МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования «Вятский государственный университет»
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

КОНТРОЛЬ ДОСТУПА

Методические указания
Лабораторная работа №7 по дисциплине
«Операционные системы»

Киров 2016

Содержание

Введение

Целью данной лабораторной работы является изучение контроля доступа в операционных системах Windows NT (XP), а также реализации дискреционной модели доступа.

В ходе выполнения данной лабораторной работы необходимо последовательно пройти задания к лабораторной работе и получить готовую модель системы безопасности для заданного варианта.

1 Теоретические сведения

1.1 Модели контроля доступа

1.1.1 Дискреционная модель

Дискреционная модель – одна из самых распространенных в мире, в системах по умолчанию имеется в виду именно эта модель. Составим математическое описание данной модели.

Пусть O – множество объектов, S – множество субъектов, U = {U1, ..., Um} – множество пользователей.

Определим отображение: Владелец: O → U.

В соответствии с этим отображением каждый объект объявляется собственностью соответствующего пользователя. Пользователь, являющийся собственником объекта, имеет все права доступа к нему, а иногда и право передавать часть или все права другим пользователям. Кроме того, собственник объекта определяет права доступа других субъектов к этому объекту, то есть модель безопасности в отношении этого объекта. Указанные права доступа записываются в виде матрицы доступа, элементы которой определяют варианты доступа субъекта S, к объекту Oi (i = 1, 2, ... ; j = 1, 2, ...). Элементы в матрице доступа имеют следующие значения: Ч – чтение, З – запись, В – выполнение, 0 – нельзя использовать, Х – Владелец (хозяин).

Таблица 1 – Матрица доступа

		O1	O2	…	Ok	S1	…	Sn
M =	S1	Х Ч	З					
	S2	ЧВ	0					
	…							
	Sn							

Элементы матрицы доступа могут содержать указатели на специальные процедуры, которые должны выполняться при обращении субъекта к объекту. Решение о доступе в этом случае осуществляется на основании результатов выполнения процедур, например:

– решение о доступе в данный момент времени основывается на анализе предыдущих доступов к другим объектам;

– решение о доступе основывается на динамике состояния системы (права доступа субъекта зависят от текущих прав доступа других субъектов);

– решение о доступе основывается на значении определенных переменных, например, на значении таймера.

Существует несколько вариантов задания матрицы доступа:

1) листы возможностей: для каждого субъекта Si создается лист (файл) всех объектов, к которому имеет доступ данный объект;

2) листы контроля доступа: для каждого объекта создается список всех субъектов, имеющих право доступа к этому объекту.

Размерность матрицы доступа зависит от количества субъектов и объектов в системе и может быть достаточно большой. Для уменьшения размерности матрицы доступа могут применяться различные методы:

– установление групп субъектов, называемых кликами, каждая из которых представляет собой группу субъектов с одинаковыми правами;

– установление групп терминалов по классам полномочий (клики терминалов);

– группировка объектов по уровням категорий (например, по уровням секретности);

– хранение списка пар вида (o, f), где o – защищаемый объект, а f – разрешение на использование его субъектом.

Перечисленные методы и другие, им подобные, могут применяться как по отдельности, так и в совокупности.

В процессе функционирования системы множества субъектов и объектов могут динамически изменяться. Такие изменения могут происходить, например, в результате появления новых субъектов и объектов, уничтожения субъектов и объектов и изменения прав доступа субъектов к объектам. Соответственно, в процессе функционирования системы должна изменяться и матрица доступа. Динамика изменения множеств субъектов и объектов, а также матрицы доступа при выполнении некоторых операций представлена на рисунке 1.1, где S – множество субъектов; O – множество объектов, причем S принадлежит O; M[s, o] – матрица доступа. Элементами матрицы M являются права доступа g, принадлежащие G. Изменившиеся множества помечены штрихом.

Исходное состояние	Операция	Результирующее состояние
S, O, M (s'Ï O)	Создание субъекта s'	S' = S U {s'}, O' = O U {s'} M'[s,o] = M[s,o], sÎS, oÎO M'[s',o] = 0, oÎO' M'[s',s] = 0, sÎS'
S, O, M (o'ÏO)	Создание объекта o'	S' = S, O' = O U {o'} M'[s,o] = M[s,o], sÎS, oÎO M'[s,o'] = 0, sÎS'
S, O, M (s'ÏS)	Уничтожение субъекта s'	S' = S\{s'}, O' = O\{s'} M'[s,o] = M[s,o], sÎS', oÎO'
S,O,M (o'ÎO), (o'ÏS)	Уничтожение объекта o'	S' = S, O' = O \{o'} M'[s,o] = M[s,o], sÎS', oÎO'
S,O,M (sÎS), (oÎO)	Введение права g в M[s,o]	S' = S, O' = O M'[s,o] = M[s,o] U {g} M'[s',o'] = M[s',o'],(s',o') <> (s,o)
S,O,M (sÎS), (oÎO)	Удаление права g из M[s,o]	S' = S, O' = O M'[s,o] = M[s,o]\{g} M'[s',o'] = M[s',o'] если (s',o')<>(s,o)

Рисунок 1.1 – Динамика изменений множеств субъектов и объектов

Динамику изменения множеств S и O, а также матрицы M, представленной в таблице, удобно рассмотреть на примере создания субъекта s'.

При создании субъекта s' этот субъект вводится в состав элементов множеств S и O. В матрице доступа появляется новая строка, соответствующая новому субъекту: M'[s,o] = M[s,o].

M'[s',o] = 0; M'[s',s] = 0, так как субъект создан, но его права по отношению к существующим субъектам и объектам не определены. Матрицы доступа в той или иной степени используются во многих защищенных системах.

Дискреционная модель, как самая распространенная, больше всего подвергалась исследованиям. Существует множество разновидностей этой модели. Однако многих проблем защиты эта модель решить не может. Одна из самых существенных слабостей этого класса моделей - то, что они не выдерживают атак при помощи «Троянского коня». Это означает, в частности, что система защиты, реализующая дискреционную модель, плохо защищает от проникновения вирусов в систему и других средств скрытого разрушающего воздействия. Покажем на примере принцип атаки «Троянским конем» в случае дискреционной модели.

Пример 1.

Пусть U1 – некоторый пользователь, а U2 – пользователь-злоумышленник, O1 – объект, содержащий ценную информацию, O2 – программа с «Троянским конем» T, и M – матрица доступа, которая представлена на рисунке 1.2.

	O1	**O2**
U1	Владелец Чтение Запись	Запись
U2		Владелец Чтение Запись

Рисунок 1.2 – Матрица доступа

Проникновение программы происходит следующим образом. Злоумышленник U2 создает программу O2 и, являясь ее собственником, дает U1 запускать ее и писать в объект O2 информацию. После этого он инициирует каким-то образом, чтобы U1 запустил эту программу. U1 запускает O2 и тем самым запускает скрытую программу T, которая обладая правами U1 (т.к. была запущена пользователем U1), списывает в себя информацию, содержащуюся в O1. После этого хозяин U2 объекта O2, пользуясь всеми правами, имеет возможность считать из O2 ценную информацию объекта O1.

Следующая проблема дискреционной модели – это автоматическое определение прав. Так как объектов много, то задать заранее вручную перечень прав каждого субъекта на доступ к объекту невозможно. Поэтому матрица доступа различными способами агрегируется, например, оставляются в качестве субъектов только пользователи, а в соответствующую ячейку матрицы вставляются формулы функций, вычисление которых определяет права доступа субъекта, порожденного пользователем, к объекту O.

Разумеется, эти функции могут изменяться во времени. В частности, возможно изъятие прав после выполнения некоторого события. Возможны модификации, зависящие от других параметров.

Одна из важнейших проблем при использовании дискреционной модели – это проблема контроля распространения прав доступа. Чаще всего бывает, что владелец файла передает содержание файла другому пользователю и тот, тем самым, приобретает права собственника на информацию. Таким образом, права могут распространяться, и даже, если исходный владелец не хотел передавать доступ некоторому субъекту S к своей информации в O, то после нескольких шагов передача прав может состояться независимо от его воли. Возникает задача об условиях, при которых в такой системе некоторый субъект рано или поздно получит требуемый ему доступ. Эта задача исследовалась в модели «take-grant», когда форма передачи или взятия прав определяются в виде специального права доступа (вместо «Владелец»).

1.1.2 Многоуровневая модель

Расширением дискреционной модели доступа является многоуровневая модель доступа (MLS). Объекты в многоуровневой модели имеют различные уровни доступа (например, уровни секретности), а субъекты - степени допуска. Разрешение допуска субъекта к объекту является функцией от степени допуска конкретного субъекта и уровня допуска конкретного объекта.

Многоуровневая модель доступа создана на основе теории алгебраических решеток. Данные могут передаваться между субъектами, если выполняются следующие правила (буквы a, b и c обозначают идентификаторы субъектов, а буквы x, y и z, соответственно, их уровни доступа):

- данные могут передаваться субъектом самому себе: $x \leq x$;
- данные могут передаваться от субъекта a к субъекту c, если они могут передаваться от субъекта a к субъекту b и от субъекта b к субъекту c: если $x \leq y$ и $y \leq z$, то $x \leq z$;
- если $x \leq y$ и $y \leq x$, то $x = y$.

Рассмотренные правила представляют, соответственно, свойства рефлективности, транзитивности и анти симметричности.

Таким образом, многоуровневая модель имеет дело с множеством информационных потоков в системе и делит их на разрешенные и неразрешенные очень простым условием. Однако эта простота касается информационных потоков, которых в системе огромное количество.

Рассмотрим класс систем с двумя видами доступов «чтение» и «запись» (хотя могут быть и другие доступы, но они либо не определяют информационных потоков, либо выражаются через «чтение» и «запись»). Пусть процесс S в ходе решения своей задачи последовательно обращается к объектам O1, O2, ..., On (некоторые из них могут возникнуть в ходе решения задачи). Пусть:

$$S \xrightarrow{r} O_1, S \xrightarrow{r} O_2, S \xrightarrow{r} O_k, S \xrightarrow{w} O_1, S \xrightarrow{w} O_{j+1}$$

Тогда при выполнении условий c(S) > c(Oit), t=l,..k, (C – класс безопасности субъекта или объекта) соответствующие потоки информации будут идти в разрешенном направлении, а при c(S) < c(Ojt), t=l,.., n-k, потоки, определяемые доступом «запись», будут идти в разрешенном направлении.

Таким образом, в результате выполнения задачи процессом S, информационные потоки, с ним связанные, удовлетворяют условиям доступа. Такого анализа достаточно, чтобы классифицировать почти все процессы и принять решение о соблюдении или нет условий доступа. Если где-то условия нарушается, то соответствующий доступ не разрешается. Причем разрешенность цепочки (1) вовсе не означает, что субъект S не может создать объект O такой, что c(S) > c(0). Однако он не может писать туда информацию. При передаче управления поток информации от процесса S или к нему прерывается (хотя в него другие процессы могут записывать или считывать информацию как в объект). При этом, если правила направления потока при чтении и записи выполняются, то условие соблюдается, если нет, то соответствующий процесс не получает доступ. Таким образом, модель управляет потоками через контроль доступов.

