=2,
  addr=network_addr,
  mask=network_mask,
  metric=fake_metric
```

```
)
# Создание пакета RIPv2
rip_packet = IP(dst=target_ip) / \
        UDP(sport=520, dport=520) / \
        RIP(cmd=2, version=2) / \
        rip_entry
```

Поддельные RIP-пакеты успешно формируются и отправляются с использованием инструмента Scapy. Маршрутизаторы начали получать некорректные записи с пониженными значениями метрик, что привело к появлению ложных маршрутов в их таблицах. Наблюдение за работой устройств в режиме отладки, показывает, что пакеты принимаются и обрабатываются маршрутизатором.

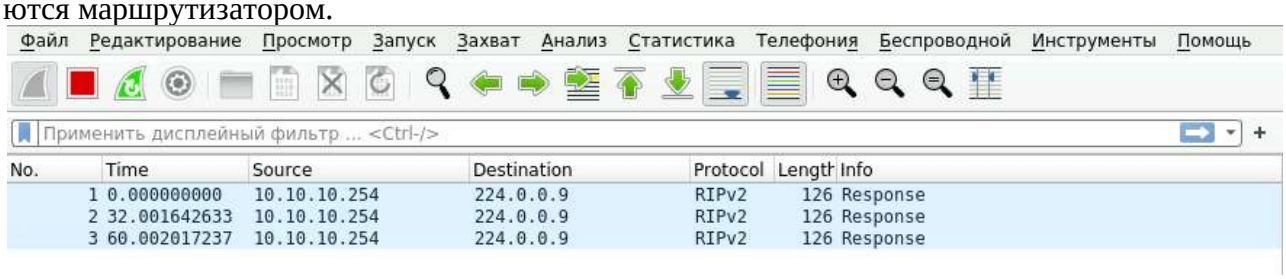


**Рис. 2. Захват трафика на атакующем ПК до атаки.**

Анализ трафика в Wireshark на атакующем ПК подтверждает изменение таблицы маршрутизации: ПК становится новым next-hop для объявленной сети. Маршрутизатор начинает перенаправлять через него соответствующий трафик, вследствие чего ICMP пакеты появляются в Wireshark.

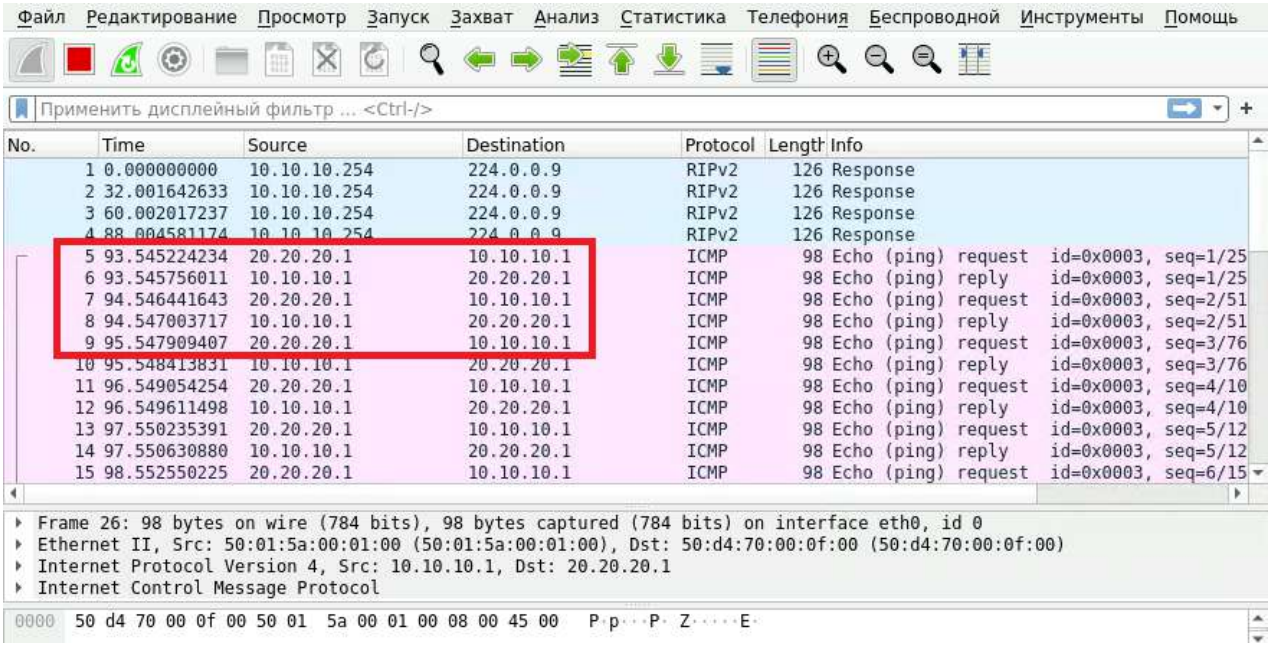


**Рис. 3. Захват трафика на атакующем ПК после атаки.**

## Реализация защитных механизмов

### Механизм 1. Аутентификация с использованием текстового пароля (Plain Text Authentication).

Одним из базовых механизмов защиты RIPv2 является применение текстовой аутентификации. В этом режиме в каждый RIP-пакет включается специальная запись, содержащая пароль в открытом виде (plain text). Маршрутизатор принимает маршрутную информацию только в том случае, если значение пароля совпадает с локально настроенным.

Команды для настройки аутентификации с использованием текстового пароля представлены в таблице 3.

**Таблица 3 – Команды для настройки аутентификации с использованием текстового пароля**

| **Блок команд** | **Cisco (Маршрутизатор 1)** | **Eltex (Маршрутизатор 2)** | **MikroTik (Маршрутизатор 3)** |
|---|---|---|---|
| Создание key chain | enable<br>configure terminal<br>key chain RIP-PLAIN<br>key 1<br>key-string testpass<br>exit | enable<br>configure terminal<br>key chain RIP-PLAIN<br>key 1<br>key-string testpass<br>exit | 1) routing rip interface add interface=ether1 authentication=plain password=testpass |
| Включение аутентификации | 1) interface GigabitEthernet0/0<br>2) ip rip authentication mode text<br>3) ip rip authentication key-chain RIP-PLAIN<br>4) exit<br>5) write memory | 1) interface gigabitethernet 0/1<br>2) ip rip authentication mode text<br>3) ip rip authentication key-chain RIP-PLAIN<br>4) exit<br>5) write | 1) routing rip interface set [find interface=ether1] authentication=plain password=testpass |

На рис. 4 изображен анализ захваченного трафика, который подтверждает корректную работу включенной защиты на основе простого текстового ключа.

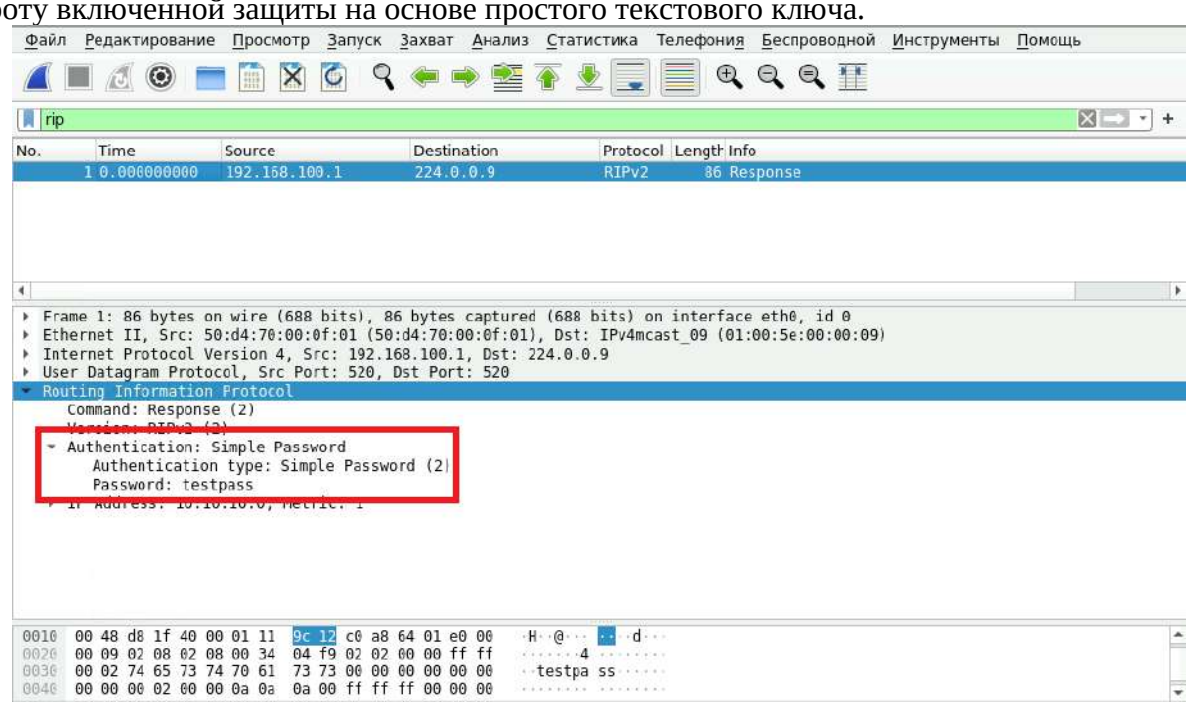


**Рис. 4. RIPv2 пакет, защищенный с помощью Plain Text Authentication.**

## Механизм 2. Аутентификация на основе MD5.

Для повышения устойчивости к подделке RIP-сообщений существует более надёжный метод – аутентификация на основе хеш-функции MD5. Этот механизм определён в RFC 2082 и значительно повышает безопасность RIPv2. Маршрутизатор проверяет хеш каждого полученного RIP-пакета. Если значение не совпадает – пакет считается поддельным и отклоняется.

