1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. **Институт компьютерных наук и кибербезопасности**

**ЛАБОРАТОРНАЯ РАБОТА № 12**

1. «Система контроля доступа»
2. по дисциплине «Основы информационной безопасности»
3. Выполнил
4. студент гр. 5131001/30003 Шевчук Н.Е.

1. Руководитель
2. асс. преподавателя Орел Е.М.

Санкт-Петербург

2024

[1. Цель работы 3](#_Toc165993248)

[2. Формулировка задания 3](#_Toc165993249)

[3. Теоретическая справка 3](#_Toc165993250)

[4. Ход работы 6](#_Toc165993251)

[4.1. Изменение прав доступа 6](#_Toc165993254)

[4.2. Тестовые утилиты 11](#_Toc165993255)

[5. Результаты работы 12](#_Toc165993256)

[6. Вывод 13](#_Toc165993257)

[7. Ответы на контрольные вопросы 13](#_Toc165993258)

[8. Листинг программы 14](#_Toc165993259)

# Цель работы

Освоить средства контроля и управления доступом пользователей к ресурсам операционной системы, приобретение навыков распределения прав на примере файловой системы NTFS в среде Windows.

# Формулировка задания

1. В файловой системе NTFS создать каталог *test\_folder*, на примере которого изучить управление правами доступов пользователей.
2. Разработать четыре тестовые утилиты, которые выполняют операции создания, чтения записи и изменения атрибутов безопасности для файла.
3. С помощью утилиты подтвердить на практике возможность доступа к файлу при разрешении и невозможность доступа при запрете.

# Теоретическая справка

Подсистема контроля и управления доступом в операционной системе Windows отличается высокой степенью гибкости, которая достигается за счет разнообразия защищаемых субъектов и объектов доступа, а также дифференциации видов доступа.

Контроль доступа выполняется централизованно с помощью компонента операционной системы - монитора безопасности (Security Reference Monitor), работающего в привилегированном режиме.

В системе Windows реализована объектная модель контроля доступа, согласно которой все субъекты и объекты доступа зарегистрированы в системе и обращение к ним регулируется с помощью множества атрибутов безопасности, наиболее важными из которых являются права доступа.

Для подсистемы безопасности операционной системы Windows характерно наличие большого количества различных предопределенных (встроенных) субъектов доступа - учетных записей пользователей и групп.

В системе имеются встроенные пользователи, например, *Администратор* *(Administrator), Гость (Guest), System* и группы *Пользователи (Users),*

*Администраторы (Administrators), Bce (Everyone).*

Встроенные пользователи и группы наделены полномочиями, заданными умолчанию, что облегчает администрирование системы. Администратор также может создавать новые группы и новых пользователей, устанавливая для них права и принадлежность группам, реализуя тем самым политику информационной безопасности.

Права доступа - множество операций, которые определены для субъектов доступа (например, пользователей и групп) по отношению к объектам доступа (например, файлам, каталогам, принтерам, процессам). Примерами прав доступа являются чтение файла, удаление каталога, печать документа, изменение прав доступа.

Права доступа, установленные группе, автоматически предоставляются всем ее участникам (членам группы), позволяя администратору рассматривать множество пользователей как учетную единицу и минимизировать свои действия по управлению правами.

При входе пользователя в систему для него создается так называемый маркер доступа (*access token*), включающий идентификатор пользователя и идентификаторы всех групп, в которые он включен. В маркере также хранится список пользовательских привилегий, учитываемых при выполнении системных действий.

Всем идентифицируемым объектам доступа, включая файлы, потоки, процессы, объекты ядра, объекты синхронизации процессов, события и пр., при создании присваивается дескриптор (описатель) безопасности.

Дескриптор безопасности содержит список контроля доступа (*Access*

*Control List, ACL*). Список контроля доступа состоит из набора записей контроля доступа (*Access Control Entry, ACE*), каждая из которых указывает идентификатор субъекта доступа (пользователя или группы), тип записи и права доступа. АСЕ может быть либо разрешающей, либо запрещающей, определяя тип прав как разрешения или запреты в маске прав доступа.

Владелец объекта - обычно пользователь, который его создал - обладает правом избирательного управления доступом к объекту, всегда может изменять *ACL* объекта, чтобы разрешить или запретить другим пользователям доступ к объекту.

При запросе процессом доступа к объекту управление передается монитору безопасности, который сравнивает идентификаторы пользователя и групп пользователей из маркера доступа с идентификаторами, хранящимися в записях *АСЕ* из *ACL* объекта. Процесс в течение сеанса работы может осуществлять доступ ко многим объектам, а количество активных процессов и проверяемых *АСЕ* в каждый момент времени довольно большое, поэтому монитор безопасности проверяет возможность доступа процесса к объекту только при его открытии (получений идентификатора), а не при каждом обращении.