В системе с двумя доступами чтение и запись модель определяется следующими правилами доступа:

$$X \xrightarrow{r} Y \Leftrightarrow c(X) \geq c(Y),$$

$$X \xrightarrow{w} Y \Leftrightarrow c(X) \leq c(Y).$$

Структура решетки очень помогает организации поддержки модели. Пусть имеется последовательная цепочка информационных потоков:

$$O_1 \xrightarrow{\quad} O_2 \xrightarrow{\quad} O_3 \xrightarrow{\quad} \cdots \xrightarrow{\quad} O_k$$

Если каждый из потоков разрешен, то свойства решетки позволяют утверждать, что разрешен сквозной поток.

Многоуровневая модель доступа в современных системах защиты реализуется через мандатный контроль (или, также говорят, через мандатную

политику). Мандатный контроль реализуется подсистемой защиты на самом низком аппаратно-программном уровне, что позволяет эффективно строить защищенную среду для механизма мандатного контроля. Устройство мандатного контроля, удовлетворяющее некоторым дополнительным, кроме перечисленных, требованиям, называется монитором обращений. Мандатный контроль еще называют обязательным, так как его проходит каждое обращение субъекта к объекту, если субъект и объект находятся под защитой системы безопасности. Организуется мандатный контроль следующим образом. Каждый объект O имеет метку с информацией о классе c(O). Каждый субъект также имеет метку, содержащую информацию о том, какой класс доступа c(S) он имеет. Мандатный контроль сравнивает метки и удовлетворяет запрос субъекта S к объекту O на чтение, если c(S) > c(0) и удовлетворяет запрос на запись, если c(S) < c(0). Многоуровневая модель доступа устойчива к атакам «Троянским конем». На чем строится защита от таких атак пояснено на примере.

Пример 2.

Пусть пользователи U1 и U2 находятся на разных уровнях, то есть c(U1) > c(U2). Тогда, если U1 может поместить в объект O1 ценную информацию, то он может писать туда и c(U2) < c(U1) < c(01), то есть c(U2) < c(O1). Тогда любой "Троянский конь" T, содержащийся в объекте O2, который может считать информацию в O1, должен отражать соотношение c(O2) ≥ c(O1). Тогда c(O2) > c(U2) и пользователь U2 не имеет право прочитать в O2, что делает съем в O1 и запись в O2 бессмысленным.

Многоуровневая модель создана, в основном, для сохранения секретности информации. Вопросы целостности при помощи этой политики не решаются или решаются как побочный результат защиты секретности.

Многоуровневая модель принята всеми развитыми государствами мира. В повседневном секретном делопроизводстве госсектор России также придерживается этой модели. Примером использования многоуровневой модели доступа является система контроля доступа, принятая в военном ведомстве США. Уровнями доступа выступают уровни секретности: НЕСЕКРЕТНО, КОНФИДЕНЦИАЛЬНО, СЕКРЕТНО, СОВЕРШЕННО СЕКРЕТНО. Внутри отдельных уровней секретности для выделения разделов данных, требующих специального разрешения на доступ к ним, определены категории: АТОМНЫЙ, НАТО и ДРУГИЕ. Для получения доступа к данным определенной категории субъект должен иметь не только доступ к данным соответствующего уровня (по секретности), но и разрешение на доступ по категории. Например, субъект, имеющий доступ к данным с уровнем СОВЕРШЕННО СЕКРЕТНО и категории НАТО, не может получить доступ к данным с категориями АТОМНЫЙ и ДРУГИЕ уровня СОВЕРШЕННО СЕКРЕТНО.

1.1.3 Модель Белла-Лападулы

Модель Белла-Лападулы состоит из следующих элементов:

- множества субъектов S;
- множества объектов O;
- множества уровней защиты L;
- множества прав доступа G;
- списка текущего доступа b;
- списка запросов Z.

Схема модели представлена на рисунке 1.3.

Рисунок 1.3 – Схема модели

Каждому субъекту sÎS сопоставляются два уровня защиты: базовый Ls(Si)ÎL, задаваемый в начале работы и остающийся неизменным, и текущий It(Si)ÎL, зависящий от уровней защиты тех объектов, к которым субъект Si имеет доступ в настоящий момент времени.

Множество объектов O наделяется структурой дерева таким образом, что каждому объекту Oj соответствует список объектов, непосредственно следующих за ним (объектов-сыновей) и, если Oj отличен от корня дерева, то существует единственный объект O(j), непосредственно предшествующий ему (отец объекта Oj).

Каждому объекту Oj приписывается уровень защиты I(Oj)ÎL. Множество L, является конечным частично упорядоченным множеством, обладавшим свойством алгебраической решетки. Возможно представление каждого уровня защиты LrÎL, в виде вектора из двух компонент: классификации и множества категорий.

Уровень защиты I1 больше уровня защиты I2, если классификация I1 больше или равна классификации I2, и множество категорий I1 содержит множество категорий I2(в формализованном виде: I1>=I2).

Множество прав доступа G имеет вид G={r, a, w, e}, где:

– r – чтение объектом субъекта (получение субъектом данных, содержащихся в объекте);

– a – модификация данных объекта субъектом без их предварительного прочтения;

– w – запись-модификация данных объекта после их предварительного прочтения субъектом;

– e – исполнение субъектом объекта (действие, не связанное ни с чтением, ни с модификацией данных).

Расширения модели допускают использование права C – управления, при котором субъект Si может передать права доступа, которыми он владеет по отношению к объекту Oj, другому субъекту Sk.

При описании модели будем рассматривать следующую схему управления передачей прав доступа: передача прав доступа к объекту Oj субъектом Si субъекту Sk связана с наличием у субъекта Si права w к "отцу" объекта Oj.

Матрица доступа M=|| Mij || не содержит пустых столбцов. Однако, наличие элемента Mij <> 0 является необходимым, но не достаточным условием разрешения доступа.

Список текущего доступа b содержит записи вида (Si , Oj , X), если субъекту Si разрешен доступ xÎG к объекту Oj и это разрешение к настоящему моменту времени не отменено. Разрешение доступа действительно до тех пор, пока субъект не обратился с запросом на отказ от доступа к монитору.

Список запросов Z описывает возможности доступа субъекта к объекту, передачи прав доступа другим субъектам, создания или уничтожения объекта. В модели Белла-Лападулы рассматриваются 11 следующих запросов:

- запрос на чтение (r) объекта;
- запрос на запись (w) в объект;
- запрос на модификацию (a) объекта;
- запрос на исполнение (e) объекта;
- запрос на отказ от доступа;
- запрос на передачу доступа к другому объекту;
- запрос на лишение права доступа другого субъекта;
- запрос на создание объекта без сохранения согласованности;
- запрос на создание объекта с сохранением согласованности;
- запрос на уничтожение объекта;
- запрос на изменение своего текущего уровня защиты.

Белл и Лападула сформулировали два условия защиты для модели:

- простое условие защиты;
- * - условие.

Простое условие защиты предложено для исключения прямой утечки секретных данных и состоит в следующем. Если субъекту Sj запрещен доступ:

- по чтению r объекта Oj, тогда Is(Si) ≤ I(Oj);
- по записи w в объект Oj, тогда Is(Si) ≤ I(Oj).

Простое условие защиты накладывает ограничения на базовые уровни защищенных объектов.

-условие защиты (-свойство) предназначено для предотвращения косвенной утечки данных. Это условие накладывает ограничения на уровни защиты тех объектов, к которым субъект может иметь доступ одновременно. Если субъект S имеет доступ X1 к объекту O1 и доступ X2 к объекту O2, то, в зависимости от вида доступа, должны выполняться соотношения между уровнями защиты, приведенные в таблице 2.

Введение *-условия предназначено для предотвращения потока данных вида "чтение объекта для переписи данных в объект с меньшим либо несравнимым уровнем защиты" (напомним, что доступ «запись» предполагает предварительное чтение).

Состояние системы считается безопасным, если соотношения между уровнями защиты объектов и субъектов удовлетворяют как простому условию защиты, так и *-условию.

Таблица 2 – Соотношения между уровнями защиты

Доступ к O1	Доступ к O2	Соотношение
Чтение – r	Дополнение – a	l(S1) ≤ l(O2)
Чтение – r	Запись – w	l(S1) ≤ l(O2)
Запись – w	Дополнение – a	l(S1) ≤ l(O2)
Запись – w	Запись – w	l(S1) ≤ l(O2)

Система защиты должна обеспечивать безопасность данных, если она не допускает перехода из безопасного состояния в состояние, не являвшееся безопасным. Для обеспечения безопасности данных необходимо и достаточно, чтобы изменение состояний системы приводило только к безопасным состояниям, если исходное состояние было безопасным.

Белл и Лападула описали правила разрешения каждого из 11 возможных запросов. Эти правила используют понятие текущего уровня защиты субъекта, которое определяется следующим образом.

Пусть Or, Ov, Oa – множества тех объектов, к которым субъект S имеет доступ r, w и a соответственно.

– если Ov <> 0, то It(s) = max I(Oj) = min I(Oj), (OjÎOv) (OjÎOv);

– если Ov = 0, то It(s) может иметь уровень защиты в пределах: max I(Oj) ≤ It(s) ≤ minI(Oj) , (OjÎOr) (OjÎOa).

В случае, если нижняя граница в этом неравенстве берется по пустому множеству, она полагается равной min I(Oj), (OjÎOv), а верхняя – Ib(s).

Рассмотрим правила выполнения каждого из 11 возможных запросов в модели Белла-Лападулы.

1) запрос на чтение субъектом Si объекта Oj разрешается, если выполняется условие: R Î Mij V Is(Si) ≥ I(Oj) V It(Si) ≥ I(Oj). При невыполнении условия запрос отвергается.

2) запрос на запись субъектом Si в объект Oj разрешается, если выполняется условие: W Î Mij V Is(Si) ≥ I(Oj) V It(Si) = I(Oj). При невыполнении условия запрос отвергается.

3) запрос на дополнение субъектом Si объекта Oj разрешается, если выполняется условие: A Î Mij V It(Si) ≤ I(Oj). При невыполнении условия запрос отвергается.

4) запрос на исполнение субъектом Si объекта Oj разрешается, если e Î Mij, и отвергается – в противном случае.

5) отказ субъекта Si от доступа x Î G к объекту разрешается безусловно. При этом происходит изменение состояния системы : b' = b \ {Si, Oi, x}.

6) передача субъекту Sk субъектом Si права на доступ x к объекту Oj разрешается, если субъект Sk имеет доступ w к «отцу» Os(j) объекта Oj, то есть если {Sk,Os(j),w} Î b, и отвергается – в противном случае. При этом происходит изменение состояния системы: M'ij = Mij U {x}, x Î G.

7) лишение субъекта Sk субъектом Si права на доступ x к объекту Oj разрешается, если субъект Sk имеет доступ w к «отцу» Os(j) объекта Oj, то есть если {Sk,Os(j),w} Î b и отвергается – в противном случае. При этом происходит изменение состояния системы: M'ij = Mij \ {x}, x Î G.