Команды для настройки аутентификации на основе MD5 представлены в таблице 4.

**Таблица 4 – Команды для настройки аутентификации на основе MD5**

| Блок команд | Cisco (Маршрутизатор 1) | Eltex (Маршрутизатор 2) | MikroTik (Маршрутизатор 3) |
|---|---|---|---|
| Создание key chain | enable<br>configure terminal<br>key chain RIP-MD5<br>key 1<br>key-string MD5secretkey<br>exit | enable<br>configure terminal<br>key chain RIP-MD5<br>key 1<br>key-string MD5secretkey<br>exit | routing rip interface add interface=ether1 authentication=md5 password=MD5secretkey |
| Включение аутентификации | interface GigabitEthernet0/0<br>ip rip authentication mode md5<br>ip rip authentication key-chain RIP-MD5<br>exit | interface gigabitethernet 0/1<br>ip rip authentication mode md5<br>ip rip authentication key-chain RIP-MD5<br>exit | routing rip interface set [find interface=ether1] authentication=md5 password=MD5secretkey |

На рис. 5 изображен анализ захваченного трафика, который подтверждает корректную работу включенной защиты на основе MD5.

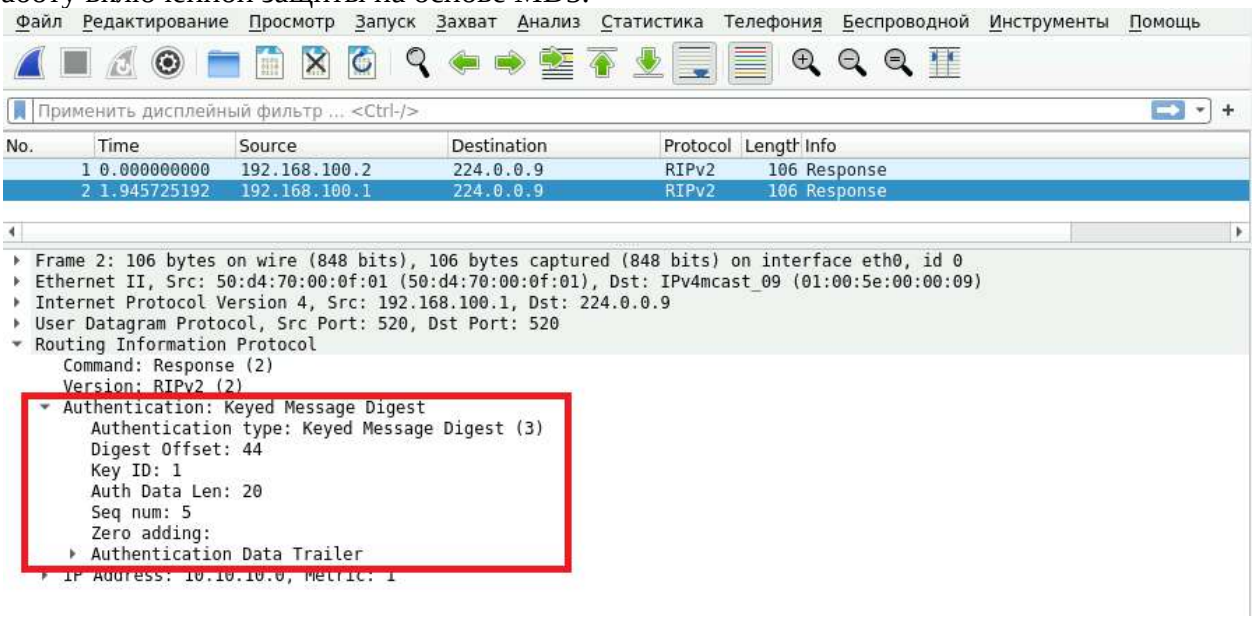


**Рис. 5. RIPv2 пакет, защищенный с помощью MD5 Authentication.**

## Результаты исследования

Экспериментальная часть работы показала, что RIPv2 крайне чувствителен к подмене количества hop’ов в пакетах, формируемых с помощью сторонних утилит, если протокол не защищён механизмами аутентификации и фильтрации. В исходной конфигурации, когда аутентификация RIPv2 отсутствовала, любой узел в общем L2-сегменте мог сгенерировать RIPv2-пакеты с произвольными маршрутами и заниженными значениями метрики. При рассылке таких пакетов по мультикаст-адресу 224.0.0.9 маршрутизатор принимал их как обычные обновления, заносил в таблицу маршрутизации новые записи и выбирал именно этот маршрут как оптимальный, поскольку количество hop’ов, установленное злоумышленником, было меньше, чем у реальных путей. В результате трафик между пользовательскими подсетями начинал проходить через атакующий узел. Связь между ПК была нарушена.

При включении текстовой аутентификации ситуация принципиально не изменилась. Анализ трафика RIPv2 показал, что строка пароля передаётся в открытом виде и без труда считывается (см. рис. 4). После этого атакующий, зная корректное значение поля аутентификации, мог воспроизвести те же самые сценарии подмены количества hop’ов, добавив соответствующую текстовую строку в формируемые пакеты. Маршрутизаторы воспринимали такие сообщения как полностью корректные и продолжали принимать ложные маршруты с

подменённой метрикой. Таким образом, текстовая аутентификация практически не повысила устойчивость сети к рассматриваемому типу атаки.

Ситуация принципиально изменилась при переходе на криптографическую аутентификацию (MD5 по RFC 2082). После настройки ключевых цепочек на всех маршрутизаторах и привязки их к интерфейсам RIPv2 пакеты, сформированные на атакующем ПК без знания секретного ключа, перестали оказывать какое-либо влияние на маршрутизацию.

Хотя MD5-аутентификация в RIPv2 значительно повышает стойкость протокола по сравнению с незащищённым текстовым паролем, описанным выше, алгоритм MD5 считается устаревшим с точки зрения криптографии. Его слабость заключается в уязвимостях его хеш-функции, в первую очередь – наличием эффективных и распространенных методов построения коллизий, когда злоумышленник может создать другой набор данных с тем же хеш-значением. Несмотря на существование более современного и криптостойкого стандарта RFC 4822, описывающего механизм аутентификации на основе HMAC-SHA, ни один из тестируемых маршрутизаторов его не поддерживает на программном уровне. Исходя из этого, практическая рекомендация к использованию MD5 для аутентификации обусловлена не его достаточной степенью защиты, а конкретными ограничениями сетевого оборудования.

Сравнение поведения устройств разных производителей (Cisco, MikroTik, Eltex) показало, что при одинаковой логике защиты их устойчивость к подмене количества hop’ов оказывается сопоставимой. Следует отметить, что заметное отличие в логике настройки было характерно только для MikroTik, в то время как интерфейсы конфигурирования Cisco и Eltex практически идентичны.

**Заключение**

Проведённый анализ протокола RIPv2, стандартов аутентификации и лабораторное моделирование атак с подменой количества hop’ов при помощи Scapy позволили сформулировать следующие выводы по заявленным гипотезам.

Гипотеза 1 подтверждена. RIPv2 без аутентификации или с использованием простой текстовой аутентификации подвержен атакам, основанным на подмене значения количества hop’ов в RIPv2-пакетах: злоумышленник может объявлять свой маршрут к целевым подсетям и нарушать доступность сервисов.

Гипотеза 2 подтверждена. MD5-аутентификация по RFC 2082 эффективно предотвращает подмену количества hop’ов с неавторизованных устройств, но её надёжность зависит от секретности ключей.

Гипотеза 3 подтверждена. Комбинация криптографической аутентификации с топологическими ограничениями и фильтрацией маршрутов существенно повышает устойчивость сети к атакам на основе подмены количества hop’ов и позволяет минимизировать последствия таких атак.

