МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный технологический университет «СТАНКИН»

ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ

ЛАБОРАТОРНАЯ РАБОТА 3.

ИДЕНТИФИКАЦИЯ И АУТЕНТИФИКАЦИЯ, АВТОРИЗАЦИЯ ОБЪЕКТОВ, АДМИНИСТРИРОВАНИЕ СУБЪЕКТОВ НА БАЗЕ РОЛЕВОГО УПРАВЛЕНИЯ ДОСТУПА В ИНФОРМАЦИОННЫХ СИСТЕМАХ.

Перед началом работы необходимо ознакомится с материалами лекции «Информационные системы и технологии», посвященной Идентификации, Аутентификации, Авторизации и Управлению доступом.

Часть 1. Изучение механизмов администрирования для легализации доступа в информационные системы на основе идентификации и аутентификации.

Основой любых способов управления легальным доступом в приложения ИС являются идентификация и аутентификация.

Все эти механизмы рассчитаны на работу с поименованными ресурсами системы, а именно, с субъектами и объектами ИС.

В качестве *субъектов* ИС могут выступать как пользователи, так и процессы, а в качестве *объектов* ИС – данные и другие информационные ресурсы системы.

Присвоение субъектам и объектам доступа личного идентификатора и сравнение его с заданным перечнем называется **идентификацией**.

Идентификация обеспечивает выполнение следующих функций:

- дальнейшее установление подлинности и определение полномочий субъекта при его допуске в систему;
- контролирование установленных полномочий в процессе сеанса работы;
- регистрация действий и др.

Аутентификацией (установлением подлинности) называется проверка принадлежности субъекту доступа предъявленного им идентификатора и подтверждение его подлинности.

Другими словами, аутентификация заключается в проверке: является ли подключающийся субъект тем, за кого он себя выдает.

Если в процессе аутентификации подлинность субъекта установлена, то система защиты информации должна определить его **полномочия**, т.е. перечень допустимых для него действий. Это необходимо для последующего контроля и разграничения доступа к интерфейсу системы.

По контролируемому компоненту системы способы аутентификации можно разделить на:

- аутентификацию партнеров по общению
- аутентификацию источника данных.

Аутентификация партнеров по общению используется при установлении соединения во время сеанса. Она служит для предотвращения определенных угроз и повтора предыдущего сеанса связи.

Аутентификация источника данных – это подтверждение подлинности источника отдельной порции данных.

По направленности аутентификация может быть односторонней (пользователь доказывает свою подлинность системе, например при входе в систему) и двусторонней (взаимной).

Обычно методы аутентификации классифицируют по используемым средствам.

В этом случае их делят на четыре группы:

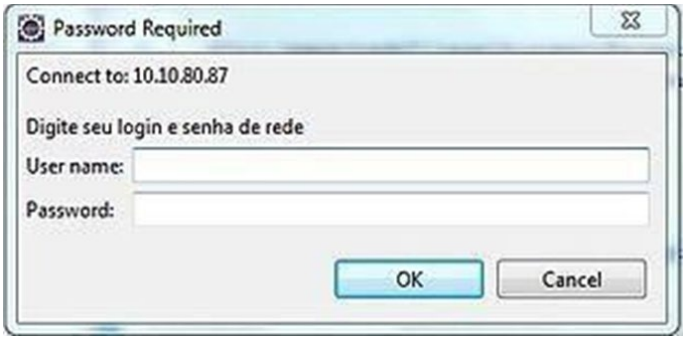
1. Методы, основанные на знании лицом, имеющим право на доступ к ресурсам системы, некоторой секретной информации – пароля.
2. Методы, основанные на использовании уникального предмета: жетона, электронной карточки и др.

3. Методы, основанные на измерении биометрических параметров человека – физиологических или поведенческих атрибутах живого организма.
4. Методы, основанные на информации, ассоциированной с пользователем, например, с его координатами.

В этой части будет подробно рассмотрена первая группа методов аутентификации. Они базируются на *паролях* – секретных идентификаторах субъектов.