Управление доступом пользователей к файловым объектам в системе Windows возможно только в рамках логических разделов файловой системы NTFS. Доступ к каталогам и файлам контролируется с помощью прав, устанавливаемых для пользователей и групп в *ACL* соответствующего файлового объекта.

В рамках одного *ACL* могут быть заданы несколько *АСЕ* для одного пользователя или группы (например, разрешающая пользователю чтение, разрешающая группе запись и чтение и запрещающая пользователю удаление), которые могут быть противоречивы. Монитор безопасности путем свертки прав вычисляет множество действующих разрешений:

1. Из разрешающих АСЕ данного пользователя (группы) составляется множество разрешений.
2. Из запрещающих АСЕ данного пользователя (группы) составляется множество запретов.
3. Результирующее множество разрешений определяется путем вычитания множества запретов из множества разрешений, поскольку запреты обладают большим приоритетом, чем разрешения.

Если для пользователя (группы) не установлено прав, то доступ запрещен. Если *ACL* не определен, то все виды доступа разрешены.

Для изменения *ACL* надо либо иметь право доступа "Изменение разрешений" (Change Permissions), либо быть владельцем объекта.

Для управления правами доступа к файловым объектам существует графический интерфейс операционной системы. Выбрав в контекстном меню файла или каталога пункт "Свойства", необходимо перейти на вкладку "Безопасность". На вкладке "Безопасность" в свойствах объекта представлен упрощенный интерфейс управления правами. При нажатии на кнопку "Дополнительно" появляется окно дополнительных параметров безопасности объекта, в котором можно изменить владельца объекта, тонко настроить *ACL* и наследование прав, рассчитать для конкретного субъекта доступа список действующих разрешений.

# Ход работы



## Изменение прав доступа

В файловой системе NTFS создан каталог test\_folder. Владельцем каталога является текущий пользователь; с каталога снят флаг наследования.

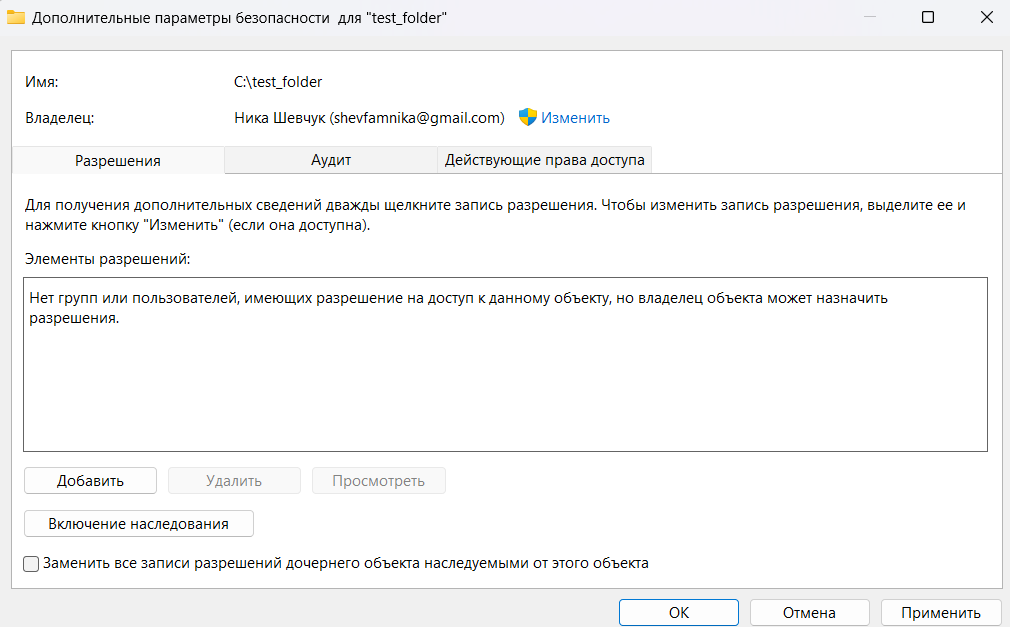
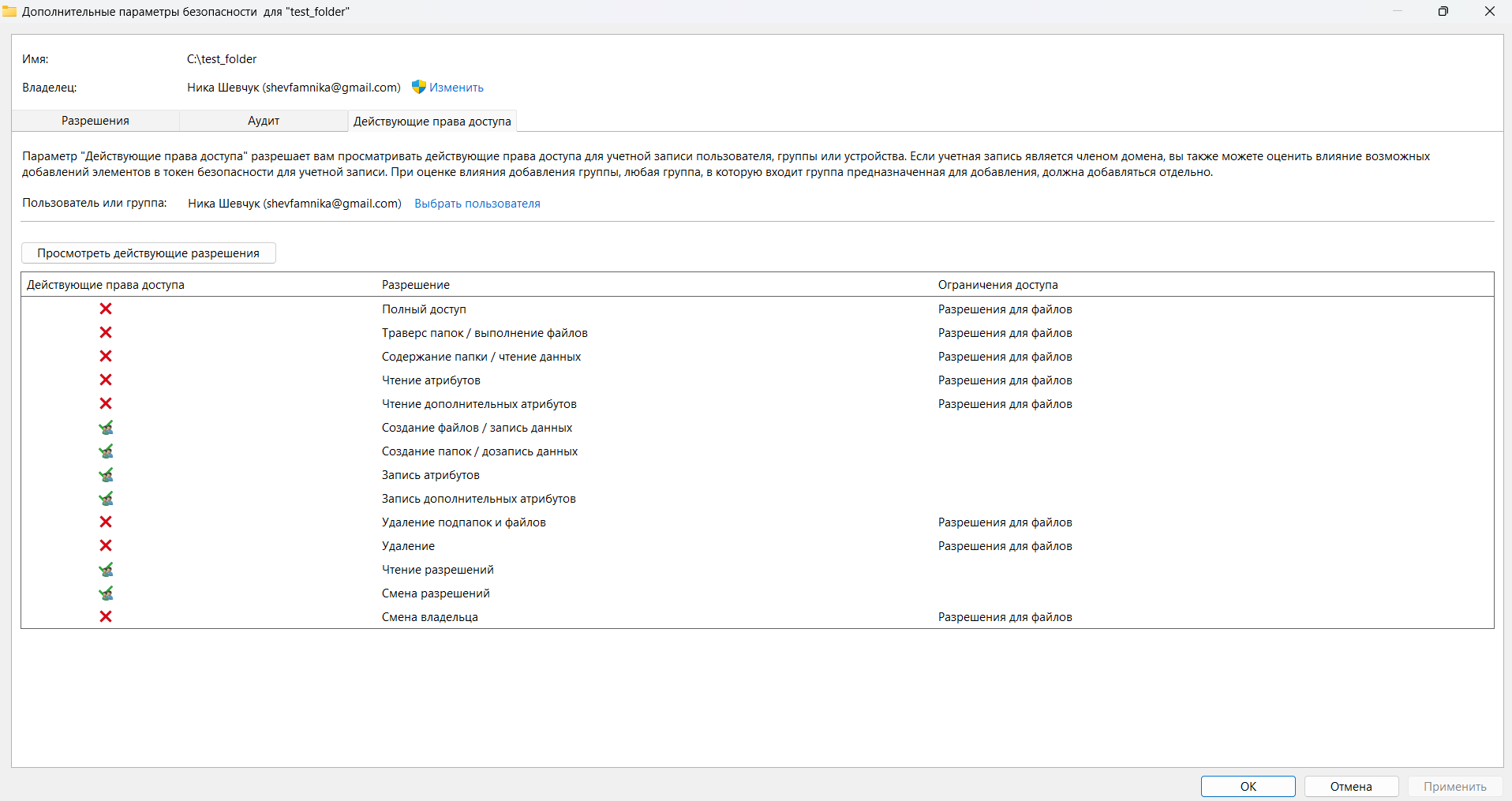
Рисунок 1 – Владелец каталога и отключение флага наследования. 