**Чек-лист для безопасного использования RIPv2:**

– анонсировать только необходимые интерфейсы, остальные перевести в режим passive-interface;
– применять MD5-аутентификацию;
– использовать key-chain для управления ключами;
– регулярно менять криптографические ключи;
– исключить возможность попадания злоумышленника в RIP-сегмент;
– использовать анализаторы трафика RIP, регулярно проверять таблицы маршрутизации, логировать изменения.

**Практическая значимость и рекомендации:**

Полученные результаты обладают прикладным характером и могут быть использованы и внедрены в корпоративных и учебных сетях, где задействован RIPv2. На основе анализа сформулированы следующие рекомендации:

– обязательное использование криптографической аутентификации RIPv2. В реальных сетях необходимо применять MD5 или более современные алгоритмы (при наличии поддержки в оборудовании);

– фильтрация маршрутов и контроль метрик. Запрет приёма маршрута по умолчанию от неподходящих соседей, а также аномальных префиксов и метрик;

– регулярный аудит конфигураций и журналов. Проверка наличия аутентификации, отсутствие подозрительных изменений таблиц маршрутизации и нестандартных значений количества hop’ов.

**Направления дальнейших исследований**

Анализ устойчивости RIPv2 аутентификации к современным методам криптоанализа и атакам подбора ключей, в том числе при использовании инструментов пакетной генерации.

Исследование комбинированных атак, совмещающих подмену количества hop’ов в RIPv2 с отравлением ARP, DHCP и другими атаками на протоколы канального и сетевого уровней.

**Библиографический список**

1. Уймин, А. Г. Применение отечественного сетевого оборудования Eltex и EcoRouter в рамках специальности 09.02.06 «Сетевое и системное администрирование». Вопросы импортозамещения и подготовки квалифицированных кадров в сетевом оборудовании / А. Г. Уймин, И. М. Толмачев // Автоматизация и информатизация ТЭК. – 2025. – № 11(628). – С. 58–62. – EDN DMHQJU.
2. Convery, D. Network Security Architectures [Электронный ресурс] / D. Convery, J. Choe // Cisco Press. – 2004. – URL: https://www.ciscopress.com/articles/article.asp?p=102157 (дата обращения: 25.11.2025).
3. Perlman, R. Interconnections: Bridges, Routers, Switches, and Internetworking Protocols / R. Perlman. – 2nd ed. – Reading, Mass. : Addison-Wesley, 2000. – 538 p.
4. Vyncke, E. LAN Switch Security: What Hackers Know About Your Switches / E. Vyncke. – Indianapolis, IN : Cisco Press, 2008. – 336 p.
5. Baker, F. RFC 2082: RIP-2 MD5 Authentication / F. Baker, R. Atkinson. – Internet Engineering Task Force, 1997. – URL: https://datatracker.ietf.org/doc/html/rfc2082 (дата обращения: 25.11.2025).
6. Malkin, G. RFC 2453: RIP Version 2 / G. Malkin. – Internet Engineering Task Force, 1998. – URL: https://datatracker.ietf.org/doc/html/rfc2453 (дата обращения: 25.11.2025).
7. Красноперов, К. Ю. Отличия между протоколами RIPv1 и RIPv2 / К. Ю. Красноперов // Академическая публицистика. – 2023. – № 10-2. – С. 88–90. – EDN BXFIGT.

**References**

1. Uymin, A. G., & Tolmachev, I. M. (2025). Application of domestic network equipment Eltex and EcoRouter within the specialty 09.02.06 "Network and System Administration": Issues of import substitution and training of qualified personnel in network equipment. Automation and Informatization of the Fuel and Energy Complex, (11), 58–62.
2. Convery, D., & Choe, J. (2004). Network security architectures. Cisco Press. Retrieved November 25, 2025, from https://www.ciscopress.com/articles/article.asp?p=102157
3. Perlman, R. (2000). Interconnections: Bridges, routers, switches, and internetworking protocols (2nd ed.). Addison-Wesley.
4. Vyncke, E. (2008). LAN switch security: What hackers know about your switches. Cisco Press.
5. Baker, F., & Atkinson, R. (1997). RIP-2 MD5 authentication (RFC 2082). Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc2082
6. Malkin, G. (1998). RIP version 2 (RFC 2453). Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc2453
7. Krasnoperov, K. Yu. (2023). Differences between RIPv1 and RIPv2 protocols. Akademicheskaya Publitsistika, (10-2), 88–90.

## Информация об авторах

Галустян Д. Г. — студент 3 курса факультета «Комплексная безопасность топливно-энергетического комплекса», ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, e-mail: galustyan.dg@gmail.com.

Соколов С. А. — студент 3 курса факультета «Комплексная безопасность топливно-энергетического комплекса», ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, e-mail: sokolov.sa@gmail.com.


# ISSUES OF ENSURING RIPv2 PROTECTION ON NETWORK DEVICES


**Galustyan D. G.[1], Sokolov S. A.[1]**

[1]National University of Oil and Gas «Gubkin University», Moscow, Russian Federation



**Abstract.** The article examines the security issues of RIPv2 routing protocol operation on network devices and focuses on attacks based on route metric spoofing. The relevance of the study is determined by the fact that RIPv2 is still used in educational, corporate and specialized network segments, while its distance-vector logic makes route selection dependent on the advertised hop count. The paper considers an attacker who has access to a common Layer 2 segment and uses Scapy to generate forged RIPv2 packets with artificially reduced metrics. The methodological basis of the research includes analysis of RIP/RIPv2 specifications, vendor documentation and laboratory modelling on Cisco, MikroTik and Eltex equipment. The experimental part compares three operating modes: RIPv2 without authentication, RIPv2 with a plain-text password and RIPv2 with MD5 cryptographic authentication according to RFC 2082. The results show that the absence of authentication and the use of a readable password do not prevent false route injection, because an attacker can reproduce valid-looking updates and influence routing tables. MD5 authentication rejects unauthorised updates and significantly reduces the risk of traffic interception or loss of connectivity, although its effectiveness depends on key secrecy and equipment support. Practical recommendations are proposed for safer RIPv2 deployment, including key-chain management, route filtering, isolation of RIP segments and regular auditing of routing tables and logs.



**Keywords:** RIPv2, routing, authentication, MD5, Scapy, network security, metric spoofing.



## Information about the authors

Galustyan D. G. — 3rd-year student, Faculty of Integrated Security of the Fuel and Energy Complex, Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: galustyan.dg@gmail.com.

Sokolov S. A. — 3rd-year student, Faculty of Integrated Security of the Fuel and Energy Complex, Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: sokolov.sa@gmail.com.

**Рубрика 3. Методология и технология профессионального образования**

**УДК 004.056:378.147**

**Губина Т. Н.[1], Шмавонян С.А.[2], Уймин А. Г.[3]**
[1]ООО «Базальт СПО», г. Москва, Российская Федерация
[2]ООО «Киберпротект», г. Москва, Российская Федерация
[3]ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, Российская Федерация

# МЕТОДИЧЕСКАЯ СТРУКТУРА КУРСА ПО DLP-СИСТЕМАМ: КОМПЕТЕНТНОСТНЫЙ ПОДХОД К ФОРМИРОВАНИЮ ПРОФЕССИОНАЛЬНЫХ КОМПЕТЕНЦИЙ В ОБЛАСТИ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ С ПРИМЕНЕНИЕМ ОТЕЧЕСТВЕННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

**Аннотация.** В статье обоснована методическая структура учебного курса, направленного на формирование профессиональных компетенций студентов в области корпоративной защиты от внутренних угроз информационной безопасности с использованием DLP-систем (Data Loss Prevention — предотвращение утечек данных). В качестве теоретической основы проектирования курса принят компетентностный подход, предполагающий единство знаний, умений, навыков, практического опыта и профессионально значимых личностных качеств обучающегося. Описана архитектура виртуального лабораторного стенда, включающего три узла: серверный (Windows Server 2025), клиентский на базе Windows 11 и клиентский на базе ОС Альт Рабочая станция 10.4 (разработчик — ООО «Базальт СПО»). Показано, что включение отечественной операционной системы в состав стенда не только соответствует контексту государственной политики импортозамещения, но и расширяет дидактические возможности курса: студенты получают практический опыт настройки и эксплуатации DLP-агента в гетерогенной среде, приближённой к реальным корпоративным условиям. Предложена трёхуровневая логика построения лабораторных работ (20 занятий, 72 академических часа): от освоения базовой инфраструктуры домена — к конфигурированию политик безопасности и далее к анализу инцидентов и цифровой криминалистике. Сформулированы педагогические условия реализации курса (содержательные, процессуальные, организационные, кадровые, нормативно-методические, технологические). Результаты апробации в двух технических вузах указывают на эффективность предложенной методики для формирования у обучающихся комплекса компетенций в области DLP-технологий.