8) создание субъектом Si объекта Ot(j) с уровнем защиты I, являющегося "сыном" объекта Oj, разрешается, если список текущего доступа b содержит записи: {Si,Oj,w} V {Si,Oj,a} и отвергается – в противном случае. При этом происходит изменение состояния системы: O' = O U {Ot(j)}; M' = M U Mt(j), где столбец Mt(j) содержит один ненулевой элемент Mit(j), значение которого зависит от дополнительного параметра, указываемого при запросе. Оно может принимать значения либо {r,a,w}, либо {r,a,w,e).

9) создание субъектом Si объекта Ot(j) с уровнем защиты I, являющегося «сыном» объекта Oj, с сохранением согласованности разрешается, если список текущего доступа b содержит записи: {Si,Oj,w} V {Si,Oj,a} V I > I(Oj), и отвергается – в противном случае. Изменение состояния системы происходит аналогично восьмому запросу.

10) уничтожение субъектом Si объекта Oj (и всех объектов Oj1,Oj2,...Ojk, являющихся «последователями» по структуре дерева) разрешается, если субъект Si имеет доступ w к «отцу» Os(j) объекта Oj, и отвергается - в противном случае. Изменение состояния системы происходит следующим образом: из списка текущего доступа b удаляются все записи, содержащие объекты Oj1,Oj2,...Ojk; из матрицы M удаляются столбцы с номерами j1,j2,...jk.

11) изменение субъектом Si своего текущего уровня защиты It(Si) на It' разрешается, если выполняются условия:

а) Is(Si) ≥ It' и

б) It' ≤ I(Oj), если субъект Si имеет доступ a к какому-либо объекту o, и

в) It' = I(Oj), если субъект Si имеет доступ w к какому-либо объекту Oj

г) It'>=I(Oj), если субъект Si имеет доступ r к какому-либо объекту Oj.

В противном случае запрос отвергается.

Монитор обработки рассмотренных 11 запросов, созданный на основе модели Белла и Лападулы, был реализован в виде программно-аппаратного ядра защиты ОС Multics.

1.2 Контроль доступа в Windows NT (XP)

1.2.1 Структура системы контроля доступа

Каждый объект в системе Windows NT имеет свои атрибуты безопасности, определяемые дескриптором безопасности SD (Security Descriptor). Дескриптор безопасности содержит информацию о владельце объекта и список управления доступом ACL, в котором описываются права доступа пользователей и групп пользователей к данному объекту.

Существует два типа ACL:

– дискреционный список (DACL) – управляется владельцем объекта и описывает права доступа отдельных пользователей и групп пользователей.

– системный список (SACL) – управляется системным администратором и позволяет применить к объекту политику безопасности на уровне системы.

Использование обоих типов списков, похоже. Однако, т.к. системный список может управляться системным администратором, большинство разрабатываемых программ определяют уровни безопасности через дискреционный список.

В дискреционном списке содержатся записи о каждом пользователе, локальной и глобальной группе, определяющие права доступа к объекту. Каждая такая запись находится в форме элемента управления доступом. Каждый элемент содержит структуру ACE_HEADER, описывающую права доступа и идентификатор безопасности. Заголовок определяет тип элемента, его размер и контрольные флаги. Привязка определенной записи к конкретному пользователю осуществляется через его идентификатор безопасности SID.

Если дискреционный список не содержит ни одной записи ACE, то любой доступ к объекту запрещен. Однако, когда объект не имеет дискреционного списка, его защита отсутствует и доступ открыт для всех. При инициализации дескриптора безопасности дискреционный список не содержит ни одной записи. Для разрешения доступа к объекту указатель на дискреционный список в дескрипторе безопасности должен быть равен NULL.

В Windows NT используется два типа дескриптора безопасности: абсолютный и относительный. Разница между ними очень существенная. Абсолютный SD содержит указатели на дискреционный список, в то время, как относительный – содержит всю информацию в блоке памяти. Функции API, управляющие безопасностью используют оба формата в разных случаях. При запросе информации о дескрипторе безопасности всегда возвращается

дескриптор в относительном формате. При этом программа может считать информацию из блока. Однако выполнение изменений в дескрипторе должно производиться в абсолютном формате. Когда программа меняет дескриптор, она должна считать его в одном формате, конвертировать в другой, произвести изменения и записать его.

1.2.2 Цели системы контроля доступа

Целями системы контроля доступа являются проверка уровня доступа и контроль над действиями клиента.

Для достижения первой цели, требуется убедиться в том, что пользователь обладает правом доступа к защищаемому объекту. Если это не так, попытка доступа должна окончиться неудачей, а пользователь должен получить сообщение «доступ запрещен» (access denied). При разработке программы, ограничивающей доступ к данным, возможно использование любого допустимого метода реагирования на подобную попытку доступа.

Вторая функция системы безопасности — аудит, то есть слежение за действиями клиента — выполняется не всегда. Аудит подразумевает документирование в файле журнала действий клиента, связанных с доступом к тому или иному защищаемому объекту. Администраторы могут включать или выключать систему аудита, они также могут настроить ее таким образом, чтобы документировались не все попытки доступа, а только те, которые были связаны с нарушением прав доступа и в результате оказались безуспешными.

Система безопасности не очень усложняет работу с операционной системой, т.к. в том случае если нет необходимости напрямую работать с механизмом безопасности, при использовании системных вызовов в качестве дескриптора безопасности можно передавать NULL. В таком случае система использует дескриптор по умолчанию. Это существенно упрощает разработку программ, не осуществляющих взаимодействия с системой безопасности Windows напрямую.

1.2.3 Права и привилегии

Система безопасности следит за выполнением тех или иных действий клиента при помощи двух основных механизмов. Эти два механизма — это права и привилегии.

Привилегия – это разрешение на выполнение некоторым пользователем некоторого действия в отношении всей системы в целом. Список привилегий приведен в таблице 3.

Таблица 3 – Список привилегий в Windows NT

SE_ASSIGNPRIMARYTOKEN_NAME	Назначение основного токена процесса
SE_AUDIT_NAME	Внесение записей в журнал аудита. Этой привилегией обычно обладают защищенные серверы
SE_BACKUP_NAME	Осуществление резервного копирования
SE_CHANGE_NOTIFY_NAME	Прием уведомлений об изменении содержимого файлов и каталогов. Эта привилегия также отменяет любые проверки на перемещения по каталогам
SE_CREATE_PAGEFILE_NAME	Создание файла виртуальной памяти
SE_CREATE_PERMANENT_NAME	Создание постоянного объекта
SE_CREATE_TOKEN_NAME	Создание первичного токена (primary token)
SE_DEBUG_NAME	Отладка процесса
SE_INC_BASE_PRIORITY_NAME	Увеличение базового приоритета процесса
SE_INCREASE_QUOTA_NAME	Увеличение квоты, назначенной процессу
SE_LOAD_DRIVER_NAME	Загрузка или выгрузка драйвера устройства
SE_LOCK_MEMORY_NAME	Блокирование физических страниц в памяти
SE_PROF_SINGLE_PROCESS_NAME	Получение информации при профилировании процесса
SE_REMOTE_SHUTDOWN_NAME	Завершение работы системы по запросу, полученному через сеть
SE_RESTORE_NAME	Выполнение восстановления файлов из резервной копии. Обладая этой привилегией, вы можете назначить владельцем объекта любого пользователя или группу с известным идентификатором SID
SE_SECURITY_NAME	Осуществление разнообразных действий, связанных с безопасностью, например управление аудитом и просмотр сообщений системы аудита. Этой привилегией обычно обладают операторы системы безопасности Завершение работы локальной системы
SE_SHUTDOWN_NAME	Завершение работы локальной системы
SE_SYSTEM_ENVIRONMENT_NAME	Модификация энергонезависимой памяти систем, которые используют этот тип памяти для хранения конфигурационной информации
SE_SYSTEM_PROFILE_NAME	Осуществление профилирования всей системы
SE_SYSTEMTIME_NAME	Изменение системного времени
SE_TAKE_OWNERSHIP_NAME	Получение прав на владение объектом, не обладая при этом правами на доступ. Эта привилегия позволяет сделать владельцем объекта только пользователя, указанного владельцем привилегии
SE_TCB_NAME	Идентифицирует владельца привилегии как часть доверенной компьютерной базы. Этой привилегией обладают некоторые части операционной системы. Эта привилегия необходима для того, чтобы обратиться к функции LogonUser
SE_UNSOLICITED_INPUT_NAME	Чтение неожиданного ввода с терминала
SE_MACHINE_ACCOUNT_NAME	Создание учетной записи компьютера

В отличие от привилегий права применяются иначе. Право – это разрешение на выполнение некоторым пользователем некоторого действия в отношении некоторого отдельного объекта. Права назначаются в отношении конкретных объектов, доступ к которым контролируется операционной системой. Этими объектами могут быть файлы, тома NTFS, учетные записи и т. п. Например, пользователь может обладать правом чтения некоторого файла, но этот же пользователь может не обладать правом записи информации в этот файл. Права могут быть так-же негативными. Например, право может запрещать пользователю чтение защищаемого объекта (например, именованного канала). Кроме того, правом могут обладать не только отдельные пользователи, но и группы пользователей.

Так как права действуют в отношении защищаемых объектов, система хранит их вместе с этими объектами. Именно поэтому для защиты файлов на дисках в формате FAT нельзя использовать большую часть механизмов безопасности Windows 2000. Управлять доступом к объектам при помощи прав чрезвычайно удобно, так как при этом приходится иметь дело с самими защищаемыми объектами. Таким образом, отпадает необходимость уделять внимание тысячам пользователей, разбросанных по всей сети.

1.2.4 Дескрипторы безопасности и контроль доступа

Для выполнения некоторых системных вызовов требуется передача структуры SECURITY_ATTRIBUTES, которая определяет политику безопасности некоторого объекта.

Структура SECURITY_ATTRIBUTES:

Nlength: Размер структуры

IpSecurityDescriptor: Дескриптор безопасности (Security Descriptor), контролирующий доступ к объекту

blnheritHandle: Разрешает наследование дескриптора (handle) дочерним процессом

Структура SECURITY_DESCRIPTOR хранит в себе информацию, связанную с защитой некоторого объекта от несанкционированного доступа. В этой структуре, которую называют дескриптором безопасности, содержится информация о том, какие пользователи обладают правом доступа к объекту и какие действия эти пользователи могут выполнить в отношении этого объекта.

Структура SECURITY_DESCRIPTOR:

Revision: Содержит уровень дескриптора безопасности. Использование уровня позволяет передавать структуру между операционными системами или сохранять на диск.

Control: Набор флагов уточняющих поля дескриптора безопасности.

Owner: Указатель на структуру SID представляющую владельца объекта.

Group: Указатель на структуру SID представляющую группу объекта.

Dacl: Указатель на дискреционный список ACL. Поле присутствует только при установке флага DaclPresent. Если флаг установлен и поле имеет пустое значение, тогда к объекту разрешен всеобщий доступ.

