Лекция 4

Модели дискреционного доступа

Общая характеристика политики дискреционного доступа

Политика дискреционного доступа охватывает самую многочисленную совокупность моделей разграничения доступа, реализованных в большинстве защищенных КС, и исторически является первой, проработанной в теоретическом и практическом плане.

Первые работы по моделям дискреционного доступа к информации в КС появились еще в 60-х годах и подробно представлены в литературе. Наиболее известные из них – модель АДЕПТ-50 (конец 60-х годов), пятимерное пространство Хартсона (начало 70-х годов), модель Хариссона-Руззо-Ульмана (середина 70-х годов), модель Take-Grant (1976 г.).

Модели дискреционного доступа непосредственно основываются и развивают субъектно-объектную модель КС как совокупность некоторых множеств взаимодействующих элементов (субъектов, объектов и т. д.).

Множество (область) безопасных доступов в моделях дискреционного доступа определяется дискретным набором троек "Пользователь (субъект)-поток (операция)-объект".

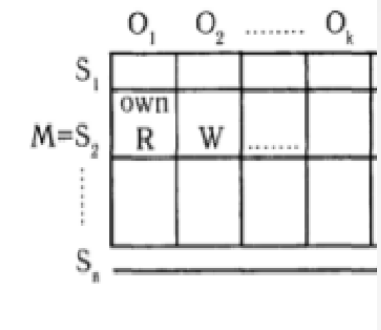
Общая характеристика политики дискреционного доступа

Множество легальных (неопасных) доступов **L** задается явным образом внешним по отношению к системе фактором в виде указания дискретного набора троек "субъект-поток(операция)-объект".

Права доступа предоставляются («прописываются» в специальных информационных объектах-стуктурах, ассоциированных с монитором безопасности), отдельно каждому пользователю к тем объектам, которые ему необходимы для работы в КС

При запросе субъекта на доступ к объекту монитор безопасности, обращаясь к ассоциированным с ним информационным объектам, в которых «прописана» политика разграничения доступа, определяет «легальность» запрашиваемого доступа и разрешает/отвергает доступ

Модели на основе матрицы доступа



субъекты \ объекты	file1	file2	dir1	file3	docA	docB	docC	docD	dir2	file4	pic1
adm	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx	rwx
pgm1	---	---	---	r-x	---	---	---	---	rwx	rwx	---
pgm2	rwx	rwx	rwx	r-x	---	---	---	---	rwx	rwx	---
pgm3	rwx	rwx	rwx	---	---	---	---	---	---	---	---
op1	---	---	rwx	r-x	---	---	---	---	---	---	---
op2	---	---	rwx	r-x	---	---	---	---	---	---	---
op3	---	---	rwx	r-x	---	---	---	---	---	---	---
us1	--x	---	r--	---	rwx	rwx	---	r--	---	---	---
us2	---	---	r--	---	r--	r--	---	r--	---	---	---
us3	--x	---	r--	---	r--	r--	rwx	rwx	---	---	rwx

Механизмы реализации дискреционного разграничения доступа

Различаются:

- в зависимости от принципов и механизмов программно-информационной структуры объекта(объектов), ассоциированных с монитором безопасности, в которых хранятся «прописанные» права доступа (тройки доступа)
- в зависимости от принципа управления правами доступа, т.е. в зависимости от того – кто и как заполняет/изменяет ячейки матрицы доступа (принудительный и добровольный принцип управления доступом).

Проблема безопасности в КС рассматривается с точки зрения анализа и исследования условий, правил, порядка и т. п. разрешений запросов на доступ, при которых система, изначально находясь в безопасном состоянии, за конечное число переходов перейдет также в безопасное состояние.

Модель АДЕПТ–50

Модель Адепт-50 — одна из первых моделей безопасности, которая рассматривает только 4 группы объектов безопасности: пользователи, задания, терминалы и файлы. Каждый объект безопасности описывается вектором (A, C, F, M), включающим следующие параметры безопасности.

Компетенция A — элемент из набора упорядоченных универсальных положений о безопасности, включающих априорно заданные возможные в ИС характеристики объекта безопасности, например, категория конфиденциальности объекта: несекретно, конфиденциально, секретно, совершенно секретно.

Категория C — рубрикатор (тематическая классификация). Рубрики не зависят от уровня компетенции. Пример набора рубрик: финансовый, политический, банковский.

Полномочия F — перечень пользователей, имеющих право на доступ к данному объекту.

Режим M — набор видов доступа, разрешенных для определенного объекта или осуществляемых объектом. Пример: читать данные, записывать данные, исполнить программу (исполнение программ понимается как порождение активной компоненты из некоторого объекта — как правило, исполняемого файла).

Четырехмерный вектор, полученный на основе прав задания (субъекта), а не прав пользователя, используется в модели для управления доступом. Такой подход обеспечивает контроль права на доступ над множеством программ и данных, файлов, пользователей и терминалов. Например, наивысшим полномочием доступа к файлу для пользователя «секретно», выполняющего задание с «конфиденциального» терминала будет «конфиденциально».

Пространство безопасности Хартсона

Пятимерное пространство на основе следующих множеств:

U - множество пользователей;

R - множество ресурсов;

E - множество операций над ресурсами;

S - множество состояний системы;

A - множество установленных полномочий, элементы множества *A* - a_{ijkl} специфицируют:

- ресурсы
- вхождение пользователей в группы;
- разрешенные операций для групп по отношению к ресурсам;

Декартово произведение *A*×*U*×*E*×*R*×*S* - область безопасного доступа

Запрос пользователя на доступ представляет собой *4-х* мерный кортеж: $q = (u,e,R',s)$, где R' - требуемый набор ресурсов

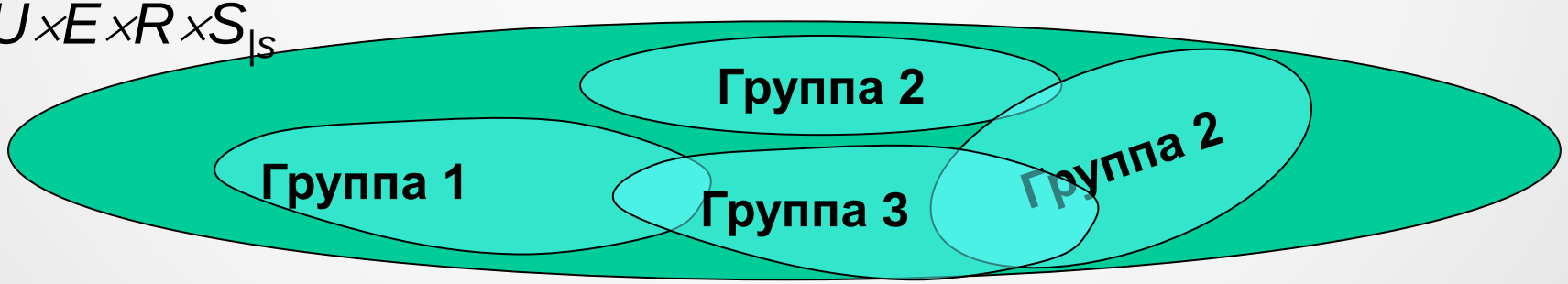
Пространство безопасности Хартсона

Процесс организации доступа по запросу осуществляется по следующему алгоритму:

1. Вызвать все вспомогательные программы для предварительного принятия решения

2. Определить те группы пользователей, в которые входит *u*, и выбрать из *A* те спецификации полномочий *P=F(u)*, которым соответствуют выделенные группы пользователей. Набор полномочий *P=F(u)* определяет т. н. привилегию пользователя