**Ключевые слова:** DLP-система, информационная безопасность, компетентностный подход, лабораторный практикум, виртуальный стенд, ОС «Альт», инсайдерская угроза.

## Введение

Цифровизация российской экономики и последовательный курс государства на технологический суверенитет формируют новые требования к подготовке специалистов в области информационной безопасности. Исследования по выявлению и прогнозированию внутренних угроз показывают устойчивую значимость инсайдерского фактора для корпоративной безопасности [1, 2]. В этих условиях системы предотвращения утечек данных (Data Loss Prevention, DLP) становятся одним из ключевых элементов защиты информации на предприятиях, в органах государственного управления и в инфраструктурах, обрабатывающих данные ограниченного доступа.

Между тем анализ требований работодателей показывает устойчивую потребность в специалистах, способных не только эксплуатировать DLP-решения, но и проектировать политики защиты, настраивать правила контентного анализа, расследовать инциденты и выявлять источники нарушений. Существующие образовательные программы по направлениям «Информационная безопасность» (10.05.03, 10.05.04) и «Компьютерная безопасность» (10.05.01) нередко ограничиваются теоретическим ознакомлением с классом DLP-решений и не всегда предлагают студентам систематизированный практикум на реальном программном обеспечении в условиях, приближённых к корпоративной среде.

Дополнительный вызов связан с переходом российских организаций с зарубежного системного программного обеспечения на отечественные альтернативы. Указ Президента Российской Федерации от 30 марта 2022 г. № 166 и ограничения на закупку иностранного программного обеспечения для государственных нужд задают нормативный контекст, в котором владение отечественными программными платформами становится значимым профессиональным требованием. Поэтому выпускник должен уметь работать с DLP-системами не

только в среде Microsoft Windows, но и в гетерогенной инфраструктуре, включающей Linux-дистрибутивы отечественного производства.

Цель настоящей статьи — обосновать методическую структуру курса практической подготовки по DLP-технологиям, в котором применение отечественной ОС «Альт Рабочая станция» (разработчик — ООО «Базальт СПО») встроено в учебный процесс как средство формирования профессиональных компетенций, а не как самоцель импортозамещения. Теоретической основой проектирования курса служит компетентностный подход, последовательно разработанный в трудах И.А. Зимней [3], Э.Ф. Зеера [4], В.А. Болотова и В.В. Серикова [5].

## 1. Теоретические основы проектирования курса

### 1.1. DLP-системы: понятие, архитектура, классификация

Термин Data Loss Prevention (DLP) обозначает класс программных решений, предназначенных для обнаружения, мониторинга и предотвращения несанкционированного использования, хранения и передачи конфиденциальной информации [6]. В прикладной литературе DLP-система рассматривается как продукт, который на основе централизованных политик идентифицирует, отслеживает и защищает данные в состоянии хранения, движения и использования посредством глубокого анализа содержимого [6]. Принципиально важно, что объектом защиты выступают именно данные в трёх состояниях: data at rest (хранящиеся), data in motion (передаваемые по сети) и data in use (обрабатываемые на рабочей станции).


**Рис. 1. Обобщённая схема работы DLP-системы**

В зависимости от точки контроля DLP-системы традиционно классифицируют на три архитектурных типа: хостовые (Endpoint DLP), сетевые (Network/Gateway DLP) и гибридные [7, 8]. Хостовый DLP-агент устанавливается непосредственно на рабочую станцию пользователя и контролирует операции с данными на уровне операционной системы: перехватывает системные вызовы, отслеживает обращения к буферу обмена, съёмным носителям, принтерам, мессенджерам и веб-браузерам. Сетевой DLP-компонент анализирует трафик на уровне шлюза, выявляя попытки передачи конфиденциальных данных по почте, в облачные хранилища и через иные сетевые каналы. Гибридные решения объединяют оба подхода, обеспечивая комплексную защиту периметра [9].

Для идентификации конфиденциальных данных современные DLP-системы используют несколько методов контентного анализа: регулярные выражения и словари, цифровые отпечатки документов (document fingerprinting), статистический анализ, методы машинного обучения [10]. Цифровой отпечаток формируется на основе математического преобразования эталонного документа и позволяет обнаружить его копию или модифицированную версию даже при частичном изменении содержимого.

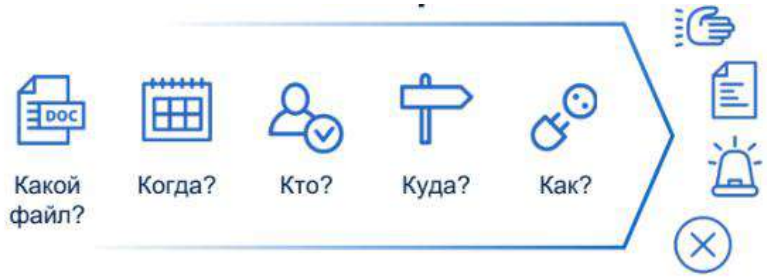


**Рис. 2. Контекстный контроль в DLP-системе**

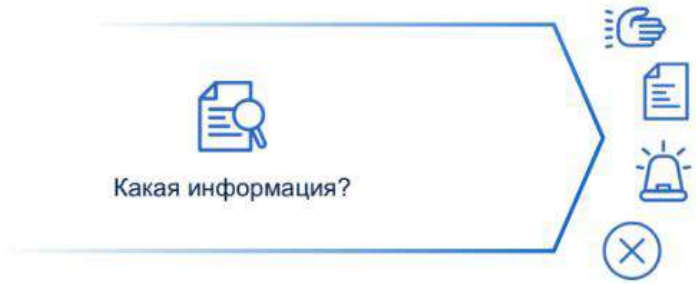


**Рис. 3. Контентно-зависимый контроль в DLP-системе**

**Таблица 1 – Классификация DLP-систем по архитектурному принципу**

| Хостовый (Endpoint DLP) | Сетевой (Network/Gateway DLP) | Гибридный (Hybrid DLP) |
|---|---|---|
| Агент на рабочей станции | Шлюз/прокси на периметре | Агент + шлюз |
| Контроль операций ОС | Анализ сетевого трафика | Комплексная защита |
| Съёмные носители, печать, буфер обмена | E-mail, HTTP/S, FTP, мессенджеры | Все каналы одновременно |
| Пример: Кибер Протего (агентский модуль) | Пример: шлюзовой модуль перехвата почты | Пример: Кибер Протего v11.0 (полное развёртывание) |

## 1.2. Компетентностный подход в подготовке специалистов по информационной безопасности

Компетентностный подход, сложившийся в российской педагогике на рубеже XX–XXI вв. в трудах И.А. Зимней, Э.Ф. Зеера, В.А. Болотова, В.И. Байденко и других учёных, постулирует, что образовательный результат должен выражаться не в объёме усвоенных знаний, а в готовности и способности применять их для решения реальных профессиональных задач [3, 4, 5]. Применительно к подготовке специалистов по информационной безопасности это означает, что выпускник должен не просто знать устройство DLP-системы, но уметь самостоятельно развернуть её, настроить политики под конкретный профиль угроз и провести расследование инцидента.

В структуре профессиональной компетенции специалиста по DLP-технологиям выделяются пять обязательных компонентов [3, 4, 11]:

– знания: понятийный аппарат информационной безопасности, принципы работы DLP-агентов, методы контентного анализа, протоколы сетевого взаимодействия, требования регуляторов (ФСТЭК, ФСБ, ГОСТ Р 57580);
– умения: развёртывать и конфигурировать DLP-систему в доменной среде, разрабатывать правила обнаружения конфиденциальных данных, настраивать каналы перехвата, анализировать журналы событий и отчёты системы;
– навыки: устойчивое выполнение типовых операций — добавление агентов в консоль управления, настройка групповых политик, экспорт теневых копий;
– практический опыт: применение знаний, умений и навыков в нестандартных ситуациях — выявление скрытых каналов утечки, расследование инцидентов по неполным данным, восстановление работоспособности DLP-компонентов и проверка целостности журналов событий;
– профессионально значимые личностные качества: аккуратность и методичность при работе с конфиденциальной информацией, стрессоустойчивость при анализе инцидентов, ответственность за принятые решения по блокировке действий пользователей.

Дополнительно в составе профессиональной компетенции специалиста ИБ необходимо учитывать гибкость и адаптивность — способность применять освоенные методы защиты в условиях смены технологического стека, включая переход с зарубежных платформ на отечественные. Именно это качество оказывается наиболее востребованным в условиях политики импортозамещения.