Sacl: Указатель на системный список ACL.

Каждый пользователь (или группа пользователей) системы Windows 2000 обладает индивидуальным идентификатором SID. Когда пользователь пытается получить доступ к тому или иному объекту, обладающему дескриптором SECURITY_DESCRIPTOR, Windows анализирует его SID, используя при этом информацию, содержащуюся в SECURITY_DESCRIPTOR объекта. Если пользователь обладает необходимым уровнем доступа, система разрешит ему обратиться к объекту. Если уровень доступа, которым он обладает, недостаточен, система откажет в доступе. Наглядно данную процедуру демонстрирует рисунок 1.4.

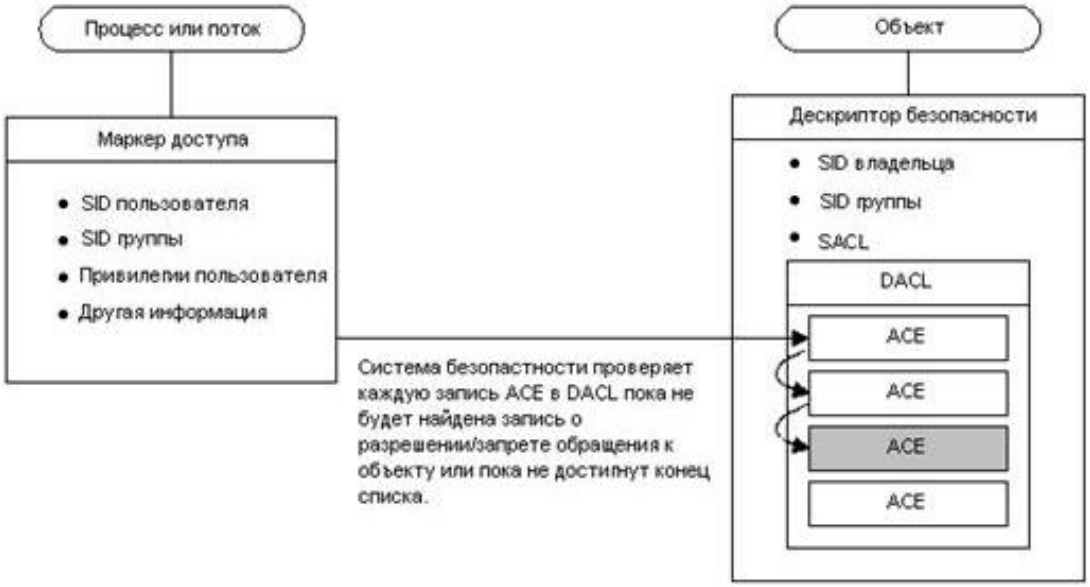

Рисунок 1.4 – Работа системы безопасности

Дескриптор безопасности объекта хранит в себе следующую информацию:

– SID владельца объекта;

– SID основной группы владельца объекта;

– дискреционный список управления доступом (Discretionary Access Control List, DACL);

– системный список управления доступом (System Access Control List, SACL);

– управляющая информация (например, сведения о том, как списки ACL пере-дают информацию дочерним дескрипторам безопасности);

Список DACL определяет, кто обладает правом доступа к объекту.

Список SACL определяет, информация о каких действиях вносится в файл журнала.

Для создания дескриптора безопасности используется системный вызов InitializeSecurityDescriptor. Этой функции следует передать указатель на дескриптор безопасности SECURITY_DESCRIPTOR и значение DWORD, которое содержит номер ревизии структуры SECURITY_DESCRIPTOR, В качестве второго аргумента следует использовать значение SECURITY_DESCRIPTOR_REVISION.

Функция InitializeSecurityDescriptor инициализирует дескриптор таким образом, что в нем отсутствует DASL, SACL, владелец и основная группа владельца, а все управляющие флаги установлены в значение FALSE.

При этом дескриптор имеет абсолютный формат. Дескриптор в абсолютном (absolute) формате содержит лишь указатели на информацию, связанную с защитой объекта. В отличие от этого дескриптор в относительном (self-relative) формате включает в себя всю необходимую информацию, которая располагается в памяти последовательно поле за полем. Таким образом, абсолютный дескриптор нельзя записать на диск (так как при последующем чтении его с диска все указатели потеряют смысл), а относительный дескриптор – можно.

Windows позволяет преобразовывать абсолютный дескриптор в относительный и обратно. Обычно это требуется лишь в случае, если вы записываете дескриптор на диск и считываете дескриптор с диска. Системные вызовы, требующие передачи указателя на дескриптор безопасности, работают только с дескрипторами в абсолютном формате.

1.2.5 Элементы управления доступом

Элемент управления доступом (Access Control Element, ACE) — это запись, которая указывает на то, что некоторый пользователь (или группа) обладает определенным правом. Записи ACE бывают трех основных разновидностей:

– ACCESS_ALLOWED_ACE_TYPE: наделяет пользователя (или группу) некоторым правом;

– ACCESS_DENIED_ACE_TYPE: отменяет это право в отношении пользователя (или группы);

– SYSTEM_AUDIT_ACE_TYPE: обозначает необходимость осуществления аудита при пользовании правом.

Список управления доступом (Access Control List, ACL) — это набор записей ACE. Дискреционный список DACL содержит записи типа ACCESS_ALLOWED_ACE_TYPE и ACCESS_DENIED_ACE_TYPE, а системный список SACL содержит записи типа SYSTEM_ AUDIT_ACE_TYPE.

Каждая запись ACE, определяющая уровень доступа к объекту, обладает 32-битной маской ACCESS_MASK. Эта маска является набором бит, которые определяют, какие права предоставляет или отменяет данная

запись ACE. Значение каждого бита определяется объектом, по отношению к которому применяется данная ACE.

Например, набор прав, которые можно назначить в отношении семафора, отличается от набора прав, которые можно назначить в отношении файла. Все же существуют четыре права, которые можно назначить в отношении абсолютно любого защищаемого объекта, это:

GR GENERIC_READ (обобщенное чтение),
GW GENERIC_WRITE (обобщенная запись),
GE GENERIC_EXECUTE (обобщенное исполнение)
GA GENERIC_ALL (все права).

Этим четырем правам всегда соответствуют одни и те же биты в маске ACCESS_MASK любой записи ACE.

Стандартный набор прав и дополнительные наборы прав для файлов, для ключей реестра, для процессов представлены на рисунках 1.5-1.8 соответственно.

Константа	Значение
DELETE	Право на удаление объекта
READ_CONTROL	Право на чтение дескриптора безопасности объекта, за исключением информации о SACL
SYNCHRONIZE	Право на использование объекта для синхронизации.
WRITE_DAC	Право на изменение DACL объекта в дескрипторе безопасности
WRITE_OWNER	Право на изменение владельца объекта в дескрипторе безопасности
STANDARD_RIGHTS_ALL	Сочетание прав DELETE, READ_CONTROL, WRITE_DAC, WRITE_OWNER, и SYNCHRONIZE access.
STANDARD_RIGHTS_EXECUTE	Аналогично READ_CONTROL.
STANDARD_RIGHTS_READ	Аналогично READ_CONTROL.
STANDARD_RIGHTS_REQUIRED	Сочетание прав DELETE, READ_CONTROL, WRITE_DAC, и WRITE_OWNER.
STANDARD_RIGHTS_WRITE	Аналогично READ_CONTROL.

Рисунок 1.5 – Стандартный набор прав

Константа	Значение
FILE_READ_ATTRIBUTES	Чтение атрибутов файла.
FILE_READ_DATA	Чтение данных из файла. Просмотр содержимого для директории
FILE_READ_EA	Чтение дополнительных атрибутов
SYNCHRONIZE	Использование дескриптора файла в функциях ожидания
STANDARD_RIGHTS_WRITE	Аналогично READ_CONTROL из стандартного набора прав
STANDARD_RIGHTS_READ	Аналогично READ_CONTROL из стандартного набора прав

Рисунок 1.6 – Дополнительный набор прав для файлов

Константа	Значение
KEY_ALL_ACCESS	Объединение прав STANDARD_RIGHTS_REQUIRED, KEY_QUERY_VALUE, KEY_SET_VALUE, KEY_CREATE_SUB_KEY, KEY_ENUMERATE_SUB_KEYS, KEY_NOTIFY, KEY_CREATE_LINK
KEY_CREATE_LINK	Создание ключа реестра
KEY_CREATE_SUB_KEY	Создание подключа рееста
KEY_ENUMERATE_SUB_KEYS	Required to enumerate the subkeys of a registry-key object.
KEY_EXECUTE	Аналогично KEY_READ
KEY_NOTIFY	Required to request change notifications for a registry key or for subkeys of a registry key.
KEY_QUERY_VALUE	Required to query a value of a registry-key object.
KEY_READ	Combines the STANDARD_RIGHTS_READ, KEY_QUERY_VALUE, KEY_ENUMERATE_SUB_KEYS, and KEY_NOTIFY values.
KEY_SET_VALUE	Required to create or set a value of a registry-key object.
KEY_WRITE	Объединение прав STANDARD_RIGHTS_WRITE, KEY_SET_VALUE, KEY_CREATE_SUB_KEY

Рисунок 1.7 – Дополнительный набор прав для ключей реестра

Константа	Значение
PROCESS_ALL_ACCESS	Объединение всех возможных наборов прав для процесса объекта
PROCESS_CREATE_PROCESS	Создание процесса
PROCESS_CREATE_THREAD	Создание потока
PROCESS_DUP_HANDLE	Диблирование хэндла
PROCESS_QUERY_INFORMATION	Получение определенной информации о процессе, например приоритет класса.
PROCESS_SET_INFORMATION	Установка определенной информации процесса, например приоритет класса.
PROCESS_TERMINATE	Уничтожение процесса
PROCESS_VM_OPERATION	Выполнение определенных операций в адресном пространстве процесса
PROCESS_VM_READ	Чтение данных из адресного пространства процесса
PROCESS_VM_WRITE	Запись данных в адресное пространство процесса
SYNCHRONIZE	Ожидание уничтожения процесса

Рисунок 1.8 – Дополнительный набор прав для процессов

Каждый системный объект или файл обладает индивидуальным списком ACL. Чтобы обеспечить защиту данных, о существовании которых система не имеет представления, необходимо самостоятельно сформировать собственный список ACL и сохранить его специальным образом.

Процесс определения прав через списки ACL представлен на рисунке 1.9.

