Аристархова А.А. студентка 3 курса факультета комплексной безопасности топливно-энергетического комплекса РГУ нефти и газа (НИУ) имени И.М. Губкина.
Чеков М.В. студент 3 курса факультета комплексной безопасности топливно-энергетического комплекса РГУ нефти и газа (НИУ) имени И.М. Губкина.

ЭФФЕКТИВНОСТЬ МЕХАНИЗМОВ ПЕСОЧНИЦЫ (SANDBOX) FLATPAK ДЛЯ ПРЕДОТВРАЩЕНИЯ УТЕЧЕК ДАННЫХ В ОС АЛЬТ

Аннотация: В данной работе проводится исследование механизмов изоляции приложений в системе Flatpak. Рассматривается поведение sandbox-приложения в базовой конфигурации, анализируются потенциальные угрозы безопасности при расширении разрешений, а также оценивается эффективность ограничительных мер после ужесточения политики доступа. В ходе эксперимента продемонстрированы возможности доступа приложения к пользовательским данным и взаимодействия с графической подсистемой, а также показано, что при корректной настройке разрешений sandbox обеспечивает эффективную защиту системы.

Ключевые слова: Flatpak, Linux, sandbox, информационная безопасность, изоляция приложений, X11, файловая система.

РАЗДЕЛ 1. ВВЕДЕНИЕ

Рост числа киберугроз и усложнение программного обеспечения повышают значимость механизмов изоляции приложений на уровне операционной системы. Одним из таких механизмов является «песочница» — sandbox, ограничивающая доступ программы к системным ресурсам и пользовательским данным. В Linux-системах для распространения и запуска настольных приложений широко применяется Flatpak, использующий контейнеризацию, пространства имен, фильтрацию системных вызовов, порталы и управление разрешениями.

Несмотря на наличие встроенных механизмов защиты, фактический уровень безопасности Flatpak-приложений зависит от конкретного профиля

разрешений. Приложения могут получать доступ к файловой системе хоста, графической подсистеме X11, Wayland или D-Bus-сервисам. При ошибочной настройке это увеличивает риск утечки данных.

Целью исследования является оценка эффективности механизмов изоляции Flatpak в ОС «Альт Рабочая станция 11.1» при предотвращении несанкционированного доступа к пользовательским данным. Для достижения цели были проанализированы базовые разрешения Flatpak-приложений, сформирована модель угроз, проведена практическая проверка доступа к файловой системе, графической подсистеме и D-Bus, а также оценено изменение поверхности атаки после ограничения разрешений.

Гипотеза исследования состоит в том, что при наличии избыточных разрешений Flatpak-приложение может сохранять риск доступа к пользовательским данным, однако корректная настройка разрешений существенно повышает уровень изоляции.

Объект исследования – процессы обеспечения информационной безопасности в операционных системах на базе Linux.

Предмет исследования – механизмы песочницы Flatpak и их влияние на предотвращение утечек данных в операционной системе Альт.

Методы исследования: анализ документации Flatpak, практический эксперимент, моделирование потенциальных угроз безопасности.

РАЗДЕЛ 2. ЛИТЕРАТУРНЫЙ ОБЗОР

Flatpak – фреймворк для сборки, распространения и запуска приложений в среде Linux. Он предоставляет разработчикам и пользователям универсальную, не зависящую от конкретного дистрибутива платформу, одновременно решая проблемы совместимости библиотек и повышая безопасность графических приложений [1]. В отличие от обычных пакетных менеджеров, Flatpak сначала задумывался как инструмент, который не просто устанавливает программу, а запускает ее в изолированном контейнере по принципу схожему с Docker, но только для обычных настольных приложений [2].

Одной из основных целей Flatpak является повышение безопасности за счет изоляции приложений с помощью механизма «песочницы». Это означает, что по умолчанию приложения, запускаемые с помощью Flatpak, имеют ограниченный доступ к среде хоста. У приложений нет доступа к большинству файлов хоста, за исключением среды выполнения, самого приложения, директорий ~/.var/app/$FLATPAK_ID, и $XDG_RUNTIME_DIR/app/ $FLATPAK_ID. При этом для записи доступны только два последних. По умолчанию отсутствует сетевой доступ. Нет доступа к каким-либо узлам устройств, кроме /dev/null и аналогичных ему. Также ограничивается доступ к процессам вне «песочницы» и к хост-службам, таким как X11, системный D-Bus. Применяется фильтрация системных вызовов, то есть приложения не могут использовать нестандартные типы сетевых сокетов или отслеживать другие процессы с помощью ptrace. Однако фактический уровень защиты зависит от набора разрешений конкретного приложения. У каждого Flatpak-приложения своя конфигурация базовых разрешений.

