22. Модель матрицы доступов Харрисона-Рузо-Ульмана. Модель системы безопасности Белла-ЛаПуды

Рассматривая вопросы защиты информации в КС, мы уже использовали ранее понятие политики безопасности. Напомним, что под политикой безопасности понимается совокупность норм и правил, регламентирующих процесс обработки информации, выполнение которых обеспечивает защиту от определенного множества угроз и составляет необходимое (а иногда и достаточное) условие безопасности системы.

Для строгого и однозначного толкования норм и правил политики безопасности обычно дается ее формализованное описание в виде соответствующей модели. Основная цель такого описания – это определение условий, которым должно подчиняться поведение системы, выработка критерия безопасности и проведение формального доказательства соответствия системы этому критерию при соблюдении установленных правил и ограничений. На практике это означает, что только соответствующим образом уполномоченные пользователи получат доступ к информации и смогут осуществить с ней только санкционированные действия.

Все существующие в настоящее время модели безопасности основаны на следующих базовых представлениях:

1. Компьютерная система является совокупностью взаимодействующих сущностей – субъектов и объектов. Объекты можно интуитивно представлять в виде контейнеров, содержащих информацию, а субъектами считать выполняющиеся программы, которые воздействуют на объекты различными способами. При таком представлении безопасность обработки информации обеспечивается путем решения задачи управления доступом субъектов к объектам в соответствии с тем набором правил и ограничений, которые образуют политику безопасности. Считается, что система безопасна, если субъекты не имеют возможности нарушать правила политики безопасности. Таким образом, общим кодом для всех моделей является именно разделение множества сущностей, образующих систему, на множества субъектов и объектов.
2. Все взаимодействия в системе моделируются установлением отношений определенного типа между субъектами и объектами. Множество типов таких отношений определяется в виде набора операций, которые субъекты могут производить над объектами.
3. Все операции между субъектами и объектами, контролируемые монитором взаимодействий, либо запрещаются, либо разрешаются в соответствии с правилами политики безопасности.
4. Политика безопасности задается в виде правил, определяющих все взаимодействия между субъектами и объектами. Взаимодействия, приводящие к нарушению этих правил, пресекаются средствами контроля доступа и не могут быть осуществлены.
5. Совокупность множеств субъектов, объектов и отношений между ними (установившихся взаимодействий) определяет состояние системы. В этом пространстве состояний каждое состояние системы является либо безопасным, либо небезопасным в соответствии с принятым в модели критерием безопасности.
6. Основной элемент модели безопасности – это доказательство того, что система, находящаяся в безопасном состоянии, не может перейти в небезопасное состояние при соблюдении всех установленных правил и ограничений.

Среди моделей политики безопасности можно выделить три основных типа:

- дискреционные (произвольные)
- мандатные (нормативные)
- ролевые

В основе этих моделей лежат, соответственно, дискреционное управление доступом (Discretionary Access Control – DAC), мандатное управление доступом (Mandatory Access Control – MAC) и ролевое управление доступом (Role-Based Access Control – RAC) .

Управление доступом

Под управлением доступом (УД) будем понимать ограничение возможностей использования ресурсов системы пользователями (процессами) в соответствии с правилами разграничения доступа.

Существует четыре основных способа разграничения доступа субъектов к совместно используемым объектам:

- физический – субъекты обращаются к физически различным объектам (однотипным устройствам, наборам данных на разных носителях и т. д.);
- временной – субъекты получают доступ к объекту в различные промежутки времени;
- логический – субъекты получают доступ к объектам в рамках единой операционной среды, но под контролем средств разграничения доступа;
- криптографический – все объекты хранятся в зашифрованном виде, права доступа определяются знания ключа для расшифрования объекта.

На практике основными способами разграничения доступа являются логический и криптографический. Рассмотрим логическое УД, которое может быть реализовано в соответствии с одной из трех формальных моделей УД (безопасности). В табл. 4.1 приведены модели и профили защиты УД.

Таблица 4.1 Модели и профили защиты УД

Модель УД	Профиль защиты	Класс безопасности по «Оранжевой книге»	Класс защищенности по классификации Гостехкомиссии РФ
Дискреционная	Контролируемый доступ	C2	5
Мандатная	Меточная защита	B1	4
Ролевая	Универсальная надстройка, применяемая с дискреционным и мандатным УД		

Модели УД

Дискреционное (произвольное, матричное, разграничительное) УД

Дискреционное УД (Discretionary Access Control – DAC) – разграничение доступа между поименованными субъектами и поименованными объектами. Оно основанное на задании владельцем объекта или другим полномочным лицом прав доступа других субъектов (пользователей) к этому объекту.

В рамках этой модели система обработки информации **представляется в виде** совокупности активных сущностей – **субъекто**в (множество S – пользователи, процессы и т. д.), которые осуществляют доступ к информации, пассивных сущностей – объектов (множество O – файлы, каталоги, процессы и т. д.), **содержащих защищаемую информацию, и конечного множества прав доступа** $R = \{r_1, ..., r_n\}$, **означающих полномочия на выполнение соответствующих действий** (например, чтение (Read – R), запись (Write – W), выполнение (Execute – E), удаление (Delete – D), владение (Ownership – O) и т. д.).

Каждый объект объявляется собственностью соответствующего субъекта (владельца). При чем в конкретный момент времени у объекта может быть только один владелец, но с течением времени они могут меняться. Владелец имеет все права доступа к объекту, и он определяет права доступа других субъектов к этому объекту.

Поведение системы моделируется с помощью понятия состояния. Пространство состояний системы образуется декартовым произведением множеств составляющих ее объектов, субъектов и прав – OxSxR. Текущее состояние системы Q в этом пространстве определяется тройкой, состоящей из множества субъектов, множества объектов и матрицы прав доступа M,

описывающей текущие права доступа субъектов к объектам, – Q=(S,O,M). Матрица доступ представлена табл. 4.2.

Таблица 4.2 Матрица доступа

		O_1	O_2	O_3	O_4	O_5
M =	S_1		R			
	S_2		RW	R		
	S_3					RW
	S_4	Own	Own	Own	Own	Own

Строки матрицы соответствуют субъектам, а столбцы — объектам, поскольку множество объектов включает в себя множество субъектов, матрица имеет вид прямоугольника. Любая ячейка матрицы M[s, o] содержит набор прав субъекта s к объекту o, принадлежащих множеству прав доступа R**. Поведение системы во времени моделируется переходами между различными состояниями.** Переход осуществляется путем внесения изменений в матрицу M с помощью команд a (x_1, …, x_k) , состоящих из элементарных операций (x_1, …, x_k – параметры команды).

В классической модели допустимы следующие элементарные операции: создание нового субъекта или объекта, удаление существующего субъекта или объекта, добавление субъекту или удаление у субъекта права для объекта.

Применение любой элементарной операции к системе, находящейся в состоянии Q=(S,O,M) влечет за собой переход в другое состояние Q'=(S',O',M'), которое отличается от предыдущего состояния Q по крайней мере одним компонентом.

«Произвольность» этой модели УД состоит в том, что права субъекта по отношению к объекту могут быть изменены в любой момент времени произвольным образом.

Достоинства дискреционного УД:

- гибкость – позволяет независимо управлять правами для любой пары «субъект»–«объект»;
- не требует никаких сложных алгоритмов реализации.

Недостатки дискреционного УД:

- дискреционные модели уязвимы по отношению к атаке с помощью «троянского коня», поскольку в них контролируются только операции доступа субъектов к объектам, а не потоки информации между ними. Поэтому, когда «троянская» программа, которую нарушитель подсунул некоторому пользователю, переносит информацию из доступного этому пользователю объекта в объект, доступный нарушителю, то формально никакое правило дискреционной политики безопасности не нарушается, но утечка информации происходит;
- сложный контроль за распространением прав доступа. Например, владелец файла передает все права или их часть на объект другому пользователю. Тот, в свою очередь, передает их третьему и т. д. В результате передачи прав доступ к объекту может получить некоторый субъект даже в том случае, если исходный владелец был против этого;
- в реальных системах процедуры по обслуживанию и поддержанию в адекватном состоянии матриц доступа оказываются весьма трудоемкими. Работа администратора защиты становится узким местом в работе систем, обладающих динамикой состава пользователей, программ, данных и т. п.

В зависимости от способа представления матрицы прав доступа в ИС различают несколько способов реализации произвольного УД.

Наиболее распространенными для операционных систем являются:

- «парольная» защита;
- списки прав доступа;
- списки полномочий субъектов;
- биты доступа.

«Парольная» защита

Осуществляется следующим образом: пользователь использует отдельный пароль для доступа к каждому объекту в системе. В большинстве реализации парольных систем существуют различные пароли для каждого объекта и для каждого типа доступа. Этот механизм был реализован в операционных системах IBM MVS и NOS.

Использование данного метода доставляет пользователю массу неудобств, т. к. запомнить пароли для каждого объекта и типа доступа невозможно, а хранить их в программах и файлах – ненадежно. Существенную проблему для применения парольной защиты представляет собой и необходимость периодической смены паролей.

Списки управления доступом (Access – ACT Control List)

При реализации произвольного управление доступом с помощью ACL с каждым объектом ассоциируется список пользователей или групп пользователей с указанием их прав доступа к объекту (рис. 4.1). При принятии решения о доступе, соответствующий объекту доступа список проверяется на наличие прав, ассоциированных с индентификатором пользователя, запрашивающего доступ, или его группы.

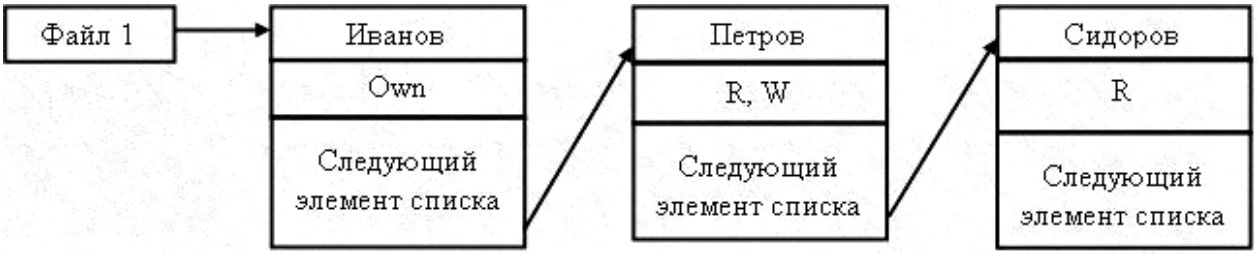


Рис. 4.1. Пример списка контроля доступа к файлам

Данный подход реализован ОС Novell Netware и Windows (точнее в NTFS – New Tchnology File System).