Исследования в области компетентностной подготовки специалистов по кибербезопасности подтверждают, что формирование устойчивых компетенций требует обязательного включения практических лабораторных занятий, воспроизводящих реальную корпоративную среду [11, 12]. A. Alammari и соавторы показали, что компетентностная оценка в области кибербезопасности должна охватывать три измерения KSA (knowledge, skills, abilities — знания, навыки, способности) и быть привязана к конкретным профессиональным задачам [11]. V.R. Kebande отмечает, что использование виртуальных лабораторий повышает вовлечённость студентов и активность обучения при дистанционном изучении кибербезопасности [12].

## 2. Методическая структура курса

### 2.1. Концепция курса и целевые компетенции

Разработанный курс «Корпоративная защита от внутренних угроз информационной безопасности с использованием современных DLP-технологий» ориентирован на обучающихся старших курсов специалитета по направлениям подготовки 10.05.03 «Информационная безопасность автоматизированных систем», 10.05.04 «Информационно-аналитические системы безопасности», а также на студентов СПО по специальностям 10.02.04 и 10.02.05. Объём курса составляет 72 академических часа, реализованных в форме 20 лабораторных работ [19]. Лекционная составляющая встроена в лабораторный формат в виде вводных теоретических блоков к каждой работе.

Курс направлен на формирование следующих профессиональных компетенций ФГОС ВО: ОПК-12 (способность применять средства защиты информации от несанкционированного доступа), ОПК-13 (способность осуществлять контроль и мониторинг защищённости информационных систем), ОПК-14 (способность проводить анализ и оценку угроз информационной безопасности) и ОПК-15 (способность разрабатывать политики и регламенты информационной безопасности). Перечисленные компетенции раскрываются через пятикомпонентный состав (знания, умения, навыки, практический опыт, личностные качества), описанный в разделе 1.2.

### 2.2. Виртуальный лабораторный стенд

Центральным педагогическим средством курса является виртуальный лабораторный стенд, развёрнутый на гипервизоре Альт Виртуализация. Выбор платформы виртуализации обусловлен её открытостью, доступностью для образовательных учреждений и возможностью развёртывания в типовой серверной конфигурации учебных вычислительных центров. Использование виртуальных лабораторий в обучении информационной безопасности обосновано в ряде исследований [12, 13, 14]: они обеспечивают изолированную среду для проведения экспериментов, снижают риск повреждения реальной инфраструктуры и позволяют воспроизводить разнообразные сценарии угроз.

Стенд включает три виртуальные машины (рис. 4, табл. 2–3):

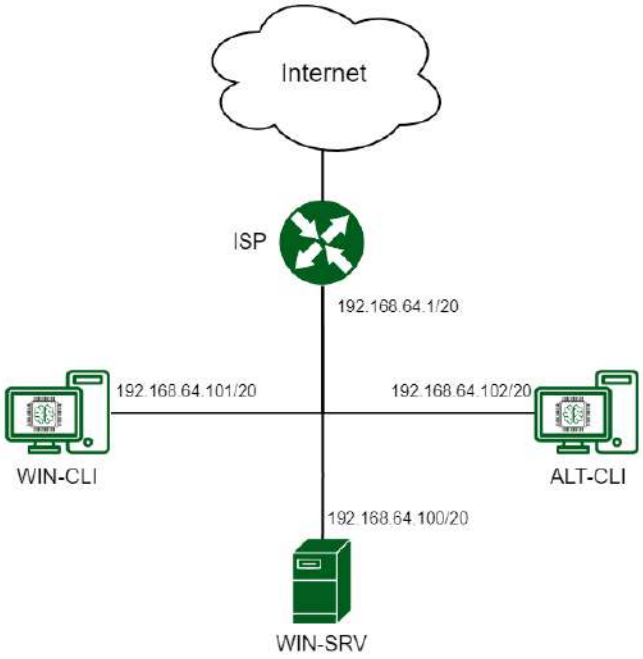


**Рис. 4. Топология виртуального лабораторного стенда**

**Таблица 2 – Состав и назначение узлов лабораторного стенда**

| Узел | ОС / Роль | IP-адрес | Компоненты DLP |
|---|---|---|---|
| WIN-SRV | Windows Server 2025<br>Контроллер домена AD DS/AD CS<br>Сервер управления DLP | 192.168.64.100 | Management Server<br>Search & Discovery Server<br>Group Policy Manager |
| WIN-CLI | Windows 11<br>Клиентская рабочая станция<br>(член домена) | 192.168.64.101 | DLP-агент (Endpoint)<br>Agent Settings Editor<br>Central Management Console |
| ALT-CLI | ОС Альт Рабочая станция 10.4<br>(Базальт СПО)<br>Клиентская рабочая станция | 192.168.64.102 | DLP-агент для Linux<br>Web Console (управление агентом через браузер) |

**Таблица 3 – Минимальные ресурсы виртуальных машин лабораторного стенда**

| Наименование | HDD, ГБ | RAM, ГБ | CPU, ядер | Операционная система |
|---|---|---|---|---|
| WIN-CLI | 80 | 4 | 2 | Windows 11 |
| WIN-SRV | 80 | 4 | 2 | Windows Server 2025 |
| ALT-CLI | 50 | 2 | 1 | Альт Рабочая станция 10.4 |

Архитектурное решение с двумя клиентскими узлами на разных операционных системах воспроизводит типовую гетерогенную корпоративную среду, в которой наряду с рабочими станциями Windows присутствуют Linux-машины разработчиков, серверы и специализированные АРМ. Такая конфигурация особенно актуальна для организаций, находящихся в процессе перехода на отечественное программное обеспечение.

### 2.3. Трёхуровневая структура лабораторных работ

Двадцать лабораторных работ курса организованы в три последовательных уровня, каждый из которых соответствует определённому этапу формирования компетенции. Логика построения опирается на принцип восходящей сложности: каждый последующий уровень предполагает не только новые знания, но и опору на навыки, сформированные на предыдущем [15, 16].

**Таблица 4 – Трёхуровневая структура лабораторных работ курса**

| Уровень / ЛР | Тематика | Формируемые компоненты | Педагогические средства |
|---|---|---|---|
| I. Инфраструктура (ЛР 1–5) 10 ч. | Установка и настройка ОС. Развёртывание AD DS, AD CS. Настройка ОС Альт, вступление в домен. | Знания: архитектура домена, роли сервера. Умения: развёртывать доменную среду. | Инструктивная карта; пошаговый лабораторный практикум; самопроверка. |
| II. Установка и конфигурация DLP (ЛР 6–14) 36 ч. | Развёртывание компонентов Кибер Протего v11.0. Настройка агентов на Windows и ОС Альт. Правила и политики DLP. | Умения: конфигурировать DLP-систему. Навыки: работа с консолью управления. Опыт: гетерогенная среда. | Ситуационные задачи; работа с реальным ПО; групповое обсуждение; отчёт по результатам. |
| III. Политики, инциденты, форензика (ЛР 15–20) 26 ч. | Контентный анализ, цифровые отпечатки. Анализ инцидентов. Досье пользователя. Теневое копирование. | Практический опыт: расследование инцидентов. Личностные качества: ответственность, аккуратность. | Кейс-метод; ролевая игра «следователь / нарушитель»; портфолио отчётов; защита результатов. |

Первый уровень (ЛР 1–5) посвящён развёртыванию базовой доменной инфраструктуры: установке и начальной настройке операционных систем на всех трёх узлах стенда, развёртыванию служб Active Directory Domain Services (AD DS) и Active Directory Certificate Services (AD CS), вступлению клиентских машин в домен, включая ОС Альт через механизм SSSD/Kerberos. На этом уровне у студентов формируются знания об архитектуре доменной среды и базовые умения её развёртывания.

Второй уровень (ЛР 6–14) — центральный и наиболее объёмный (36 часов). Студенты последовательно устанавливают и настраивают компоненты DLP-системы Кибер Протего v11.0 (разработчик — ООО «Киберпротект»): сервер управления (Management Server), поисковый сервер (Search and Discovery Server), агент на Windows-клиенте, агент на ОС «Альт» с управлением через веб-консоль. Особое внимание уделяется конфигурированию политик контроля, аудита и перехвата: настройке правил для каналов e-mail, USB-носителей, буфера обмена, печати и мессенджеров. На этом уровне формируются умения и навыки работы с DLP-консолью, а также практический опыт эксплуатации системы в гетерогенной среде.


**Рис. 5. Каналы контроля DLP-политик в Кибер Протего**

Третий уровень (ЛР 15–20) ориентирован на формирование практического опыта анализа инцидентов и цифровой криминалистики. Студенты создают базу цифровых отпечатков конфиденциальных документов, настраивают правила контентного анализа, моделируют сценарии утечки данных и проводят их расследование с использованием досье пользователя, теневых копий и журналов событий. Применение кейс-метода и ролевых игр на этом уровне обеспечивает формирование профессионально значимых личностных качеств: аккуратности, ответственности и способности работать в условиях неполноты данных.