На практике эффективность «песочницы» Flatpak часто снижается из-за некорректных настроек разрешений. Исследования показывают [3], что почти 42% проанализированных Flatpak-пакетов либо переопределяют предполагаемую изоляцию, либо имеют ошибки в конфигурации «песочницы». Это может приводить к избыточным привилегиям или к путям потенциального «побега». В частности, для обеспечения обратной совместимости и функциональности приложения часто запрашивают права на доступ к графической подсистеме X11. Но это известный фактор риска и это может позволить вредоносному программному обеспечению обойти ограничения [4].

Известные уязвимости Flatpak и связанных компонентов, представленные в Банке данных угроз безопасности информации ФСТЭК России [13], показывают, что риски изоляции могут быть связаны не только с архитектурой песочницы, но и с обработкой разрешений, порталов, символических ссылок и взаимодействия с хостовой системой. Поэтому при

анализе безопасности Flatpak важно учитывать как общую модель изоляции, так и фактический набор разрешений конкретного приложения.

Проблема изоляции пользовательских приложений в Linux рассматривается в ряде современных исследований. В работе [10] проведен анализ политик песочницы для Flatpak и Snap, где показано, что уровень защиты зависит не только от самой технологии контейнеризации, но и от конкретного набора разрешений, выданных приложению. Авторы исследования проанализировали политики песочницы для приложений Flatpak и Snap, что подтверждает актуальность оценки разрешений не на уровне общей архитектуры, а на уровне конкретных пакетов.

Вопросы безопасности контейнеризованных приложений также рассматриваются в рекомендациях NIST SP 800-190 [12]. В них подчеркивается необходимость контроля конфигураций, ограничения избыточных привилегий и анализа рисков среды выполнения. Для настольных Linux-приложений данная проблема очень важна, так как графические приложения часто требуют доступа к пользовательским файлам, оконной системе, устройствам и межпроцессному взаимодействию.

РАЗДЕЛ 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

Для эксперимента была подготовлена виртуальная среда на базе Oracle VirtualBox. В качестве операционной системы использовалась «Альт Рабочая станция» 11.1. В качестве тестовых приложений были выбраны org.gnome.gedit и org.gnome.Calculator. Основным тестовым было org.gnome.gedit так как оно относится к обычным пользовательским программам и работает с текстовыми файлами.

В работе рассматривается сценарий запуска потенциально недоверенного Flatpak-приложения от имени обычного пользователя. Нарушитель не обладает правами суперпользователя, но может использовать разрешения, уже заданные в профиле приложения или выданные пользователем вручную. Основными защищаемыми объектами являются

пользовательские файлы, конфигурационные данные, пользовательский ввод, содержимое окон и граница между sandbox и хостовой системой.

Таблица 1 – Модель угроз

Угроза	**Защищаемые данные**	**Вектор реализации**	**Проверяемое разрешение**	**Критерий успешности**
Чтение файлов пользователя	Домашний каталог	Обращение к файлам через shell внутри Flatpak	*filesystems=host*	Успешное чтение файлов пользователя
Изменение файлов пользователя	Целостность пользовательских данных	Запись файла из sandbox в /home/user	*filesystems=host*	Появление файлов в реальном домашнем каталоге
Взаимодействие с окнами	Пользовательский ввод и интерфейс	Использование xdotool в X11	*sockets=x11*	Ввод команды в активное окно
Взаимодействие с хостом	Граница sandbox/host	Использование flatpak-spawn --host	*org.freedesktop.Flatpak=talk*	Выполнение команды вне sandbox

В данной работе основными проверяемыми механизмами являются: доступ к файловой системе, D-Bus-разрешения, графические подсистемы x11 и Wayland. Все действия выполнялись от обычного пользователя «user».

После установки выполнен просмотр разрешений у установленных приложений командой *flatpak info --show-permissions …* (Рис. 1).

