МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Факультет информационных технологий и компьютерной безопасности

(факультет)

Кафедра Систем автоматизированного проектирования и информационных систем

КУРСОВОЙ ПРОЕКТ

по дисциплине: Операционные системы

Тема: «Вывод списка разрешений для подраздела реестра»

Расчетно-пояснительная записка

Разработал студент А.Д Мальков

Подпись, дата Инициалы, фамилия

Руководитель К.А. Маковий

Подпись, дата Инициалы, фамилия

Члены комиссии

Подпись, дата Инициалы, фамилия

Подпись, дата Инициалы, фамилия

Нормоконтролер

Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка

дата

2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ», ВГТУ)

Кафедра Систем автоматизированного проектирования и информационных систем

ЗАДАНИЕ

на курсовой проект

по дисциплине: Операционные системы

Тема: «Вывод списка разрешений для подраздела реестра »

Студент бИСТ-221 Мальков Александр Дмитриевич

Группа, фамилия, имя, отчество

Вариант 22

Технические условия процессор AMD Ryzen 5 2600 3.9 ГГц, операционная система Windows 10, ОЗУ 16 384 МБ

Содержание и объем работы (графические работы, расчеты и прочее):

26 стр, 10 рисунков, 1 приложение, 2 таблицы

Сроки выполнения этапов: анализ и постановка задачи (01.03-15.03); изучение теоретического обоснования работы (15.03-15.04); реализация программного решения (15.04-15.05); оформление пояснительной записки (15.05-05.06).

Срок защиты курсового проекта:

Руководитель К.А. Маковий

Подпись, дата Инициалы, фамилия

Задание принял студент А.Д. Мальков

Подпись, дата Инициалы, фамилия

Замечания руководителя

**СОДЕЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc136947865)

[1. Организация контроля доступа 6](#_Toc136947866)

[1.1 Дескрипторы защиты 6](#_Toc136947867)

[1.2 Списки контроля доступа для подраздела реестра 8](#_Toc136947868)

[2. Разработка скрипта вывода списка ACL для подраздела реестра 17](#_Toc136947869)

[2.1 Постановка задачи курсового проекта 17](#_Toc136947870)

[2.2 Обоснование выбора среды разработки 17](#_Toc136947871)

[2.3 Описание работы системы 18](#_Toc136947872)

[ЗАКЛЮЧЕНИЕ 23](#_Toc136947873)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 24](#_Toc136947874)

[ПРИЛОЖЕНИЕ А 26](#_Toc136947875)

# ВВЕДЕНИЕ

Данный курсовой проект направлен на изучение работы подраздела реестра. В работе используется интегрированная среда сценариев Windows PowerShell Integrated Scripting Environment— это поставляемая с пакетом PowerShell оболочка разработки, дополненная отладчиком и редактором сценариев. ISE была впервые представлена в Windows PowerShell версии 2 и переработана в PowerShell версии 3. ISE поддерживается во всех поддерживаемых версиях Windows PowerShell до версии 5.1 включительно. Однако ISE находится в режиме обслуживания, и вероятнее всего новые функции добавляться не будут. Также не поддерживается ISE c PowerShell версии 6 и более поздних [5].

Для управления доступом к файлам, папкам, ключам и веткам реестра в Windows на каждый объект файловой системы NTFS назначается специальный ACL (Access Control List, что означает список контроля доступа). В ACL объекта задаются доступные операции (разрешения), которые может совершать с этим объектом пользователь и/или группы.

Целью данного курсового проекта является получение доступа к реестру и вывод списка пользователей, имеющих права на это.

# 1. Организация контроля доступа

## **1.1 Дескрипторы защиты**

Объекты, к которым могут получать доступ процессы, имеют специальный атрибут – дескриптор защиты, содержащий информацию обо всех пользователях, которым разрешен или запрещен доступ к объекту.

Маркеры, которые идентифицируют удостоверения пользователя, являются лишь частью выражения, описывающего защиту объектов. Другая его часть — информация о защите, сопоставленная с объектом и указывающая, кому и какие действия разрешено выполнять над объектом.

Структура данных, хранящая эту информацию, называется дескриптором защиты (security descriptor). **SID (Security IDentifier)** – это уникальный идентификатор, который присваивается пользователям, группам, компьютерам или другим объектам безопасности при их создании в Windows или Active Directory. Windows использует SID, а не имя пользователя для контроля доступа к различным ресурсам: сетевым папкам, ключам реестра, объектам файловой системы, принтерам и т.д. Дескриптор защиты включает следующие атрибуты:

1. SID владельца - идентификатор защиты владельца;
2. SID группы - идентификатор защиты основной группы для данного объекта;
3. список управления избирательным доступом (discretionary access-control list, DACL) - указывает, кто может получать доступ к объекту и какие виды доступа;
4. системный список управления доступом (system access-control list, SACL) - указывает, какие операции и какие пользователи должны регистрироваться в журнале аудита безопасности [1].