Рисунок 2 – разрешения, которыми наделен текущий пользователь.

Групповые права распространяются на всех членов группы. По умолчанию все пользователи находятся в группе «Пользователи», поэтому достаточно установить разрешения этой группе.

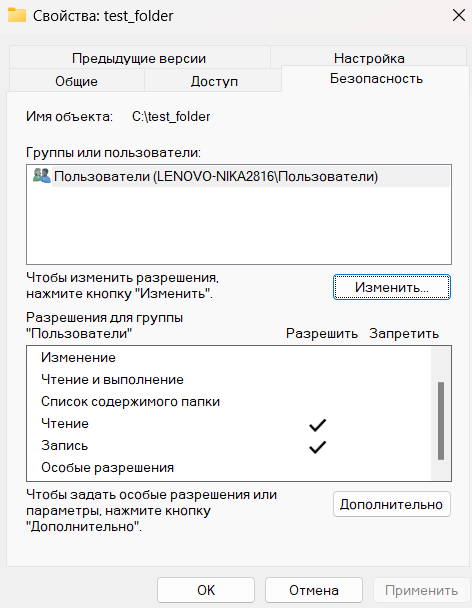
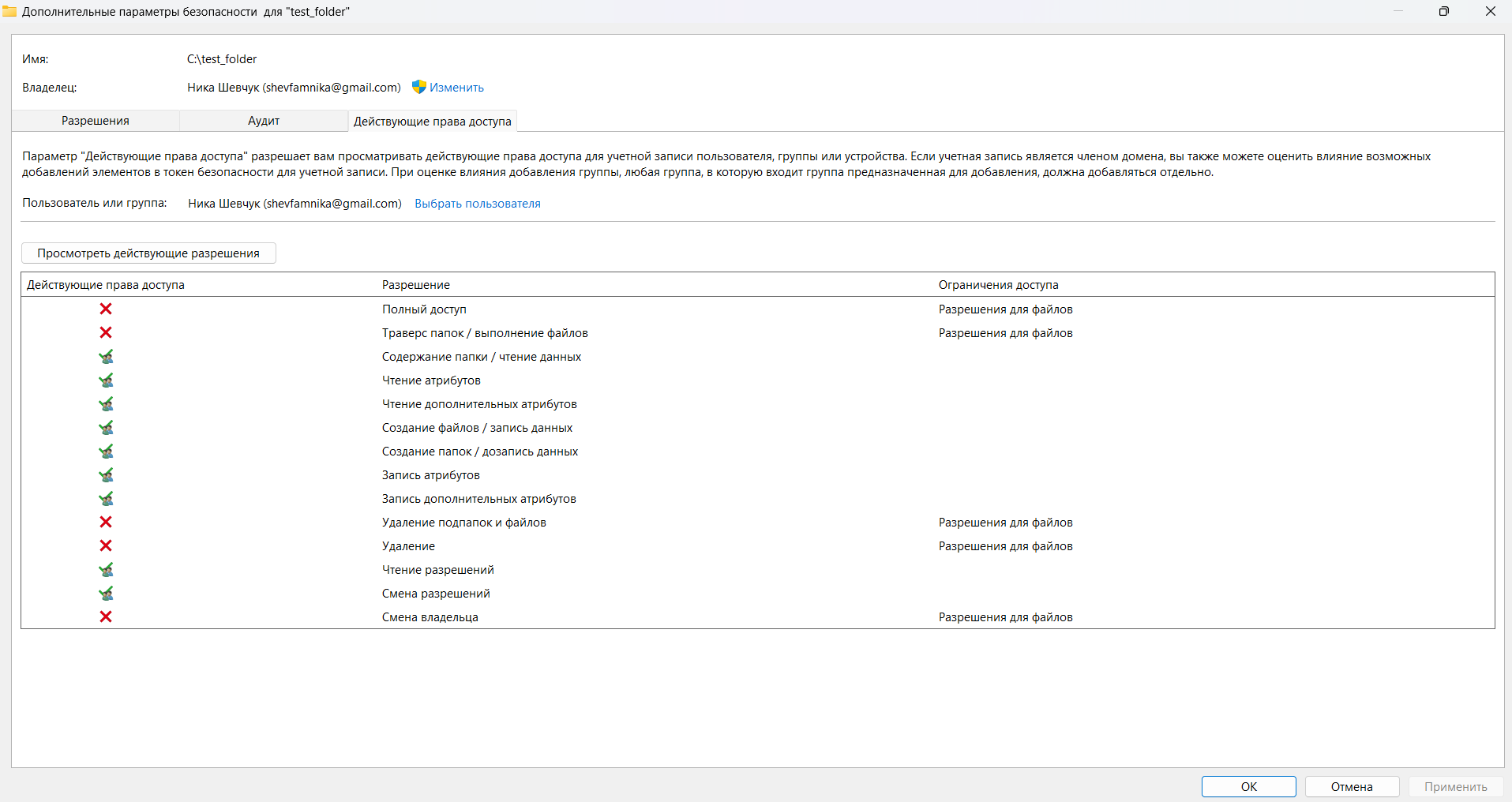


Рисунок 3 – разрешения группы «Пользователи»

Рисунок 4 – права доступа текущего пользователя

Запрещающие права имеют приоритет над разрешающими. Если для группы «Пользователи» запретить права на запись, а текущему пользователю разрешить полный доступ, то он получит запрет на запись.

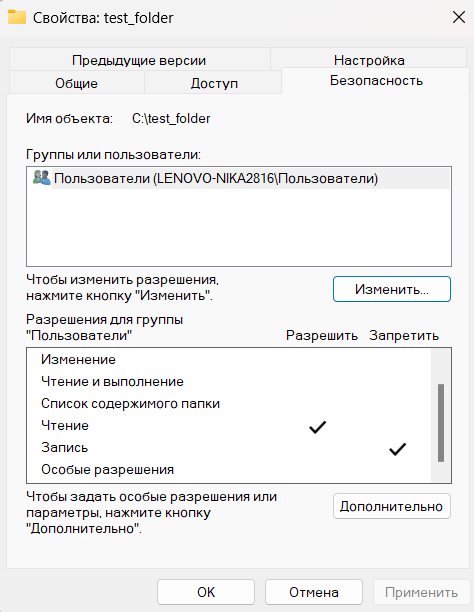


Рисунок 5 – установка запрета на запись группе «Пользователи»