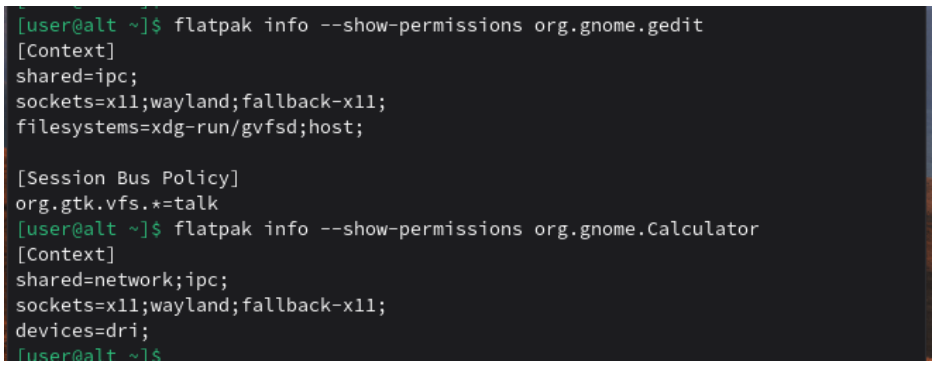

Рисунок 1 – Просмотр разрешений

На основе увиденного можно сделать вывод, что у каждого приложения свой индивидуальный базовый набор разрешений. Для того чтобы показать различие профилей разрешений у разных Flatpak-приложений, проведен аудит разрешений разных категорий (Таблица 2).

Таблица 2 – Аудит базовых разрешений

Приложение	**Категория**	**Файловый доступ**	**Граф. подсистема**	**Сеть**	**D-Bus/ особенности**	**Риск**
org.gnome.gedit	Текстовый редактор	host, gvfsd	X11, Wayland	нет	Gtk.vfs	Высок.

org.gnome.Calculator	Калькулятор	нет	X11, Wayland	да	нет	Сред.
org.mozilla.firefox	Браузер	xdg-download, persistent=.mozilla	X11, Wayland	да	FileManager, NetworkManager, a11y	Высок.
org.videolan.VLC	Медиаплеер	host, gvfsd	X11	да	KDE, secrets, MPRIS	Высок.
org.gimp.GIMP	Графический редактор	host, /tmp, gvfsd	X11, Wayland	да	Screenshot, FileManager, Gtk.vfs	Высок.

В таблице приведены объединенные группы разрешений. Длинные списки D-Bus-разрешений объединены по функциональному назначению. Такой формат позволяет сравнить профили приложений без перегрузки таблицы техническими строками.

Анализ показал, что приложения разных категорий имеют различные профили разрешений. Наиболее рискованными являются приложения с доступом к host, так как они могут работать с файлами хостовой системы в пределах прав пользователя. Дополнительного внимания требуют приложения с сетевым доступом и большим числом D-Bus-разрешений.

Полный эксперимент с проверкой доступа выполнялся на примере приложения org.gnome.gedit. Разберем базовые разрешения.

У приложения org.gnome.gedit были выявлены разрешения, влияющие на уровень изоляции: shared=ipc, доступ к X11 и Wayland, файловый доступ filesystems=xdg-run/gvfsd; host, а также D-Bus-разрешения org.gtk.vfs.*=talk. Наличие filesystems=host снижает уровень изоляции, так как приложение может работать с файлами хоста в пределах прав текущего пользователя. Для демонстрации взаимодействия с хостовой системой дополнительно было выдано разрешение org.freedesktop.Flatpak=talk (Рис. 2).

Рисунок 2 – Выдача дополнительного разрешения

Перейдем к тестированию изоляции приложения org.gnome.gedit. Для запуска используем команду *flatpak run --command=bash org.gnome.gedit*.

На первом этапе проверялся доступ к пользовательским и системным файлам. В домашнем каталоге был создан файл secret.txt, после чего его наличие и содержимое были успешно прочитаны из Flatpak-окружения. Также была проверена возможность чтения системного файла /etc/passwd. Этот факт не является самостоятельной уязвимостью, так как данный файл доступен обычному пользователю в Linux. Проверка показала, что при наличии filesystems=host приложение может работать с файлами, доступными текущему пользователю. Результат представлен на рисунке 3.

Рисунок 3 – Демонстрация доступа к файловой системе

Следующим этапом стала проверка взаимодействия Flatpak-приложения с графической подсистемой. Тип текущей сессии определялся командой echo $XDG_SESSION_TYPE. Для тестирования использовались инструменты xdotool, запускаемые через flatpak-spawn --host, что позволило оценить возможность обращения приложения к интерфейсу хостовой системы.

В среде X11 приложение смогло получить сведения о положении курсора, обнаружить активные окна и инициировать ввод команды в целевое окно. Это показывает, что при наличии доступа к X11 и разрешения org.freedesktop.Flatpak=talk sandbox-приложение может влиять на

пользовательский интерфейс хоста. Демонстрация выполнения команд представлена на рисунке 4.