В NTFS 5.0 существует набор из 6 стандартных прав (в терминологии Windows – разрешений):

- полный доступ (включает все права);
- изменение;
- чтение и выполнение;
- список содержимого папки (только для папок);
- чтение;
- запись.

Разрешения перечислены по мере убывания возможностей по работе с объектом (рис. 4.2).

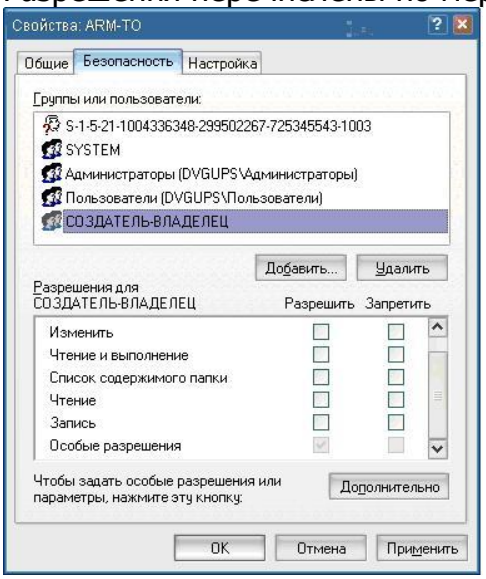


Рис. 4.2. Задание стандартных разрешений

При указании прав для объекта можно сразу выбрать несколько стандартных разрешений.

Каждое из стандартных разрешений в свою очередь представляется комбинацией более мелких, специальных прав, общее число которых равно 13 (рис. 4.3).

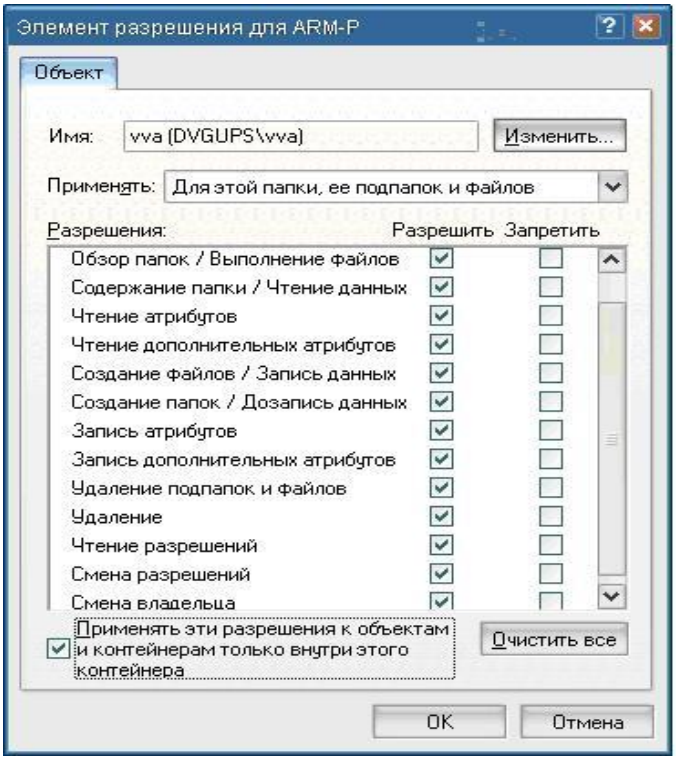


Рис. 4.3. Задание специальных разрешений

Кроме прав чтения, записи и т. п. объекта, они включают в себя также разрешения для работы с его атрибутами и с назначением прав доступа других субъектов по отношению к объекту. Менять стандартные или специальные разрешения может только владелец и субъекты, обладающий специальным правом «Смена разрешений».

Следует отметить, что владельцем объекта является субъект его создавший, и в один и тот же момент времени у объекта может быть только один владелец. В тоже время, группе (учетной записи) «Администраторы» по умолчанию назначается право «Смена владельца». Хотя администратор может вступить в права владения, он не может передавать это право другим пользователям. Такое ограничение необходимо для того, чтобы администратор нес ответственность за свои действия.

Если стандартные комбинации (права) специальных прав, не удовлетворяет потребностям защиты, то можно создать произвольный набор специальных прав. В этом случае, такой набор в списке стандартных разрешений будет называться «Особые разрешения».

Кроме применения ACL для разрешения доступа к файлам и папкам, в Windows они также используются для разрешения доступа к разделам реестра, службам, принтерам, объектам Active Directory и т. п.

Следует отметить, что сама по себе NTFS не шифрует данные на диске, поэтому они могут читаться в обход ОС и, следовательно, в обход установленных прав доступа. Существует драйвера NTFS для MS-DOS и Linux. Таким образом при наличии физического доступа к компьютеру и возможности его перезагрузить в другой ОС, злоумышленник получит доступ ко всем данным на диске безо всякой аутентификации и проверки прав. Избежать такой угрозы безопасности можно за счет применения EFS (Encrypting File System, шифрующей файловой системы), работающей только поверх NTFS 5.0, или за счет применения других криптографических средств.

Основным недостатком применения ACL являются большие временные затраты на обработку списков.

Списки полномочий субъектов (списки возможностей)

В данной модели с каждым субъектом ассоциируется список прав доступа для всех объектов, к которым он имеет доступ.

Пример списка полномочий субъектов приведен на рис. 4.4.

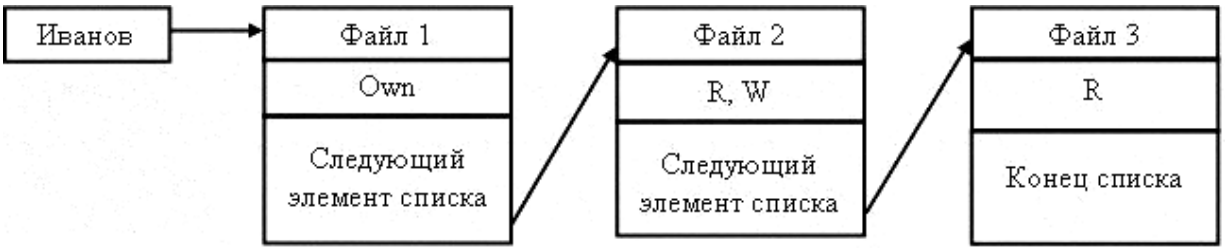


Рис. 4.4. Пример списка полномочий субъектов

Биты защиты

Данный подход реализован в большинстве защищенных ОС, ведущих свое происхождение от UNIX. Вместо списка пользователей, которым разрешен доступ к объекту, с объектом связываются биты защиты.

Владелец			Группа			Все пользователи		
Чтение	Запись	Выполнение	Чтение	Запись	Выполнение	Чтение	Запись	Выполнение
1	2	3	4	5	6	7	8	9

Рис. 4.5. Биты защиты UNIX

В ОС UNIX биты защиты указывают, права доступа для трех категорий пользователей: все пользователи (world), члены группы владельца (group) и владелец объекта (owner). При этом биты защиты может изменять только владелец объекта и администратор.

Недостатком использования механизма битов защиты является неполная реализация произвольного УД, т.к. доступ к объекту нельзя разрешить или запретить для отдельных пользователей. Сложные комбинации прав доступа могут быть установлены путем создания индивидуальных групп для каждого файла и копированием файлов, но это очень сложно организовать для большого количества пользователей.

Мандатное (принудительное) УД

Мандатное УД (Mandatory Access Control – MAC) – разграничение доступа субъектов к объектам, основанное на характеризуемой меткой конфиденциальности информации, содержащейся в объектах, и официальном разрешении (допуске) субъектов обращаться к информации такого уровня конфиденциальности. Оно основано на сопоставлении атрибутов безопасности субъекта (уровня допуска пользователя) и объекта (грифа секретности информации).

Мандатная модель управления доступом основана на правилах секретного документооборота, принятых в государственных и правительственных учреждениях многих стран.

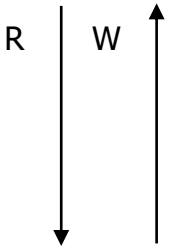
Основным положением данной ПБ, взятым из реальной жизни, является назначение всем участникам процесса обработки защищаемой информации и документам, в которых она содержится, специальной метки, получившей название уровень безопасности (метка безопасности). Метка субъекта описывает его благонадежность, а метка объекта – степень закрытости, содержащейся в нем информации. Уровни секретности, поддерживаемые системой, образуют множество, упорядоченное с помощью отношения доминирования. Такое множество может выглядеть следующим образом: сов. секретно, секретно, конфиденциально, несекретно и т. д.

Система в мандатной модели представляется в виде множеств субъектов S, объектов O, решетки уровней безопасности L и матрицы доступа M.

С помощью решетки уровней безопасности (табл. 4.3) задается соотношение между уровнями безопасности, субъектами и объектами.

Таблица 4.3 Решетка уровней безопасности

Уровень безопасности	Субъекты	Объекты
Совершенно секретно	S_1, S_2	O_5
Секретно	S_3	O_1, O_2
Конфиденциально	S_4	O_4
Несекретно	S_5, S_6	O_3, O_6



В данной модели набор прав ограничен двумя: **read (чтение)** и **write (запись)**. При этом контроль доступа осуществляется в зависимости от уровней безопасности взаимодействующих сторон на основании двух простых правил:

1. **Уполномоченное лицо (субъект) имеет право читать только те документы, уровень безопасности которых не превышает его собственный уровень безопасности.** Данное правило обеспечивает защиту информации, обрабатываемой более доверенными (высокоуровневыми) лицами, от доступа со стороны менее доверенных (низкоуровневых).
2. **Уполномоченное лицо (субъект) имеет право заносить информацию только в те документы, уровень безопасности которых не ниже его собственного уровня безопасности.** Это правило предотвращает утечку информации (сознательную или несознательную) со стороны высокоуровневых участников процесса обработки информации к низкоуровневым.