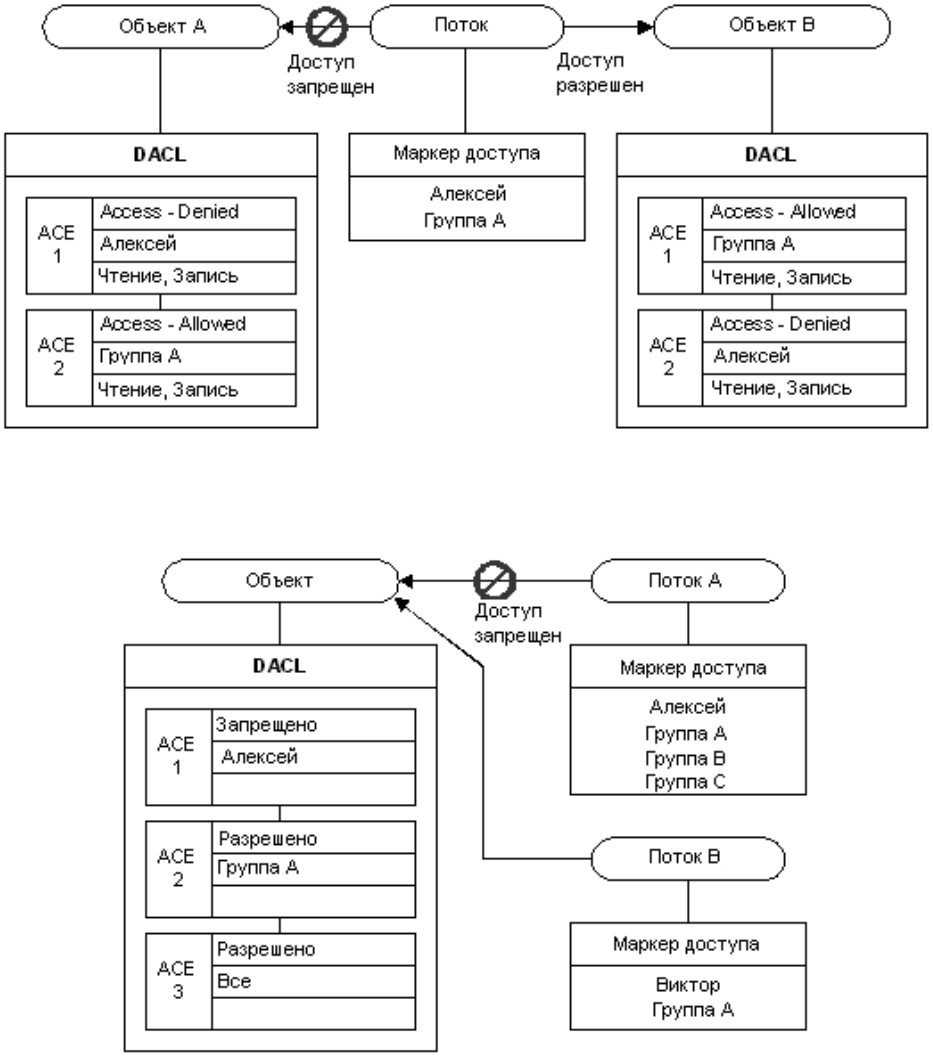

Рисунок 1.9 – Процесс определения прав

Для всех ниже перечисленных объектов имеется возможность установки дополнительных наборов прав:

- файлы и директории (при использовании NTFS);
- каналы;
- процессы;
- потоки;
- файлы, отображенные в память;
- маркеры доступа;
- объекты рабочего стола;
- реестр;
- службы;
- локальные или сетевые принтеры;
- общие ресурсы;
- события, мьютексы, семафоры, таймеры;

1.2.6 Маркеры доступа

Выполнение действий от другого имени (impersonation) — важная составляющая системы безопасности Windows. Благодаря этому механизму сервер, обладающий широкими полномочиями в отношении собственных ресурсов, может выполнять действия от имени клиента, который обладает ограниченными полномочиями. Если клиент обращается к серверу с просьбой выполнить то или иное действие, сервер выполняет это действие, используя уровень безопасности клиента, а не уровень безопасности сервера. Таким образом, клиент, обратившийся к серверу, не сможет работать с объектами на стороне сервера, используя при этом уровень безопасности сервера.

Если процесс (поток) намерен выполнять действия от имени другого процесса (потока), он должен использовать вызов наподобие ImpersonateNamedPipeClient или ImpersonateLoggedOnUser. При этом создается так называемый токеп персонализации (impersonation token), который присваивается вызывающему потоку. Некоторые вызовы требуют предварительного создания токена персонализации даже в случае, если программа обращается к ним от своего собственного имени. Чтобы создать такой токен, следует использовать вызов ImpersonateSelf. После обращения к этому вызову процесс становится обладателем токена персонализации, который соответствует ему самому.

Для того, чтобы выступить от имени другого пользователя необходимо использовать вызов: ImpersonateLoggedOnUser. В качестве аргумента этому вызову следует передать токен пользователя, от имени которого будет выступать вызывающий процесс. Токен пользователя опередеяется при помощи вызова LogonUser. Для этого необходимо знать регистрационное имя и пароль интересующего пользователя.

Токен – это набор атрибутов безопасности, соответствующий некоторому процессу.

Основной токен (primary token) создается модулем Executive операционной системы Windows и содержит конфигурацию атрибутов безопасности процесса, которому он соответствует. Токен персонализации (impersonate token) создается для того, чтобы сообщить операционной системе о том, что соответствующий такому токену процесс или поток выполняет действия от имени другого процесса или потока.

Получить токен процесса или потока можно при помощи вызовов OpenProcessToken и OpenThreadToken.

2 Описание лабораторной установки

Описание редактора объектов системы безопасности, редактора субъектов системы безопасности и редактора действий представлены на рисунках 2.1-2.3.