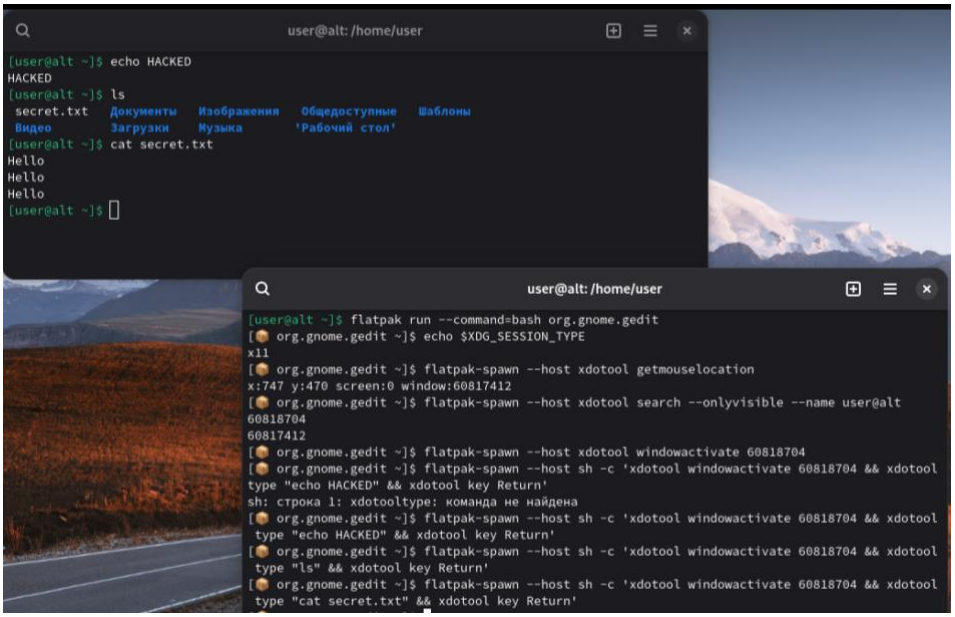

Рисунок 4 – Взаимодействие с графической подсистемой X11

При повторении проверки в Wayland аналогичный сценарий оказался ограничен. Приложение не смогло получить такой же уровень доступа к окнам других программ. Это связано с более строгой моделью изоляции окон и пользовательского ввода в Wayland. (Рис. 5).

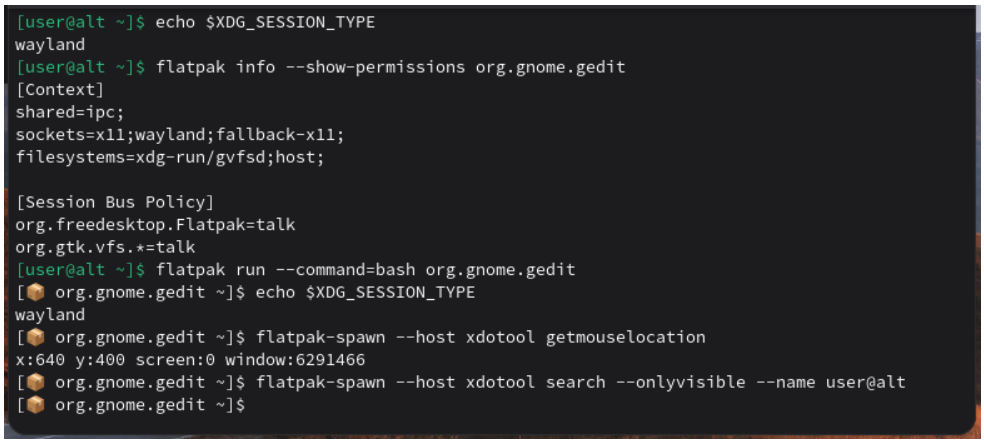

Рисунок 5 – Взаимодействие с графической подсистемой Wayland

Результаты проверки показывают, что X11 является более рискованной графической подсистемой для недоверенных Flatpak-приложений, тогда как Wayland обеспечивает более строгую изоляцию взаимодействия между окнами.

После демонстрации рисков были ограничены потенциально опасные разрешения: доступ к файловой системе хоста, X11, Wayland и D-Bus-сервису org.freedesktop.Flatpak. Изменение профиля выполнялось с помощью flatpak override --user. Выполнение изолирующих команд представлено на рисунке 6.