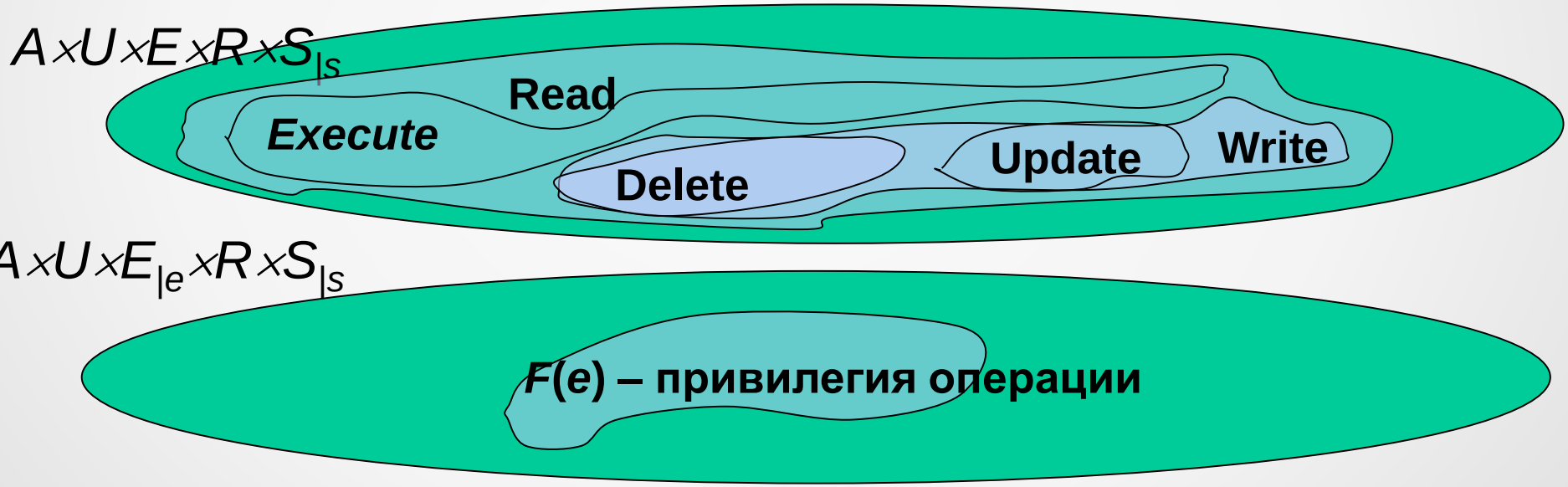
$A \times U \times E \times R \times S_{|s}$



$A \times U_{|u} \times E \times R \times S_{|s}$

Пространство безопасности Хартсона

3. Определить из множества *A* набор полномочий *P=F(e)*, которые устанавливают *e*, как основную операцию. Набор полномочий *P=F(e)* определяет привилегию операции.

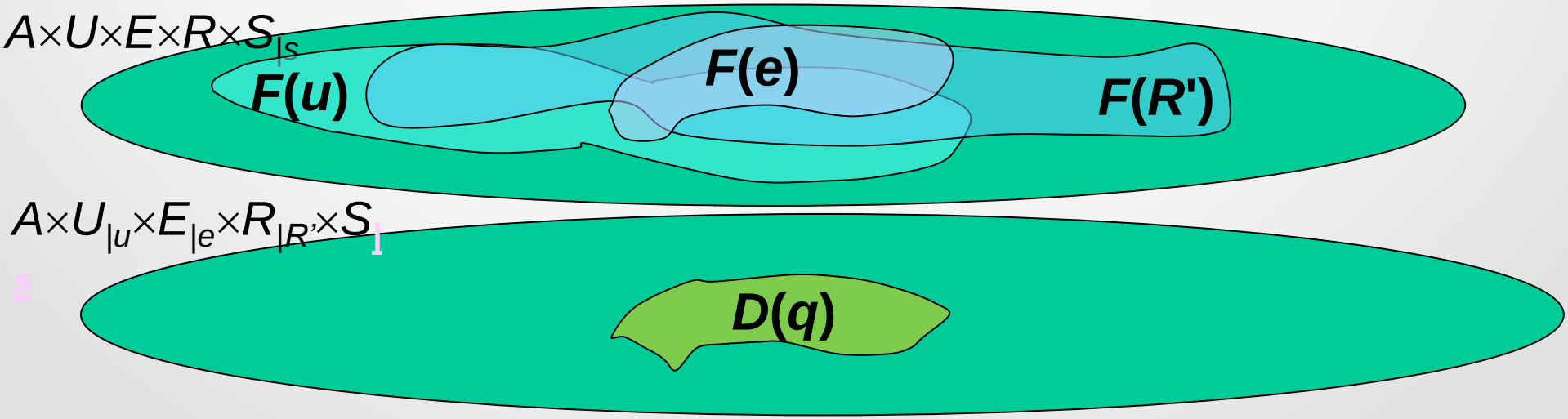


Пространство безопасности Хартсона

4.Определить из множества *A* набор полномочий *P=F(R')*, разрешающих доступ к набору ресурсов *R'*. Набор полномочий *P=F(R')* определяет привилегию ресурсов.

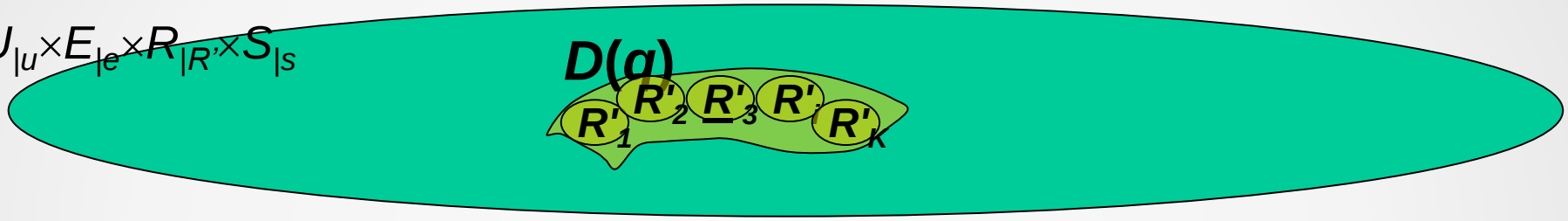
На основе *P=F(u)*, *P=F(e)* и *P=F(R')* образуется т.н. домен полномочий запроса:

$$D(q)= F(u)\cap F(e)\cap P=F(R')$$



Пространство безопасности Хартсона

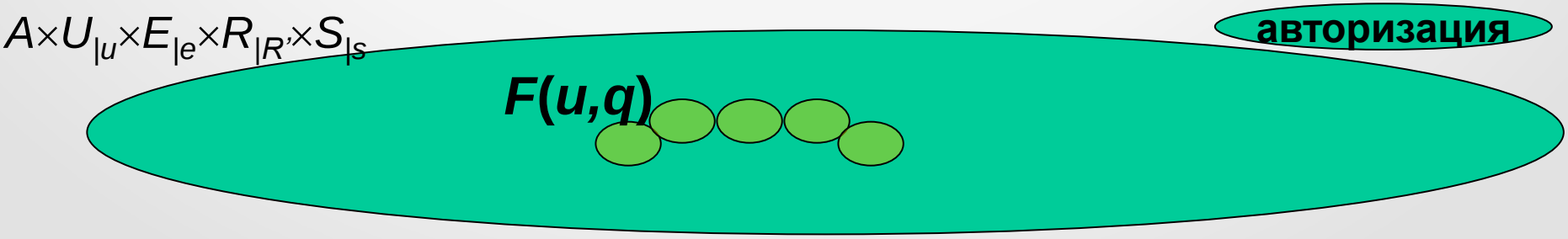
5. Убедиться, что запрашиваемый набор ресурсов R' полностью содержится в домене запроса $D(q)$, т.е. Любой r'_i из набора R' хотя бы один раз присутствует среди элементов $D(q)$.



6. Осуществить разбиение $D(q)$ на эквивалентные классы, так, чтобы в один класс попадали полномочия (элементы $D(q)$), когда они специфицируют один и тот же ресурс r из набора R'.

В каждом классе произвести операцию логического «ИЛИ» элементов $D(q)$ с учетом типа операции e.

В результате формируется новый набор полномочий на каждую единицу ресурса, указанного в $D(q)$ - $F(u,q)$.Набор $F(u,q)$ называется привилегией пользователя u по отношению к запросу q.



Пространство безопасности Хартсона

7. Вычислить условие фактического доступа, соответствующее запросу q , через операции логического ИЛИ по элементам полномочий $F(u,q)$ и запрашиваемым ресурсам r из набора R', и получить тем самым набор R'' - набор фактически доступных по запросу ресурсов

8. Оценить условие фактического доступа и принять решение о доступе:
   - разрешить доступ, если R'' и R' полностью перекрываются;
   - отказать в доступе в противном случае.

9. Произвести запись необходимых событий

10. Вызвать все программы, необходимые для организации доступа после "принятия решения".

11. Выполнить все вспомогательные программы, вытекающие для каждого случая по п.8.

12. При положительном решении о доступе завершить физическую обработку.

Пространство безопасности Хартсона

При всей своей наглядности модель Хартсона обладает одним, но существенным недостатком – безопасность системы на основе данной модели строго не доказана.

Пользователи, осуществляя законный доступ к ресурсам, могут изменять состояния системы, в том числе, изменять множество ресурсов ***R.***

Тем самым может изменяться и сама область безопасности.

Сохранится ли в таком случае безопасность системы?

Модели на основе матрицы доступа

Обозначения:
- w – "изменение объекта";
- r – "чтение объекта";
- e – "запуск объекта на выполнение".

Модели на основе матрицы доступа

При *централизованном подходе* *матрица доступа создается как от*дельный самостоятельный объект с особым порядком размещения и доступа к нему, она может достигать очень больших величин, и, кроме того, подвержено динамическому изменению.

В большинстве систем при *централизованном подходе* строки матрицы доступа характеризуют не субъектов, а непосредственно самих пользователей и их группы, зарегистрированные для работы в системе.