## 3. Роль отечественной операционной системы в дидактической структуре курса

### 3.1. Импортозамещение как педагогический контекст

Включение ОС «Альт Рабочая станция» 10.4 в состав лабораторного стенда продиктовано не только нормативным контекстом, но и педагогической логикой. Начиная с 2022 года переход на отечественное системное программное обеспечение приобрёл характер государственной задачи. Указ Президента РФ от 30 марта 2022 г. № 166 и Постановление Правительства РФ от 16 ноября 2015 г. № 1236 с последующими изменениями ограничивают использование иностранного программного обеспечения в ряде государственных и критически значимых контуров. В образовательном процессе это означает необходимость готовить выпускников к работе в смешанных инфраструктурах, где отечественные ОС сосуществуют с Windows-решениями.

Бурняшов в своём исследовании показал, что переход российских вузов на отечественное программное обеспечение в учебном процессе сопряжён с рядом системных трудностей: дефицитом методических материалов, необходимостью переподготовки преподавательского состава и сопротивлением со стороны студентов, привыкших к зарубежным платформам [17]. Разработанный курс решает эту проблему иначе: ОС «Альт» вводится не как замена Windows, а как органичная часть профессионального контекста — корпоративной среды, в которой сосуществуют разные платформы, и специалист по ИБ обязан уметь работать с каждой из них.

Продуктовая линейка Базальт СПО — ОС «Альт Рабочая станция», ОС «Альт Сервер»— представлена в реестре отечественного программного обеспечения Минцифры России и применяется в государственных, образовательных и корпоративных организациях. Использование этого дистрибутива в учебном курсе повышает соответствие практической подготовки будущей профессиональной деятельности выпускников [18].

### 3.2. Дидактические преимущества гетерогенной среды

Ключевой дидактический эффект от включения ОС «Альт» в состав стенда состоит в формировании у студентов более глубокого понимания механизмов работы DLP-системы. Если установка и настройка DLP-агента в Windows-среде во многом сводится к запуску установочного пакета и выбору параметров в графическом интерфейсе, то настройка агента в ОС «Альт» требует работы с командной строкой, понимания структуры файловой системы Linux, механизмов systemd, правил SELinux/AppArmor и процедур аутентификации через SSSD/Kerberos.

Таким образом, Linux-клиент выполняет в курсе двойную дидактическую функцию: во-первых, он является объектом практики по освоению отечественной ОС; во-вторых, он помогает явно раскрыть архитектурные особенности DLP-системы, которые остаются менее заметными при работе исключительно в среде Windows. Студент, успешно настроивший DLP-агент под управлением веб-консоли на ОС «Альт», демонстрирует более глубокое понимание серверно-клиентского взаимодействия в DLP-системе, чем студент, ограничившийся только Windows-агентом.

Опыт апробации курса показал, что задания, связанные с ОС «Альт» (ввод машины в домен, развёртывание Linux-агента, настройка веб-консоли и применение политики к Linux-клиенту), оказались наиболее значимыми с точки зрения диагностики реального уровня профессиональной компетентности. Студенты, демонстрировавшие поверхностные знания в Windows-ориентированных работах, чаще всего обнаруживали пробелы именно при переходе к Linux-среде.

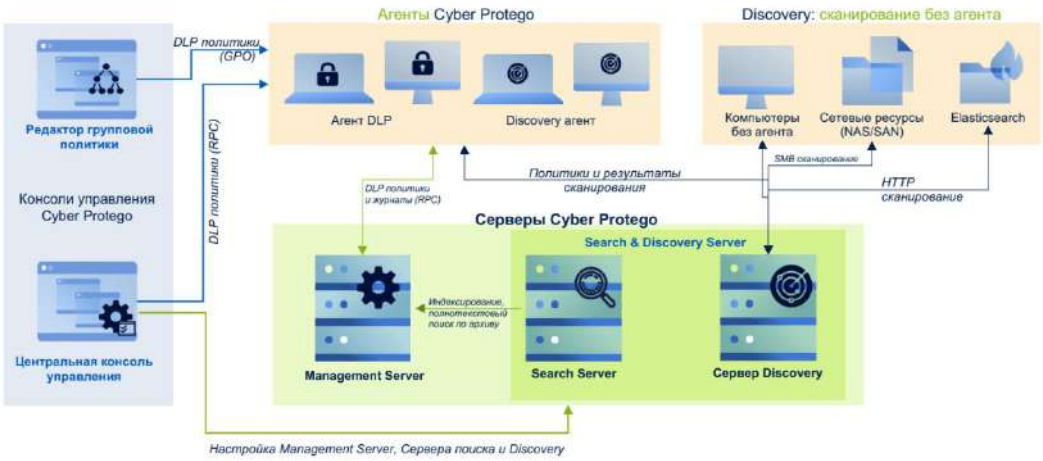


**Рис. 6. Состав и взаимодействие компонентов Кибер Протего**

**Таблица 5 – Функциональное распределение компонентов Кибер Протего на узлах лабораторного стенда**

| Узел стенда | Операционная система и роль | Компоненты Кибер Протего | Основные взаимодействия |
|---|---|---|---|
| WIN-SRV (192.168.64.100) | Windows Server 2025; контроллер домена AD DS/AD CS; серверный узел DLP | Management Server; Search & Discovery Server; Group Policy Manager | Распространение политик, сбор агентских отчётов, хранение журналов и теневых копий |
| WIN-CLI (192.168.64.101) | Windows 11; клиентская рабочая станция, член домена | Endpoint Agent; Agent Settings Editor; Central Management Console | Получение политик, контроль локальных каналов, отправка событий на сервер управления |
| ALT-CLI (192.168.64.102) | ОС «Альт Рабочая станция» 10.4; Linux-клиент | Linux Agent; Web Console для управления агентом | Работа через HTTPS-веб-консоль, получение политик, передача агентских отчётов |

## 4. Педагогические условия реализации курса

Эффективность курса обеспечивается совокупностью педагогических условий, которые охватывают все аспекты образовательного процесса: от содержания учебного материала до нормативной базы и технологического оснащения. В соответствии с устоявшейся традицией профессиональной педагогики выделяется шесть видов педагогических условий [3, 4].

### 1. Содержательные условия

Содержание курса структурировано на основе реальных профессиональных задач специалиста по защите информации от инсайдерских угроз. Каждая лабораторная работа имеет чёткую профессиональную направленность: студент не «изучает систему ради системы», а решает конкретную учебно-профессиональную задачу — развернуть защищённую доменную среду, расследовать инцидент утечки, создать базу цифровых отпечатков конфиденциальных документов. Включение учебно-профессиональных задач разной степени сложности — от типовых до нестандартных — обеспечивает формирование всех пяти компонентов профессиональной компетенции.

### 2. Процессуальные условия

Процесс формирования компетенций разворачивается в определённой последовательности: диагностическое тестирование входного уровня → вводный теоретический блок → выполнение лабораторной работы → защита отчёта → рефлексивное обсуждение. Диагностика уровня сформированности компетенции осуществляется на трёх срезах: после первого уровня (ЛР 1–5), после второго уровня (ЛР 6–14) и по завершении курса (ЛР 15–20). Инструменты диагностики — практические задания в виде сценариев-инцидентов, которые студент должен воспроизвести и расследовать на стенде в ограниченное время.

**3. Организационные условия**

Курс реализуется в условиях цифровой образовательной среды учебного вычислительного центра, оснащённого серверным оборудованием, достаточным для одновременного функционирования лабораторных стендов для всей учебной группы. Обязательным организационным условием является обеспечение изолированности сетевой среды каждого стенда во избежание взаимного влияния лабораторных работ разных студентов. Привлечение представителей работодателей (компаний в сфере ИБ) к разработке сценариев инцидентов третьего уровня повышает профессиональную значимость учебных задач.

**4. Кадровые условия**

Преподаватель курса должен обладать актуальным практическим опытом работы с корпоративными DLP-системами и знанием как Windows-экосистемы, так и Linux. Рекомендуется прохождение дополнительной профессиональной подготовки по программам сертификации продукта (ООО «Киберпротект») и повышение квалификации не реже одного раза в три года с учётом обновлений версий используемого программного обеспечения. Желательно наличие у преподавателя опыта работы в корпоративных службах информационной безопасности.

**5. Нормативно-методические условия**

Курс разработан в соответствии с требованиями ФГОС ВО по направлениям 10.05.03 и 10.05.04, а также ФГОС СПО 10.02.04 и 10.02.05. Тематика лабораторных работ согласована с приказами ФСТЭК России, регламентирующими требования к средствам защиты от несанкционированного доступа и мониторинга информационной безопасности. Локальная нормативная база включает рабочую программу дисциплины, методические указания к каждой лабораторной работе и фонд оценочных средств, охватывающий все заявленные компетенции.