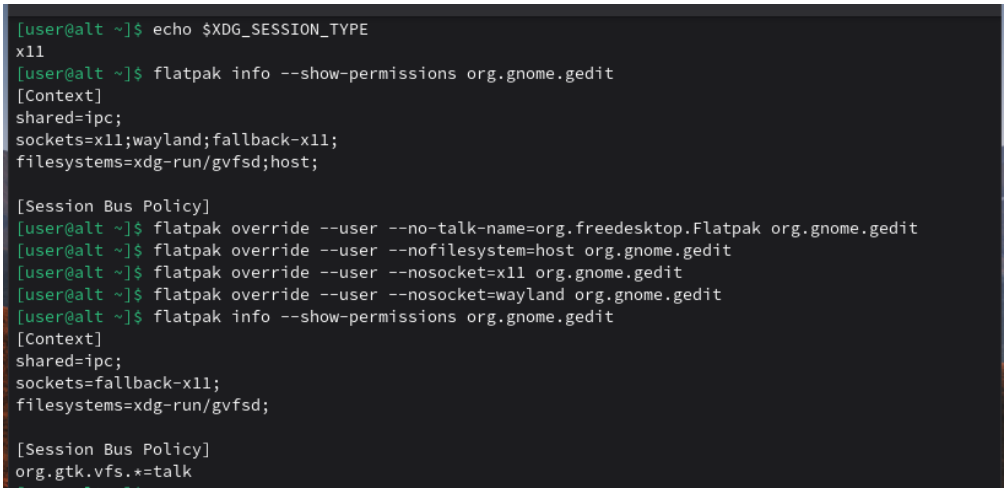

Рисунок 6 – Выполнение изолирующих команд

Приступим к проверке после изоляции. Для начала проверим доступ к файловой системе. Для этого создадим новый файл *proverka.txt* с содержимым *«Hello!!!»*. Файл успешно создался и даже с помощью команды *cat* мы смогли получить его содержимое. В результате проверки файл не появился в реальном домашнем каталоге. Мы видим лишь созданный при первой проверке файл *secret.txt* (Рис. 7).

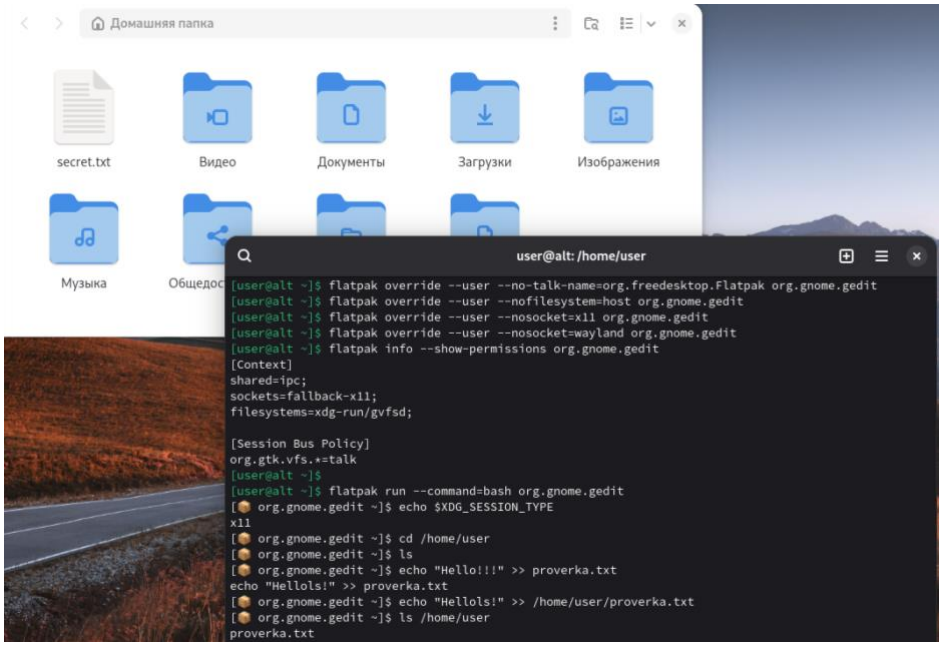

Рисунок 7 – Попытка создания файла в домашнем каталоге

Это доказывает эффективность изоляции. Новый созданный файл *proverka.txt* создался внутри изолированного окружения.

Аналогично проверена работа с системными каталогами. При выводе содержимого */etc* внутри песочницы (sandbox) отображается лишь копия Flatpak runtime необходимая для работы самого Flatpak (Рис. 8).