Для уменьшения количества столбцов матрицы, объекты доступа КС могут агрегироваться в две группы – группу объектов, доступ к которым не ограничен (т. е. разрешен любым пользователям по любым операциям), и группу объектов собственно дискреционного (избирательно разграничительного) доступа.

Наиболее характерным и известным примером такого подхода являются т. н. "биты доступа" в UNIX-системах.

Модели на основе матрицы доступа

При *распределенном подходе* *матрица доступа как отдельный объект* не создается, а представляется или так называемыми "*списками доступа*", распределенными по объектам системы, или так называемыми "*списками возможностей*", *распределенными по субъектам доступа.*

Через "списки возможностей" (маркеры доступа) субъектов реализуется широко используемый на практике *механизм привилегий, наделяющий субъектов правами* выполнять определенные операции над всеми объектами или их определенными группами (типами объектов), что формирует дополнительные аспекты дискреционной политики разграничения доступа

И централизованный, и распределенный принцип организации матрицы доступа имеет свои преимущества и недостатки, присущие в целом централизованному и децентрализованному принципам организации и управления.

Модели на основе матрицы доступа

Системы с принудительным управлением доступа.
Системы с добровольным управлением доступом.

Принудительное управление доступом (СУБД)

- вводится т.н.доверенный субъект (администратор доступа), который и определяет доступ субъектов к объектам (централизованный принцип управления)

- в таких системах заполнять и изменять ячейки матрицы доступа может только администратор

Добровольное управление доступом (ОС)

- вводится т.н. владение (владельцы) объектами и доступ субъектов к объекту определяется по усмотрению владельца (децентрализованный принцип управления)
- в таких системах субъекты посредством запросов могут изменять состояние матрицы доступа (право владения объектом его прежним владельцем может быть передано другому пользователю)

Модели на основе матрицы доступа

1. множество исходных объектов $O = (o_1, o_2, \ldots, o_M)$
2. множество исходных субъектов $S = (s_1, s_2, \ldots, s_N)$, при этом $S \subseteq O$
3. множество пользователей $U \subseteq S$
4. множество операций (действий) $Op = (Op_1, Op_2, \ldots, Op_L)$ над объектами
5. множество прав, которые м.б. даны субъектам по отношению к объектам $R (r_1, r_2, \ldots, r_K)$ – т.н. "общие права"
6. NxM матрица доступа A, в которой каждому субъекту соответствует строка, а каждому объекту - столбец
7. в ячейках матрицы располагаются права *r* соотв. субъекта над соотв. объектом в виде набора разрешенных операций Op_i $A[s_i, oj] = a_{ij}$ - право *r из R* (т.е. не общее, а конкретное. право) $r_k \sim (Op_{1k}, Op_{2k}, \ldots, Op_{Jk})$

Модель Харрисона-Руззо-Ульмана (модель HRU)

В модели *Харрисона-Руззо-Ульмана* помимо элементарных операций доступа *Read*, *Write, Own* и т.д., вводятся так же т.н. примитивные операции POp_k по *изменению* субъектами матрицы доступа:

- *Enter r into (s,o)* - ввести право *r* в ячейку (*s,o*)
- *Delete r from (s,o)* - удалить право *r* из ячейки (*s,o*)
- *Create subject s* - создать субъект *s* (т.е. новую строку матрицы *A*)
- *Create object o* - создать объект *o* (т.е. новый столбец матрицы *A*)
- *Destroy subject s* - уничтожить субъект *s*
- *Destroy object o* - уничтожить объект *o*

Состояние системы *Q* изменяется при выполнении команд $C(\alpha_1, \alpha_2, \ldots)$, изменяющих состояние матрицы доступа *A*. Команды инициируются пользователями-субъектами

Модель Харрисона-Руззо-Ульмана (модель HRU)

Примитивный оператор модели **HRU**	Условия выполнения	Новое состояние системы
Enter r *into* $A[s,o]$	$s \in S$, $o \in O$	$S'=S$, $O'=O$, $A'[s, o] = A[s, o] \cup \{r\}$, если $(s',o') \neq (s,o) \Rightarrow A'[s',o']=A[s,o]$
Delete r *from* $A[s,o]$	$s \in S$, $o \in O$	$S'=S$, $O'=O$, $A'[s,o]= A[s,o] \setminus \{r\}$, если $(s',o') \neq (s,o) \Rightarrow A'[s',o']=A[s,o]$
Create subject s'	$s' \notin S$	$S'=S \cup \{s'\}$, $O'=O \cup \{s'\}$, если $(s,o) \in S \times O \Rightarrow A'[s,o]=A[s,o]$, если $o \in O' \Rightarrow A'[s',o] = \varnothing$, если $s \in S' \Rightarrow A'[s, s'] = \varnothing$
Create object o'	$o' \notin O$	$S'= S$, $O'=O \cup \{o'\}$, если $(s,o) \in S \times O \Rightarrow A'[s,o]=A[s,o]$, если $s \in S' \Rightarrow A'[s,o] = \varnothing$
Destroy subject s'	$s' \in S$	$S'=S \setminus \{s'\}$, $O'=O \setminus \{s'\}$, если $(s,o) \in S' \times O' \Rightarrow A'[s,o]=A[s,o]$
Destroy object o'	$o' \in O \setminus S$	$S'=S$, $O'=O \setminus \{o'\}$, если $(s,o) \in S' \times O' \Rightarrow A'[s,o]=A[s,o]$

Модель Харрисона-Руззо-Ульмана (модель HRU)

Примеры команд -

Command "создать файл"(*s, f*):
Create object f ;
Enter "own" into (*s, f*) ;
Enter "read" into (*s, f*) ;
Enter "write" into (*s, f*) ;
end

Command «ввести право чтения"(*s,s',f*):
if *own* ⊂ (*s, f*) ;
then
Enter r "read" into (*s', f*) ;
end

Основной критерий безопасности -

Состояние системы с начальной конфигурацией Q_0 безопасно по праву *r*, если не существует (при определенном наборе команд и условий их выполнения) последовательности запросов к системе, которая приводит к записи права *r* в ранее его не содержащую ячейку матрицы *A*[*s,o*]

Формулировка проблемы безопасности для модели Харрисона-Руззо-Ульмана:

Существует ли какое-либо достижимое состояние, в котором конкретный субъект обладает конкретным правом доступа к конкретному объекту? (т.е. всегда ли возможно построить такую последовательность запросов при некоторой исходной конфигурации когда изначально субъект этим правом не обладает?)

Модель Харрисона-Руззо-Ульмана (модель HRU)

Определение 1. Будем считать, что возможна утечка права r в результате выполнения команды c, если при переходе системы в состояние Q' выполняется примитивный оператор, вносящий r в элемент матрицы доступов M, до этого r не содержавший.

Определение 2. Начальное состояние Q_0 называется безопасным по отношению к некоторому праву r, если невозможен переход системы в такое состояние Q, в котором может возникнуть утечка права r.

Определение 3. Система называется монооперационной, если каждая команда выполняет один примитивный оператор.

Модель Харрисона-Руззо-Ульмана (модель HRU)

Теорема . Существует алгоритм, который проверяет, является ли исходное состояние монооперационной системы безопасным для данного права r.

Для доказательства достаточно показать, что число последовательностей команд системы, которые следует проверить, ограничено. В этом случае алгоритм проверки безопасности есть алгоритм тотального перебора всех последовательностей и проверки конечного состояния для каждой из них на отсутствие утечки права r.

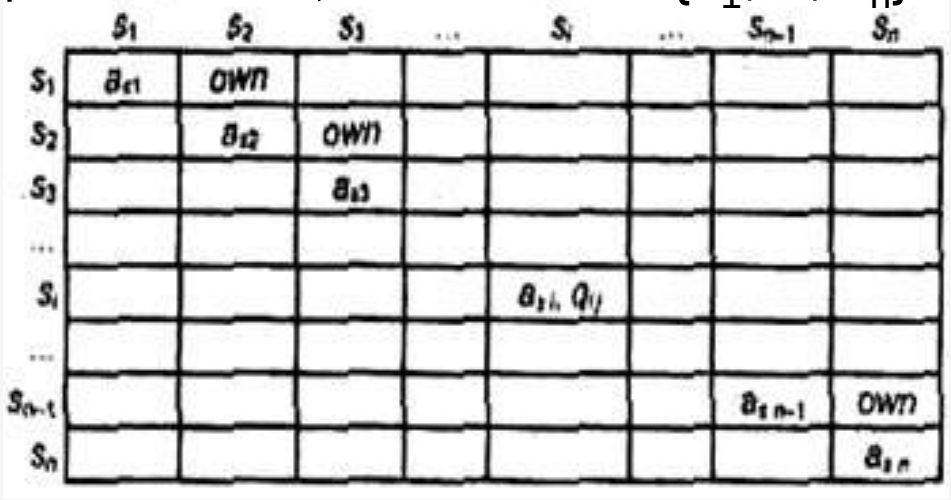
- Заметим, что нет необходимости рассматривать в последовательностях команды "удалить" и "уничтожить".
- Заметим, что нет необходимости рассматривать последовательности команд, содержащих более одного оператора "создать".
- Число различных операторов "внести" равно
  $n = |R| * (|So| + 1)*(|O_0| + 1)$.