Список управления доступом (access-control list, ACL) состоит из заголовка и может содержать элементы (access-control entries, АСЕ). Существует два типа ACL: DACL и SACL. B DACL каждый ACE содержит SID и маску доступа (а также набор флагов). Функция маски доступа заключается в описании прав доступа в компактной форме, а причем ACE могут быть четырех типов: «доступ разрешен» (access allowed), «доступ отклонен» (access denied), «разрешенный объект» (allowed-object) и «запрещенный объект» (denied-object). Если в дескрипторе защиты нет DACL (DACL = null), любой пользователь получает полный доступ к объекту. Если DACL пуст (т. е. в нем нет АСЕ), доступа к объекту не получает никто. АСЕ, используемые в DACL, также имеют набор флагов, контролирующих и определяющих характеристики АСЕ, связанные с наследованием. SACL состоит из ACE двух типов: системного аудита (system audit ACE) и объекта системного аудита (system audit-object АСЕ) [1]. Эти ACE определяют, какие операции, выполняемые над объектами конкретными пользователями или группами, подлежат аудиту. При SACL, равном null, аудит объекта не ведется.

Флаги дескриптора защиты представляют собой 32-битную маску, отдельные биты которой имеют следующий смысл:

0 — установлен, если идентификатор владельца объекта был установлен по умолчанию при создании объекта и с тех пор не менялся;

1 — установлен, если идентификатор первичной группы владельца объекта был установлен по умолчанию при создании объекта и с тех пор не менялся;

2 — установлен, если в дескрипторе защиты присутствует DACL. Если данный флаг сброшен, дескриптор защиты допускает полный доступ к данному объекту всех субъектов доступа;

3 — установлен, если DACL объекта назначен по умолчанию при создании объекта на основе маркера доступа субъекта-создателя;

4 — установлен, если в дескрипторе защиты присутствует SACL. Если данный флаг сброшен, дескриптор защиты не предусматривает генерации сообщений аудита при доступе субъектов к данному объекту;

5 — установлен, если SACL объекта назначен по умолчанию при создании объекта на основе маркера доступа субъекта-создателя;

8 — установлен, если любые изменения в DACL объекта должны быть автоматически унаследованы существующими дочерними объектами;

9 — установлен, если любые изменения в SACL объекта должны быть автоматически унаследованы существующими дочерними объектами;

10 — установлен, если DACL объекта подвергался процедуре автоматического наследования изменений в DACL объекта-родителя;

11 — установлен, если SACL объекта подвергался процедуре автоматического наследования изменений в SACL объекта-родителя;

12 — установлен, если DACL объекта не должен автоматически наследовать изменений в DACL объекта-родителя;

13 — установлен, если SACL объекта не должен автоматически наследовать изменений в SACL объекта-родителя;

15 — установлен, если дескриптор защиты представлен в упакованном формате (для хранения на диске или в другом внешнем хранилище, но не для чтения или модификации). Большинство операций над упакованным дескриптором защиты требуют его предварительной распаковки. Распаковка дескриптора защиты (как и упаковка) является чисто технической процедурой и никак не влияет на порядок управления доступом к объекту, к которому относится дескриптор защиты.

Если объект не имеет дескриптора защиты, при обращениях к нему субъектов никакие права доступа не проверяются. В этом случае любой субъект имеет абсолютные права на данный объект [2].

## **1.2 Списки контроля доступа для подраздела реестра**

Реестр - это унифицированная база данных, в которой Windows NT хранит всю инфор­мацию о конфигурации оборудовании и программного обеспечения локального компь­ютера. Реестр управляет ОС Windows NT, предоставляя информацию, используемую при запуске приложений и загрузке компонентов, например, драйверов устройств и сетевых протоколов.

Windows постоянно обращается к этой базе. Она необходима для нормальной и связанной работы всех аппаратных частей компьютера и программного обеспечения. Реестр формируется в процессе установки windows, и в дальнейшем при каждом изменение (даже малом), это изменение вносится в реестр. Он заменяет собой большинство текстовых ini-файлов, которые использовались в Windows 3.x, а также файлы конфигурации MS-DOS (например, Autoexec.bat и Config.sys). В наше время реестр – это как бы «сердце» Windows [3]. Куст реестра - это группа разделов, подразделов и параметров реестра с набором вспомогательных файлов, содержащих резервные копии этих данных. Расширения имен файлов в этих папках указывают на тип содержащихся в них данных. Максимальная длина имени раздела составляет 255 символов [4].