Так, при вводе субъектом своего логина (имя пользователя) и пароля подсистема аутентификации сравнивает его с внутренним списком связок логин-пароль. Пароли хранятся в зашифрованном виде. Для этого используются различные процедуры хэширования (например, MD5, SHA1, SHA256 и др.).

В случае совпадения логина и пароля подсистема аутентификации разрешает доступ к ресурсам ИС.



Парольные методы следует классифицировать по степени изменяемости паролей:

- методы, использующие **постоянные** (многократно используемые) пароли;
- методы, использующие **одноразовые** (динамично изменяющиеся) пароли.

В большинстве ИС используются *многоразовые пароли*. В этом случае пароль пользователя не изменяется от сеанса к сеансу в течение установленного администратором системы времени его действительности. Это упрощает процедуры администрирования, но повышает угрозу рассекречивания пароля. Известно множество способов вскрытия пароля: от подсмотра через плечо до перехвата сеанса связи. Вероятность вскрытия злоумышленником пароля повышается, если пароль несет смысловую нагрузку (год рождения, номер телефона или машины), небольшой длины, набран на одном регистре, не имеет ограничений на период существования и т. д.

Важно, разрешено ли вводить пароль только в диалоговом режиме илиесть возможность обращаться из программы. В последнем случае, возможно запустить программу по подбору паролей – «дробилку».

Более надежный способ – использование *одноразовых или динамически меняющихся паролей*. Известны следующие методыпарольной защиты, основанные на одноразовых паролях:

- **метод модификации схемы простых паролей** - пользователю выдается список паролей. При аутентификации система запрашивает у пользователя пароль, порядковый номер в списке которого определен по случайному закону. Если пользователь не угадал текущий пароль из известногоему списка, то система предлагает несколько попыток.

- **метод «запрос-ответ»** - при использовании метода «запрос-ответ» система задает пользователю некоторые вопросы общего характера, правильные ответы на которые известны только конкретному пользователю.
- **функциональный метод** - функциональные методы основаны на использовании специальной функции парольного преобразования ***f(x).***

Это позволяет обеспечить возможность изменения (по некоторой формуле или алгоритму) паролей пользователя во времени. Указанная функция должна удовлетворять следующим требованиям:

- для заданного пароля x легко вычислить новый пароль y=f(x) ;
- зная x и y, сложно или невозможно определить функцию f(x).

Наиболее известными примерами функциональных методов являются:

- метод функционального преобразования
- метод «рукопожатия».

Метод функционального преобразования состоит в периодическом изменении самой функции f(x). Последнее достигается наличием в функциональном выражении динамически меняющихся параметров, например, функции от некоторой даты и времени. Пользователю сообщается исходный пароль, собственно функция и периодичность смены пароля.

Метод «рукопожатия» состоит в следующем:

Функция парольного преобразования известна только пользователю и системе защиты. При входе в ИС подсистема аутентификации генерирует случайную последовательность *x*, которая передается пользователю. Пользователь вычисляет результат функции *y=f(x)* и возвращает его в систему. Система сравнивает собственный вычисленный результат с полученным от пользователя. При совпадении указанных результатов подлинность пользователя считается доказанной. Достоинством метода является то, что передача какой-либо информации, которой может воспользоваться злоумышленник, здесь сведена к минимуму.

В ряде случаев пользователю может оказаться необходимым проверить подлинность другого удаленного пользователя или некоторой ИС, к которой он собирается осуществить доступ.

Наиболее подходящим здесь является метод «рукопожатия», так как никто из участников информационного обмена не получит никакой конфиденциальной информации.

Отметим, что методы аутентификации, основанные на одноразовых паролях, также не обеспечивают абсолютной защиты. Например, если злоумышленник имеет возможность подключения к сети и перехватывать передаваемые пакеты, то он может посылать последние как собственные.

Часть 2. Изучение методов и процедур авторизации объектов в информационных системах.

После успешной регистрации, а затем идентификации и аутентификации пользователя, информационная система должна осуществить авторизацию - предоставление субъекту прав на доступ к объекту.