Матрица доступа для приведенного в табл. 4.4 примера выглядит следующим образом.

Таблица 4.4 Матрица доступа

	O_1	O_2	O_3	O_4	O_5	O_6
S_1	R	R	R	R	RW	R
S_2	R	R	R	R	RW	R
S_3	RW	RW	R	R	W	R
S_4	W	W	R	RW	W	R
S_5	W	W	RW	W	W	RW
S_6	W	W	RW	W	W	RW

Таким образом можно констатировать следующий факт – мандатное УД является частным случаем дискреционного УД, с более строгими правилами разграничения доступа. Использование столь жесткого подхода, не позволяющего осуществлять гибкое управление доступом, объясняется тем, что в мандатной модели контролируются не операции, осуществляемые субъектом над объектом, а потоки информации, которые могут быть только двух видов: либо от субъекта к объекту (запись), либо от объекта к субъекту (чтение). Остальные права моделируются через эти две базовые операции.

Достоинства мандатного УД:

- экономия памяти, так как элементы матрицы доступа не хранятся, а динамически вычисляются при попытке доступа для конкретной пары субъект-объект на основе их меток;
- удобство корректировки базы данных защиты, то есть модификации меток;
- принудительное УД хорошо согласуется с работой государственных, правительственных и военных организаций, так как переносит общепринятые и хорошо отработанные принципы обращения с бумажными секретами на современную основу работы с документами.

Недостатки мандатного УД:

- затруднено задание прав доступа конкретного субъекта к конкретному объекту
- каждый субъект и объект должен быть помечен и при любых операциях с данными метками

В особенности это относится к экспорту и импорту данных. Например, печатный документ должен открываться заголовком, содержащим текстовое и/или графическое представление метки безопасности.

Аналогично, при передаче файла по каналу связи должна передаваться и ассоциированная с ним метка, причем в таком виде, чтобы удаленная система могла ее правильно трактовать, несмотря на возможные различия в уровнях секретности и наборе категорий.

Наиболее известными мандатными моделями являются ММ Белла-Лападула (Дэвид Белл и Леонардо ЛаПадула), ММ Биба, решетчатая модель Д. Деннинга, ММ совместного доступа с уполномоченными субъектами и т. д.

Ролевое УД

Ролевое УД (Role-Based Access Control – RAC) – универсальная надстройка (каркас), применяемая с дискреционным и мандатным УД и предназначенная для упрощения функций администрирования систем с большим количеством субъектов и объектов.

Суть ролевого УД состоит в том, что между пользователями и их правами доступа к объектам появляются промежуточные сущности – роли. Для каждого пользователя одновременно могут быть активными несколько ролей, каждая из которых дает ему определенные права доступа к объекту и, наоборот, несколько пользователей может выступать в одной роли по отношению к одному объекту.

Между ролями могут быть установлены связи, аналогичные отношению наследования в ООП. Таким образом может быть построена иерархия ролей, используя которую можно существенно сократить количество контролируемых (администрируемых) связей. Пример применения ролевого УД приведен на рис. 4.5.

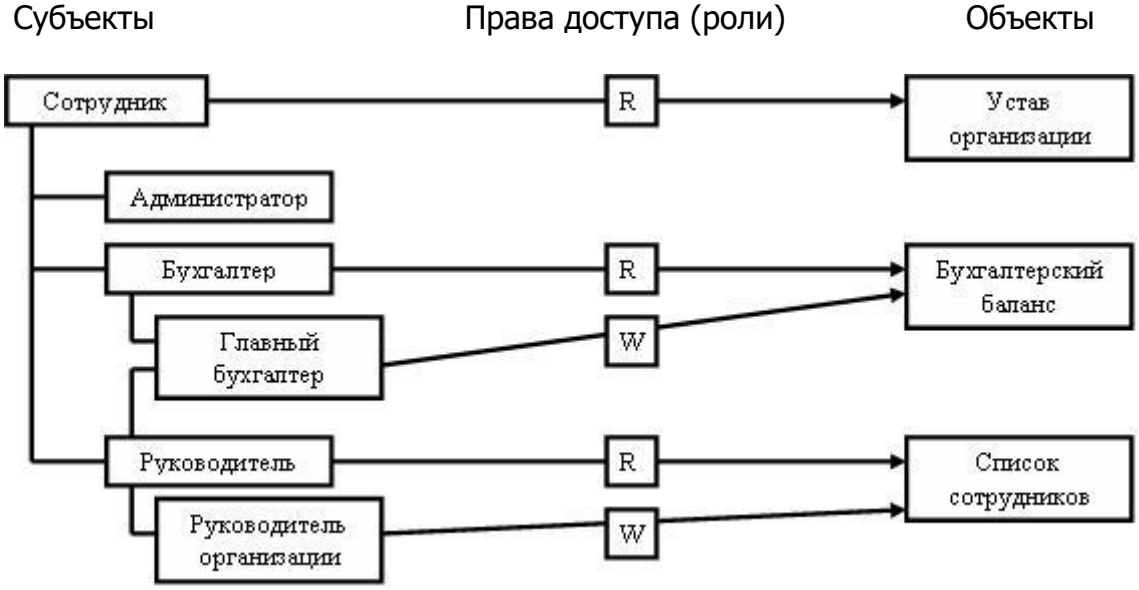


Рис. 4.5. Пример применения ролевого УД

Основополагающими принципами организации ролевого УД являются принципы минимизации привилегий и разделения обязанностей.

Принцип минимизации привилегий гласит о том, что для роли целесообразно назначить только такие права, которые необходимы для выполнения ее служебных обязанностей. В связи с этим, дерево иерархии строят, наращивая (расширяя) права ролей. При построении этой иерархии допускается множественное наследование ролей (например, роль «Главный бухгалтер» наследует права от непосредственных родителей «Бухгалтер» и «Руководитель»).

Разделения обязанностей делится на два вида:

- статическое;
- динамическое.

Статическое разделение обязанностей налагает ограничения на приписывание ролей пользователям. В простейшем случае членство в некоторой роли запрещает приписывание пользователя определенному множеству других ролей. Например, может существовать пять бухгалтерских ролей, но политика УД допускает членство не более чем в двух таких ролях. При наличии наследования ролей ограничение приобретает более сложный вид, но суть остается простой: при проверке членства в ролях нужно учитывать приписывание пользователей ролям-наследницам.

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.

Существуют модель политики безопасности АДЕПТ – 50, формальные модели монитора безопасности объектов: дискреционные модели Харрисона-Руззо-Ульмана и Типизированная матрица доступа, мандатная модель Белла-Лападулы, модель ролевой политики безопасности, модель политики безопасности распределенной компьютерной системы.

В качестве классических примеров моделей этих типов можно назвать дискреционную модель Харрисона-Руззо- Ульмана (модель HRU) и мандатную модель Белла-Лападула (модель БЛ).

В рамках модели Белла-Лападула доказывается важное утверждение, указывающее на принципиальное отличие систем, реализующих мандатную защиту, от систем с дискреционной защитой: если начальное состояние системы безопасно, и все переходы системы из состояния в состояние не нарушают ограничений, сформулированных политикой безопасности, то любое состояние системы безопасно.

Модель политики безопасности адепт – 50

Модель АДЕПТ относится к одной из первых дискреционных моделей политики безопасности МБО.

В модели рассматриваются:

- множество объектов компьютерной системы $O=[o_j]$, $j = 1,\dots n$
- множество субъектов компьютерной системы $S=[s_j]$, $i=1,\dots m$
- множество прав доступа субъектов к объектам $R=[r_g]$, $k = 1, \dots k$
- множество, содержащее перечень предметных областей, к которым относятся выполняемые в компьютерной системе процессы $E=[e_l]$, $l = 1,\dots q$.
- множество, содержащее категории безопасности субъектов и объектов $D=[d_x]$, $x = 1,\dots v$.

Субъекты в компьютерной системе характеризуются:

- предметной областью, к которой они принадлежат
- правами, которыми они наделяются
- категорией безопасности, которая им назначается.

Объекты компьютерной системы характеризуются:

- предметной областью, к которой они принадлежат
- категорией безопасности, которая им назначается.

Критерий безопасности

Субъект наследует категорию безопасности объекта, с которым он связан (ассоциирован).

Описание политики безопасности

Если в компьютерной системе инициализируется поток Stream $(s_j, o_j) \rightarrow o_i$, то

1. субъект s_j должен принадлежать множеству S: $s_j \in S$;
2. объект o_i должен принадлежать множеству O: $o_i \in O$;

3. субъект s_j получает право доступа r_{sj} к объекту o_i, если право доступа субъекта s_j (r_{sj})принадлежит множеству его прав доступа к объекту o_i: $r_{sj} \in R_{oi}$;
4. предметная область субъекта sj принадлежит предметной области объекта oi: $e_{sj} \in e_{oi}$;
5. категория безопасности субъекта sj больше или равна категории безопасности объекта o_i: $d_{oi} \in d_{sj}$.

Дискреционная модель харрисона-руззо-ульмана

Формальная модель Харрисона- Руззо-Ульмана – это классическая дискреционная модель, которая реализует произвольное управление доступом субъектов к объектам и осуществляет контроль за распределением прав доступа.

В модели рассматриваются:

- конечное множество объектов компьютерной системы O=[o_j], 1,…n;
- конечное множество субъектов компьютерной системы S=[s_j], 1,…m.
- конечное множество прав доступа R=[r_g], 1, …k.

Матрица прав доступа, содержащая права доступа субъектов к объектам A=[a_{ij}], I=1, …m, j=1,…n+m, причем элемент матрицы a_{ij} рассматривается как подмножество множества R.

Каждый элемент матрицы a_{ij} содержит права доступа субъекта s_i к объекту o_j.

Конечное множество команд C=[c_z (аргументы)], z=1, l, аргументами команд служат идентификаторы объектов и субъектов.