Далее, рассмотрим, как и кто может получать доступ к параметру реестра, может её изменять, получить полные права. Некоторые разделы системного реестра Windows не доступны для редактирования, даже если ваша учётная запись относится к группе «Администраторы». Прав доступа может не быть по нескольким причинам:

1. у текущего пользователя, авторизованного в системе, нет прав администратора;
2. группа «Администраторы» является владельцем раздела, но не имеет полных прав на него. В этом случае достаточно просто выдать группе «Администраторы» полные права;
3. владельцем раздела является системная служба TrustedInstaller;
4. владельцем раздела является системная учетная запись «Система»;

В таблице 1 перечислены стандартные разделы реестра.

Таблица 1 – Перечень стандартных разделов реестра

|  |  |
| --- | --- |
| Папка/стандартный раздел | Описание |
| HKEY\_CURRENT\_USER | Данный раздел является корневым для данных настройки пользователя, вошедшего в систему в настоящий момент. Здесь хранятся папки пользователя, цвета экрана и настройки панели управления. Эти данные называются профилем пользователя. |
| HKEY\_USERS | Данный раздел содержит все профили пользователей компьютера. HKEY\_CURRENT\_USER является подразделом HKEY\_USERS. |
| HKEY\_LOCAL\_MACHINE | Раздел содержит данные настройки, относящиеся к данному компьютеру (для всех пользователей). |
| HKEY\_CLASSES\_ROOT | Данный раздел является подразделом HKEY\_LOCAL\_MACHINE\Software. Хранящиеся здесь сведения обеспечивают открытие необходимой программы при открытии файла с помощью проводника. |
| HKEY\_CURRENT\_CONFIG | Данный раздел содержит сведения о профиле оборудования, используемом локальным компьютером при запуске системы. |

Таблица иллюстрирует особенности стандартных разделов реестра, каждый из которых отвечает за определенные задачи и содержит необходимые сведения.

Параметры или ключи реестра имеют имена, представленные в обычном текстовом виде и значения, которые хранятся в виде стандартизированных записей определенного типа. Допустимые типы данных реестра:

* REG\_BINARY - двоичный параметр. Большинство сведений об аппаратных компонентах хранится в виде двоичных данных и выводится в редакторе реестра в шестнадцатеричном формате;
* REG\_DWORD - двойное слово. Данные представлены в виде значения, длина которого составляет 4 байта (32-разрядное целое). Этот тип данных используется для хранения параметров драйверов устройств и служб. Значение отображается в окне редактора реестра в двоичном, шестнадцатеричном или десятичном формате;
* REG\_QWORD - Данные, представленные в виде 64-разрядного целого. Начиная с Windows 2000, такие данные отображаются в окне редактора реестра в виде двоичного параметра;
* REG\_SZ - строковый параметр;
* REG\_EXPAND\_SZ - Расширяемая строка данных. Многострочный параметр. Многострочный текст. Этот тип, как правило, имеют списки и другие записи в формате, удобном для чтения. Записи разделяются пробелами, запятыми или другими символами.
* REG\_RESOURCE\_ LIST - Двоичный параметр. Последовательность вложенных массивов. Служит для хранения списка ресурсов, которые используются драйвером устройства или управляемым им физическим устройством. Обнаруженные данные система сохраняет в разделе \ResourceMap. В окне редактора реестра эти данные отображаются в виде двоичного параметра в шестнадцатеричном формате;
* REG\_RESOURCE\_ REQUIREMENTS\_LIST - двоичный параметр. Последовательность вложенных массивов. Служит для хранения списка драйверов аппаратных ресурсов, которые могут быть использованы определенным драйвером устройства или управляемым им физическим устройством. Часть этого списка система записывает в раздел \ResourceMap. Данные определяются системой. В окне редактора реестра они отображаются в виде двоичного параметра в шестнадцатеричном формате;
* REG\_FULL\_RESOURCE\_DESCRIPTOR - двоичный параметр. Последовательность вложенных массивов. Служит для хранения списка ресурсов, которые используются физическим устройством. Обнаруженные данные система сохраняет в разделе \HardwareDescription. В окне редактора реестра эти данные отображаются в виде двоичного параметра в шестнадцатеричном формате;
* REG\_NONE - Данные, не имеющие определенного типа. Такие данные записываются в реестр системой или приложением. В окне редактора реестра отображаются в виде двоичного параметра в шестнадцатеричном формате;
* REG\_LINK - Символическая ссылка в формате Юникод;
* REG\_DWORD\_BIG\_ENDIAN (самый младший байт хранится в памяти в последнем числе) [13].

При изменении какого-либо подраздела в реестре и недостатке прав доступа к нему, пользователь получит сообщение об ошибке, представленное на рисунке 1.