Средства авторизации контролируют доступ легальных пользователей к ресурсам системы, предоставляя каждому из них именно те права, которые были определены администратором, а также осуществляют контроль возможности выполнения пользователем различных системных функций.

Информационная система включает набор субъектов (процессы, пользователи) и объектов. Под объектами понимается как ресурсы оборудования (процессор, сегменты памяти, принтер, диски, флэш-память), так и программные (файлы, программы).

Каждый объект имеет уникальное имя, отличающее его от других объектов в системе, и каждый из них может быть доступен через хорошо определенные и значимые операции.

Объекты - абстрактные типы данных. Операции зависят от объектов. Например, процессор может только выполнять команды. Сегменты памяти, а также флэш-память могут быть записаны и прочитаны. Файлы данных могут быть записаны, прочитаны, переименованы и т.д.

Очевидно, что процессу может быть разрешен доступ только к тем ресурсам, к которым он имеет авторизованный доступ.

Желательно добиться того, чтобы он имел доступ только к тем ресурсам, которые ему нужны для выполнения его задачи. Это требование имеет отношение только к принципу *минимизации* привилегий, полезному с точки зрения ограничения количества повреждений, которые процесс может нанести системе.

Различают *дискреционный* (избирательный) способ управления доступом и *полномочный* (мандатный). При дискреционном доступе определенные операции над определенным ресурсом запрещаются или разрешаются субъектам или группам субъектов. С концептуальной точки зрения текущее состояние прав доступа при дискреционном управлении описывается матрицей, в строках которой перечислены субъекты, а в столбцах - объекты.

Полномочный подход заключается в том, что вся информация делится на уровни в зависимости от степени секретности, а все пользователи также делятся на группы, образующие иерархию в соответствии с уровнем допуска к этой информации.

2.1 Дискреционный способ управления доступом

Большинство информационных систем реализуют именно дискреционное управление доступом. Главное его достоинство - гибкость, основные недостатки - рассредоточенность управления и сложность централизованного контроля, а также оторванность прав доступа от данных, что позволяет копировать секретную информацию в общедоступные файлы.

2.1.1 Домены безопасности

Чтобы подробно изучить эту схему рассмотрим концепцию домена безопасности (protection domain). Процесс оперирует с доменом безопасности,

который специфицирует ресурсы, к которым процесс может иметь доступ (см рис.) Каждый домен определяет набор объектов и типов операций, которые могут быть осуществлены над каждым объектом. Возможность выполнять операции над объектом есть права доступа. Домен есть набор прав доступа, каждое из которых есть упорядоченная пара <object-name, rights-set>. Например, если домен D имеет права доступа <file F, {read, write}>, это означает, что процесс, выполняемый в домене D, может читать или писать в файл F, но не может выполнять других операций над этим объектом.

Объект / Домен	F1	F2	F3	Printer
D1	read			
D2				print
D3		read	execute	
D4	read write		read write	

Рис. Специфицирование прав доступа к ресурсам

Связь процессов с доменами может быть статической и динамической. Организация динамической связи сложнее. Заметим, что домен может быть реализован различными способами:

- каждый *пользователь* может быть доменом. В этом случае набор объектов, к которым может быть организован доступ, зависит от идентификации пользователя. Переключение между доменами имеет место, когда меняется пользователь (один входит в систему, другой выходит из нее);
- каждый *процесс* может быть доменом. В этом случае набор доступных объектов определяется идентификацией процесса.

Переключение между доменами происходит, когда один из процессов посылает сообщение другому и ждет отклика;

- каждая *процедура* может быть доменом. В этом случае набор доступных объектов соответствует локальным переменным, определенным внутри процедуры. Переключение между доменами происходит, когда процедура выполнена.

2.1.2 Матрица доступа

Модель безопасности, специфицированная в предыдущем подразделе, имеет вид матрицы, которая называется матрицей доступа. Какова может быть эффективная реализация матрицы доступа? В общем случае она будет разреженной, то есть большинство ее клеток будут пустыми. Хотя существуют структуры данных для представления разреженной матрицы, они не слишком полезны для приложений, использующих возможности защиты. Поэтому на практике матрица доступа применяется редко. Эту матрицу можно разложить по столбцам, в результате чего получаются списки прав доступа (Access control list - ACL). В результате разложения по строкам получаются мандаты возможностей (Capability list или Capability ticket).