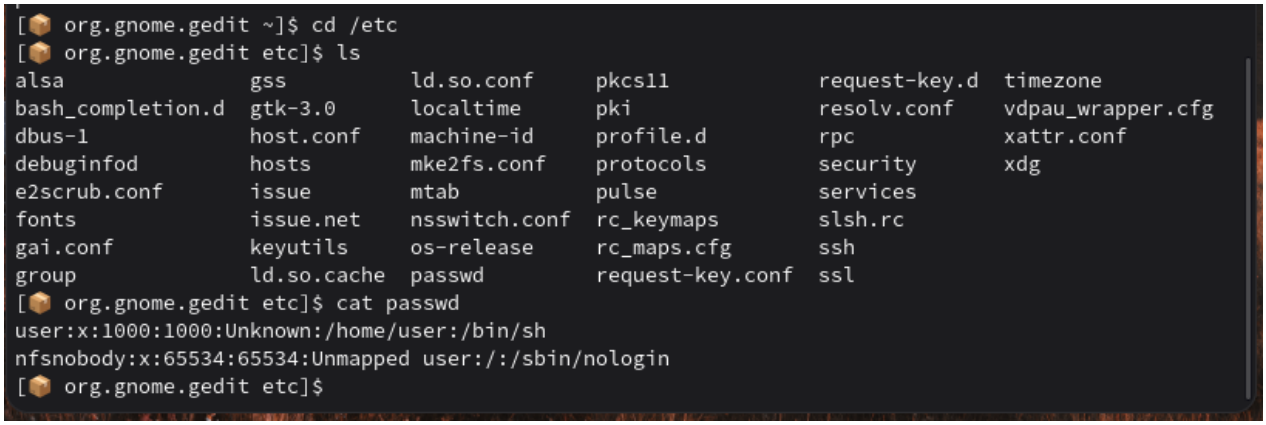

Рисунок 8 – Проверка доступа к */etc*

Теперь приступим к проверке возможности использовать команду *flatpak-spawn --host* на двух графических подсистемах (Рис. 9-10).

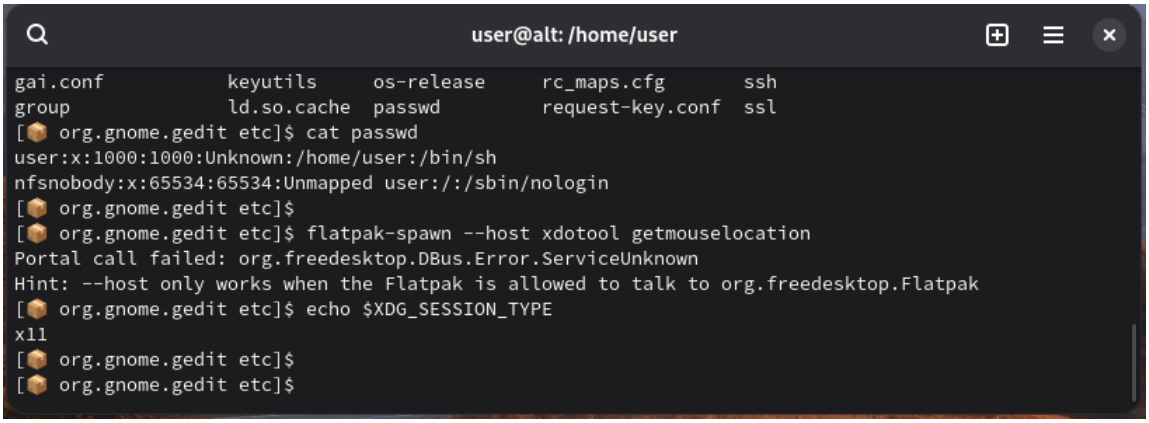

Рисунок 9 – Проверка на X11

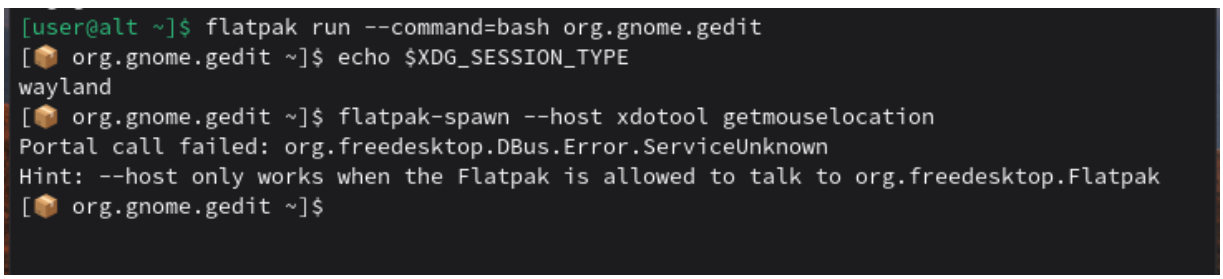

Рисунок 10 – Проверка на Wayland

На рисунках видно, что команда не работает. В подтверждение этому выведенное сообщение.