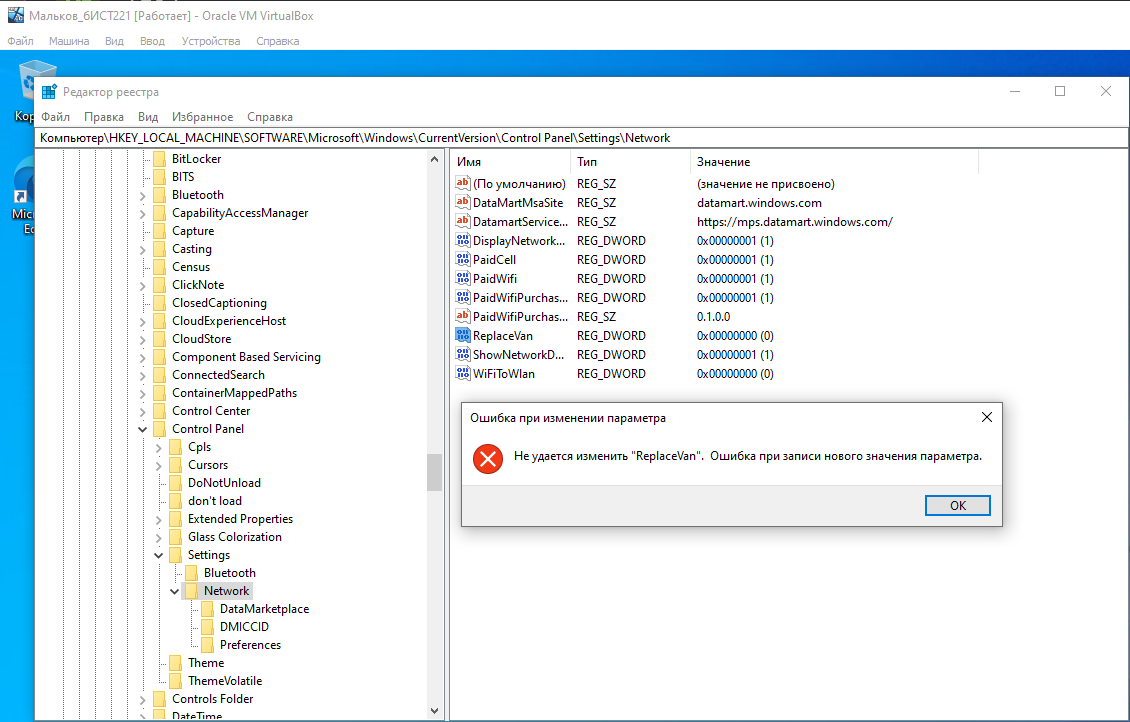


Рисунок 1 – Попытка изменения подраздела реестра

Чтобы избежать данную ошибку, необходимо группе «Администраторы» предоставить «Полный доступ». Допустимые права доступа для разделов реестра включают права delete, READ\_CONTROL, WRITE\_DAC и WRITE\_OWNER. В таблице 2 перечислены определенные права доступа для объектов раздела реестра [14].

Таблица 2 - Права доступа для объектов раздела реестра.

|  |  |
| --- | --- |
| Права доступа | Описание |
| KEY\_ALL\_ACCESS | Объединяет права доступа STANDARD\_RIGHTS\_REQUIRED, KEY\_QUERY\_VALUE, KEY\_SET\_VALUE, KEY\_CREATE\_SUB\_KEY, KEY\_ENUMERATE\_SUB\_KEYS, KEY\_NOTIFY и KEY\_CREATE\_LINK. |
| KEY\_CREATE\_LINK | Зарезервировано для системного использования. |
| KEY\_CREATE\_SUB\_KEY | Требуется для создания подраздела раздела реестра. |
| KEY\_ENUMERATE\_SUB\_KEYS | Требуется для перечисления подразделов раздела реестра. |
| KEY\_EXECUTE | Эквивалентно KEY\_READ. |
| KEY\_NOTIFY | Требуется для запроса уведомлений об изменениях для раздела реестра или подразделов раздела реестра. |
| KEY\_QUERY\_VALUE | Требуется для запроса значений раздела реестра. |
| KEY\_READ | Объединяет значения STANDARD\_RIGHTS\_READ, KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS и KEY\_NOTIFY. |
| KEY\_SET\_VALUE | Требуется для создания, удаления или задания значения реестра. |
| KEY\_WOW64\_32KEY | Указывает, что приложение на 64-разрядной Windows должно работать в 32-разрядном представлении реестра. Этот флаг игнорируется 32-разрядным Windows. |
| KEY\_WOW64\_64KEY | Указывает, что приложение на 64-разрядной Windows должно работать в 64-разрядном представлении реестра. Этот флаг игнорируется 32-разрядным Windows. |
| KEY\_WRITE | Объединяет STANDARD\_RIGHTS\_WRITE, KEY\_SET\_VALUE и KEY\_CREATE\_SUB\_KEY права доступа. |