2.1.3 Список прав доступа. Access control list

Каждая колонка в матрице может быть реализована как список доступа для

одного объекта. Очевидно, что пустые клетки могут не учитываться. В результате для каждого объекта имеем список упорядоченных пар <domain, rights-set>, который определяет все домены с непустыми наборами прав для данного объекта.

2.1.4 Мандаты возможностей. Capability list

Если матрицу доступа хранить по строкам, то есть каждый субъект хранит список объектов и для каждого объекта список допустимых операций, то такой способ хранения называется Capability list.

Иногда применяется комбинированный способ. Например, в операционной системе Unix на этапе открытия файла происходит анализ ACL. В случае благоприятного исхода (у процесса были соответствующие права) файл заносится в список открытых файлов, и при последующих операциях чтения и записи проверки прав доступа не происходит. Список открытых файлов можно рассматривать как Capability list.

2.1.5 Механизм Lock-Key

Механизм Lock-Key представляет компромисс между Access list и Capability list. Каждый объект имеет список уникальных битовых шаблонов patterns, называемых lock. Аналогично, каждый домен имеет список уникальных битовых шаблонов, называемых ключами key. Процесс, выполняющийся в домене, может иметь доступ к объекту, только если домен имеет key, который соответствует одному из lock объекта.

Как и в случае Capability list, список ключей для домена должен управляться администратором ИС. Пользователям не разрешено проверять или модифицировать списки key и lock непосредственно.

2.2 Полномочный (мандатный) способ управления доступом

Полномочный подход заключается в следующем :

- все субъекты и объекты информационной системы должны быть однозначно идентифицированы;
- каждому объекту системы присвоена метка критичности, определяющая ценность (секретность) содержащейся в нем информации;
- каждому субъекту системы присвоен уровень прозрачности, определяющий максимальное значение метки критичности объектов, к которым субъект имеет доступ.

Когда совокупность меток имеет одинаковые значения, говорят, что они принадлежат к одному уровню безопасности. Организация меток имеет иерархическую структуру и, таким образом, в системе можно реализовать иерархически восходящий поток информации (например, от рядовых исполнителей к руководству). Чем важнее объект или субъект, тем выше его метка критичности. Поэтому наиболее защищенными оказываются объекты с наиболее высокими значениями метки критичности.

Каждый субъект, кроме уровня прозрачности, имеет текущее значение уровня безопасности, которое может изменяться от некоторого минимального значения до значения его уровня прозрачности. Основное назначение полномочного (мандатного) способа управления доступом -- регулирование доступа субъектов системы к объектам с различным уровнем критичности (секретности) и предотвращение утечки информации с верхних уровней должностной иерархии в нижние, а также блокирование возможного проникновения с нижних уровней в

верхние. Отсюда должны неукоснительно выполняться следующие правила:

- субъект может читать информацию только из объекта, уровень критичности (секретности) которого не выше уровня прозрачности субъекта. (например: топ-менеджер читает документы менеджера, а не наоборот);
- субъект может записывать информацию в объекты только своего уровня или более высоких уровней критичности (секретности) т.е. топ-менеджер не может случайно разгласить подчиненному персоналу конфиденциальную информацию.

Часть 3. Изучение методов и процедур администрирования субъектов в информационных системах.

При большом количестве пользователей (субъектов) вышеописанные способы управления доступом к ресурсам ИС становятся крайне сложными для администрирования. Число связей в них пропорционально произведению количества пользователей на количество объектов. Необходимы решения на более высоком, объектно-ориентированном уровне, которые способны эти сложности обойти.

Таким решением является **ролевое управление доступом.**

Суть его в том, что между пользователями и их привилегиями появляются промежуточные сущности - роли. Для каждого пользователя одновременно могут быть активными несколько ролей, каждая из которых дает ему определенные права

Рис. Пользователи, объекты и роли.