Так как порядок операций "внести" в последовательности команд не важен, то с учетом одной операции "создать" число последовательностей команд ограничено величиной 2n+1.

Модель Харрисона-Руззо-Ульмана (модель HRU)

Теорема. Задача проверки безопасности произвольных систем алгоритмически неразрешима.

Каждой ячейке ленты МТ поставим в соответствие субъекта модели HRU, при этом будем считать, что O = S = {s_1,..., s_n}. Зададим матрицу доступов M.



Для каждой команды машины Тьюринга зададим соответствующую ей команду модели HRU.

Если машина Тьюринга останавливается в своем конечном состоянии q_0, то в соответствующей системе, построенной на основе модели HRU, происходит утечка права доступа q_0.

Модель Харрисона-Руззо-Ульмана (модель HRU)

В теории вычислимости проблема остановки — это проблема разрешимости, которая может неформально быть поставлена в виде:

Даны описание алгоритма и его начальные входные данные, требуется определить, сможет ли выполнение алгоритма с этими данными завершиться когда-либо. Альтернативой этому является то, что он работает всё время без остановки.

Не существует общего алгоритма для решения проблемы остановки. Другими словами, проблема остановки неразрешима на машине Тьюринга.

Модель Харрисона-Руззо-Ульмана (модель HRU)

Выводы по модели Харрисона-Руззо-Ульмана:

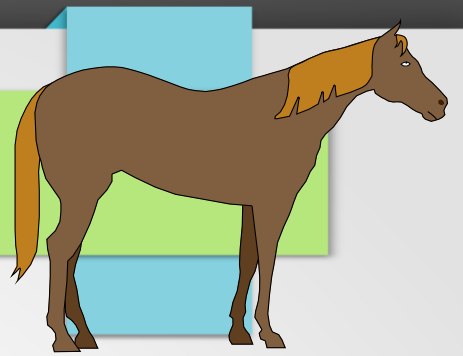
-Данная модель в ее полном виде позволяет реализовать множество политик безопасности, но при этом проблема безопасности становится неразрешимой.

-Разрешимость проблемы безопасности для монооперационных систем приводит к слабости такой модели для реализации большинства политик безопасности (т.к. нет операции автоматического наделения своими правами дочерних объектов, ввиду чего по правам доступа они изначально не различимы)

-Харрисон, Руззо и Ульман показали также, что безопасными могут быть монотонные системы (не содержащие операций *Delete и Destroy),* системы, не содержащие операций *Create, и моно-условные системы (в ко*торых запросы содержат только одно условие).

- Помимо проблем с неопределенностью распространения прав доступа в системах на основе модели HRU была подмечена еще одна серьезная проблема – отсутствие контроля за порождением потоков информации, и, в частности, контроля за порождением субъектов.

Проблема «троянских» программ



Пример 1: Пусть U_1 - некоторый пользователь, а U_2 - пользователь-злоумышленник, O_1 - объект, содержащий ценную информацию, O_2 - программа с «троянским конем» T, и M - матрица доступа, которая имеет вид:

	O_1	O_2
U_1	own r w	w
U_2		own r w

Проникновение программы происходит следующим образом. Злоумышленник U_2 создает программу O_2 и, являясь ее собственником, дает U_1 запускать ее и писать в объект O_2 информацию. После этого он инициирует каким-то образом, чтобы U_1 запустил эту программу (например, O_2 - представляет интересную компьютерную игру, которую он предлагает U_1 для развлечения). U_1 запускает O_2 и тем самым запускает скрытую программу T, которая обладая правами U_1 (т.к.была запущена пользователем U_1), списывает в себя информацию, содержащуюся в O_1. После этого хозяин U_2 объекта O_2, пользуясь всеми правами, имеет возможность считать из O_2 ценную информацию объекта O_1.

Расширения модели HRU

Типизованная матрица доступа (Модель TAM) R. Sandhu, 1992г.

КС представляется четверкой сущностей:
- множеством исходных объектов $O = (o_1, o_2, \ldots, o_M)$
- множеством исходных субъектов $S = (s_1, s_2, \ldots, s_N)$, при этом $S \subseteq O$;
- матрицей доступа **A**, каждая ячейка которой специфицирует права доступа к объектам из конечного набора прав доступа $R (r1, r2, \ldots, rK)$, т. е. $A[s,o] \subseteq R$;
- множеством (конечным набором) типов безопасности T $(t_1, t_2, \ldots, t_L)$, ***с одним из которых создается любой объект*** *(вклю-*
чая субъекты). Тип объекта впоследствии не меняется (т. н. *сильная организация типов). Таким образом, в системе задана функ*ция f_t : $O \rightarrow T$, *ставящая в соответствие каждому объекту некото*рый тип.

Типизованная матрица доступа

Примитивные опреаторы:
- *Enter* ***r into A[s,o]*** *– ввести право r в ячейку A[s,o];*
- *Delete* ***r from A[s,o]*** *– удалить право r из ячейки A[s,o];*
- *Create subject* ***s of type t****– создать субъект s типа t;*
- *Create object* ***o of type t*** *– создать объект o типа t;*
- *Destroy subject* ***s –*** *уничтожить субъект s;*
- *Destroy object* ***o*** *– уничтожить объект o.*

Состояния системы так же, как и в модели **HRU**, изменяются под воздействием запросов на модификацию матрицы доступа в виде команд, формат которых с учетом типизованного характера объектов системы имеет следующий вид:

Command $\alpha(x_1:t_1, x_2:t_2, \ldots, x_k:t_k)$
if r_1 ***in*** $A[x_{s1}, x_{o1}]$ ***and***
r_2 ***in*** $A[x_{s2}, x_{o2}]$ ***and***
:
:
r_m ***in*** $A[x_{sm}, x_{om}]$
then
$op_1, op_2, \ldots, op_n$

Таким образом, при выполнении команд на модификацию матрицы доступа вводится контроль типов фактических параметров команды, т. е. контроль типов объектов и субъектов.

Типизованная матрица доступа

*Тип t_k является **дочерним** типом в команде создания $\alpha(x_1{:}t_1, x_2{:}t_2, \ldots, x_k{:}t_k)$, если и только если имеет место один из следующих элементарных операторов:* ***"Create subject x_k of type t_k"*** *или* ***"Create object x_k of type t_k"****. В противном случае тип t_k является **родительским** типом.*

Основываясь на понятии родительских и дочерних типов, можно описать взаимосвязи между различными типами с помощью графа отношений "наследственности" при операциях порождения сущностей (субъектов и объектов). Такой граф является ориентированным (орграфом) и называется "графом создания". Множество вершин графа образуется множеством типов безопасности ***T.***

Как и в модели **HRU**, используется понятие монотонной (**MTAM)** системы, которая не содержит примитивных операторов *Delete и Destroy. Можно показать* что критерий безопасности Хариссона-Руззо-Ульмана *разрешим для ациклических реализаций («граф создания» не содержит циклов)* системы с монотонной **TAM.**

На основе специальной системы типов (например, типы "файл", "программа" для объектов; "user", "administra-tor", "auditor" для субъектов) в КС на основе **TAM возможна организация эффективного контроля за порождением субъектов** с нейтрализацией проблемы "троянских" программ.

Теоретико-графовая модель (Джонс, Липтон, Шнайдер, 1976г)

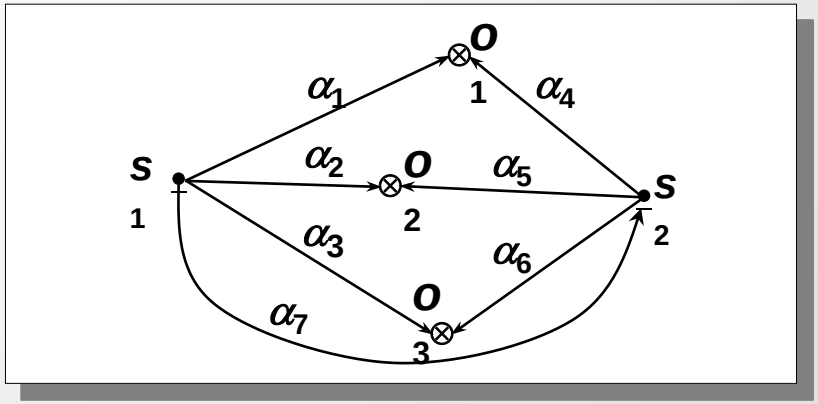
Также как и в модели HRU система защиты представляет совокупность следующих множеств:

- множество исходных объектов $O\ (o_1,o_2,\ldots,o_M)$

- множество исходных субъектов $S\ (s_1,s_2,\ldots,s_N)$, при этом $S \subseteq O$

- множество прав, которые м.б. даны субъектам по отношению к объектам $(r_1,r_2,\ldots,r_K) \cup \{t, g\}$, в том числе с двумя специфическими правами – правом *take* (*t – право брать права доступа у какого-либо объекта по отношению к другому объекту*) и правом *grant* (*g – право предоставлять права доступа к определенному объекту другому субъекту*)

-множеством E установленных прав доступа (x, y, α) субъекта x к объекту y с правом α из конечного набора прав.