У каждого пользователя есть определённые разрешения:

1. List Folder Contents (Просмотр содержимого директории) — позволяет зайти в папку и просмотреть ее содержимое;
2. Read (Чтение) — дает право на открытие файла\папки на чтение, без возможности изменения;
3. Read & execute (Чтение и выполнение) — позволяет открывать файлы на чтение, а также запускать исполняемые файлы;
4. Write (Запись) — разрешает создавать файлы\папки и редактировать файлы, без возможности удаления;
5. Modify (Изменение) — включает в себя все вышеперечисленные разрешения. Имея разрешение Modify можно создавать, редактировать и удалять любые объекты файловой системы;
6. Full Control (Полный доступ) — включает в себя разрешение Modify, кроме того позволяет изменять текущие разрешения объекта.

При попытке добавления пользователя в список разрешений необходимо открыть файл regedit.exe [15], нажать правой кнопкой мыши на любую из веток (HKLM, например), далее «Разрешения (см. Рисунок 2).

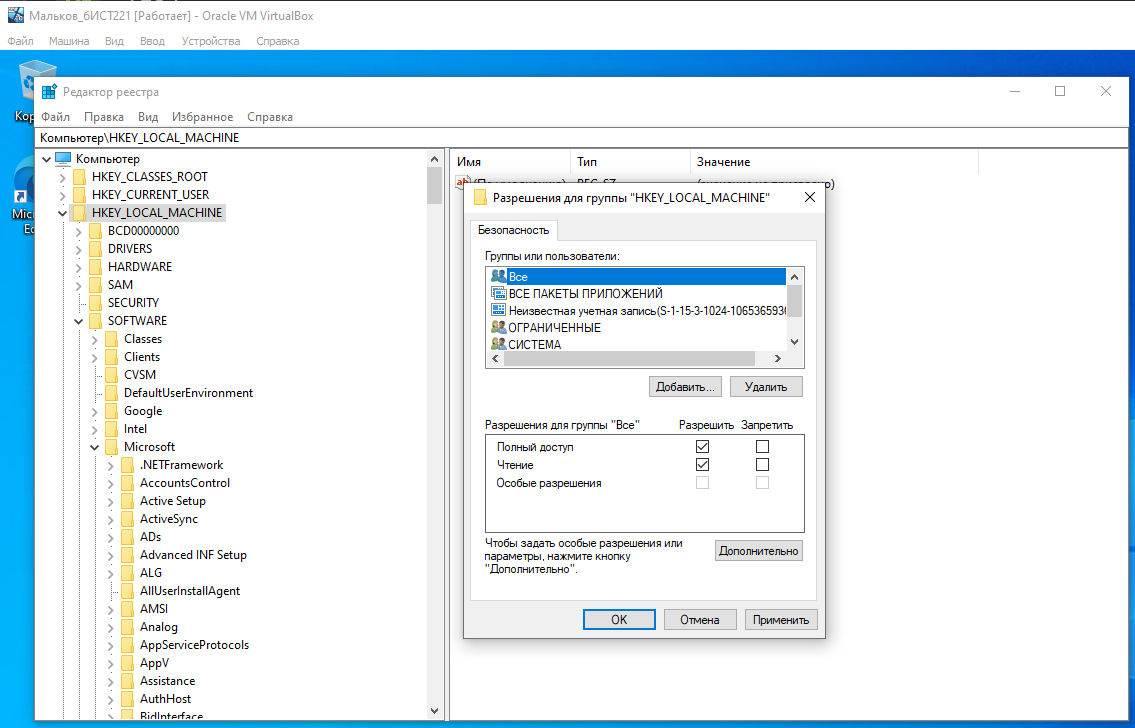


Рисунок 2 – Список разрешений для HKLM

Для добавления пользователя необходимо нажать «Добавить», далее «Дополнительно» и выбрать пользователя, которого надо добавить в список разрешений для данной ветки, но бывает так, что все пользователи имеют право доступа к данной ветке и добавлять некого, что показано на рисунке ниже.

|  |
| --- |
|  |
| Рисунок 3 – Добавление пользователей в список разрешений для ветки |

Пользователей также можно и удалять, в окне «Разрешения», далее «Удалить». Пользователям также можно не только давать права доступа, но и забирать. Явные запреты используются достаточно редко, поскольку для управления доступом вполне хватает обычных разрешений. Более того, запреты не рекомендуется использовать без крайней необходимости. Запрещающие правила настраиваются аналогично разрешающим — необходимо зайти в редактор реестра, открыть разрешения ветки, далее поставить галочки в столбце «Запретить». У запрета всегда приоритет над разрешением, система безопасности Windows предупредит пользователя об этом, что и представлено на рисунке 4 [5].

|  |
| --- |
|  |
| Рисунок 4 – Запрет на доступ к ветке реестра |

Существуют такие подразделы, к которым права доступа нет ни у одного пользователя, и если администратор, не входит в ACL, то ни один пользователь не сможет этого сделать. Также стоит заметить, что при работе с правами доступа незначительное изменение может серьёзно повредить операционную систему.