Каждая команда включает условия выполнения команды и элементарные операции, которые могут быть выполнены над субъектами и объектами компьютерной системы и имеют следующую структуру:

- Command c_z (аргументы)
  - Условия выполнения команды
  - Элементарные операции
- End.

Поведение системы моделируется с помощью понятия состояние. Состояние системы определяется:

- конечным множеством субъектов (S);
- конечным множеством объектов (O), считается, что все субъекты системы одновременно являются и ее объектами ($S \subset O$);
- матрицей прав доступа (A).

Если система находится в состоянии Q, то выполнение элементарной операции переводит ее в некоторое другое состояние Q' ,которое отличается от предыдущего состояния хотя бы одним компонентом.

В модели Харрисона-Руззо-Ульмана определены следующие элементарные операции, при выполнении которых система может перейти из одного состояния в другое.

Добавление субъекту s_i права r_g для объекта o_j. : Enter r_g into a_{ij}

При выполнении этой операции множество субъектов и множество объектов не изменяются. Подмножеству прав доступа добавляется новое право, остальные подмножества не изменяются. Если право уже содержится в подмножестве, то это подмножество также не изменяется.

Операция добавления права называется монотонной, поскольку она только добавляет права в матрицу доступа, но ничего не изменяет.

Удаление у субъекта s_i права r_g для объекта o_j : Delete r_g from a_{ij}

При выполнении этой операции множество субъектов и множество объектов не изменяются. Из подмножества прав доступа удаляется право r_g, остальные подмножества не изменяются. Если удаляемое право не содержится в подмножестве, то это подмножество также не изменяется.

Создание нового субъекта s_i : Create subject s_i

При выполнении этой операции изменяются следующие состояния системы:

- к множеству объектов системы добавляется новый объект;
- к множеству субъектов системы добавляется новый субъект;
- множество прав доступа субъекта s_i к объектам системы становится пустым;
- множество прав доступа всех субъектов системы к объекту o_j становится пустым.

Остальные подмножества матрицы доступа не изменяются.

Удаление существующего субъекта s_i : Destroy subject s_i

При выполнении этой операции изменяются следующие состояния системы

- из множества объектов системы удаляется объект o_j ;
- из множества субъектов системы удаляется субъект s_i ;
- множество прав доступа субъекта s_i к объектам системы становится пустым;
- права доступа всех субъектов системы к объекту o_j становится пустым;

Остальные подмножества матрицы доступа не изменяются.

Создание в системе нового объекта o_j : Create object o_j

При выполнении этой операции изменяются следующие состояния системы:

- к множеству объектов системы добавляется новый объект o_j ;
- множество субъектов системы не изменяется ;
- множество прав доступа всех субъектов к новому объекту системы становится пустым .

Остальные подмножества матрицы доступа не изменяются.

Удаление существующего объекта o_j из системы : Destroy object o_j

При выполнении этой операции изменяются следующие состояния системы:

- из множества объектов системы удаляется объект o_j ;
- множество субъектов системы не изменяется ;
- права доступа всех субъектов системы к объекту o_j становится пустым.

Остальные подмножества матрицы доступа не изменяются.

Описание модели

Поведение системы во времени моделируется последовательностью состояний Q_1, Q_2,…Q_n, в которой каждое последующее состояние является результатом применения команды из множества C к предыдущему состоянию.

Критерий безопасности

Начальное состояние системы считается безопасным относительно права доступа r_g, если не существует применимой к Q_0 последовательности команд, в результате выполнения которых право доступа r_g будет приобретено субъектом s_i для объекта o_j, если это право не принадлежит подмножеству a_{ij} в состоянии Q_0.

Харрисон-Руззо-Ульман доказали, что в общем случае не существует алгоритма, позволяющего решить является ли конфигурация системы, соответствующая ее начальному состоянию $Q_0 = (S_0, O_0, A_0)$ безопасной.

Указанная задача может быть разрешена в одном из следующих случаев:

- команды cz (аргументы), z = 1, 2,… l являются моно операционными, т.е. состоят только из одной операции;
- команды cz (аргументы), z = 1, 2,… l являются одно условными и монотонными, т. е. содержат не более одного условия и не содержат операций Destroy и Delete;
- команды cz (аргументы), z = 1, 2,… l не содержат команды Create.

Вывод: дискреционная модель Харрисона-Руззо-Ульмана в своей общей постановке не дает гарантий безопасности системы, однако именно она послужила основой для целого класса моделей политик безопасности, которые используются для управления доступом и контролем за распространением прав доступа во всех современных системах.

Типизированная матрица доступа

Дискреционная модель Type Access Matrix (TAM) – типизированная матрица доступа – является развитием модели Харрисона- Руззо-Ульмана. Модель TAM дополняет модель Харрисона- Руззо-Ульмана концепцией типов, что позволяет смягчить те условия, для которых возможно доказательство безопасности системы.

В модели TAM рассматриваются:

- конечное множество объектов компьютерной системы O = [o_j], 1,…n;
- конечное множество субъектов компьютерной системы S = [s_i], 1, …m.
- конечное множество прав доступа R = [rg], 1, …k.

Матрица прав доступа, содержащая права доступа субъектов к объектам A = [a_{ij}], I = 1, …m, j = 1,…n+m, причем элемент матрицы a_{ij} рассматривается как подмножество множества R.

Каждый элемент матрицы a_{ij} содержит права доступа субъекта s_i к объекту o_j.

Множество типов, которые могут быть поставлены в соответствие объектам и субъектам системы T = [t_b], b = 1,…w.

Конечное множество команд C=[c_z (аргументы с указанием типов)], z=1,...I, включающих условия выполнения команд и их интерпретацию в терминах элементарных операций. Элементарные операции TAM отличаются от элементарных операций дискреционной модели Харрисона- Руззо-Ульмана использованием типизированных аргументов.

Структура команды

- Command c_z (аргументы и их типы)
    - Условия выполнения команды
    - Элементарные операции
- End

Смысл элементарных операций совпадает со смыслом аналогичных операций, используемых в модели Харрисона-Руззо-Ульмана с точностью до использования типов.

Поведение системы моделируется с помощью понятия состояние. Состояние системы определяется:

- конечным множеством субъектов (S);
- конечным множеством объектов (O), считается, что все субъекты системы одновременно являются и ее объектами (S·O);
- матрицей прав доступа (A)

И далее описывается тройкой Q (S, O, A).

Выполнение элементарных операций переводит систему, находящуюся в состоянии Q в другое состояние Q'.

В модели TAM определены следующие элементарные операции:

- Добавление субъекту s_i права r_g для объекта o_j.
- Enter r_g into a_{ij}
- Удаление у субъекта s_i права r_g для объекта oj.
- Delete r_g from a_{ij}

Выполнение этих элементарных операций приводит к тем же изменениям в состоянии системы, как и их выполнение в модели Харрисона-Руззо-Ульмана.

Типы объектов и субъектов при выполнении этих операций остаются без изменения

Создание нового субъекта s_i с типом t_{sx} : Create subject s_i of type t_{sx}

Выполнение этой элементарной операции приводит к тем же изменениям в состоянии системы, как и ее выполнение в модели Харрисона-Руззо-Ульмана.

Субъекту s_x и объекту o_x ставится в соответствие тип t_{sx} из множества T, типы остальных субъектов и объектов остаются без изменения.

Удаление существующего субъекта s_i с типом t_{sx}: Destroy subject s_x of type t_{sx}

Выполнение этой элементарной операций приводит к тем же изменениям в состоянии системы, как и ее выполнение в модели Харрисона-Руззо-Ульмана.

Типы субъекта s_x и соответствующего ему объекта o_x становятся неопределенными, типы остальных объектов и субъектов остаются без изменения.

Создание в системе нового объекта o_j типом t_{oy}: Create object o_y of type t_{oy}

Выполнение этой элементарной операции приводит к тем же изменениям в состоянии системы, как и их выполнение в модели Харрисона- Руззо-Ульмана.

Объекту o_y ставится в соответствие тип t_{oy} из множества T, типы остальных объектов и субъектов при выполнении этой операций остаются без изменения.

Удаление существующего объекта o_y с типом t_{oy} : Destroy object o_y of type t_{oy}

Выполнение этой элементарной операции приводит к тем же изменениям в состоянии системы, как и их выполнение в модели Харрисона- Руззо-Ульмана.

Тип объекта o_y становится неопределенным, типы остальных объектов и субъектов при выполнении этой операций остаются без изменения.

Таким образом, TAM является обобщением модели Харрисона-Руззо-Ульмана, которую можно рассматривать как частный случай TAM с одним единственным типом, к которому относятся все объекты и субъекты.

Строгий контроль соответствия типов в модели TAM позволяет смягчить требование одноусловности, заменив его ограничением на типы аргументов команд, при выполнении которых происходит создание новых объектов и субъектов.

Для регулирования этого ограничения вводятся понятия родительского и дочернего типов. Тип аргументов команды - C_z ($s_j : t_{sj}$, $o_i : t_{oi}$, $s_x : t_{sx}$, $o_y : t_{oy}$) , считается дочерним, если в этой команде используются элементарные операции вида: Create Subject s_j of type t_{sj} ;

Create Object o_i of type t_{oi} ,

Т. е. элементарные операции создания субъекта или объекта типа, который указан для этих субъекта и объекта в аргументах команды. В противном случае тип аргументов в команде считается родительским. При этом предполагается, что объект или субъект не могут быть созданы, если для них отсутствует родительский тип.

В рассматриваемой команде:

- t_{sj} и t_{oi} считаются дочерними типами;
- t_{sx} и t_{oy} считаются родительскими типами.

Другими словами, для того, чтобы создать объект o_i типа t_{oi} или субъект s_j типа t_{sj} в системе должен быть объект o_y типа t_{oy} или субъект s_x типа t_{sx}.

Следует отметить, что в одной команде тип может быть одновременно и родительски и дочерним. Например,

- Command_1 ($s_1 : t_1$, $s_2 : t_1$)
    - Create Subject s_2 of type t_1
- End.

В этой команде тип t_1 считается родительским относительно субъекта s_1 и дочерним относительно субъекта s_2.

Связи между родительскими и дочерними типами описываются с помощью графа создания, определяющего отношение наследственности.