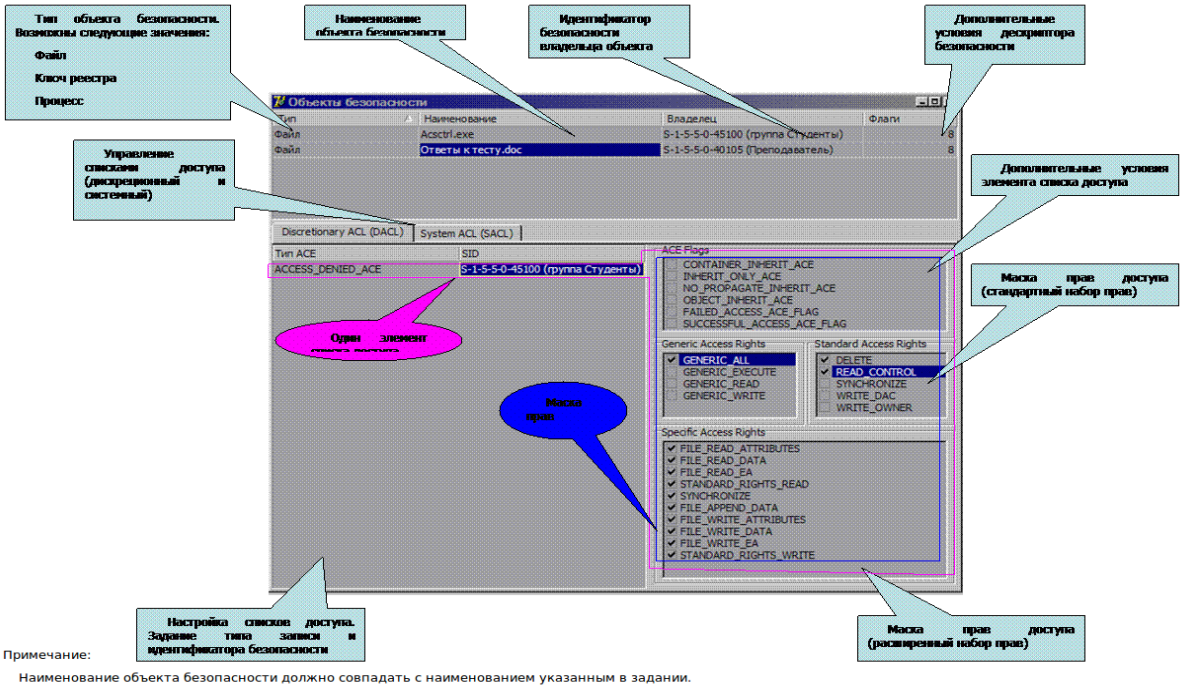

Рисунок 2.1 – Редактор объектов системы безопасности

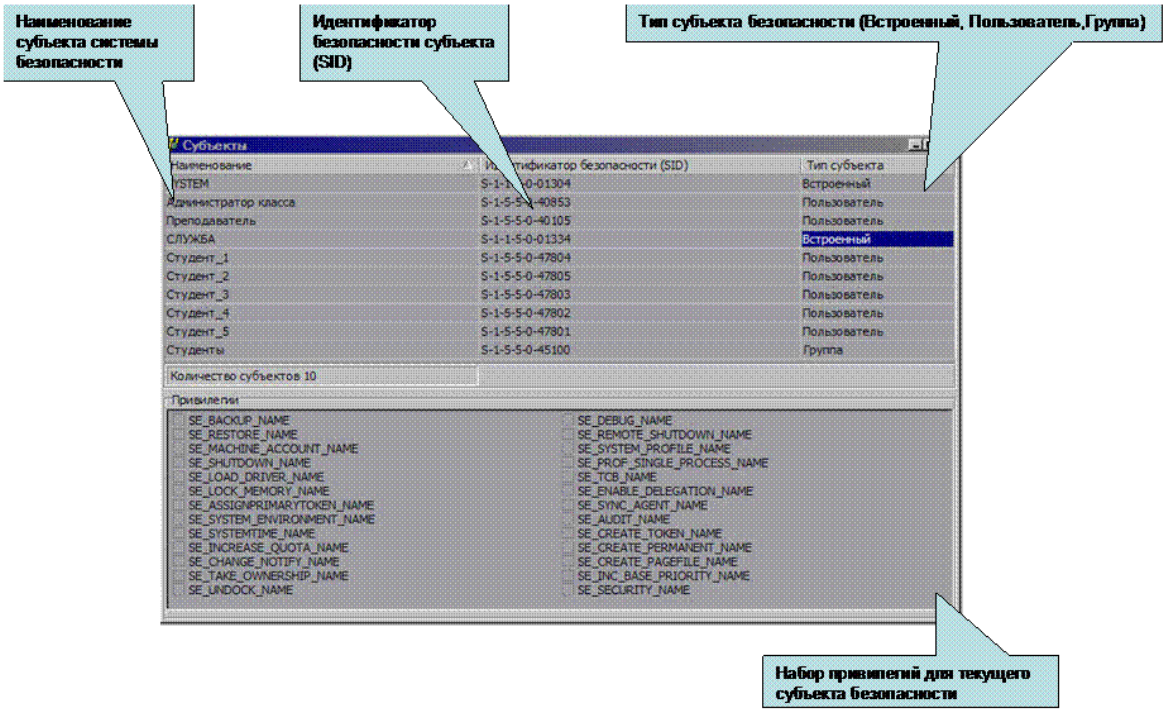

Рисунок 2.2 – Редактор субъектов системы безопасности

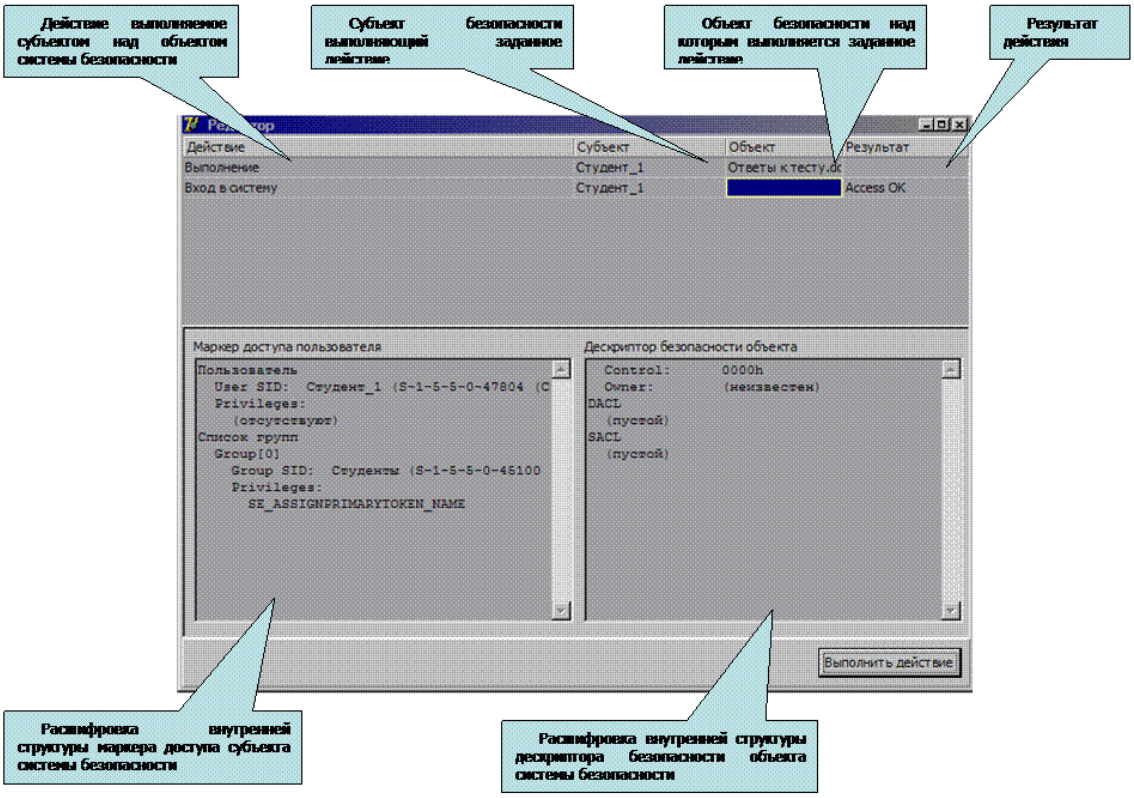

Рисунок 2.3 – Редактор действий

Замечания:

– сохранение текущего проекта происходит автоматически;

– содержимое редактора выполняемых действий и журнал аудита не сохраняется.

3 Выполнение лабораторной работы

После запуска лабораторного практикума необходимо создать новый проект и выбрать вариант задания (рисунок 3.1) Файл проекта содержит все объекты, созданные в ходе выполнения лабораторной работы.

Далее необходимо приступить к созданию субъектов и объектов моделируемой системы. Для создания и изменения субъектов безопасности предназначен редактор субъектов безопасности. Все субъекты должны быть созданы в соответствии с заданием на лабораторную работу.

Каждая строка таблицы соответствует одному субъекту моделируемой системы. При создании субъекта необходимо задать его характеристики:

- наименование субъекта;
- идентификатор безопасности (используется в списках доступа);
- тип субъекта (возможны три значения: пользователь, группа или встроенная учетная запись).

При создании субъектов необходимо указать необходимый набор привилегий для каждого субъекта. Список перечисленных в задании действий, который должен выполнять субъект в отношении операционной системы должен совпадать со списком установленных привилегий.

Добавление нового субъекта безопасности и назначение привилегий представлено на рисунках 3.2 - 3.4.

После создания всех необходимых субъектов моделируемой операционной системы следует приступить к созданию объектов (разделяемых ресурсов) моделируемой операционной системы.

Редактор объектов безопасности визуально разделен на две части:

- верхняя часть (таблица) используется для определения объектов (разделяемых ресурсов);
- нижняя часть используется для заполнения списков доступа, указания прав доступа к объектам, а так же указания дополнительной (служебной) информации относящейся к списку доступа.

При создании объекта необходимо указать его характеристики в верхней части редактора:

- тип объекта;
- наименование объекта;
- владелец объекта;
- дополнительные атрибуты объекта (дескриптора безопасности).

После того как требуемые объекты созданы, следует приступить к заполнению списков доступа находящихся в нижней части формы.

Пользователь может редактировать два списка: дискреционный и системный. Дискреционный список, как было описано выше, предназначен для ограничения доступа перечисленных в нем субъектов к объекту, которому принадлежит список. Системный список предназначен для

настройки аудита при обращении к объекту. Необходимо указать список прав доступа к объекту необходимых для реализации контроля доступа.

Добавление нового объекта и назначение прав доступа представлено на рисунках 3.5 – 3.8.

Далее необходимо заполнить таблицу в редакторе действий. Пример показан на рисунке 3.9.

В табличную часть редактора вводится список требуемых действий над операционной системой или над объектами. Кроме того, необходимо указать дополнительную информацию:

- субъект, осуществляющий доступ к объекту или ресурсам операционный системы;
- объект, к которому обращается субъект (в случае обращения к ресурсам операционной системы поле «объект» следует оставить пустым).

Выполнение действия выполняется нажатием кнопки «Выполнить действие». Для этого необходимо установить курсор на требуемую строку таблицы и нажать соответствующую кнопку.

Результат выполнения действия будет отображен в четвертой колонке таблицы. В случае успешного выполнения действия значение в ячейке будет «Access OK», в противном случае «Access denied». Так же на форме отображены внутренние представления структур объекта и субъекта в системе контроля доступа. При изучении представленной информации можно определить место ошибки.