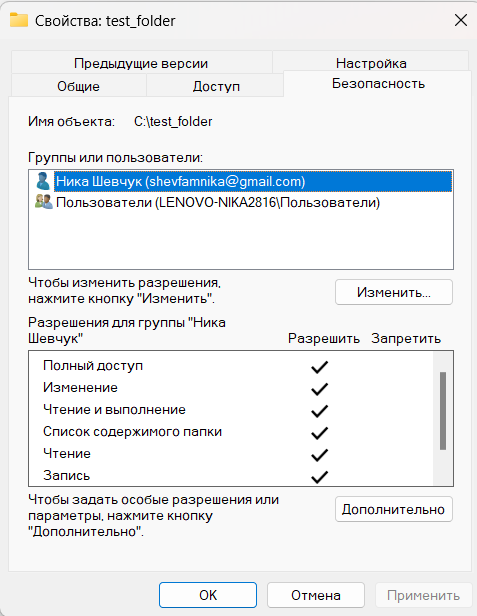
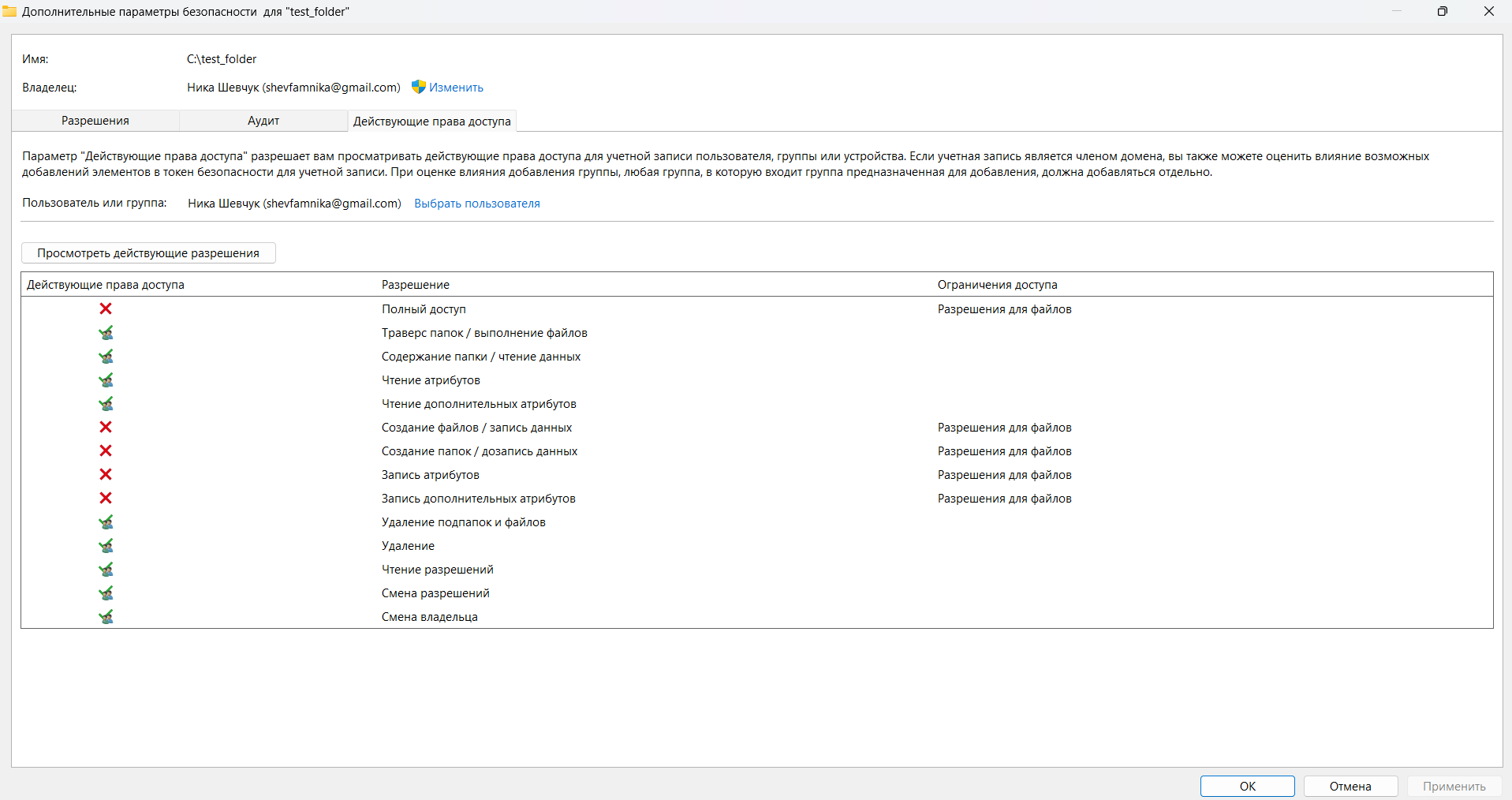


Рисунок 6 – установка полного доступа текущему пользователю

Рисунок 7 – права доступа текущего пользователя

Для двух и более *ACE* происходит суммирование разрешающих прав. Группе «Пользователи» представлено разрешение на запись, а текущему пользователю – на чтение. Тогда в действующих разрешениях текущий пользователь получает права и на чтение, и на запись.