**6. Технологические условия**

Виртуализация на платформе Альт Виртуализация обеспечивает воспроизводимость учебной среды: перед началом каждого занятия стенд восстанавливается из эталонного снапшота, что гарантирует одинаковые стартовые условия для всех студентов. Использование ОС «Альт Рабочая станция» 10.4 соответствует контексту государственной политики импортозамещения и обеспечивает доступность лицензионного программного обеспечения для образовательных учреждений. DLP-система Кибер Протего v11.0 включена в единый реестр российских программ для ЭВМ и баз данных Минцифры России, что подтверждает возможность её использования в государственных образовательных организациях.

**5. Результаты апробации**

Курс прошёл апробацию в двух технических университетах. Апробация проводилась в формате опытно-экспериментальной работы с выделением контрольной и экспериментальной групп. Контрольная группа изучала тематику DLP-систем в традиционном формате (лекции + единичные лабораторные работы на демонстрационном стенде преподавателя). Экспериментальная группа проходила полный курс из 20 лабораторных работ на индивидуальных виртуальных стендах.

Диагностика уровня сформированности компетенций проводилась по четырём критериям: мотивационно-ценностному (отношение к задачам профессиональной защиты информации), когнитивному (знание архитектуры и принципов работы DLP-систем), деятельностному (способность самостоятельно выполнить типовую задачу на стенде) и рефлексивному (способность оценить результат и скорректировать действия). Уровни сформированности: базовый, практический, продвинутый.

По результатам итоговой диагностики в экспериментальной группе доля студентов, достигших практического и продвинутого уровней, составила 78 %, в контрольной — 31 %; по когнитивному компоненту приведены 82 % и 61 %. Авторы рассматривают эти показатели как описательные результаты пилотной апробации, а не как доказательство причинного эффекта, поскольку размеры групп, абсолютные значения, диагностический инструмент и статистическая обработка в исходных материалах не приведены.

Включение ОС «Альт» в состав стенда первоначально вызвало затруднения у части студентов, однако после выполнения заданий по доменному подключению, развёртыванию

Linux-агента и настройке веб-консоли понимание архитектуры DLP-системы заметно улучшалось. В анкетах студенты отмечали, что именно работа с Linux-клиентом помогла им понять, как DLP-агент взаимодействует с операционной системой на уровне ниже графического интерфейса.

**Заключение**

В статье обоснована методическая структура курса практической подготовки специалистов по DLP-технологиям, отличительными чертами которого являются: компетентностная ориентированность (единство знаний, умений, навыков, практического опыта и личностных качеств), практикоориентированность (20 лабораторных работ на индивидуальном виртуальном стенде), трёхуровневая прогрессия сложности и гетерогенная программно-аппаратная среда, включающая отечественную операционную систему ОС «Альт».

Показано, что применение ОС «Альт Рабочая станция» 10.4 в учебном стенде выполняет двойную функцию: нормативную (соответствие требованиям государственной политики импортозамещения) и дидактическую (углублённое понимание архитектурных механизмов DLP-системы через практику в Linux-среде). Именно сочетание этих двух функций отличает предложенный подход от простого «замещения» одной платформы другой.

Результаты апробации в двух технических университетах показывают выраженные различия в уровне сформированности деятельностного компонента профессиональной компетенции между экспериментальной и контрольной группами. Перспективы дальнейших исследований связаны с разработкой методики оценки рефлексивного компонента компетенции, масштабированием курса на программы СПО и адаптацией сценариев инцидентов к отраслевой специфике предприятий — потенциальных работодателей.

**Библиографический список**

1. Sarhan, B. B. Insider Threat Detection Using Machine Learning Approach / B. B. Sarhan, N. Altwaijry // Applied Sciences. – 2022. – Vol. 13, № 1. – P. 259. – DOI: 10.3390/app13010259.
2. Gheyas, I. A. Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis / I. A. Gheyas, A. E. Abdallah // Big Data Analytics. – 2016. – Vol. 1, № 1. – P. 6. – DOI: 10.1186/s41044-016-0006-0.
3. Зимняя, И. А. Ключевые компетенции — новая парадигма результата образования / И. А. Зимняя // Высшее образование сегодня. – 2003. – № 5. – С. 34–42.
4. Зеер, Э. Ф. Психология профессионального образования / Э. Ф. Зеер. – Москва : Академия, 2009. – 384 с.
5. Болотов, В. А. Компетентностная модель: от идеи к образовательной программе / В. А. Болотов, В. В. Сериков // Педагогика. – 2003. – № 10. – С. 8–14.
6. Liu, S. Data Loss Prevention / S. Liu, R. Kuhn // IT Professional. – 2010. – Vol. 12, № 2. – P. 10–13. – DOI: 10.1109/MITP.2010.52.
7. Alsuwaie, A. Data Leakage Prevention Adoption Model & DLP Maturity Level Assessment / A. Alsuwaie, B. Habibnia, P. Gladyshev // Proceedings of the International Symposium on Computer Science and Intelligent Controls. – IEEE, 2021. – P. 396–405.
8. Balinsky, H. System Call Interception Framework for Data Leak Prevention / H. Balinsky, D. Subiros Perez, S. J. Simske // Proceedings of the IEEE 15th International Enterprise Distributed Object Computing Conference. – IEEE, 2011. – P. 139–148. – DOI: 10.1109/EDOC.2011.25.
9. Data Loss Prevention Solution for Linux Endpoint Devices // Proceedings of the ACM Conference on Data and Application Security and Privacy. – ACM, 2023. – DOI: 10.1145/3600160.3605036.
10. Gupta, K. A Learning Oriented DLP System based on Classification Model / K. Gupta, A. Kush // arXiv preprint. – 2023. – arXiv:2312.13711.
11. Alammari, A. Developing and evaluating cybersecurity competencies for students in computing programs / A. Alammari, O. Sohaib, S. Younes // PeerJ Computer Science. – 2022. – Vol. 8. – e827. – DOI: 10.7717/peerj-cs.827.
12. Kebande, V. R. The Impact of Virtual Laboratories on Active Learning and Engagement in Cybersecurity Distance Education / V. R. Kebande // arXiv preprint. – 2024. – arXiv:2404.04952.

13. Willems, C. Online assessment for hands-on cyber security training in a virtual lab / C. Willems, C. Meinel // Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON). – IEEE, 2012. – DOI: 10.1109/EDUCON.2012.6201104.
14. Pirta-Dreimane, R. Application of intervention mapping in cybersecurity education design / R. Pirta-Dreimane [et al.] // Frontiers in Education. – 2022. – Vol. 7. – DOI: 10.3389/feduc.2022.998335.
15. Švábenský, V. Scalable Learning Environments for Teaching Cybersecurity Hands-on / V. Švábenský [et al.] // Proceedings of the 2021 IEEE Frontiers in Education Conference. – IEEE, 2021. – DOI: 10.1109/FIE49875.2021.9637180.
16. Alsmadi, I. Practical Information Security: A Competency-Based Education Course / I. Alsmadi. – Cham : Springer, 2018. – DOI: 10.1007/978-3-319-72119-4.
17. Бурняшов, Б. А. Импортозамещение программного обеспечения в образовательном процессе российских вузов / Б. А. Бурняшов // Информатика и образование. – 2022. – Т. 37, № 1. – С. 27–36. – DOI: 10.32517/0234-0453-2022-37-1-27-36.
18. Бурняшов, Б. А. Отечественные облачные пакеты офисных приложений в образовательном процессе вузов / Б. А. Бурняшов // Информатика и образование. – 2023. – Т. 38, № 2. – С. 5–15. – DOI: 10.32517/0234-0453-2023-38-2-5-15.
19. Уймин, А. Г. Практикум. Корпоративная защита от внутренних угроз информационной безопасности с использованием современных DLP-технологий : учебное пособие / А. Г. Уймин. – Москва : РГУ нефти и газа (НИУ) имени И. М. Губкина, 2025.

**References**

1. Sarhan, B. B., & Altwaijry, N. (2022). Insider threat detection using machine learning approach. Applied Sciences, 13(1), 259. https://doi.org/10.3390/app13010259
2. Gheyas, I. A., & Abdallah, A. E. (2016). Detection and prediction of insider threats to cyber security: A systematic literature review and meta-analysis. Big Data Analytics, 1(1), 6. https://doi.org/10.1186/s41044-016-0006-0
3. Zimnyaya, I. A. (2003). Key competencies as a new paradigm of educational outcomes. Higher Education Today, (5), 34–42.