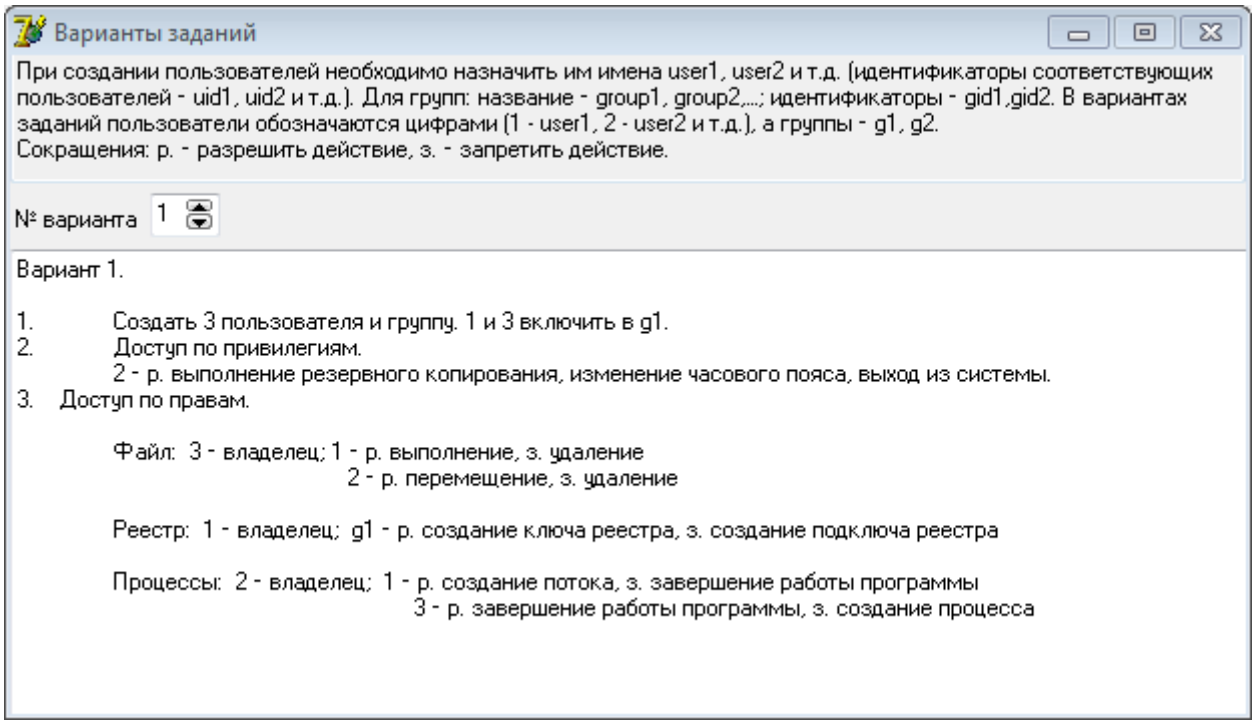

Рисунок 3.1 – Вариант задания

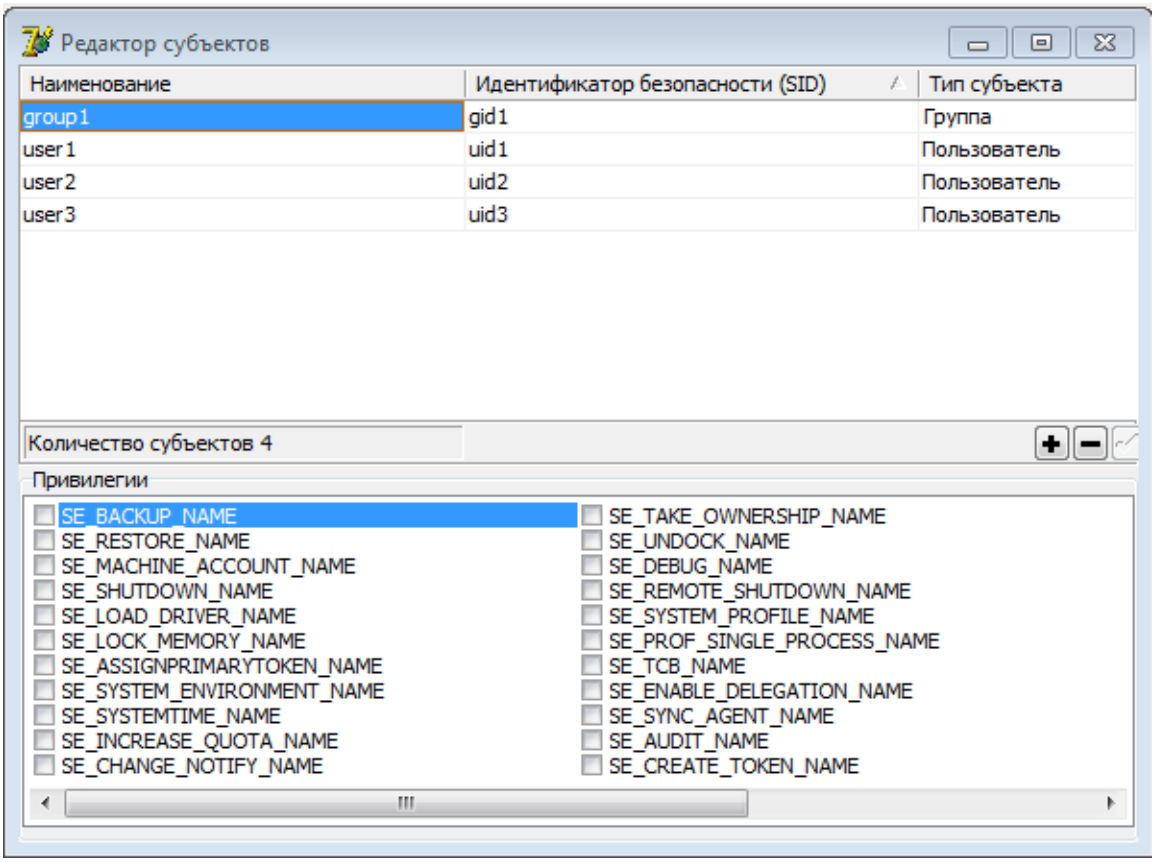

Рисунок 3.2 – Редактор субъектов

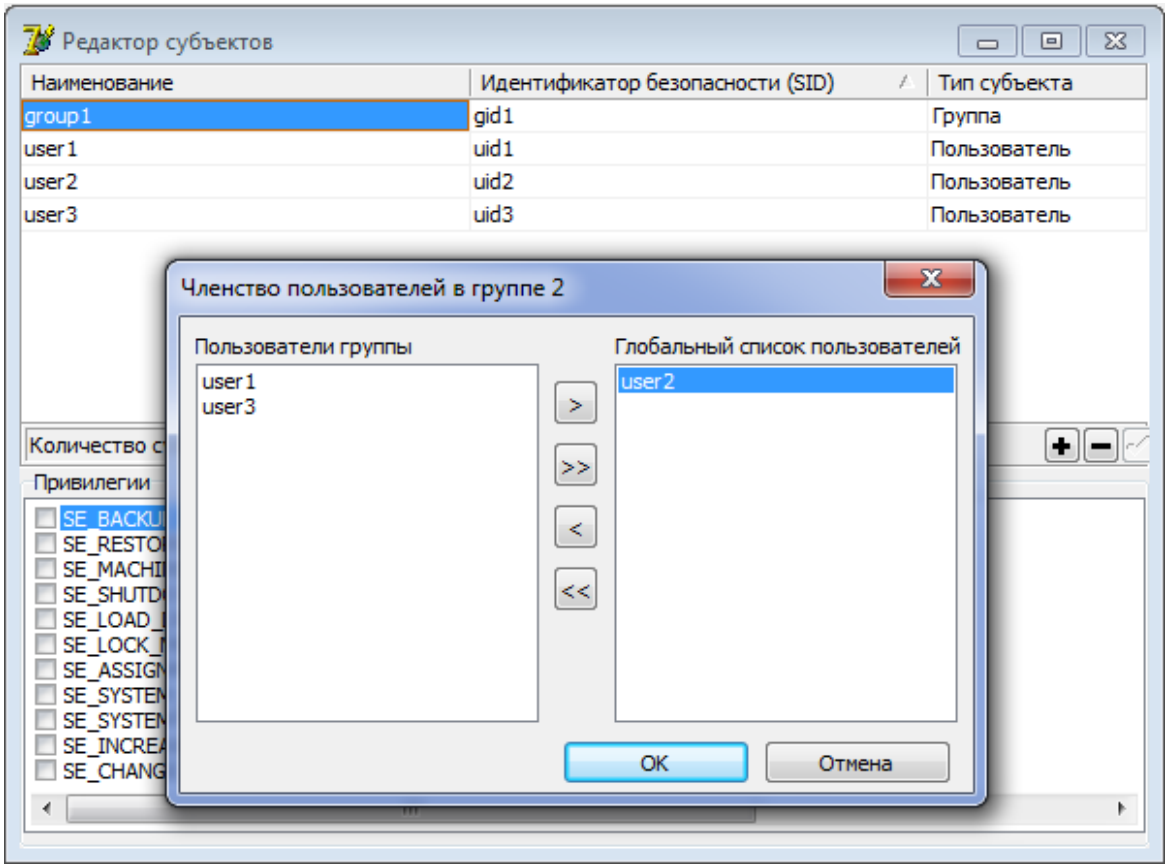

Рисунок 3.3 – Добавление пользователей в группу

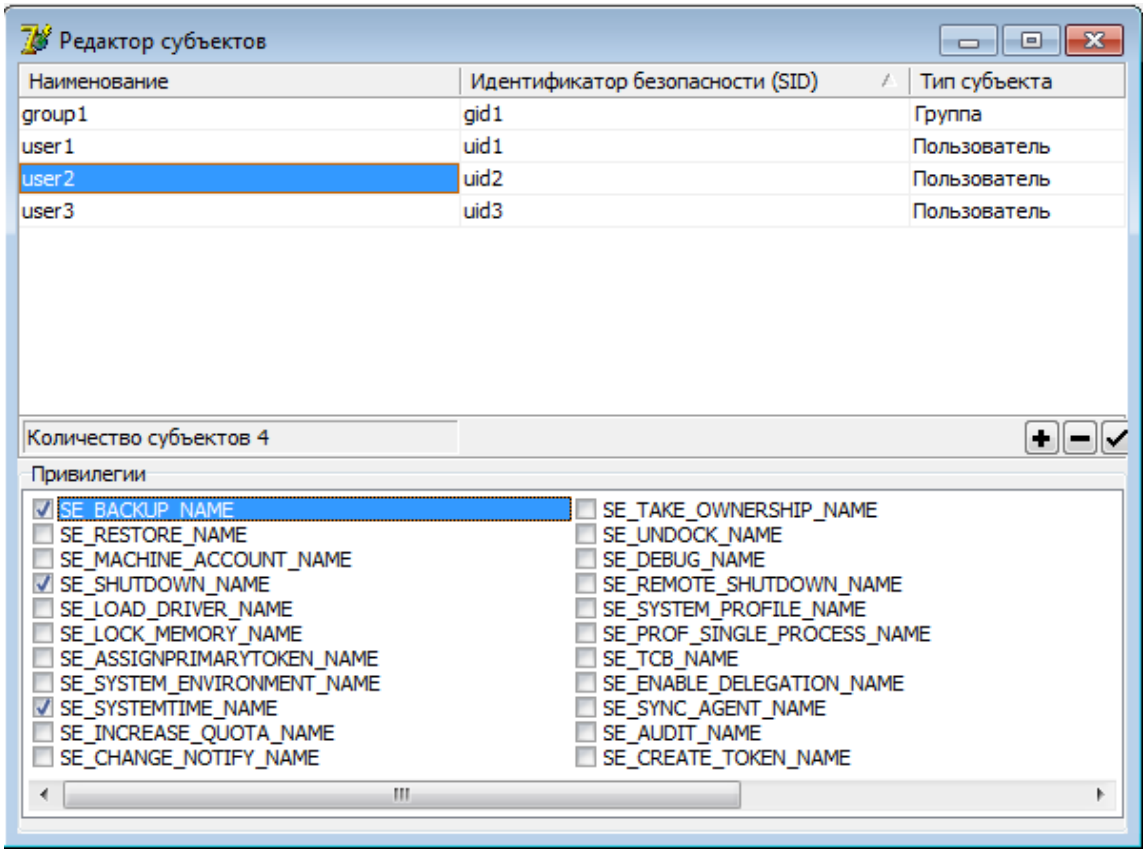

Рисунок 3.4 – Добавление привилегий

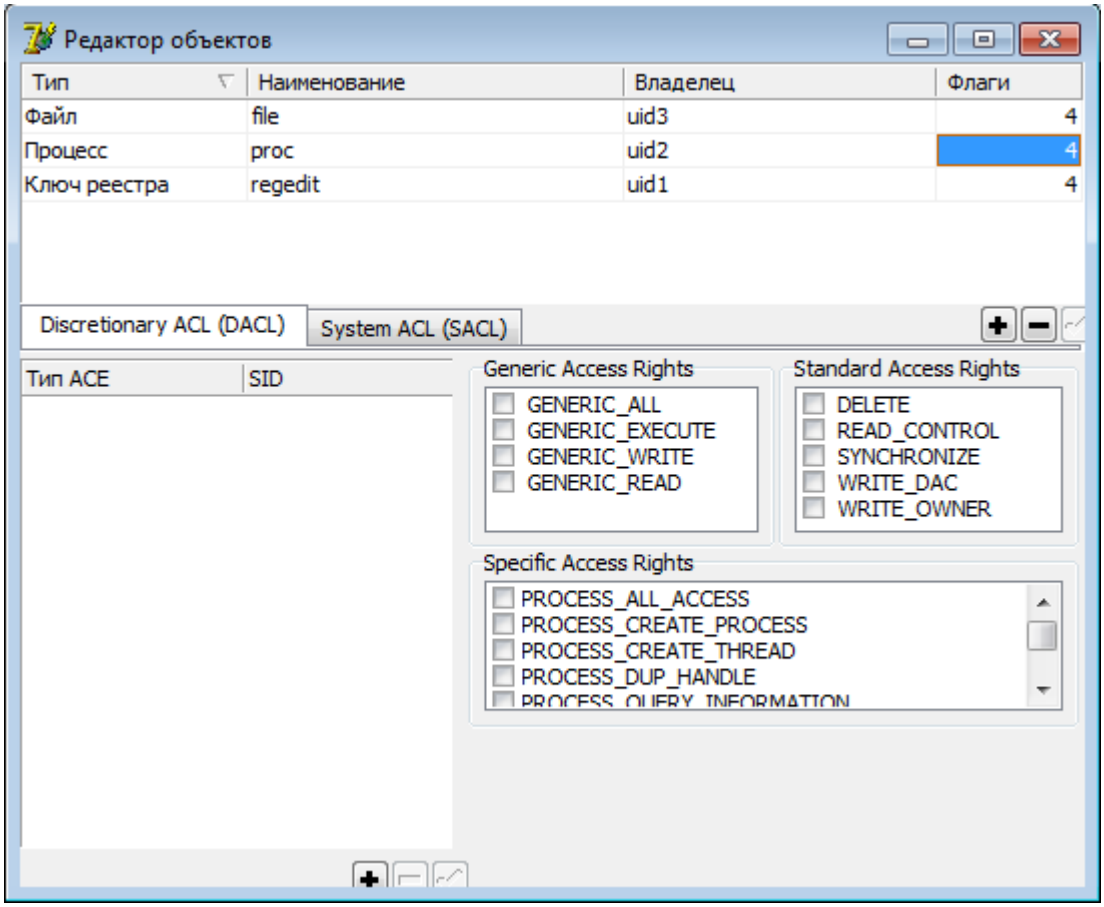