# 2. Разработка скрипта вывода списка ACL для подраздела реестра

## **2.1 Постановка задачи курсового проекта**

Задачей курсового проекта является вывод списка разрешений для подраздела реестра. Для выполнения задачи выбран скрипт, который будет:

* Находить ветку, заданную пользователем;
* Выводить список ACL.

Для выполнения этих задач была использована среда сценариев Powershell ISE. Она позволяет выполнять команды в области консоли, а также поддерживает области, позволяющие одновременно просматривать исходный код сценария, и другие подключаемые к этой среде средства [7].

Можно даже одновременно открыть несколько окон сценария. Это особенно удобно при отладке сценария, который использует функции, определенные в других сценариях или модулях. В нем вы можете печатать команды в интерактивном режиме, немедленно просматривать результаты и вставлять эти результаты в скрипт для долгосрочного использования. Или вы можете набирать команды непосредственно в скрипт, выделять необходимые и затем с помощью клавиши F8 требовать у оболочки выполнения выделенных задач.

## **2.2 Обоснование выбора среды разработки**

Ранее описаны средства, в которых можно разрабатывать данную программу. Рассмотрим каждую из них.

Если использовать какую-либо программу, то удобнее всего выбрать язык C# с платформой пользовательского интерфейса для создания приложений Windows Presentation Foundation (WPF) или Windows Forms(WF), так как интерфейс всегда приятен в использовании, а обычное консольное приложение как в С, например, очень трудно в написании и в обеспечении правильной работоспособности программы [12].

Если говорить о проводнике, он был использован при описании списков контроля доступа, находящемся в пункте 1.2.

Для разработки программы выбран скрипт. PowerShell ISE – это среда разработки скриптов PowerShell. В ней реализован богатый функционал по созданию и отладке скриптов. PowerShell ISE позволяет лишь ускорить, упростить процесс написания и отладки скрипта. ISE не обязателен к использованию. Запустить его можно из меню пуск или щелкнув правой кнопкой мыши по иконке PowerShell в панели задач [8].

## **2.3 Описание работы системы**

При открытии программы - скрипта в Powershell ISE, выводится окно, представленное на рисунке 5.

Разработанный скрипт - это пример использования PowerShell для создания графического интерфейса пользователя (GUI) с помощью Windows Forms.

Скрипт состоит из нескольких частей:

1. [сначала он загружает две библиотеки .NET: System.Windows.Forms и System.Drawing, которые содержат классы для работы с элементами GUI](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/);
2. затем он создает объект формы (Form), который является основным окном приложения. [Он задает ему текст, размер и положение на экране](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/);
3. далее он создает объект текстового поля (TextBox), который позволяет пользователю вводить и отображать текст. Он задает ему многострочный режим, размер, положение, полосы прокрутки и шрифт. [Он также добавляет его в коллекцию элементов управления формы (Form.Controls)](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/);
4. потом он создает объект кнопки (Button), который позволяет пользователю запускать действия по нажатию. Он задает ему размер, положение и текст. Он также добавляет к нему обработчик события Click, который выполняется при нажатии на кнопку. [В этом обработчике он запрашивает у пользователя путь к подразделу реестра (Read-Host), получает его права доступа (Get-Acl), форматирует их в виде таблицы (Format-Table) и выводит их в текстовое поле (Out-String) и в файл (Set-Content)](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/). [Он также добавляет кнопку в коллекцию элементов управления формы](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/);
5. затем он создает объект метки (Label), который позволяет отображать текстовую информацию. Он задает ему положение, размер и текст. [Он также добавляет его в коллекцию элементов управления формы](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/);
6. [после того, как все элементы GUI созданы и настроены, скрипт отображает форму на экране с помощью метода ShowDialog.](https://learn.microsoft.com/en-us/training/modules/script-with-powershell/)