Граф создания представляет собой направленный граф с множеством вершин T, в котором ребро от t_i к t_j существует тогда и только тогда, когда в системе имеется команда создания субъекта или объекта, в которой t_i является родительским типом, а t_j– дочерним.

Этот граф для каждого типа позволяет определить:

- объекты и субъекты, каких типов должны существовать в системе, чтобы в ней мог появиться объект или субъект заданного типа;
- объекты и субъекты, каких типов могут быть порождены при участии объектов и субъектов заданного типа.

В теории компьютерной безопасности рассматривается модифицированная ТАМ – МТАМ (монотонная типизированная матрица доступа), **в которой отсутствуют немонотонные элементарные операции: Delete, Destroy Subject, Destroy Object.**

Реализация МТАМ называется ациклической, если ее граф создания не содержит циклов, в противном случае реализация МТАМ называется циклической.

Доказано, что критерий безопасности, предложенный Харрисоном-Руззо-Ульманом, разрешим для ациклических реализаций ТАМ, и что требование одно условности команд можно заменить требованием ацикличности графа создания. Смысл этой замены состоит в том, что последовательность состояний системы должна определяться некоторым маршрутом соответственно графу создания, т. к. невозможно появление субъектов и объектов дочерних типов, если в системе отсутствуют родительские типы, которые должны участвовать в их создании.

Отсутствие циклов на графе создания позволяет избежать зацикливания при доказательстве критерия безопасности, т. к. количество путей на графе без циклов является ограниченным. Это означает, что поведение системы становится предсказуемым, поскольку в любом состоянии можно определить субъекты и объекты каких типов могут появиться в системе, а каких – нет.

Однако доказано, что сложность проверки критерия безопасности для МТАМ с ростом в системе количества субъектов и объектов значительно увеличивается. Время на решение этой задачи растет в степенной зависимости от количества субъектов и объектов. Этот недостаток может быть преодолен с помощью тернарной ТАМ, в которой команды могут иметь не более трех аргументов.

Тернарная МТАМ является монотонной версией тернарная ТАМ. Для тернарной МТАМ доказательство безопасности системы значительно упрощается, поскольку запись условий в команде ограничивается небольшим фрагментом матрицы доступа.

Доказано, что при ациклической реализации МТАМ время, затрачиваемое на проверку критерия безопасности, растет полиноминально в зависимости от размерности начальной матрицы доступа.

Вывод: введение строгого контроля типов в дискреционную модель Харрисона-Руззо-Ульмана позволило доказать критерий безопасности компьютерных систем для более приемлемых ограничений, что расширило область ее применения.

Мандатная модель Белла-Лападулы

Модель Белла-Лападулы была предложена в 1975 году. Возможность ее использования в качестве формальной модели отмечена в критерии TCSES ("Оранжевая книга").

Мандатная модель Белла – Лападулы основана на правилах секретного документооборота, которые приняты в государственных и правительственных учреждениях большинства стран. Согласно этим правилам всем участникам процесса обработки критичной информации и документам, в которых она содержится, присваивается специальная метка, которая называется уровнем безопасности.

Мандатное управление доступом подразумевает, что:

- задан линейно упорядоченный набор меток секретности (например, секретно, совершенно секретно и т. д.);
- каждому объекту системы присвоена метка секретности, определяющая ценность содержащейся в нем информации, т. е. его уровень секретности в КС;
- каждому субъекту системы присвоена метка секретности, определяющая уровень доверия к нему в КС или, иначе, его уровень доступа.

Все уровни безопасности упорядочиваются с помощью установленного отношения доминирования. Контроль доступа к документам осуществляется в зависимости от уровней безопасности **на основании двух простых правил:**

1. Уполномоченное лицо имеет право читать только те документы, уровень безопасности которых не превышает его собственный уровень безопасности. Уровень безопасности

уполномоченного лица должен доминировать над уровнем безопасности читаемого им документа.

Это правило обеспечивает защиту информации, обрабатываемую высокоуровневым лицом, от доступа со стороны низкоуровневых лиц.

2. Уполномоченное лицо может записывать информацию только в документы, уровень безопасности которых не ниже его собственного уровня безопасности. Уровень безопасности документа должен доминировать над уровнем безопасности уполномоченного лица.

Это правило предотвращает утечку информации со стороны высокоуровневых участников процесса обработки документов к низкоуровневым.

В формальной модели Белла – Лападулы рассматриваются:

- конечное множество объектов компьютерной системы $O=[o_j]$, 1,…n;
- конечное множество субъектов компьютерной системы $S=[s_i]$, 1, …m, считается, что все субъекты системы одновременно являются и ее объектами ($S \subset O$).;
- права доступа read и write.

Матрица прав доступа, содержащая права доступа субъектов к объектам $A=[a_{ij}]$, I=1, …m, j=1,…n+m.

Множество запросов на выполнение потоков типа read или write $R=[r_g]$, 1, …k.

Функция уровня безопасности F, которая ставит в соответствие каждому объекту и субъекту системы определенный уровень безопасности, принадлежащий множеству уровней безопасности D, на котором определена решетка.

Функция перехода T, которая при выполнении запроса на запись или чтение, переводит систему из состояния Q в состояние Q'.

Состояние системы характеризуется состоянием матрицы A, содержащей права доступа, и функцией уровня безопасности F. Множество состояний системы представляется в виде упорядоченных пар (F, A).

Начальное состояние системы обозначается как Q_0. Выполнение запроса r_0 из множества R переводит систему в состояние Q_1 с помощью функции перехода T. Система, находящаяся в состоянии Q_1, при получении следующего запроса r_1 из множества R переходит в следующее состояние Q_2 и т. д.

Состояние Q_k достижимо тогда и только тогда, когда существует последовательность:
(Q_0, r_0), (Q_1, r_1), ... (Q_{k-1}, r_{k-1}) такая, что T (Q_{k-1}, r_{k-1}) = Q_k.

Считается, что для любой системы состояние Q_0 тривиально достижимо.

Следует отметить, что в мандатных моделях контролируются не операции, которые выполняются субъектами над объектами, а потоки информации. **Потоки типа:**

- Stream(s_i, o_i) $\rightarrow$ o_j (поток типа чтение),
- Stream(s_i, o_j) $\rightarrow$ o_i (поток типа запись).

Решетка уровней безопасности

Решетка уровней безопасности – это формальная алгебра, использование которой позволяет упорядочить потоки информации в компьютерной системе.

Решетка уровней безопасности представлена:

- множеством уровней безопасности D;
- оператором отношения $\leq$;
- оператором, позволяющим определить наименьшую верхнюю границу для пары уровней безопасности;
- оператором, позволяющим определить наибольшую нижнюю границу для пары уровней безопасности.

Смысл этих операций заключается в том, что для каждой пары элементов множества D всегда можно указать единственный элемент, ограничивающий ее сверху или снизу таким образом, что между ними и этим элементом не будет других элементов.

Множество уровней безопасности может быть представлено как целыми числам, так и более сложными составными элементами. Например, в государственных организациях в

качестве уровней безопасности используются комбинации, состоящие из уровня безопасности, представленного целым числом, и набора категорий из некоторого множества (1. секретно, 1. совершенно секретно и т.п.)

Пусть имеется множество субъектов S и множество объектов O. В модели Белла-Лападулы каждому субъекту и объекту системы из множества D функцией уровня безопасности F назначается соответствующий уровень безопасности.

Естественно, что некоторые субъекты и объекты могут иметь один и тот же уровень безопасности. Подмножества, в которых субъекты и объекты имеют одинаковый уровень безопасности, называются классами. Пусть имеются два класса X и Y. Рассмотрим применение основных положений теории решеток применительно к этим классам.

В теории решеток рассматривается отношение порядка $\leq$, использование этого отношения в теории компьютерной безопасности позволяет установить направление потоков информации.

Отношение порядка обладает следующими свойствами:

- рефлексивности;
- антисимметричности;
- транзитивности.

Вывод: мандатная модель управляет доступом неявным образом – с помощью назначения всем сущностям системы уровней безопасности, которые определяют все допустимые взаимодействия между ними.

Классическая мандатная модель Белла-Лападулы

В классической мандатной модели Белла – Лападулы состояния системы делятся на:

- безопасные, в которых информационные потоки не противоречат установленным в модели правилам;
- небезопасные, в которых эти правила нарушаются, и происходит утечка информации.

Белл и Лападула предложили следующее определение безопасного состояния:

1. Состояние (F, A) называется безопасным по чтению (или просто безопасным) тогда и только тогда, когда для каждого субъекта si, который реализует в этом состоянии поток типа read к объекту oj, уровень безопасности субъекта si доминирует над уровнем безопасности объекта oj

   $\forall s_i \in S, \forall o_j \in O, read \in a_{ij} \rightarrow F(s_i) \geq F(o_j), \quad i = 1,\dots m, j = 1,\dots,n$
2. Состояние (F, A) называется безопасным по записи (или *- безопасным) тогда и только тогда, когда для каждого субъекта si, который реализует в этом состоянии поток типа write к объекту oj, уровень безопасности объекта oj доминирует над уровнем безопасности субъекта si

   $\forall s_i \in S, \forall o_j \in O, write \in a_{ij} \rightarrow F(o_j) \geq F(s_i), \quad i = 1,\dots m, j = 1,\dots,n$
3. Состояние системы безопасно тогда и только тогда, когда оно безопасно и по чтению, и по записи.

В соответствии с определением безопасного состояния критерий безопасности системы формулируется следующим образом.

Система (Q_0, R, T) безопасна тогда и только тогда, когда ее начальное состояние Q_0 безопасно и все состояния, достижимые из Q_0 в результате применения конечной последовательности запросов из R безопасны.

Белл и Лападула доказали теорему, формально определяющую безопасность системы при соблюдении необходимых условий. Эта теорема называется Основной теоремой безопасности.