Рисунок 3.5 – Редактор объектов

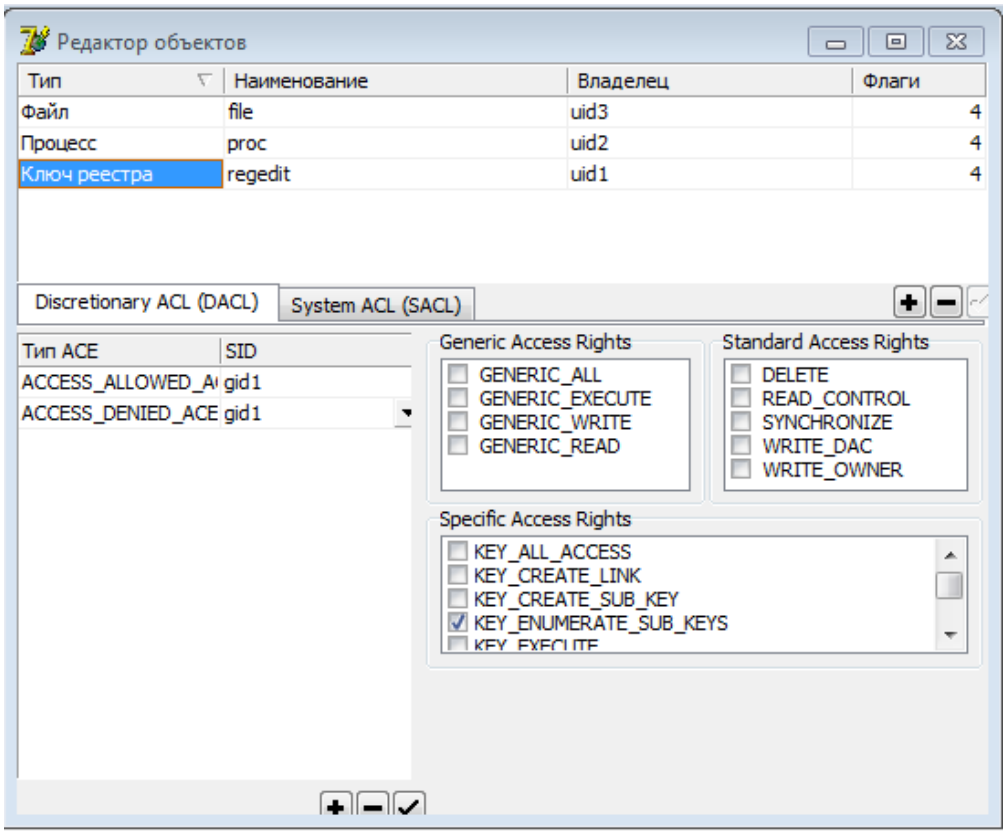

Рисунок 3.6 – Добавление прав доступа для ключа реестра

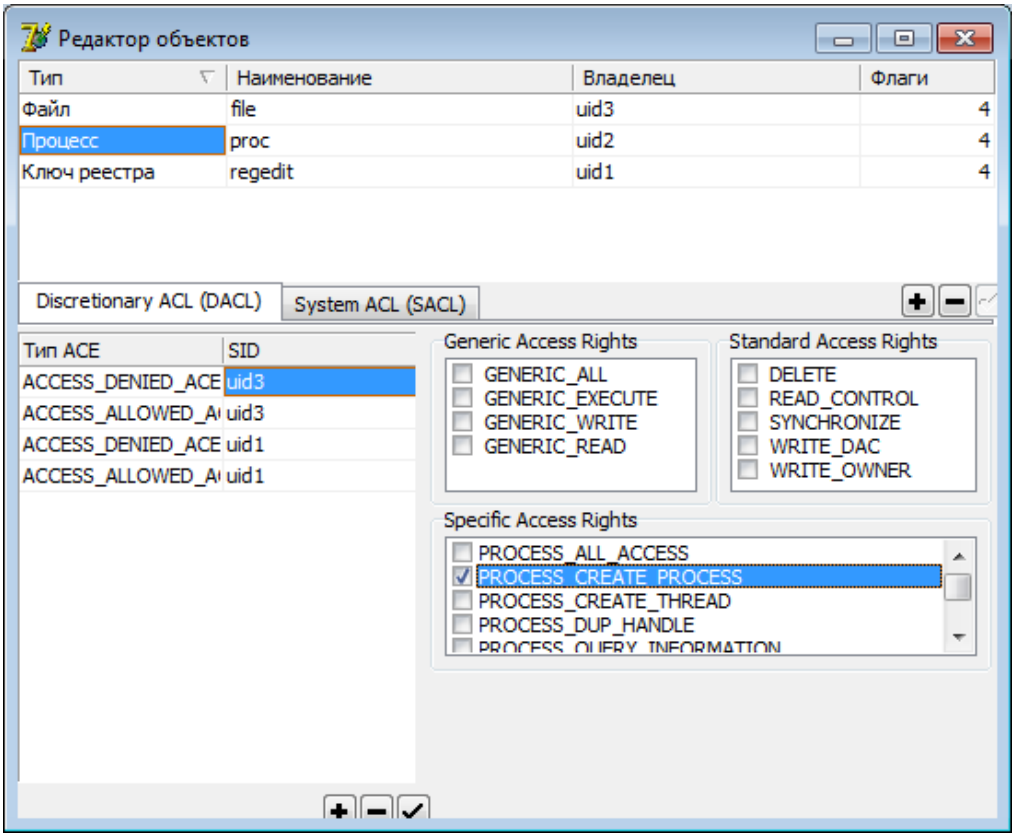

Рисунок 3.7 – Добавление прав доступа для процесса

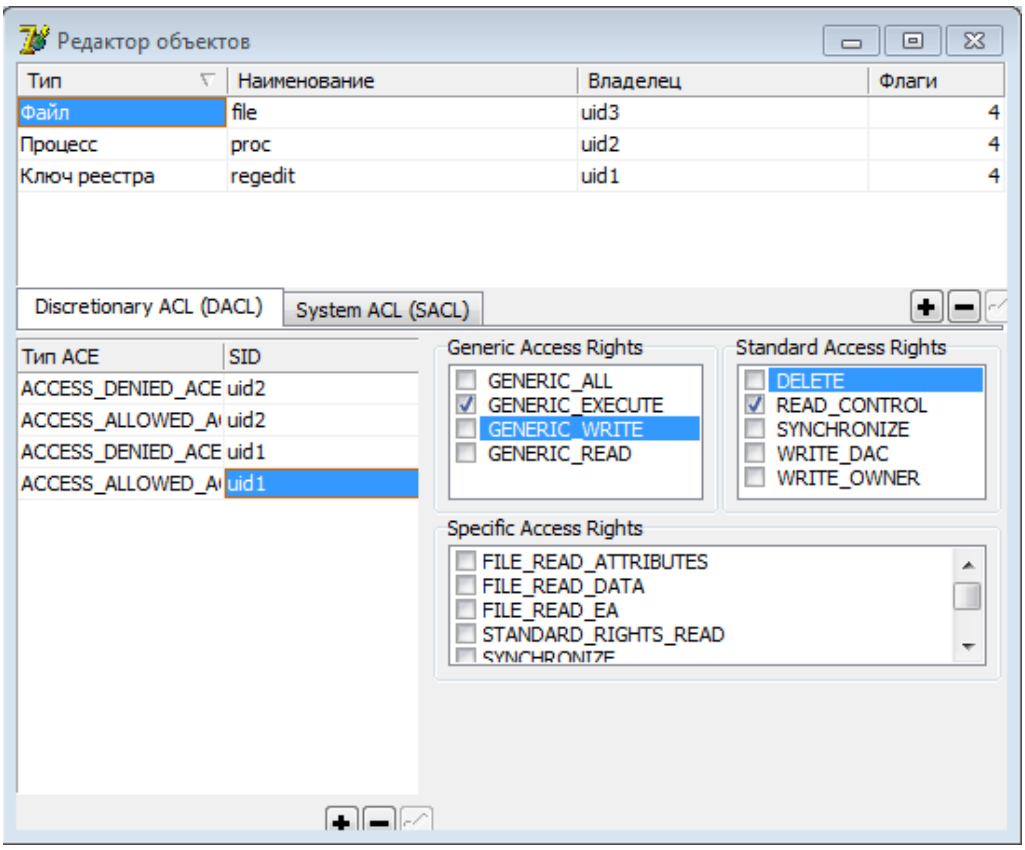

Рисунок 3.8 – Добавление прав доступа для файла

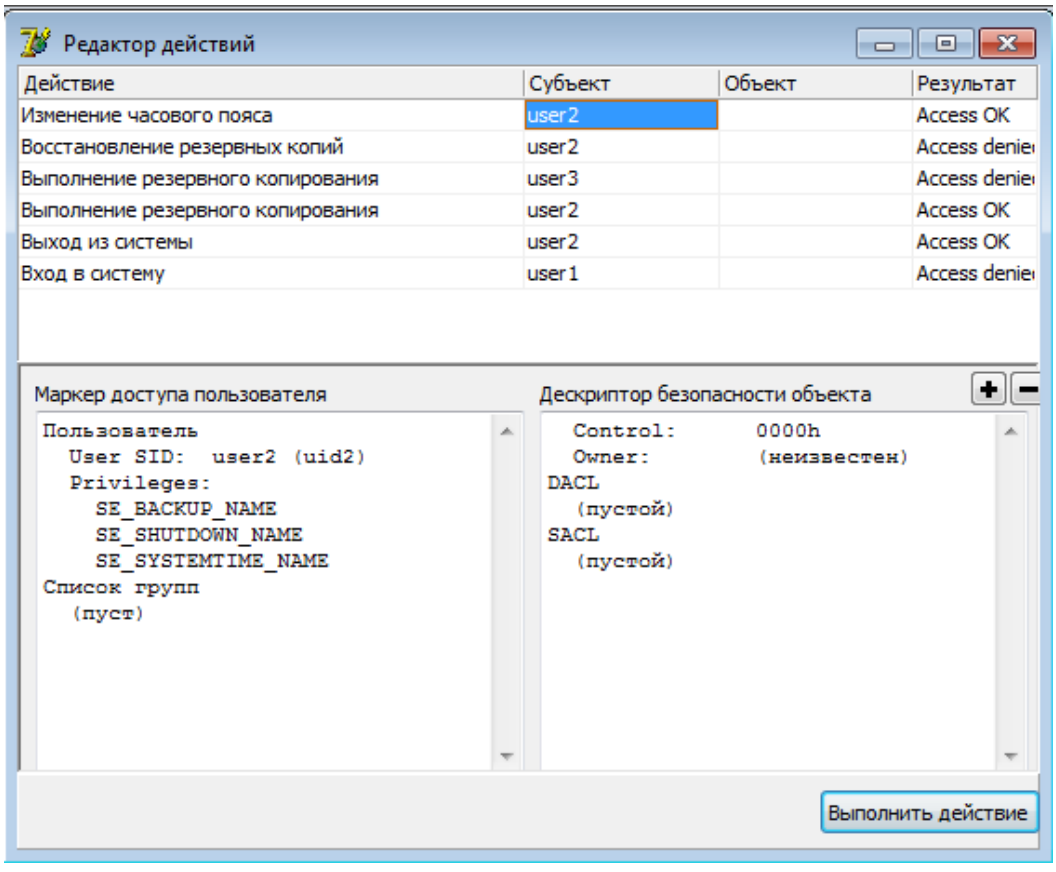

Рисунок 3.9 – Редактор действий