4. Zeer, E. F. (2009). Psychology of professional education. Akademiya.
5. Bolotov, V. A., & Serikov, V. V. (2003). Competency model: From idea to educational program. Pedagogy, (10), 8–14.
6. Liu, S., & Kuhn, R. (2010). Data loss prevention. IT Professional, 12(2), 10–13. https://doi.org/10.1109/MITP.2010.52
7. Alsuwaie, A., Habibnia, B., & Gladyshev, P. (2021). Data leakage prevention adoption model & DLP maturity level assessment. In Proceedings of the International Symposium on Computer Science and Intelligent Controls (pp. 396–405). IEEE.
8. Balinsky, H., Subiros Perez, D., & Simske, S. J. (2011). System call interception framework for data leak prevention. In Proceedings of the IEEE 15th International Enterprise Distributed Object Computing Conference (pp. 139–148). IEEE. https://doi.org/10.1109/EDOC.2011.25
9. Data loss prevention solution for Linux endpoint devices. (2023). In Proceedings of the ACM Conference on Data and Application Security and Privacy. ACM. https://doi.org/10.1145/3600160.3605036
10. Gupta, K., & Kush, A. (2023). A learning oriented DLP system based on classification model. arXiv. https://arxiv.org/abs/2312.13711
11. Alammari, A., Sohaib, O., & Younes, S. (2022). Developing and evaluating cybersecurity competencies for students in computing programs. PeerJ Computer Science, 8, e827. https://doi.org/10.7717/peerj-cs.827
12. Kebande, V. R. (2024). The impact of virtual laboratories on active learning and engagement in cybersecurity distance education. arXiv. https://arxiv.org/abs/2404.04952

13. Willems, C., & Meinel, C. (2012). Online assessment for hands-on cyber security training in a virtual lab. In Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON). IEEE. https://doi.org/10.1109/EDUCON.2012.6201104
14. Pirta-Dreimane, R., et al. (2022). Application of intervention mapping in cybersecurity education design. Frontiers in Education, 7. https://doi.org/10.3389/feduc.2022.998335
15. Švábenský, V., et al. (2021). Scalable learning environments for teaching cybersecurity hands-on. In Proceedings of the 2021 IEEE Frontiers in Education Conference. IEEE. https://doi.org/10.1109/FIE49875.2021.9637180
16. Alsmadi, I. (2018). Practical information security: A competency-based education course. Springer. https://doi.org/10.1007/978-3-319-72119-4
17. Burnyashov, B. A. (2022). Import substitution of software in the educational process of Russian universities. Informatics and Education, 37(1), 27–36. https://doi.org/10.32517/0234-0453-2022-37-1-27-36
18. Burnyashov, B. A. (2023). Domestic cloud office application suites in the educational process of universities. Informatics and Education, 38(2), 5–15. https://doi.org/10.32517/0234-0453-2023-38-2-5-15
19. Uymin, A. G. (2025). Practicum. Corporate protection against internal information security threats using modern DLP technologies: Textbook. Gubkin Russian State University of Oil and Gas.

### Информация об авторах

Губина Татьяна Николаевна — кандидат педагогических наук, руководитель направления по работе с образовательными организациями, ООО «Базальт СПО», г. Москва, e-mail: gubina@basealt.ru.

Шмавонян Саркис Артушевич — руководитель группы по работе с образовательными организациями, ООО «Киберпротект», г. Москва, Российская Федерация, e-mail: Sarkis.Shmavonyan@cyberprotect.ru.

Уймин Антон Григорьевич — старший преподаватель кафедры безопасности информационных технологий, ФГАОУ ВО «РГУ нефти и газа (НИУ) имени И. М. Губкина», г. Москва, e-mail: au-mail@ya.ru.

## METHODOLOGICAL STRUCTURE OF A COURSE ON DLP SYSTEMS: A COMPETENCY-BASED APPROACH TO DEVELOPING PROFESSIONAL COMPETENCIES IN INFORMATION SECURITY USING DOMESTIC SOFTWARE

**Gubina T. N.[1], Shmavonyan S.A.[2], Uymin A. G.[3]**

[1]BaseALT LLC, Moscow, Russian Federation
[2]Cyberprotect LLC, Moscow, Russian Federation
[3]Gubkin Russian State University of Oil and Gas (National Research University), Moscow, Russian Federation

**Abstract.** This paper presents the methodological structure of an academic course aimed at developing professional competencies in protection against insider threats to corporate information security using DLP (Data Loss Prevention) systems. A competency-based approach is adopted as the theoretical foundation for course design, encompassing the unity of knowledge, skills, practical abilities, professional experience, and professionally significant personal qualities of the learner. The architecture of a virtual laboratory stand is described, comprising three nodes: a server running Windows Server 2025, a client workstation running Windows 11, and a client workstation running Alt Workstation 10.4 (developed by BaseALT LLC). The paper shows that incorporating a domestic operating system into the stand not only corresponds to the state import-substitution policy context but also expands the didactic capabilities of the course: students gain practical experience in configuring and operating a DLP agent in a heterogeneous environment close to real corporate conditions. A three-level logic for constructing laboratory assignments is proposed (20 sessions, 72 academic hours): from mastering basic domain infrastructure to configuring security policies and then to incident analysis and digital forensics. Pedagogical conditions for implementing the course are formulated (content-related, procedural, organisational, personnel-related, regulatory-methodological, and technological). Pilot results from two technical universities indicate the effectiveness of the proposed methodology for developing students' competencies in DLP technologies.

**Keywords:** DLP system, information security, competency-based approach, laboratory practicum, virtual stand, Alt OS, insider threat.

## Information about the authors

Gubina Tatiana Nikolaevna — Candidate of Pedagogical Sciences, Head of Educational Organizations Relations, BaseALT LLC, Moscow, e-mail: gubina@basealt.ru.

Shmavonyan Sarkis Artushevich — Head of the group for work with educational organizations, Cyberprotect LLC, Moscow, Russian Federation, e-mail: Sarkis.Shmavonyan@cyberprotect.ru.

Uymin Anton Grigorevich — Senior Lecturer of the Department of Information Technology Security, Gubkin Russian State University of Oil and Gas (National Research University), Moscow, e-mail: au-mail@ya.ru.

## УСЛОВИЯ И ПОРЯДОК НАПРАВЛЕНИЯ МАТЕРИАЛОВ

Редакция научного журнала «ПРОФЕССИОНАЛИТЕТ» принимает к рассмотрению статьи, исправленные версии рукописей, ответы на замечания редакции и рецензентов, а также сопроводительную переписку по конкретной статье. Направление материалов осуществляется в электронной форме в соответствии с внутренним регламентом редакции.

Каждое письмо должно содержать информативную тему и заполненный сопроводительный текст. Переписка ведется в официально-деловом стиле. В теме письма указываются тип обращения, фамилия автора, краткое название статьи и, при необходимости, номер итерации исправления.

В тексте письма обязательно приводятся полное название статьи, сведения обо всех авторах, дисциплина (если применимо), номер итерации исправления и контактные данные ответственного автора. Для исправленных версий дополнительно рекомендуется указывать, на какое письмо редакции или рецензии дается ответ, и кратко обозначать внесенные исправления.

Файл рукописи направляется в формате .docx, на кириллице, по установленной форме наименования. Исправленные версии направляются исключительно ответом на письмо редакции с обязательным указанием номера итерации в теме письма, тексте письма и имени файла.

Материалы, оформленные с нарушением установленных требований, могут быть возвращены без рассмотрения до устранения технических замечаний.

**Контактные данные редакционной коллегии**

Почтовый адрес для корреспонденции: Москва, Ленинский пр-кт, д. 65/1, отделение почтовой связи № 119296.

До востребования.

Получатель: Павловский Владимир Владимирович.

Тел. +7(950) 632-04-38

e-mail: organizers@au-team.ru

главный редактор – Уймин Антон Григорьевич;

ответственный секретарь - Уймина Ольга Ивановна;

технический редактор - Козлов Глеб Васильевич.

**ПРОФЕССИОНАЛИТЕТ, 2025, № 3 (007)**

Научное электронное издание.

Сведения о программном обеспечении, использованном для создания электронного издания:
LibreOffice — набор, вёрстка текста, генерация PDF
https://ru.libreoffice.org

Техническая обработка и подготовка материалов выполнены авторами.

Подписано к использованию: 30.09.2025.

Объём издания: 76,6 Мб.

Комплектация издания: pdf.

Запись на физический носитель: Уймин А. Г., тел. +7 (950) 632-04-38.

Издатель — редакция научного журнала «ПРОФЕССИОНАЛИТЕТ».

Место издания: Москва.

Электронная версия подготовлена редакцией журнала для распространения в локальной и сетевой форме.

Носитель электронного издания: URALOLIMP.WEBSITE