Ролевой доступ нейтрален по отношению к конкретным видам прав и способам их проверки; его можно рассматривать как объектно- ориентированный каркас, облегчающий администрирование, поскольку он позволяет сделать подсистему разграничения доступа управляемой при сколь угодно большом числе пользователей, прежде всего за счет установления между ролями связей, аналогичных наследованию в объектно-ориентированных системах. Кроме того, ролей должно быть значительно меньше, чем пользователей. В результате число администрируемых связей становится пропорциональным сумме (а не произведению) количества пользователей и объектов.

Ролевое управление доступом оперирует следующими основными понятиями:

- пользователь (человек, процесс и т.д.);
- сеанс работы пользователя;
- роль (обычно определяется в соответствии с организационной структурой);
- объект (сущность, доступ к которой разграничивается; например,файл, реляционная таблица или экран интерфейса);

- операция (зависит от объекта; для файлов - чтение, запись, выполнение и т.д.; для таблиц - вставка, удаление и т.д., для экранов интерфейса – наличие (отсутствие) различных функциональных возможностей;
- право доступа (разрешение выполнять определенные операции над определенными объектами).

Ролям приписываются пользователи и права доступа, т.е. можно считать, что роли описывают отношения "многие ко многим" между пользователями и правами. Роли могут быть приписаны многим пользователям; один пользователь может быть приписан нескольким ролям. Во время сеанса работы пользователя активизируется подмножество ролей, которым он приписан, в результате чего он становится обладателем объединения прав, приписанных активным ролям. Одновременно пользователь может открыть несколько сеансов.

Между ролями может быть установлено отношение определенного порядка, называемое наследованием. Если роль r2 является наследницей r1, то все права r1 приписываются r2, а все пользователи r2 приписываются r1. Отношение наследования является иерархическим, при этом права доступа и пользователи распределяются по уровням иерархии. В общем случае наследование является множественным, то есть у одной роли может быть несколько предшественниц и, естественно, несколько наследниц.

Можно представить себе формирование иерархии ролей, начиная с минимума прав (и максимума пользователей), приписываемых роли "сотрудник", с постепенным уточнением состава пользователей и добавлением прав (роли "системный администратор", "бухгалтер" и т.д.), вплоть до роли "руководитель".

Для ролевого управления доступом также вводится понятие разделения обязанностей, причем в двух видах: **статическом** и **динамическом**.

Статическое разделение обязанностей налагает ограничения на приписывание пользователей ролям.

В простейшем случае членство в некоторой роли запрещает приписывание пользователя определенному множеству других ролей.

В общем случае данное ограничение задается как пара "множество ролей - число" (где множество состоит, по крайней мере, из двух ролей, а число должно быть больше 1). Например, может существовать пять бухгалтерских ролей, но политика безопасности допускает членство не более чем в двух таких разных ролях (здесь число=1+2).

Динамическое разделение обязанностей отличается от статического только тем, что рассматриваются роли, одновременно активные (быть может, в разных сеансах) для данного пользователя (а не те, которым пользователь статически приписан). Например, один пользователь может играть роль и кассира, и

контролера, но не одновременно; чтобы стать контролером, он должен сначала закрыть кассу.

Рассматриваемый подход может содержать три категории функций, необходимых для администрирования ролевым управлением доступом:

- **административные функции** (создание и сопровождение ролей и других атрибутов ролевого доступа): создать/удалить роль/пользователя, приписать пользователя/право роли или ликвидировать существующую ассоциацию, создать/удалить отношение наследования между существующими ролями, создать новую роль и сделать ее наследницей/предшественницей существующей роли, создать/удалить ограничения для статического/динамического разделения обязанностей.
- **вспомогательные функции** (обслуживание сеансов работы пользователей): открыть сеанс работы пользователя с активацией подразумеваемого набора ролей; активировать новую роль, деактивировать роль; проверить правомерность доступа.
- **информационные функции** (получение сведений о текущей конфигурации с учетом отношения наследования). Здесь проводится разделение на обязательные и необязательные функции. К числу первых принадлежат получение списка пользователей, приписанных роли, и списка ролей, которым приписан пользователь. Все остальные функции отнесены к разряду необязательных. Это получение информации о правах, приписанных роли, о правах заданного пользователя (которыми он обладает как член множества ролей), об активных, в данный момент сеанса, ролях и правах, об операциях, которые роль/пользователь правомочны совершить над заданным объектом, о статическом/динамическом разделении обязанностей.