Основная теорема безопасности

Система (Q_0, R, T) безопасна тогда и только тогда, когда

а) начальное состояние системы Q_0 безопасно

b) для любого состояния Q¢, достижимого из Q_0 в результате выполнения конечной последовательности запросов из R таких, что: T(Q, r) = Q′; Q = (F, A); Q′ = (F′, A′)

Для каждого $s_i \in S$ и $O_j \in O$ (i = 1, …, m, j = 1,…, n) выполняются следующие условия:

1. если read $\in a_{ij}' \wedge$ read $\notin a_{ij}$, то $F'(s_i) \geq F'(o_j)$
2. если read $\in a_{ij} \wedge F'(s_i) < F'(o_j)$, то read $\notin a_{ij}'$
3. если write $\in a_{ij}' \wedge$ write $\notin a_{ij}$, то $F'(o_j) \geq F'(s_i)$
4. если write $\in a_{ij} \wedge F'(o_j) < F'(s_i)$, то write $\in a_{ij}'$ ▪

Доказательство

Теорема утверждает, что система с безопасным начальным состоянием Q_0 является безопасной тогда и только тогда, когда при любом переходе системы из одного состояния в другое не возникает никаких новых и не сохраняется никаких старых отношений доступа (потоков), которые будут небезопасны по отношению к функции уровня безопасности нового состояния.

Формально эта теорема определяет все необходимые и достаточные условия, которые должны быть выполнены для того, чтобы система, начиная свою работу в безопасном состоянии, никогда не достигла небезопасного состояния.

Безопасная функция перехода. Теорема безопасности Мак – Лина

Недостаток основной теоремы безопасности Белла–Лападулы состоит в том, что ограничения, накладываемые теоремой на функцию перехода, совпадают с критериями безопасности состояния системы. Поэтому данная теорема является избыточной по отношению к определению безопасного состояния.

Из теоремы так же следует, что все состояния, в которые может перейти система из безопасного состояния, при определенных условиях безопасны. Но при этом ничего не говорится о том, что могут ли в процессе перехода изменятся уровни безопасности субъектов и объектов.

Если в процессе перехода системы из одного состояния в другое уровни безопасности субъектов и объектов могут изменяться, то это может привести к потере свойств безопасности системы.

Действительно можно представить гипотетическую систему (в литературе оно получила название Z – системы), в которой при попытке субъекта с низким уровнем безопасности прочитать информацию из объекта, имеющего более высокий уровень безопасности, будет происходить понижение уровня безопасности объекта до уровня безопасности субъекта и осуществляться чтение.

В этом случае функция перехода Z – системы удовлетворяет ограничениям (условиям) основной теоремы безопасности, и все состояния такой системы также являются безопасными в смысле критерия Белла–Лападулы, но вместе с тем в этой системе любой субъект может реализовать поток типа read к любому объекту, что, очевидно, не совместимо с безопасностью в обычном понимании.

Следовательно, необходимо сформулировать теорему, в которой не только бы констатировалась безопасность всех достижимых состояний в системе при выполнении определенных условий, но и гарантировалась бы безопасность в процессе осуществления переходов между состояниями. Для этого необходимо регламентировать изменения уровней безопасности при переходе системы из одного состояния в другое с помощью дополнительных правил.

Такую интерпретацию мандатной модели осуществил Мак-Лин. Он предложил свою формулировку основной теоремы безопасности, основанную не на понятии безопасного состояния, а на понятии безопасного перехода.

При таком подходе функция уровня безопасности представляется в виде двух функций:

- F_s – которая ставит каждому субъекту системы в соответствие определенный уровень безопасности из множества D;

- F_o – которая ставит каждому объекту системы в соответствие определенный уровень безопасности из множества D.

Функция перехода T считается безопасной по чтению, если для любого перехода T(Q, r) = T′ выполняются следующие условия:

1. если read $\in a_{ij}' \wedge$ read $\notin a_{ij}$, (возникает новое отношение доступа), то
   $F_s'(s_i) \geq F_o'(o_j)$; $F = F'$
2. если $F_s \neq F_s'$ (изменяются уровни безопасности субъекта), то
   $A = A'$, $F_o = F_o'$ и
   для $\forall s_i$ и o_j, для которых $F_s'(s_i) < F_o'(o_j)$,
   read $\notin a_{ij}'$, i = 1…m, j=1…n.
3. если $F_o \neq F_o'$ (изменяется уровень безопасности объекта), то
   $A = A'$, $F_s = F_s'$ и
   для $\forall s_i$ и o_j, для которых $F_s'(s_i) < F_o'(o_j)$,
   read $\notin a_{ij}'$, i = 1…m, j=1…n.

Функция перехода T считается безопасной по записи, если для любого перехода T(Q,r) = T′ выполняются следующие три условия:

1. если write $\in a_{ij}' \wedge$ read $\notin a_{ij}$, (возникает новое отношение доступа), то
   $F_o'(o_j) \geq F_s'(s_i)$; $F = F'$
2. если $F_s \neq F_s'$ (изменяются уровни безопасности субъекта), то
   $A = A'$, $F_o = F_o'$ и
   для $\forall s_i$ и o_j, для которых $F_s'(s_i) > F_o'(o_j)$,
   write $\notin a_{ij}'$, i = 1…m, j=1…n.
3. если $F_o \neq F_o'$ (изменяется уровень безопасности объекта), то
   $A = A'$, $F_s = F_s'$ и
   для $\forall s_i$ и o_j, для которых $F_s'(s_i) > F_o'(o_j)$,
   write $\notin a_{ij}'$, i = 1…m, j=1…n.

Функция перехода является безопасной тогда и только тогда, когда она одновременно безопасна и по чтению, и по записи.

Смысл введения перечисленных ограничений и их принципиальное отличие от условий теоремы Белла–Лападулы состоит в том, что нельзя изменить одновременно более одного компонента состояния системы.

В процессе перехода:

- либо возникает новое отношение доступа;
- либо изменяется уровень безопасности субъекта;
- либо изменяется уровень безопасности объекта.

Следовательно, функция перехода является безопасной тогда и только тогда, когда она изменяет только один из компонентов состояния, и изменения не приводят к нарушению безопасности системы.

Поскольку безопасный переход из состояния в состояние позволяет изменяться только одному элементу и так как этот элемент может быть изменен только способами, сохраняющими безопасность состояния, была доказана следующая теорема о свойствах безопасной системы.

Теорема безопасности Мак-Лина

Система безопасна в любом состоянии и в процессе переходов между ними, если ее начальное состояние является безопасным, а ее функция перехода удовлетворяет критерию Мак-Лина.

Однако система может быть безопасной по определению Белла-Лападулы, но не иметь безопасной функции перехода.

Такая формулировка основной теоремы безопасности предоставляет в распоряжение разработчиков защищенных систем базовый принцип их построения, в соответствии с которым для обеспечения безопасности системы, как в любом состоянии, так и в процессе

перехода между ними, необходимо реализовать для нее такую функцию перехода, которая соответствует указанным условиям.

Выводы: классическая модель Белла-Лападулы построена для анализа систем защиты, реализующих мандатное (полномочное) разграничение доступа.

Ролевая политика безопасности

Ролевая политика безопасности представляет собой существенно усовершенствованную модель Харрисона–Руззо–Ульмана, однако ее нельзя отнести ни к дискреционным, ни к мандатным моделям.

В ролевой политике безопасности классическое понятие субъекта заменяется понятиями пользователь и роль.

Пользователь – это человек, работающий с системой и выполняющий определенные служебные обязанности, а с понятием роли связывается набор полномочий, необходимых для выполнения этих служебных обязанностей.

При использовании ролевой политики безопасности управление доступом осуществляется в две стадии:

- каждому пользователю назначается список доступных эму ролей;
- для каждой роли указывается набор полномочий, представляющий набор прав доступа к объектам.

Причем полномочия назначаются ролям в соответствии с принципом наименьших привилегий, из которого следует, что каждый пользователь должен обладать только минимально необходимым для выполнения своей роли набором полномочий.

В модели ролевой политики безопасности используются следующие множества:

- U – множество пользователей;
- R – множество ролей;
- P – множество полномочий на доступ к объектам компьютерной системы, представленное, например, в виде матрицы доступа;
- S – множество сеансов работы пользователей с системой.

Для перечисленных множеств определены следующие отношения:

- $UA \subseteq U \times R$ – отображает множество пользователей на множество ролей, определяя для каждого пользователя набор доступных ему ролей,
- $RA \subseteq P \times R$ – отображает множество полномочий на множество ролей, устанавливает для каждой роли набор присвоенных ей полномочий.

Правила управления доступом ролевой политики безопасности определяются следующими функциями:

- User: $S \rightarrow U$ – эта функция для каждого сеанса S определяет пользователя, который осуществляет этот сеанс работы с системой: $User(s) = u \in U$,
- Roles: $S \rightarrow P(R)$ – эта функция для каждого сеанса s определяет набор ролей из множества R, которые могут быть одновременно доступны пользователю в этом сеансе:
- $Roles: (s) = \{ r_i \mid (user(s), r_i) \in UA \}$
- Permission: $S \rightarrow P$ – эта функция для каждого сеанса задает набор доступных в нем полномочий, который определяется как совокупность полномочий всех ролей, задействованных в этом сеансе:
- $Permission (s) = U_{r \in roles(s)} \{ p_i \mid (p_i, r) \in RA \}$

Критерий безопасности ролевой модели определяется следующим правилом:

Система считается безопасной, если любой пользователь системы, работающий в сеансе s, может выполнять действия, требующие полномочия p, только в том случае, если $p \in Permission (s)$.

Из формулировки критерия безопасности ролевой модели следует, что управление доступом осуществляется главным образом не с помощью назначения полномочий ролям, а определением отношения UA, с помощью которого назначаются роли пользователям, и функции Roles (s), которая определяет доступный в сеансе набор ролей.

Существуют различные интерпретации ролевой модели, различающиеся видом функций User( ), Roles ( ), Permission ( ), а так же ограничениями, накладываемыми на отношения RA и UA.

Иерархическая организация ролей

Иерархическая организация ролей представляет собой наиболее распространенный тип ролевой модели, поскольку она очень точно отражает установившиеся реальном мире отношения подчиненности между участниками процесса обработки информации и разделение между ними сфер ответственности.

В реальной жизни, как правило, пользователи жестко упорядочены по степени ответственности, соответствующей уровню полномочий, которыми они обладают. Причем, более доверенные пользователи, стоящие на служебной лестнице выше, всегда обладают всеми полномочиями, менее доверенных, подчиненных им.