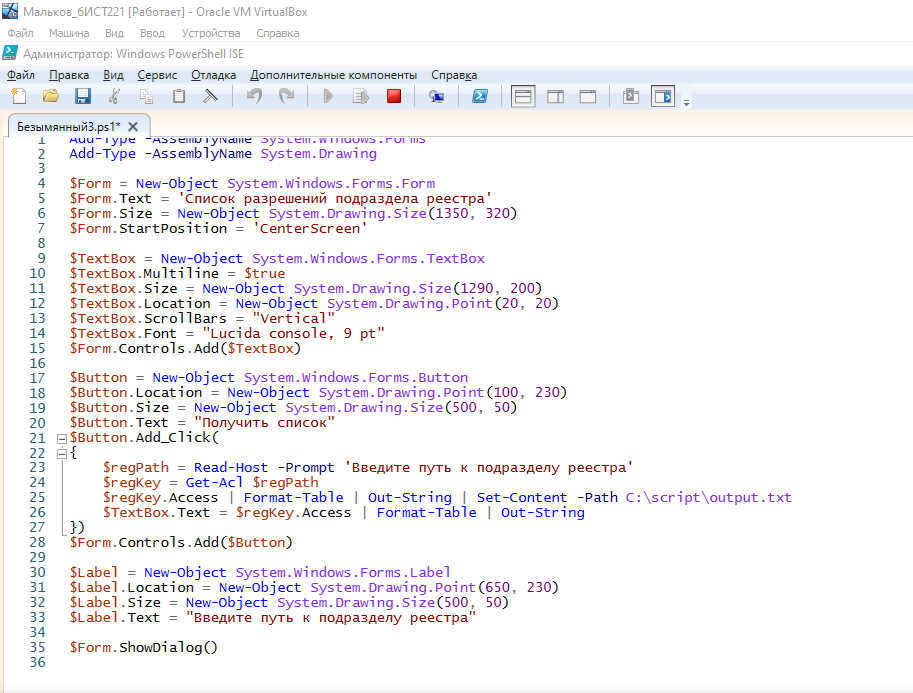


Рисунок 5 – Рабочее окно скрипта

Обычно в процессе работы PowerShell генерируются некоторые выходные данные. Они записываются в текстовый файл. Если же в файле до этого была какая-то информация, скрипт автоматически перезаписывает необходимые данные [9]. Для работы программы необходимо нажать кнопку «Получить список» (см. Рисунок 6).

|  |
| --- |
|  |
| Рисунок 6 – оконное приложение для получения и вывода списка разрешений |

Далее нам нужно будет ввести путь к подразделу реестра, откуда будут браться списки разрешений (см. Рисунок 7).

|  |
| --- |
|  |
| Рисунок 7 – ввод пути к подразделу реестра |

После ввода мы нажимаем клавишу Enter и список выводится в оконном приложении (см. Рисунок 8).

|  |
| --- |
|  |
| Рисунок 8 – вывод списка в оконном приложении |

Скрипт запускается и передаёт результат выходному файлу. Результат работы представлен на рисунке 9.

|  |
| --- |
|  |
| Рисунок 9 – Список разрешений для HKLM:\Software\Microsoft |

|  |
| --- |
|  |
| Рисунок 10 – script и выходной .txt файл |

На данном рисунке изображен скрипт и выходной текстовый файл в диске C:\script.

# ЗАКЛЮЧЕНИЕ

При выполнении данного курсового проекта было осуществлено несколько задач, а именно создание программы, выводящей список разрешений для подраздела реестра, исследована работа подраздела реестра.

Изучены:

1. среда сценариев для разработки скриптов Powershell ISE;
2. безопасность операционной системы Windows 10.

Благодаря данной работе существенно улучшено написание скриптов и знания об операционной системе Windows 10.