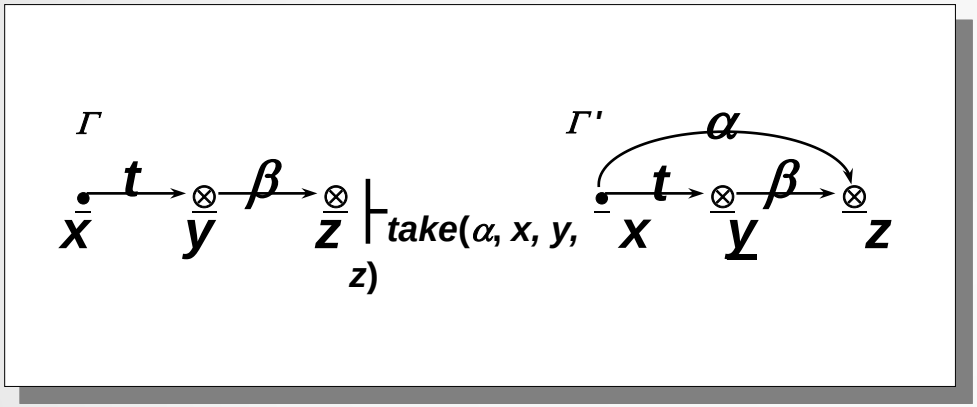


При этом состояние системы представляется графом доступов ***Г(O,S,E)***

Теоретико-графовая модель TAKE-GRANT

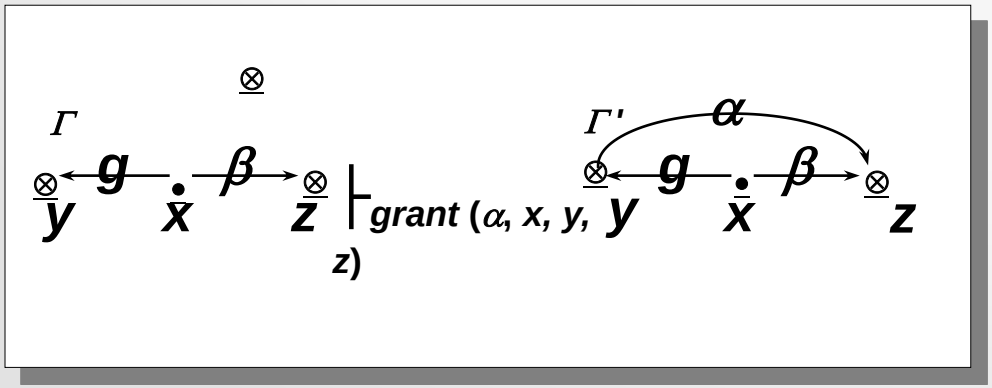
Состояние системы (Графа доступов) изменяется под воздействием элементарных команд 4-х видов

Команда "*Брать*" – *take*(α, *x*, *y*, *z*)



субъект *x* берет права доступа $\alpha \subseteq \beta$ на объект *z* у объекта *y* (обозначения: $\vdash_c$ – переход графа *Г* в новое состояние *Г'* по команде *c* ; $x \in S$; $y, z \in O$)

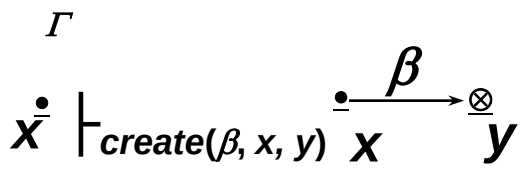
Команда «*Давать*" – *grant*(α, *x*, *y*, *z*)



субъект *x* дает объекту *y* право $\alpha \subseteq \beta$ на доступ к объекту *z (обозначения:* $\vdash_c$ *– переход графа Г в новое состояние Г' по команде c ;* $x \in S$; $y, z \in O$)

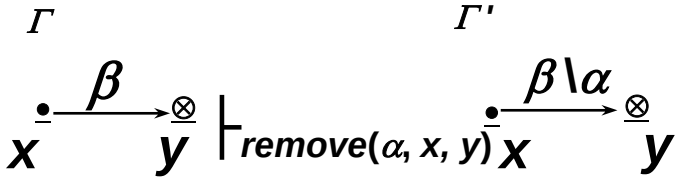
Теоретико-графовая модель TAKE-GRANT

Команда "*Создать*" – *create*(β, *x*, *y*)



субъект *x* создает объект *y* с правами доступа на него $\beta \subseteq R$ (*y* – новый объект, $O' = O \cup \{y\}$), в т. ч. с правами *t*, или *g*, или {*t*, *g*}.

Команда «*Изъять*" – *remove*(α, *x*, *y*)