При иерархической организации ролей роли упорядочиваются по уровню предоставляемых полномочий. Чем выше роль пользователя в иерархии, тем больше с ней связано полномочий, поскольку считается, что если пользователю присвоена некоторая роль, то ему автоматически назначаются и все подчиненные ей по иерархии роли. Иерархия ролей допускает множественное наследование.

Иерархическая модель отличается от классической модели следующим:

- вводится дополнительное отношение $RH \subseteq R \times R$, которое отображает частичное отношение порядка на множестве R. Это отношение определяет иерархию ролей и задает на множестве ролей оператор доминирования $\leq$, такой что, если $r_2 \leq r_1$, то r_1 находится в иерархии выше, чем r_2;
- отношение UA записывается в виде $UA^h \subseteq U \times R$

С помощью этого отношения каждому пользователю назначается набор ролей, причем вместе с каждой ролью в него включаются и все роли, подчиненные ей по иерархии, т. е.
для $\forall r_1, r_2 \in R, u \in U: r_2 \leq r_1 \wedge (u, r_1) \in UA^h \Rightarrow (u, r_2) \in UA^h$

- функция Roles: $S \rightarrow P(R)$ принимает вид - $Roles^h : S \rightarrow P(R)$

Эта функция назначает каждому сеансу s набор ролей из иерархии ролей пользователя, работающих в этом сеансе: $Roles^h(s) \subseteq \{ r_i \mid \exists r_i \leq r_1 (user(s), r_1) \in UA^h \}$

- функция Permission: S ® P заменяется функцией $Permission^h$: $S \rightarrow P$

Эта функция определяет полномочия сеанса как совокупность полномочий всех задействованных в нем ролей и полномочий всех ролей подчиненных им

$Permission^h(s) = U_{r1 \in roles^h(s)} \{ p_i \mid \exists r_2 \leq r_1 (p_i, r_2) \in RA^h \}$

Таким образом, каждому пользователю назначается некоторое подмножество иерархии ролей, а в каждом сеансе является доступной совокупность полномочий ролей, которая составляет фрагмент этой иерархии.

Такой подход позволяет существенно упростить управление доступом за счет неявного назначения полномочий.

Взаимоисключающие роли

В этом случае множество ролей пользователей разбивается на подмножества, объединяющее роли, которые не могут быть назначены пользователю одновременно и эти подмножества считаются несовместимыми. Таким образом, пользователю может быть назначено только по одной роли из каждого подмножества несовместимых ролей.

Для определения отношения несовместимости на множестве ролей R задается функция : Exclusive: $R \rightarrow P(R)$, которая для каждой роли определяет множество несовместимых с ней ролей. В этом случае на отношение UA, отображающее множество ролей на множество пользователей, накладывается следующее ограничение: если $(u, r_1) \in UA \wedge r_2 \in Exclusive(r_1) \Rightarrow (u, r_2) \notin UA$.

Взаимоисключающие роли реализуют так называемое статическое разделение обязанностей, когда конфликт несовместимости полномочий решается на стадии назначения ролей.

Ограничение на одновременное использование ролей в рамках одного сеанса

В этом случае множество ролей также разбивается на подмножество несовместимых ролей, но отношение UA может назначить пользователю любую комбинацию ролей. Однако в ходе сеанса работы с системой пользователь может одновременно активизировать не более одной роли из каждого подмножества несовместимых ролей.

В этом случае на функцию Roles: $S \rightarrow P(R)$, которая назначает каждому сеансу s некоторый набор ролей, накладывается следующее ограничение: если $\forall r_1, r_2 \in R: r_1 \in Roles(s) \wedge r_2 \in Exclusive(r_1) \Rightarrow r_2 \notin Roles(s)$.

Поскольку в процессе сеанса пользователь может переключаться между различными ролями, то он должен избегать конфликтов несовместимости между ними. Эта политика получила название динамического разделения обязанностей.

Количественные ограничения при назначении ролей и полномочий

Эта модель предназначена для тех случаев, когда роль может быть назначена только ограниченному числу пользователей, и/или представление некоторых полномочий допускается только для ограниченного числа ролей.

Модель использует функцию $Cardinality^r: R \rightarrow N$, которая определяет какому числу пользователей может быть назначена каждая роль, и функцию $Cardinality^p: P \rightarrow N$, которая для каждого полномочия определяет скольким ролям оно может быть присвоено.

На отношения UA и RA накладываются следующие ограничения для $\forall r_i \{ | u_j | (u_j, r_i) \in UA | \} \leq Cardinality^r(r_i)$ и для $\forall p_i \{ | r_j | (p_i, r_j) \in RA | \} \leq Cardinality^p(p_i)$

Смысл данных ограничений состоит в том, что благодаря ограничению количества пользователей, осуществляющих те или иные операции, сужается круг лиц, на которых лежит ответственность за совершение соответствующих действий.

Группировка ролей по полномочиям

Роли и полномочия, которые дополняют друг друга, и назначение которых по отдельности не имеет смысла, объединяются в группы, которые могут быть назначены только целиком. Для этого вводятся дополнительные правила, в соответствии с которыми:

- **любая роль может быть назначена пользователю только в том случае, если ему уже присвоен определенный набор ролей**, для этого на множестве ролей R задается функция $Prerequisite^r: R \rightarrow P(R)$, определяющая для каждой роли подмножество ролей, которые должны быть назначены пользователю прежде, чем он получить эту роль, а для отношения UA, назначающего пользователям роли, вводится ограничение, требующее наличия у пользователя всех ролей, принадлежащих одной группе: для $\forall (u, r) \in UA \wedge r' \in Prerequisite^r(r) \Rightarrow (u, r') \in UA$
- **роль может быть наделена полномочиями только тогда, когда с ней уже связан определенный набор полномочий**, для этого на множестве P задается функция $Prerequisite^p: P \rightarrow P(P)$, определяющая для каждого полномочия подмножества полномочий, которые должны быть присвоены роли перед назначением полномочия, а на отношение RA, отображающее множество полномочий на множество ролей, накладывается ограничение, в соответствии с которым роли сразу присваиваются все полномочия из группы для $\forall (p, r) \in PA \wedge p' \in Prerequisite^p(p) \Rightarrow (p', r) \in RA$.

Введение подобных ограничений упрощает администрирование системы в тех случаях, когда полномочия должны предоставляться определенным набором, или когда назначение ролей должно производиться в определенной последовательности.

Вывод: Ролевая политика представляет широкий выбор для разработчиков систем управления доступом – с одной стороны, использование матрицы прав доступа может превратить ее в разновидность дискреционной модели, но, с другой стороны, применение жестких правил распределения ролей между сеансами и пользователями, а также полномочий между ролями, позволяет построить на ее основе полноценную нормативную политику.

Модель политики безопасности распределенной компьютерной системы

Распределенная компьютерная система состоит из двух сегментов:

- локального, который включает в себя один компьютер или сегмент локальной вычислительной сети;
- внешнего, который представляет собой компьютеры либо сегменты локальной вычислительной сети, не включенные в локальный сегмент.

Удаленным субъектом называется субъект, принадлежащий множеству субъектов внешнего сегмента компьютерной сети.

Удаленным объектом называется объект, принадлежащий множеству объектов внешнего сегмента компьютерной сети.

В распределенной компьютерной системе существует четыре вида потоков:

- потоки между локальными субъектами и локальными объектами;
- потоки между локальными субъектами и удалёнными субъектами;
- потоки между удаленными субъектами и локальными объектами;
- потоки между удаленными субъектами и удаленными объектами.

Примечание: Поток между субъектом и объектом означает поток между ассоциированным объектом субъекта и объектом.

В дальнейшем будут рассматриваться только потоки между локальными субъектами и локальными объектами при участии в потоке локального субъекта.

Передача данных на расстояние требует использования телекоммуникационного программного обеспечения (ТПО), с помощью которого осуществляется совместная работа прикладных программных средств и аппаратуры передачи данных.

В объектно-субъектной модели телекоммуникационное программное обеспечение рассматривается как субъект $s_{ком}$, относящийся к подмножеству субъектов локального сегмента компьютерной системы.

Обозначим поток от ассоциированного объекта $o_{уд}$ удаленного субъекта $s_{уд}$ к ассоциированному объекту $o_{ком}$ субъекта $s_{ком}$ как $Stream\ (s_{уд}, o_{уд}) \rightarrow o_{ком}$.

Предположим, что свойства субъекта $s_{ком}$ таковы, что возможно существование потока

$Stream\ (s_{ком}, o_{ком}) \rightarrow o_{лок}$, тогда по свойству транзитивности потоков имеет место доступ субъекта $s_{уд}$ к объекту $o_{лок}$ через субъект $s_{ком}$.

Метод межсетевого экранирования

Недостатки классической модели политики безопасности с полным проецированием прав пользователя на всё множество субъектов привели к появлению методов защиты, связанных с "экранированием" локального сегмента компьютерной системы от внешнего сегмента компьютерной системы.

Суть экранирования состоит в том, что поток между удаленным субъектом $s_{уд}$ и телекоммуникационным субъектом $s_{ком}$ осуществляется через дополнительный объект o_f, ассоциированный с субъектом-анализатором s_f.

В этой системе существуют следующие потоки:

- $Stream(s_{уд}, o_{уд}) \rightarrow o_f$
- $Stream(s_f, o_f) \rightarrow o_{ком}$
- $Stream(s_{ком}, o_{ком}) \rightarrow o_{лок}$

и потоки обратного направления

- $Stream(s_{ком}, o_{лок}) \rightarrow o_{ком}$
- $Stream(s_{ком}, o_{ком}) \rightarrow o_f$
- $Stream(s_f, o_f) \rightarrow o_{уд}$.

В этой модели субъект s_f рассматривается как некоторый фильтр, который может определить факт доступа субъекта $s_{уд}$ к объекту $o_{лок}$, или зафиксировать поток между $o_{уд}$ и $o_{ком}$. Допускается, что поток может рассматриваться на различном уровне относительно объекта $o_{лок}$.