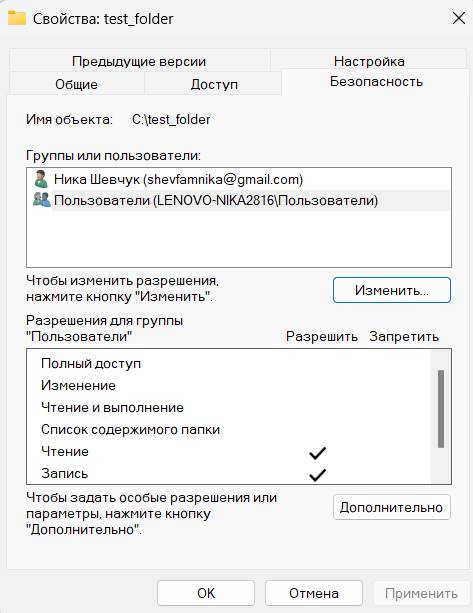
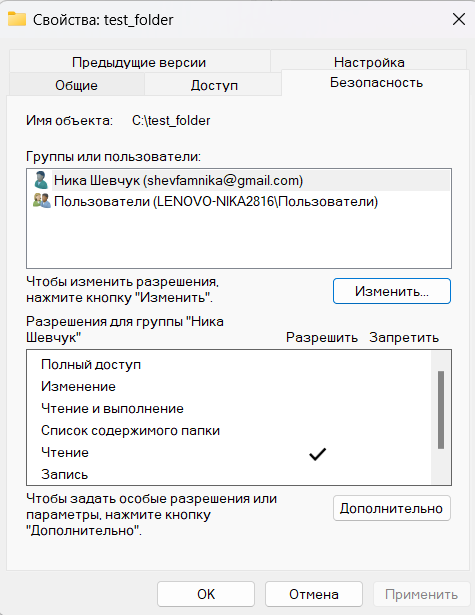


Рисунок 8 – разрешение на запись группе «Пользователи»



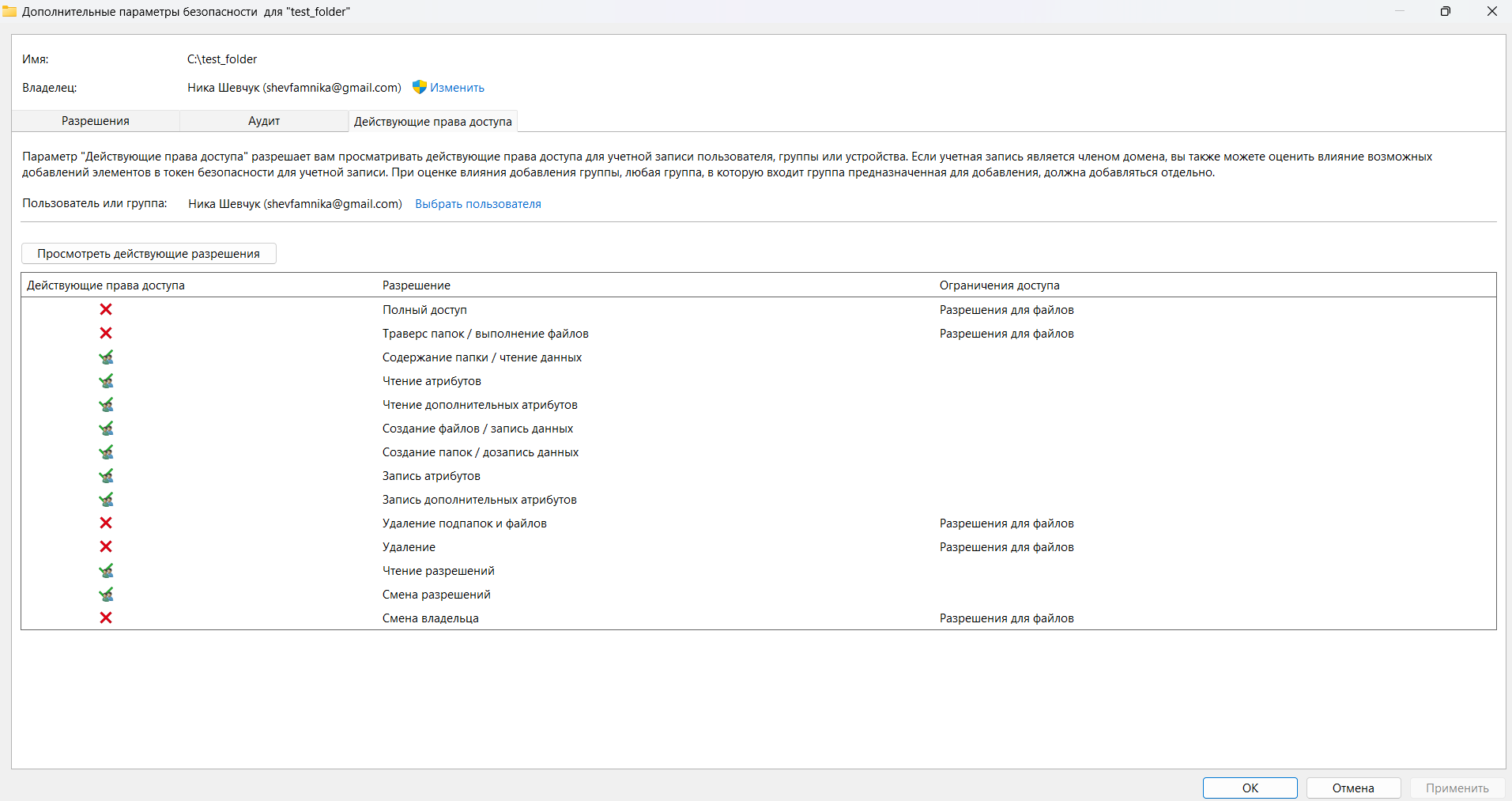
Рисунок 9 – разрешение на чтение текущему пользователю