РАЗДЕЛ 4. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

Для оценки эффективности механизмов изоляции выполнено сравнение состояний приложения org.gnome.gedit до и после ограничения разрешений.

Таблица 3 – Сравнение состояний приложения до и после ограничений

Проверка	**Влияющее разрешение**	**До ограничения**	**После ограничения**	**Влияние на поверхность атаки**
Чтение пользовательского файла	*filesystems=host*	Успешно	Нет доступа	Риск снижен
Запись в домашний каталог	*filesystems=host*	Файл создается в /home/user	Файл создается внутри sandbox	Риск снижен
Чтение системных папок	*filesystems=host*	Доступно	Доступна копия внутри runtime	Не является самостоятельной уязвимостью
Получение координат мыши	*org.freedesktop.Flatpak=talk*	Успешно через *flatpak-spawn --host*	Ошибка D-Bus	Риск снижен
Вывод команд через X11	*sockets=x11*	Успешно	Заблокировано	Риск снижен
Проверка в Wayland	*sockets=wayland*	Ограничено	Заблокировано	Риск снижен

На основе результатов эксперимента можно предложить политику управления разрешениями Flatpak-приложений, основанную на принципе минимально необходимых привилегий. Простым утилитам не следует предоставлять доступ к filesystems=host, filesystems=home, сети и лишним D-Bus-разрешениям. Текстовым редакторам целесообразно разрешать доступ только к выбранным файлам или каталогам. Для браузеров допустимы сетевой доступ и доступ к каталогу загрузок, но нежелателен полный доступ к файловой системе хоста. Для недоверенных приложений следует запрещать X11, широкие D-Bus-разрешения, devices=all и полный доступ к домашнему каталогу.

Практический аудит Flatpak-приложения целесообразно выполнять по следующей схеме: определить назначение приложения, проверить текущие разрешения, выявить потенциально опасные права, удалить избыточные разрешения, проверить работоспособность и зафиксировать итоговую конфигурацию.

Для упрощения проверки правил приложений был разработан автоматический аудит приложений. Рабочий код и демонстрация представлены на рисунках 11-12.

```
#!/bin/bash

app="$1"

echo "Flatpak permission audit"
echo "========================"
echo "Application: $app"

perms=$(flatpak info --show-permissions "$app")

if echo "$perms" | grep -E "filesystems=.*(host|home)" > /dev/null; then
    echo "RISK: filesystem access to host/home"
fi

if echo "$perms" | grep -E "sockets=.*x11" > /dev/null; then
    echo "RISK: X11 socket access"
fi

if echo "$perms" | grep -E "org.freedesktop.Flatpak=talk" > /dev/null; then
    echo "RISK: access to org.freedesktop.Flatpak"
fi

if echo "$perms" | grep -E "devices=all" > /dev/null; then
    echo "RISK: access to all devices"
fi

if echo "$perms" | grep -E "shared=network" > /dev/null; then
    echo "NOTICE: network access"
```

Рисунок 11 – Код для автоматического аудита

```
[root@alt 52]# chmod +x flatpak_audit.sh

[root@alt 52]# ./flatpak_audit.sh org.gnome.gedit

Flatpak permission audit
========================
Application: org.gnome.gedit
RISK: filesystem access to host/home
RISK: X11 socket access
[root@alt 52]# ./flatpak_audit.sh org.gnome.Calculator
Flatpak permission audit
========================
Application: org.gnome.Calculator
RISK: X11 socket access
NOTICE: network access
[root@alt 52]# ./flatpak_audit.sh org.mozilla.firefox

Flatpak permission audit
========================
Application: org.mozilla.firefox
RISK: X11 socket access
RISK: access to all devices
NOTICE: network access
[root@alt 52]#
```

Рисунок 12 – Демонстрация

Данный аудит не заменяет полноценный аудит безопасности, однако позволяет быстро выявить приложения с потенциально опасными разрешениями.

РАЗДЕЛ 5. ВЫВОД

Проведенное исследование показало, что Flatpak является эффективным инструментом изоляции приложений, однако уровень защиты

зависит от фактически выданных разрешений. Наличие filesystems=host, доступа к X11 и разрешения org.freedesktop.Flatpak=talk повышает риск доступа к пользовательским данным и взаимодействия с хостовой системой. Практический эксперимент на примере org.gnome.gedit подтвердил, что при широких разрешениях приложение может работать с файлами пользователя в пределах его прав и взаимодействовать с пользовательским интерфейсом через X11. В Wayland аналогичный сценарий оказался ограничен из-за более строгой модели взаимодействия между окнами.