Рассмотрим случай передачи объекта o_j удаленному субъекту $s_{уд}$

При использовании режима пакетной передачи данных объект o_j разбивается на подобъекты:

$o_j = \{ o_j^{t1}, o_j^{t2}, \ldots o_j^{ti}, \ldots o_j^{tk} \}$ (осуществляется декомпозиция на последовательность подобъектов)

Тогда телекоммуникационный субъект $s_{ком}$ должен инициализировать поток:

$Stream(s_{ком}, o_j^{ti}) \rightarrow o_f^{ti}$, где $t_i \in T = \{ t_1, t_2, \ldots, t_i, \ldots t_k \}$, t_i – промежуток времени.

При чем объекты o_j^{ti} и o_j^{ti} должны быть тождественными, а за период времени T через субъект $s_{ком}$ должен пройти весь объект o_j.

При пакетной передаче данных объект o_j^{ti} представляется в виде:

$o_j^{ti} = o_j^{ti}(адр) \,||\, o_j^{ti}(инф)$, где $o_j^{ti}(адр)$ – называется адресной частью подобъекта o_j^{ti} ; $o_j^{ti}(инф)$ – информационной частью подобъекта o_j^{ti}

Тогда объект o_j можно представить в вид:

$o_j = (o_j^{t1}(адр), o_j^{t2}(адр), \ldots, o_j^{ti}(адр), \ldots, o_j^{t1}(инф), o_j^{t2}(инф), \ldots, o_j^{ti}, \ldots)$ (как последовательность слов, определяющих адресные и информационные составляющие подобъектов).

Очевидно, что только последовательность подобъектов

$o_j^{t1}(инф)$, $t_i \in T$ представляет объект o_j, а дополнительные подобъекты $o_j^{ti}(адр)$, $t_i \in T$, необходимые для передачи информационной части подобъекта по соответствующему адресу.

Утверждение (о существовании декомпозиции на подобъекты)

Если существует поток:

$Stream(s_{уд}, o_j^{ti}) \rightarrow o_{уд}$, где $o_{уд}$ – ассоциированный объект удаленного субъекта $s_{уд}$ и объекты o_j^{ti} и $o_{уд}$ тождественны, то для любого субъекта $s_{уд}$ существует декомпозиция каждого объекта $o_j^{ti} = o_j^{ti}(адр) \,||\, o_j^{ti}(инф)$ при которой $||\, o_j^{ti}(инф)$ (конкатенация всех информационных подобъектов) составляет целый объект o_j.

Доказательство

Из утверждения также следует, что на произвольном уровне r (одном из семи) поток подобъектов, проходящих через субъект-фильтр, содержит полную информацию о всем объекте o_j. Однако структура объекта o_j^{ti} зависит от конкретного субъекта.

При рассмотрении любого множества подобъектов, составляющих o_j, получение полной информации о том, к какому именно объекту локального сегмента компьютерной сети происходит доступ, не представляется возможным. В связи с этим как минимально необходимую задачу реализации политики безопасности в субъекте-фильтре необходимо рассмотреть сборку полного пакета (объекта) из подпакетов (подобъектов).

Отметим, что политика безопасности реализуется на уровне целого объекта, а не составляющих его подобъектов.

Сформулируем задачу корректного экранирования на некотором уровне r.

Субъект s_f называется корректно экранирующим (или корректно фильтрующим) на вход относительно субъекта $s_{ком}$, если для любого объекта o_j при потоке $Stream(s_{ком}, o_f) \rightarrow o_j$

По последовательности $o_f^{t1}, o_f^{t2}, \ldots o_f^{ti}, \ldots o_f^{tk}$ можно однозначно восстановить объект o_j.

Субъект s_f называется корректно экранирующим (или корректно фильтрующим) на выход относительно субъекта $s_{ком}$, если для любого объекта o_j при потоке $Stream(s_{ком}, o_{j)} \rightarrow o_f$ по последовательности $o_f^{t1}, o_f^{t2}, \ldots o_f^{ti}, \ldots o_f^{tk}$ можно однозначно восстановить o_j.

Субъект s_f называется корректным фильтром, если он является корректно фильтрующим кок на вход, так и на выход. \

Утверждение (основная теорема о корректном экранировании)

Экранирующий субъект s_f, участвующий в потоке подобъектов уровня r, будет корректным на вход и на выход тогда и только тогда, когда для любого $s_{ком}$ и для любого o_j по последовательности $o_j^{t1}, o_j^{t2}, \ldots o_j^{ti}, \ldots o_j^{tk}$ уровня r однозначно определяется объект o_j.

Доказательство

Основная теорема о корректном экранировании хотя и является критерием, но тем не менее недостаточна конструктивна. Кроме того, субъект-фильтр не производит разделения потоков на легальные и нелегальные.

Отметим два существенных момента:

- субъект-фильтр должен иметь информацию о самих объектах o_j для осуществления сравнения;
- субъект-фильтр должен иметь информацию о разрешенных и запрещенных потоках между объектами $o_{уд}$ и $o_{ком}$.

Гарантированно изолирующим фильтром называется корректный фильтр, который разрешает порождение потоков $Stream(s_{уд}, o_{j)} \rightarrow o_{уд}$ и $Stream(s_{уд}, o_{уд)} \rightarrow o_j$ (только для потоков, принадлежащих множеству легальных потоков).

На практике субъект-фильтр не имеет доступа к множеству объектов локального сегмента компьютерной системы. В этом случае задача восстановления объекта o_j по последовательности подобъектов o_j^{ti} не может быть решена в явном виде.

Утверждение (необходимое условие гарантированной изоляции для субъекта-фильтра)

Для того чтобы фильтр был гарантировано изолирующим, необходимо обеспечить существование потоков $Stream(s_f, o_{j)} \rightarrow o_f$ и выполнить условие тождественности объектов o_j и o_f, где o_f – ассоциированный объект субъекта s_f, который служит для сравнения объекта o_j с объектом, восстановленным по последовательности подобъектов объекта o_j.

Доказательство

В зарубежных разработках вводится понятие сервиса – субъекта, в котором реализованы конкретные алгоритмы декомпозиции. При чем эти алгоритмы порождают последовательности подобъектов, свойственные только данному субъекту.

В этом случае для описания доступа из внешней среды выделяется множество доступных сервисов. Эти сервисы описывают множество субъектов, для которых разрешается поток к произвольному объекту локального сегмента компьютерной системы.

При этом сформулированные выше замечания относительно политики безопасности, реализованной в локальном сегменте компьютерной системы, остаются действительными.

Любая политика с полным проецированием прав также будет некорректна относительно сервиса, допускающего доступ удаленного субъекта к объекту локального сегмента компьютерной системы.

Однако, телекоммуникационный объект может обладать такими свойствами, которые позволят исключить опасные для защищенности локального сегмента компьютерной системы потоки между $o_{уд}$ и $o_{ком}$. Следовательно, ограничение доступных сервисов имеет смысл для построения защиты.

С другой стороны, фильтрация сервисов ограничивает множество локальных субъектов, которые могут иметь доступ к объектам локального сегмента компьютерной системы (т.к. для каждого из этих субъектов должен быть свой алгоритм декомпозиции объекта над подобъекты.

Утверждение (о тождестве фильтра сервисов и изолированной программной среды в рамках локального сегмента компьютерной системы).

Возможности внешнего злоумышленника по отношению к объектам локального сегмента компьютерной системы одинаковы как в случае существования фильтрующего субъекта s_f, так и генерации изолированной программной среды с включением субъектов:
$s_1 \ldots s_m$ из s_n, где $s_1 \ldots s_m$ – допустимые сервисы, s_f – фильтр сервисов, который допускает существование только сервисов $s_1 \ldots s_m$ из s_n.

Доказательство

Из утверждения следует, что методы защиты, связанные с разрешенными сервисами, в принципе эквивалентны методу генерации изолированной программной среды для локального сегмента компьютерной системы, в которую включены локальные субъекты, обеспечивающие телекоммуникационное взаимодействие.

В сущности, уменьшение множества субъектов, как методом генерации изолированной программной среды, так и методом фильтрации сервисов является только гарантией

выполнения политики безопасности, реализованной в субъектах локального сегмента компьютерной системы, либо субъекте-фильтре.

Далее следует отметить, что свойства произвольного субъекта $s_{уд}$ внешнего сегмента относительно s_f могут быть произвольными.

Аксиома

При произвольном составе субъектов внешнего сегмента компьютерной системы возможно формирование подобъектов на уровне ассоциированных объектов o_f субъекта-фильтра s_f для потока $Stream(s_{уд}, o_{уд)} \rightarrow o_f$ с произвольной адресной и информационной частью.

Из аксиомы следует, что фильтрация подобъектов изолированно от содержания объектов локального сегмента компьютерной системы в общем случае потенциально ненадежна относительно любых критериев фильтрации при возможности управления телекоммуникационным субъектом локального сегмента компьютерной системы со стороны злоумышленника.

Пусть в субъекте-фильтре однозначно выделяются информационные подобъекты и реализация потока $Stream(s_{ком}, o_{j)} \rightarrow o_f$ является тождественным отображением (технически это означает безошибочную передачу в тракте "фильтр-компютер").

Для всех объектов локального сегмента компьютерной системы вычисляется хэш-функция: $H(o_{j,} s_{удi}) = h_{ji}$ (хэш-функция может зависеть от индивидуальной информации пользователя $s_{удi}$) **и гарантируется их доступность для субъекта-фильтра**.

Процедура фильтрации на выход (относительно существующих объектов) формулируется следующим образом:

1. по последовательности подобъектов o_j^{t1}, o_j^{t2},… o_j^{ti},… o_j^{tk} восстанавливается объект o_z.
2. вычисляется хэш-функция $H(o_{z,} s_{удi}) = h_{zi}$
3. вычисленное значение h_{zi} сравнивается с h_{ji}.

В случае совпадения проверяются права доступа к объекту o_j.

В случае доступности объекта для передачи во внешний сегмент компьютерной системы разрешается передача подобъектов, соответствующих декомпозиции объекта, во внешнюю сеть.

В случае несовпадения передача запрещается.

Указанный метод может быть дополнен фильтрацией сервисов для обеспечения достоверного восстановления объекта по последовательности подобъектов.