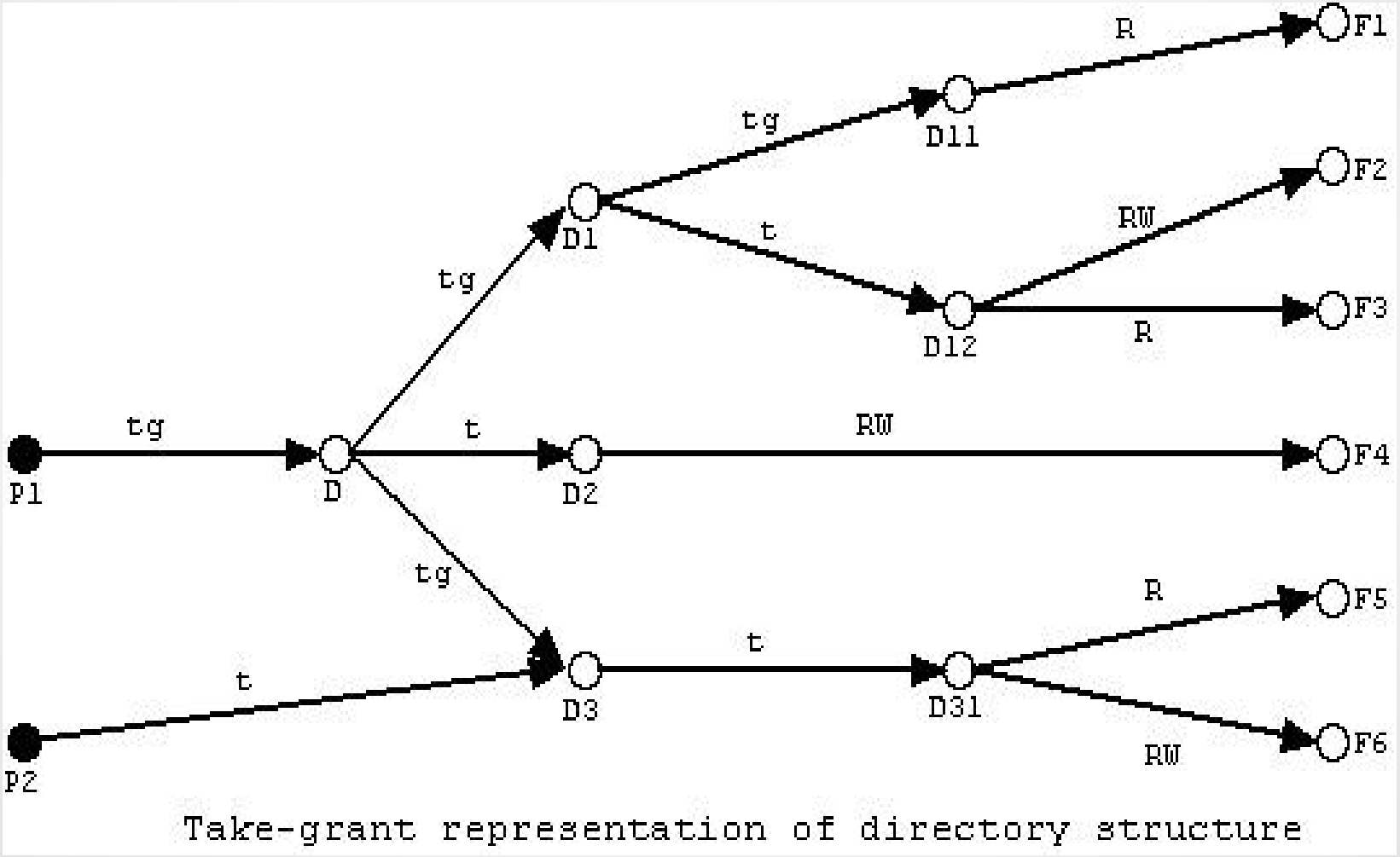


субъект *x* удаляет права доступа $\alpha \subseteq \beta$ на объект *y*

Теоретико-графовая модель TAKE-GRANT

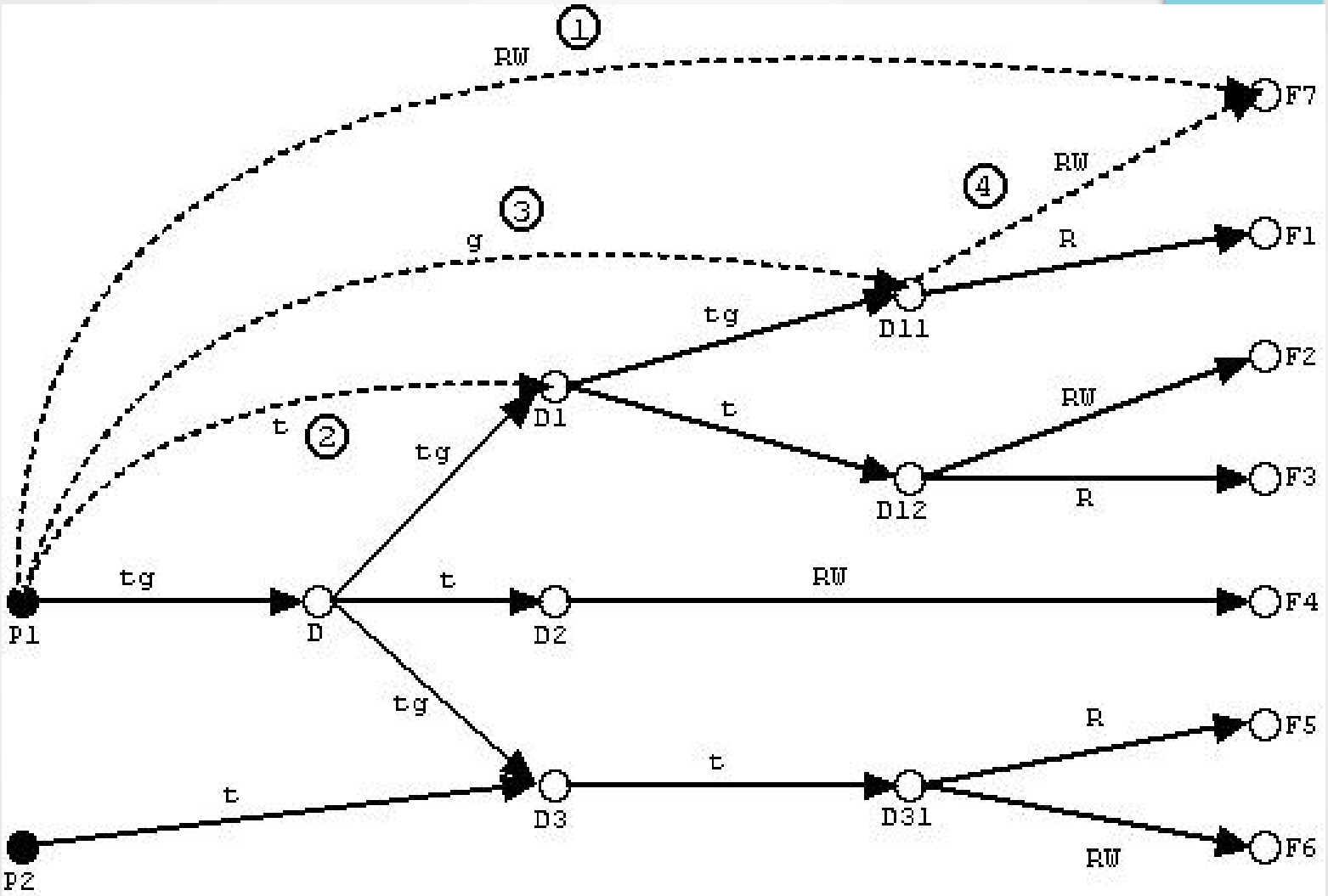
Команда модели **TAKE-GRANT**	Условия выполнения	Новое состояние системы
take(α, x, y, z)	$x \in \boldsymbol{S}$, $(x, y, t) \in \boldsymbol{E}$, $(y, z, \beta)^1 \in \boldsymbol{E}$ $x \neq z$, $\alpha \subseteq \beta$	$\boldsymbol{S'}=\boldsymbol{S}$, $\boldsymbol{O'}=\boldsymbol{O}$, $\boldsymbol{E}=\boldsymbol{E'} \cup \{(x, z, \alpha)\}$
grant(α, x, y, z)	$x \in \boldsymbol{S}$, $(x, y, g) \in \boldsymbol{E}$, $(y, z, \beta) \in \boldsymbol{E}$ $x \neq z$, $\alpha \subseteq \beta$	$\boldsymbol{S'}=\boldsymbol{S}$, $\boldsymbol{O'}=\boldsymbol{O}$, $\boldsymbol{E}=\boldsymbol{E'} \cup \{(y, z, \alpha)\}$
create(β, x, y)	$x \in \boldsymbol{S}$, $y \notin \boldsymbol{O}$	$\boldsymbol{O'}=\boldsymbol{O} \cup \{y\}$, $\boldsymbol{S'}=\boldsymbol{S} \cup \{y\}$, если y – субъект $\boldsymbol{E}=\boldsymbol{E'} \cup \{(y, z, \beta)\}$
remove(α, x, y)	$x \in \boldsymbol{S}$, $y \in \boldsymbol{O}$, $(x, y, \beta) \in \boldsymbol{E}$, $\alpha \subseteq \beta$	$\boldsymbol{S'}=\boldsymbol{S}$, $\boldsymbol{O'}=\boldsymbol{O}$, $\boldsymbol{E}=\boldsymbol{E'} \setminus \{(x, y, \alpha)\} \cup \{(x, y, \beta)\}$

Теоретико-графовая модель TAKE-GRANT



Take-grant representation of directory structure

Теоретико-графовая модель TAKE-GRANT



Adding a new file F7 to directory D11

Теоретико-графовая модель TAKE-GRANT

Безопасность системы рассматривается с точки зрения возможности получения каким-либо субъектом прав доступа к определенному объекту (в начальном состоянии Γ_0 (O_0, S_0, E_0) такие права отсутствуют) при определенной кооперации субъектов путем последовательного изменения состояния системы на основе выполнения элементарных команд.

Рассматриваются две ситуации – условия санкционированного, т.е. законного получения прав доступа, и условия «похищения» прав доступа

Теоретико-графовая модель TAKE-GRANT

- *Для исходного состояния системы $\Gamma_0 (O_0, S_0, E_0)$ и прав доступа $\alpha \subseteq R$* ***предикат "возможен доступ(α, x, y, Γ_0)"*** *является истинным тогда и только тогда, когда существуют графы доступов системы $\Gamma_1 (O_1, S_1, E_1)$, $\Gamma_2 (O_2, S_2, E_2)$, …, $\Gamma_N (O_N, S_N, E_N)$, такие, что:*
  $\Gamma_0 (O_0, S_0, E_0) \vdash c_1 \Gamma_1 (O_1, S_1, E_1) \vdash c_2 \ldots \vdash c_N \Gamma_N (O_N, S_N, E_N)$
  и $(x, y, \alpha) \in E_N$, *где* $c_1, c_2, \ldots, c_N$ *– команды переходов*

- *Вершины графа доступов являются* ***tg-связными*** *(соединены tg-путем), если в графе между ними существует такой путь, что каждая дуга этого пути выражает право t или g (без учета направления дуг)*