После удаления избыточных разрешений поверхность атаки была снижена: файлы создавались внутри sandbox, доступ к файловой системе хоста был ограничен, а выполнение команд на стороне хоста через flatpak-spawn --host было заблокировано. Таким образом, гипотеза исследования подтверждена: избыточные разрешения повышают риск утечки данных, но корректная настройка Flatpak позволяет существенно усилить изоляцию приложений.

Для повышения защищенности рекомендуется проверять разрешения Flatpak-приложений, избегать выдачи filesystems=host без необходимости, ограничивать доступ к X11 для недоверенных приложений, по возможности использовать Wayland и применять принцип минимально необходимых привилегий. Перспективным направлением дальнейшей работы является сравнение Flatpak с Snap, AppImage, Firejail и systemd-nspawn, а также развитие автоматического аудита разрешений.

СПИСОК ЛИТЕРАТУРЫ

1) Flatpak Documentation: Introduction to Flatpak // GNOME Developer Documentation. – URL: https://developer.gnome.org/documentation/introduction/flatpak.html (дата обращения: 26.04.2026).

2) Flatpak overview // Gentoo Wiki. – URL: https://wiki.gentoo.org/wiki/Flatpak (дата обращения: 26.04.2026).

3) Whittaker G. When Flatpak’s Sandbox Cracks: Real-Life Security Issues Beyond the Ideal // Linux Journal. – 2025. – URL: https://www.linuxjournal.com/content/when-flatpaks-sandbox-cracks-real-life-security-issues-beyond-ideal (дата обращения: 26.04.2026).

4) Security vulnerability in X11 socket access for Flatpak // GitHub: flatpak/flatpak Issues (#5744). – URL: https://github.com/flatpak/flatpak/issues/5744 (дата обращения: 26.04.2026).

5) Anderson E. Flatpak Permission Survey [Электронный ресурс]. – 2024. – URL: https://ejona.ersoft.org/archive/2024/03/03/flatpak-perm-survey/ (дата обращения: 02.05.2026).

6) Разрешения песочницы // Flatpak Documentation. – URL: https://docs.flatpak.org/ru/latest/sandbox-permissions.html (дата обращения: 02.05.2026).

7) Sandbox в Flatpak – объясните на пальцах // Linux.org.ru. – 2020. – URL: https://www.linux.org.ru/forum/talks/15757064 (дата обращения: 02.05.2026).

8) CODE RED 2026: Актуальные киберугрозы для российских организаций [Электронный ресурс] / Я. Авезова, Р. Резников, В. Беседина ; Positive Technologies. — 2025. — 8 октября. — Режим доступа: https://ptsecurity.com/research/analytics/russia-cyberthreat-landscape-2026/#id1 (дата обращения: 26.04.2026).

9) Уймин, А. Г. Демонстрационный экзамен базового уровня. Сетевое и системное администрирование: Практикум. Учебное пособие для вузов / А. Г. Уймин. – Санкт-Петербург : Издательство "Лань", 2024. – 116 с. – (Высшее образование). – ISBN 978-5-507-48647-2. – EDN BZJRIQ.

10) Dunlap T., Enck W., Reaves B. «A Study of Application Sandbox Policies in Linux» // Proceedings of the 27th ACM on Symposium on Access Control Models and Technologies (SACMAT '22). − New York : ACM, 2022. − P. 19–30. − DOI 10.1145/3532105.3535016.

11) Терских М. Г. ТЕХНОЛОГИИ ИЗОЛЯЦИИ ПРИЛОЖЕНИЙ И ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ДЛЯ УПРАВЛЕНИЯ КОНТЕЙНЕРАМИ // Теория и практика современной науки. 2017. №6 (24). URL: https://cyberleninka.ru/article/n/tehnologii-izolyatsii-prilozheniy-i-instrumentalnye-sredstva-dlya-upravleniya-konteynerami (дата обращения: 01.05.2026).

12) Souppaya M., Morello J., Scarfone K. Application Container Security Guide. − Gaithersburg : National Institute of Standards and Technology, 2017. − 63 p. − (NIST Special Publication ; 800-190). − DOI 10.6028/NIST.SP.800-190.

13) Банк данных угроз безопасности информации ФСТЭК России [Электронный ресурс] // Федеральная служба по техническому и экспортному контролю. – URL: https://bdu.fstec.ru/threat (дата обращения: 03.05.2026).