Рисунок 10 – права доступа текущего пользователя

## Тестовые утилиты

В программе обращаемся к атрибутам каталога с помощью структуры данных WIN32\_FIND\_DATA, которая описывает файл, найденный функцией FindFirstFile. С помощью функции dwFileAttributes можно устанавливать и управлять следующими значениями атрибутов фалов:

* FILE\_ATTRIBUTE\_ARCHIVE - архивный файл или каталог. Приложения используют этот атрибут, чтобы отметить файлы для резервного копирования или перемещения.
* FILE\_ATTRIBUTE\_HIDDEN - скрытые. Они не включаются в обычный перечень файлов каталога.
* FILE\_ATTRIBUTE\_NORMAL - файл или каталог не имеют других установленных атрибутов. Этот атрибут допустим только в том случае, если используется как единственный.
* FILE\_ATTRIBUTE\_READONLY - файл или каталог только для чтения. Приложения могут читать этот файл, но не могут записать в него или удалить его. Что касается каталога, приложения не могут удалить его.

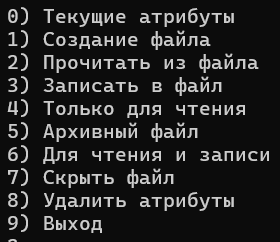


Рисунок 11 – интерфейс программы

Программа позволяет создать файл в каталоге, установить различные права доступа (разрешающие и запрещающие) и видоизменять их.

# Результаты работы

Разработаны четыре тестовые утилиты, которые выполняют операции создания, чтения, записи и изменения атрибутов безопасности для файла. С помощью утилит была подтверждена на практике возможность доступа к файлу при разрешении и невозможность доступа при запрете

# Вывод

При выполнении данной работы были получены навыки работы с доступом прав на примере файловой системы NTFS. Были разработаны утилиты для работы с файлами и изменения атрибутов безопасности файла. Выполнены многие проверки этих утилит, получены результаты действий при различных правах доступа.

# Ответы на контрольные вопросы

1. **Что такое множество действующих разрешений?**

Множество действующих разрешений – это множество действий с файлами, каталогами, которые разрешены пользователю или группе.

1. **Почему проверка прав доступа к файлу осуществляется только при открытии файла, а не при обращении к нему?**

Это выполнено в целях производительности. Каждому объекту присвоен свой *ACL*, в который входит большое количество записей *ACE*. Если бы проверка проходила при каждом обращении к файлу, то это привело бы снижению производительности.

1. **Можно ли запретить администратору систему доступы к какому-либо файлу? Можно ли он обойти это ограничение?**

Администратору системы можно запретить доступ к какому-нибудь файлу, однако он обладает исключительными правами и может отменить все запреты, восстановив доступ к файлу.

1. **Как изменить владельца объекта в среде Windows?**

Чтобы изменить владельца объекта в среде Windows нужно перейти в свойства файла, открыть окно «Дополнительные параметры безопасности», перейти во вкладку «Владелец» и изменить.

1. **Имеет ли владелец какие-либо права к файлу, если существует ACE, запрещающая полный доступ к этому файлу на имя владельца?**

Владелец файла имеет исключительный доступ к файлу, поэтому он может получить доступ к этому файлу, изменить *ACE* объекта.

# Листинг программы

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <Windows.h>

#include <clocale>

int remove\_file\_attribute(const char\* filename, DWORD attribute) {

DWORD currentAttributes = GetFileAttributesA(filename);

if (currentAttributes == INVALID\_FILE\_ATTRIBUTES) {

printf("Ошибка получения\n");

return 0;

}

// Проверяем, установлен ли атрибут, который мы хотим удалить

if (currentAttributes & attribute) {

currentAttributes &= ~attribute; // Удаляем атрибут

// Применяем измененные атрибуты

if (SetFileAttributesA(filename, currentAttributes)) {

printf("Атрибут файла удален\n");

return 1;

}

else {

printf("Ошибка удаления\n");

return 0;

}

}

else {

printf("Атрибут не установлен, удаление не требуется\n");

return 1;