Теоретико-графовая модель TAKE-GRANT

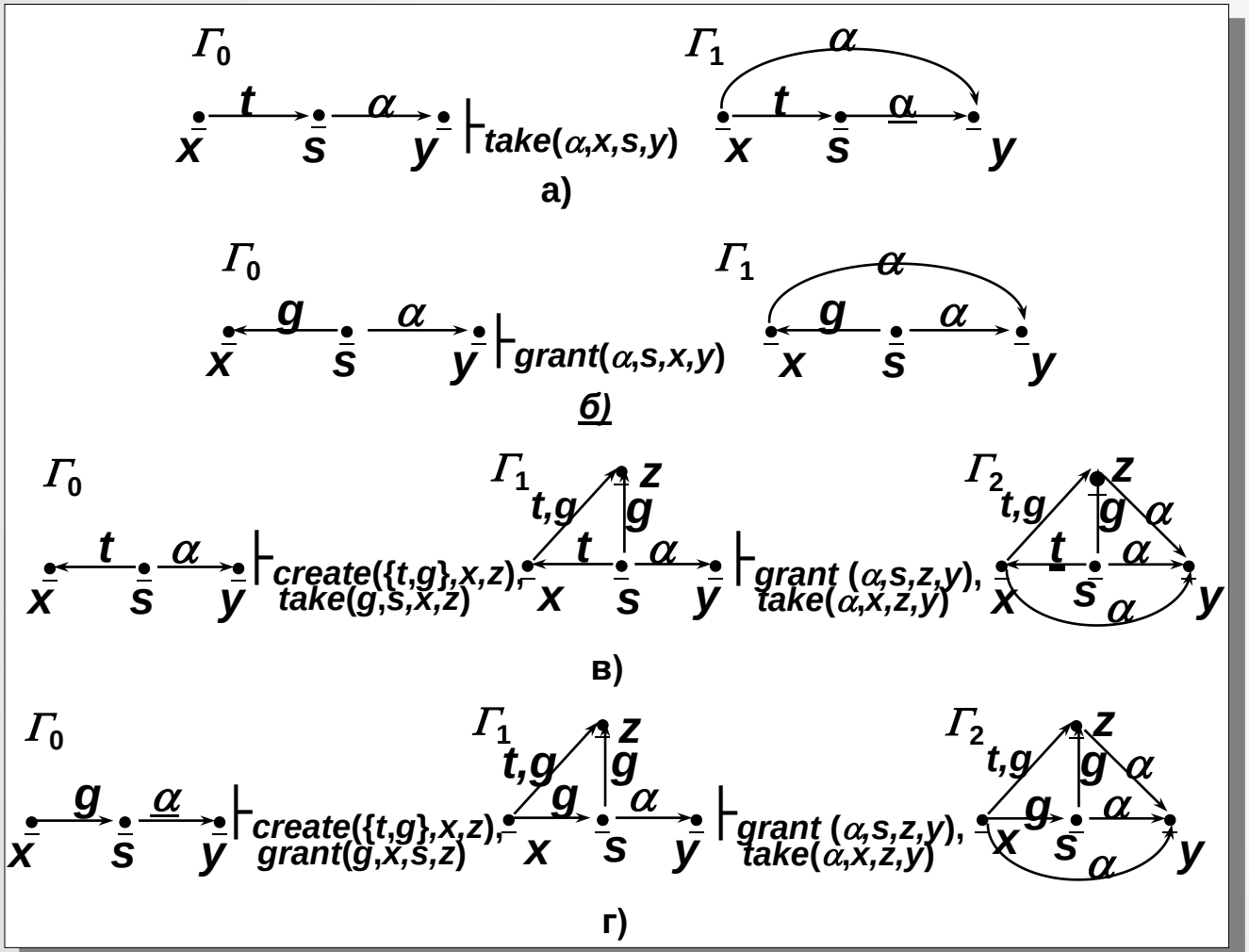
Теорема.

В графе доступов $\Gamma_0 (O_0, S_0, E_0)$ содержащем только вершины-субъекты, предикат "возможен доступ(α, x, y, Γ_0)" истинен тогда и только тогда, когда выполняются следующие условия:

- существуют субъекты $s_1,\ldots,s_m$ такие, что $(s_i, y, \gamma_i) \in E_0$ для i=1, …, m и $\alpha = \gamma_1 \cup \ldots \cup \gamma_m$.

- субъект *x* соединен в графе Γ_0 tg-путем с каждым субъектом s_i для i=1, …, m

Теоретико-графовая модель TAKE-GRANT



Доказательство: получение прав α доступа субъектом *x* у субъекта *s* на объект *y* при различных вариантах непосредственной *tg*-связности

Теоретико-графовая модель TAKE-GRANT

Определение *Островом в произвольном графе доступов* ***Г*** (***O, S, E***) *называется его максимальный* ***tg-связный*** *подграф, состоящий только из вершин субъектов.*

Определение. *Мостом в графе доступов* ***Г*** (***O, S, E***) *называется* ***tg–путь****, концами которого являются вершины-субъекты; при этом словарная запись* ***tg–пути*** *должна иметь вид*

$$\vec{t}^{\,*}, \overleftarrow{t}^{\,*}, \vec{t}^{\,*}\vec{g}\,\overleftarrow{t}^{\,*}, \vec{t}^{\,*}\overleftarrow{g}\,\overleftarrow{t}^{\,*}$$

*где символ * означает многократное (в том числе нулевое) повторение.*

Определение *Начальным пролетом моста в графе доступов* ***Г*** (***O, S, E***) *называется* ***tg-путь****, началом которого является вершина-субъект; при этом словарная запись* ***tg-пути:***

$$\vec{t}^{\,*}\vec{g}$$

Определение *Конечным пролетом моста в графе досту-пов* ***Г*** (***O, S, E***) *называется* ***tg-путь****, началом которого является вершина-субъект; при этом словарная запись* ***tg-пути:***

$$\vec{t}^{\,*}$$

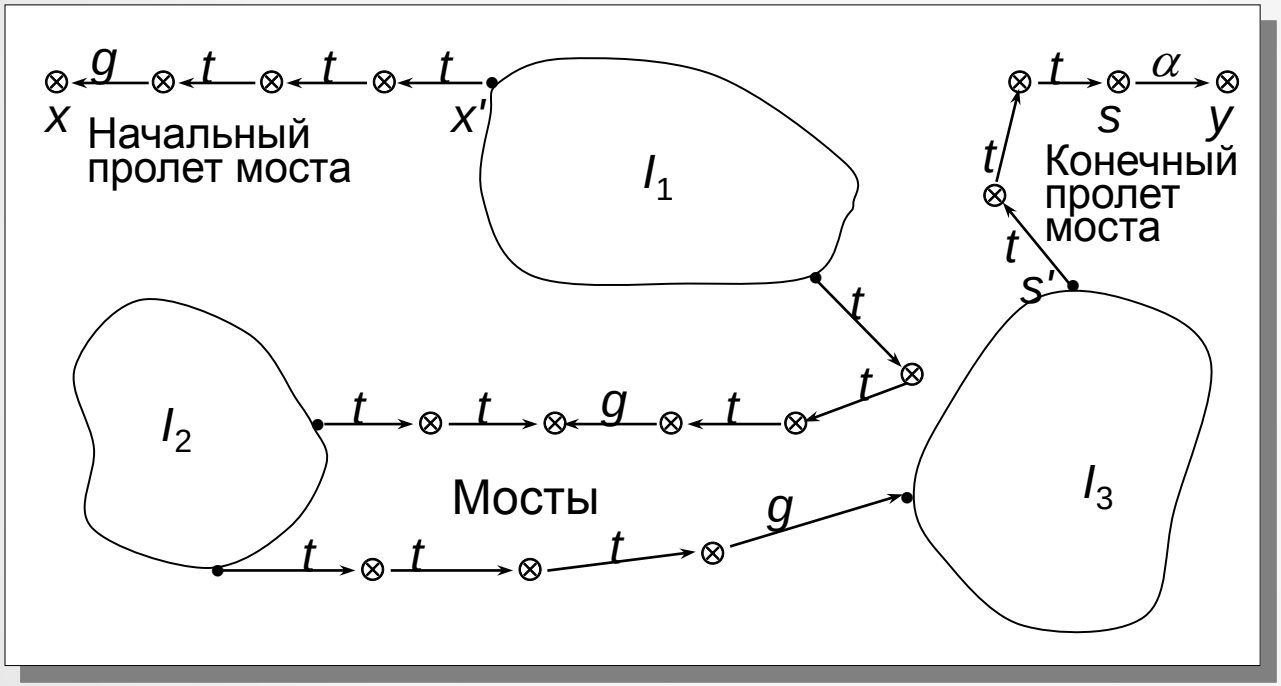
Теоретико-графовая модель TAKE-GRANT

Теорема.