# 

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дескрипторы защиты и управление доступом //Сайт ereading - URL: [https://www.ereading.club/chapter.php/89561/12/Russinovich%2C\_Solomon\_-\_3.Vnutrennee\_ustroiistvo\_Windows\_%28gl.\_8-11%29.html](https://www.ereading.club/chapter.php/89561/12/Russinovich,_Solomon_-_3.Vnutrennee_ustroiistvo_Windows_(gl._8-11).html) – (Дата обращения: 19.04.2023)
2. Дескриптор защиты объекта – Защита в операционных системах//Сайт ozlib - URL: <https://ozlib.com/809549/informatika/deskriptor_zaschity_obekta> – (Дата обращения: 20.05.2023)
3. Руссинович М., Соломон Д., Алекс И. Внутреннее устройство Microsoft Windows. 6-е изд. – Издательский дом "Питер", 2013.
4. Реестр Windows – описание, настройка и полезные ветки реестра – Статья: Что такое реестр Windows //Сайт info-comp - URL: <https://info-comp.ru/vseowindowsst/44-sisreestr.html> – (Дата обращения: 20.04.2023)
5. Как изменить права доступа к разделу реестра //Сайт safezone - URL: <https://safezone.cc/threads/kak-izmenit-prava-dostupa-k-razdelu-reestra.18323/> – (Дата обращения: 20.05.2023)
6. Попов А. В. Введение в Windows PowerShell. – БХВ-Петербург, 2012.
7. Станек В. Windows PowerShell 2.0. Карманный консультант администратора. - Microsoft Press, 2009.
8. Введение в PowerShell //Сайт get-powershell – URL: <http://get-powershell.ru/2015/vvedenie-v-powershell/> – (Дата обращения: 20.05.2023)
9. Франк Кох (БЕРН) Введение в технологии языка сценариев для пользователей без базовых знаний.: Швейцария, 2007. – с. 15.:
10. Работа с разделами реестра //Обучающий сайт Microsoft Learn – URL: https://learn.microsoft.com/ru-ru/powershell/scripting/samples/working-with-registry-keys?view=powershell-7.3&viewFallbackFrom=powershell-7 – (Дата обращения: 20.04.2023)
11. Обзор PowerShell //Сайт docs.microsoft – URL: [https://docs.microsoft.com/rupowershell/scripting/gallery/overview?view=powershell-7](https://docs.microsoft.com/ru-ru/powershell/scripting/gallery/overview?view=powershell-7) – (Дата обращения: 20.05.2023)
12. Интегрированная среда сценариев Windows PowerShell //Сайт docs.microsoft – URL: <https://docs.microsoft.com/ru-ru/powershell/scripting/windows-powershell/ise/introducing-the-windows-powershell-ise?view=powershell-7> – (Дата обращения: 20.05.2023)
13. Что такое PowerShell //Сайт docs.microsoft – URL: <https://docs.microsoft.com/ru/powershell/scripting/overview?view=powershell-7> – (Дата обращения: 20.05.2023)
14. Политика выполнения скриптов PowerShell //Сайт windowsnotes – URL: <https://windowsnotes.ru/powershell-2/politika-vypolneniya-skriptov-powershell/> – (Дата обращения: 20.05.2023)
15. Глава 1-Начало работы с PowerShell //Сайт docs.microsoft – URL: <https://docs.microsoft.com/ru-ru/powershell/scripting/learn/ps101/01-getting-started?view=powershell-7> – (Дата обращения: 20.05.2023)

# ПРИЛОЖЕНИЕ А

ЛИСТИНГ ПРОГРАММЫ ПОЛУЧЕНИЯ СПИСКА РАЗРЕШЕНИЯ ПОДРАЗДЕЛА РЕЕСТРА:

Add-Type -AssemblyName System.Windows.Forms

Add-Type -AssemblyName System.Drawing

$Form = New-Object System.Windows.Forms.Form

$Form.Text = 'Список разрешений подраздела реестра'

$Form.Size = New-Object System.Drawing.Size(1350,500)

$Form.StartPosition = 'CenterScreen'

$Global:directF

$TextBox = New-Object System.Windows.Forms.TextBox

$TextBox.Multiline = $true

$TextBox.Size = New-Object System.Drawing.Size(1290, 300)

$TextBox.Location = New-Object System.Drawing.Point(20, 20)

$TextBox.ScrollBars = "Vertical"

$TextBox.Font = "Lucida console, 9 pt"

$Form.Controls.Add($TextBox)

$Button = New-Object System.Windows.Forms.Button

$Button.Location = New-Object System.Drawing.Point(100, 350)

$Button.Size = New-Object System.Drawing.Size(170, 60)

$Button.Text = "Получить список"

$Button.Add\_Click(

{

$regPath = Read-Host -Prompt 'Введите путь к подразделу реестра'

$regKey = Get-Acl $regPath

$regKey.Access | Format-Table | Out-String | Set-Content -Path D:\script\output.txt

$TextBox.Text = $regKey.Access | Format-Table | Out-String

})

$Form.Controls.Add($Button)

$Button2 = New-Object System.Windows.Forms.Button

$Button2.Location = New-Object System.Drawing.Point(300, 350)

$Button2.Size = New-Object System.Drawing.Size(200, 60)

$Button2.Text = "Выберите каталог реестра"

$Button2.Add\_Click{

$path = "C:\Windows\System32"

$object = New-Object -comObject Shell.Application

$regDirectory = $object.BrowseForFolder(0, "Выберите каталог", 0, $path)

if ($regDirectory -ne $null)

{

$Global:directF = $regDirectory

$tableRegF.RowCount[0].ColumnCount[0].Value = $regDirectory.Self.Path

$tableRegF.RowCount[0].ColumnCount[1].Value = $regFold = New-Object System.Security.Principal.NTAccount (Get-Acl -Path $regDirectory.Self.Path).$regFold | foreach { $\_.Value}

}

}

$Form.Controls.Add($Button2)

$Form.ShowDialog()