}

}

void set\_file\_attributes(const char\* filename, DWORD attributes) {

DWORD currentAttributes = GetFileAttributesA(filename);

if (currentAttributes == INVALID\_FILE\_ATTRIBUTES) {

printf("Ошибка получения\n");

return;

}

// Добавляем новые атрибуты

int result = SetFileAttributesA(filename, currentAttributes | attributes);

if (result)

printf("Атрибуты файла изменены\n");

else

printf("Ошибка установки\n");

printf("\n");

}

void print\_file\_attributes(const char\* filename) {

WIN32\_FILE\_ATTRIBUTE\_DATA file\_info;

if (GetFileAttributesExA(filename, GetFileExInfoStandard, &file\_info)) {

printf("Размер файла: %lld байт\n", (long long)file\_info.nFileSizeLow);

if (file\_info.dwFileAttributes & FILE\_ATTRIBUTE\_HIDDEN)

printf("Скрытый файл\n");

if (file\_info.dwFileAttributes & FILE\_ATTRIBUTE\_ARCHIVE)

printf("Архивный файл\n");

if (file\_info.dwFileAttributes & FILE\_ATTRIBUTE\_READONLY)

printf("Файл только для чтения\n");

else

printf("Файл для записи и чтения\n");

}

else {

printf("Ошибка получения\n");

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

setlocale(LC\_ALL, "Russian");

const char\* filename = "C:/test\_folder/text.txt";

wchar\_t File[] = TEXT("C:/test\_folder/text.txt");

int n;

while (1) {

printf("0) Текущие атрибуты\n1) Создание файла\n2) Прочитать из файла\n3) Записать в файл\n4) Только для чтения\n5) Архивный файл\n6) Для чтения и записи\n7) Скрыть файл\n8) Удалить атрибуты\n9) Выход\n");

scanf\_s(" %d", &n);

if (n == 0) {

const char\* final = "C:/test\_folder/text.txt";

print\_file\_attributes(final);

printf("\n");

}

if (n == 1) {

WIN32\_FIND\_DATAA findData; ///Структура WIN32\_FIND\_DATA описывает файл, найденный функцией FindFirstFile

HANDLE hFile = FindFirstFileA("C:/test\_folder/text.txt", &findData);/// Функция FindFirstFile ищет каталог файла или подкаталог, название которого соответствует указанному имени файла

if (findData.dwFileAttributes & FILE\_ATTRIBUTE\_ARCHIVE)

printf("Архивный файл\n");

else {

FILE\* file1 = fopen("C:/test\_folder/text.txt", "w");

printf("Файл создан\n");

fclose(file1);

}

}

if (n == 2) {

WIN32\_FIND\_DATAA findData;

HANDLE hFile = FindFirstFileA("C:/test\_folder/text.txt", &findData);

if (findData.dwFileAttributes & FILE\_ATTRIBUTE\_READONLY || findData.dwFileAttributes & FILE\_ATTRIBUTE\_NORMAL) {

FILE\* file3 = fopen("C:/test\_folder/text.txt", "r");

char a[100] = { 0 };

char c;

while ((c = fgetc(file3)) != EOF)

printf("%c", c);

printf("\n");

fclose(file3);

}

}

if (n == 3) {

WIN32\_FIND\_DATAA findData;

HANDLE hFile = FindFirstFileA("C:/test\_folder/text.txt", &findData);

if (findData.dwFileAttributes & FILE\_ATTRIBUTE\_READONLY)

printf("Файл только для чтения\n");

else {

FILE\* file2 = fopen("C:/test\_folder/text.txt", "w");

char m[100];

scanf("%99s", &m);

fputs(m, file2);

fclose(file2);

}

}

if (n == 4) set\_file\_attributes(filename, FILE\_ATTRIBUTE\_READONLY);

if (n == 5) set\_file\_attributes(filename, FILE\_ATTRIBUTE\_ARCHIVE);

if (n == 6) SetFileAttributes(File, FILE\_ATTRIBUTE\_NORMAL);

if (n == 7) set\_file\_attributes(filename, FILE\_ATTRIBUTE\_HIDDEN);

if (n == 8) {

while (true) {

int ch;

printf("Удаление атрибута:\n1) Только для чтения\n2) Архивный файл\n3) Скрытый файл\n");

scanf("%d", &ch);

if (ch == 1) {

if (!remove\_file\_attribute(filename, FILE\_ATTRIBUTE\_READONLY))

printf("Атрибут 'Только для чтения' не найден\n");

}

if (ch == 2) {

if (!remove\_file\_attribute(filename, FILE\_ATTRIBUTE\_ARCHIVE))

printf("Атрибут 'Архивный файл' не найден\n");

}

if (ch == 3) {

if (!remove\_file\_attribute(filename, FILE\_ATTRIBUTE\_HIDDEN))

printf("Атрибут 'Скрытый файл' не найден\n");

}

break;

}

}

if (n == 9)

break;

}

return 0;

}