АЛГОРИТМ ВЫПОЛНЕНИЯ ЗАДАНИЯ

1. Лабораторная работа №3 выполняется командой, разрабатывается единое приложение, члены команды самостоятельно распределяют работы, результаты выполнения лабораторной работы представляются преподавателю командой, каждый член команды должен быть готов ответить на вопросы и сообщить о индивидуальном вкладе в командный проект.

ЧАСТЬ 1.

2. Для выполнения Части 1 лабораторной работы №3 необходимо выбрать оптимальный для командного проекта вариант идентификации пользователей и сообщить преподавателю об обоснованном выборе:
    - Разработать и отладить процедуру, выполняющую следующие действия: идентификацию пользователя, аутентификацию пользователя на базе метода, использующего постоянные, многократно используемые пароли.
    - Разработать и отладить процедуру, выполняющую следующие действия: идентификацию пользователя, аутентификацию пользователя, использующую одноразовые, динамично изменяющиеся пароли на базе схемы простых паролей.
    - Разработать и отладить процедуру, выполняющую следующие действия: идентификацию пользователя, аутентификацию пользователя, использующую одноразовые, динамично изменяющиеся пароли на базе метода «запрос-ответ».
    - Разработать и отладить процедуру, выполняющую следующие действия: идентификацию пользователя, аутентификацию пользователя, использующую одноразовые, динамично изменяющиеся пароли на базе метода функционального преобразования.
    - Разработать и отладить процедуру, выполняющую следующие действия: идентификацию пользователя, аутентификацию пользователя, использующую одноразовые, динамично изменяющиеся пароли на базе функционального метода «рукопожатия».
3. Разработать приложение в строгом соответствии с типами пользователей, описанными командой в диаграммах IDEF0 и DFD.

ЧАСТЬ 2.

4. Для выполнения Части 2 лабораторной работы №3 необходимо выбрать оптимальный для командного проекта вариант авторизации пользователей и сообщить преподавателю об обоснованном выборе:
    - Разработать и отладить процедуру, реализующую дискреционный способ авторизации с использованием матрицы доступа.
    - Разработать и отладить процедуру, реализующую дискреционный способ авторизации с использованием списка прав доступа Access control list.
    - Разработать и отладить процедуру, реализующую дискреционный способ авторизации с использованием мандатов возможностей Capability list.
    - Разработать и отладить процедуру, реализующую дискреционный способ авторизации с использованием механизма Lock-Key.
    - Разработать и отладить процедуру, реализующую полномочный

(мандатный) способ авторизации.

5. Разработать приложение в строгом соответствии с типами пользователей, описанными командой в диаграммах IDEF0 и DFD.

ЧАСТЬ 3.

6. Для выполнения Части 3 лабораторной работы №3 необходимо выбрать оптимальный для командного проекта вариант ролевого управления пользователей и сообщить преподавателю об обоснованном выборе:
    - Разработать и отладить локальное приложение, реализующее ролевое управление интерфейсом приложения.
    - Разработать и отладить локальное приложение, реализующее ролевое управление со статическим разделением обязанностей (один пользователь – одна роль).
    - Разработать и отладить локальное приложение, реализующее ролевое управление со статическим разделением обязанностей (один пользователь – заданное число ролей).
    - Разработать и отладить локальное приложение, реализующее ролевое управление с сеансовым динамическим разделением обязанностей.
    - Разработать и отладить локальное приложение, реализующее иерархическое ролевое управление с наследованием прав.
7. Сформировать отчет в формате Microsoft Word, состоящий из скриншотов моделей по разделам с титульной страницей и названием файла в формате «Отчет_Фамилия_Л3.docx»:
8. Ответом на задание прикрепить в Электронную образовательную среду Станкина файл: Отчет_Фамилия_Л3.docx