В произвольном графе доступов $\Gamma_0 (O_0, S_0, E_0)$ *предикат "**возможен доступ**(*α*, x, y,* Γ_0 *)" истинен тогда и только тогда, когда выполняются условия*:

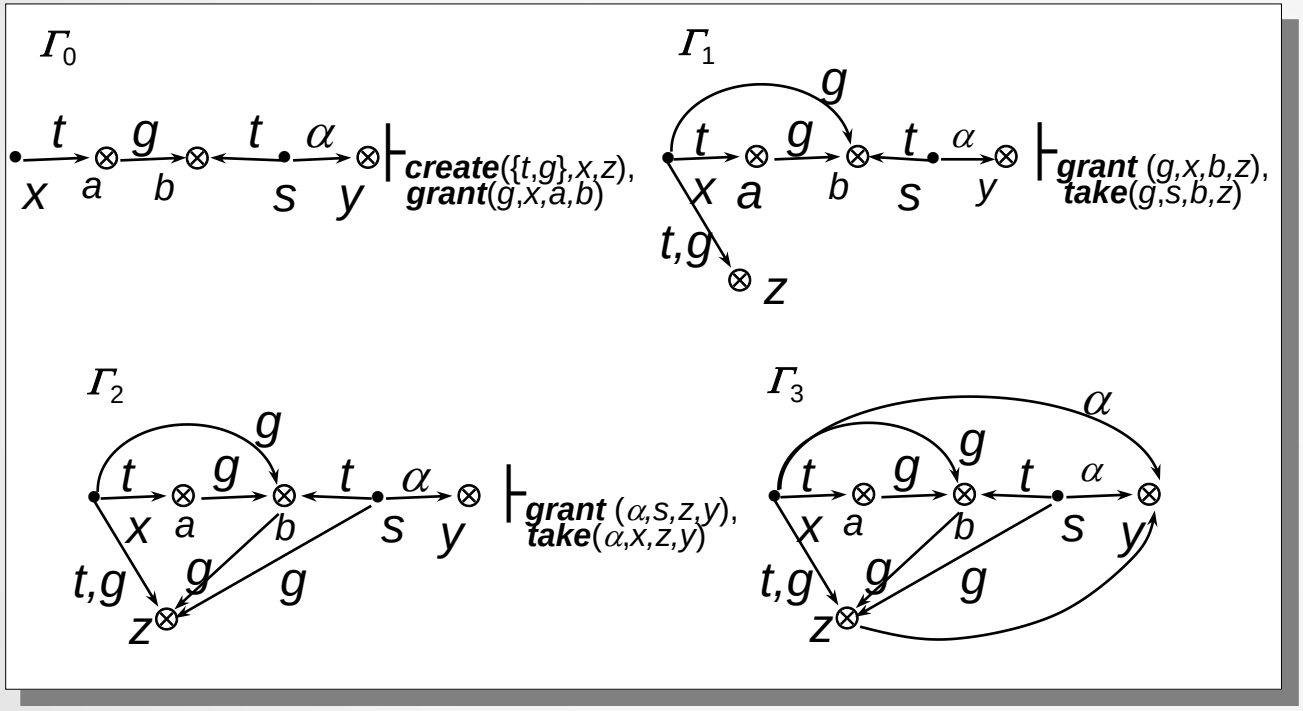
1. *существуют объекты* $s_1,\ldots,s_m$ *такие, что* $(s_i, y, \gamma_i) \in E_0$ *для* $i=1, \ldots, m$ *и* $\alpha = \gamma_1 \cup \ldots \cup \gamma_m$.
2. *существуют вершины-субъекты* $x_1',\ldots,x_m'$ *и* $s_1',\ldots,s_m'$ *такие, что*:
   - $x = x_i'$ *или* x_i' *соединен с* x *начальным пролетом моста для* $i=1, \ldots, m$;
   - $s_i = s_i'$ *или* s_i' *соединен с* s_i *конечным пролетом моста для* $i=1, \ldots, m$.
   - *для каждой пары* (x_i' , s_i'), $i = 1,\ldots, m$, *существуют острова* $l_{i1} ,\ldots, l_{iu_i}$, $u_i \geq 1$, *такие, что* $x_i' \in l_{i1}$, $s_i' \in l_{iu_i}$, *и мосты между островами* l_{i_j} *и* $l_{i_{j+1}}$.

Теоретико-графовая модель TAKE-GRANT



Пример графа доступов с возможностью передачи объекту ***x*** прав доступа α на объект ***y***

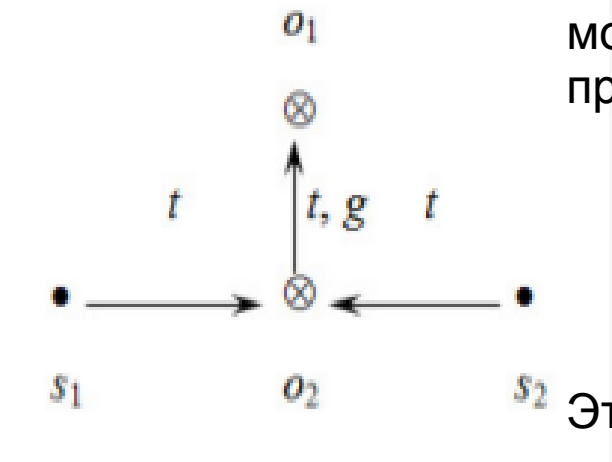
Теоретико-графовая модель TAKE-GRANT



Достаточность:

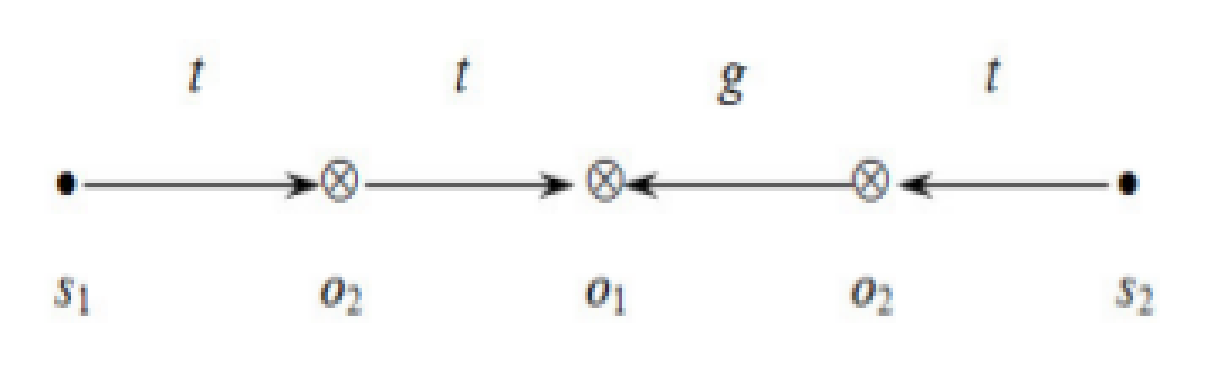
Пример передачи прав доступа по мосту

T-образный мост



мостами могут являться tg-пути, неоднократно проходящие через один и тот же объект:

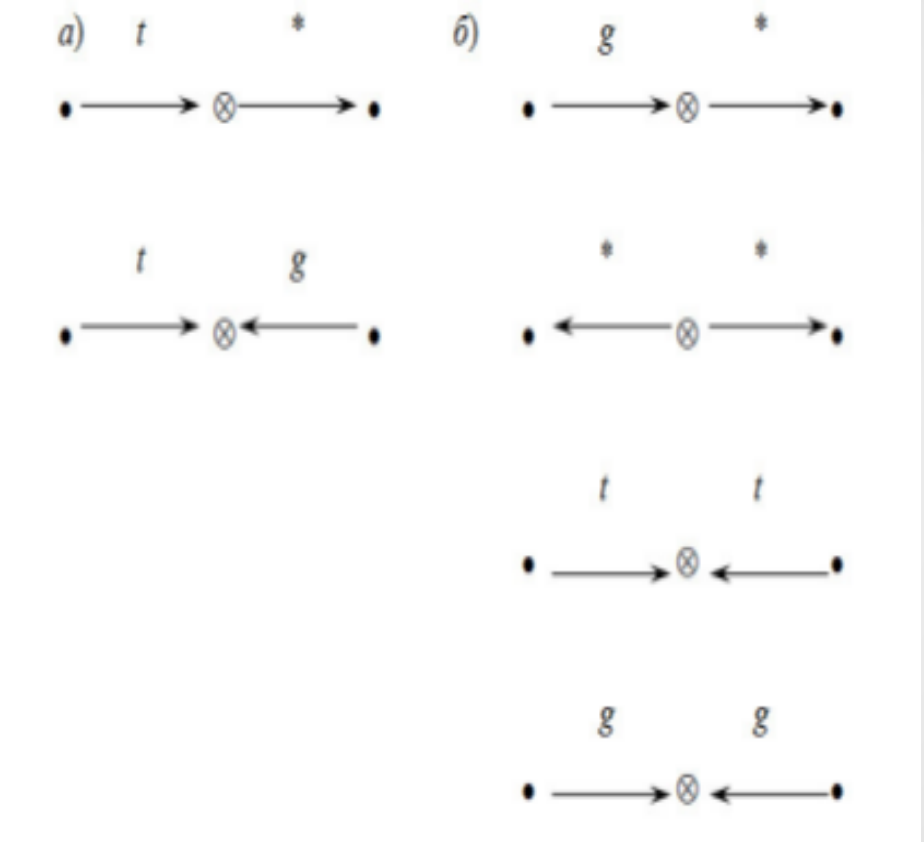
Этот T-образный мост эквивалентен:



Мосты

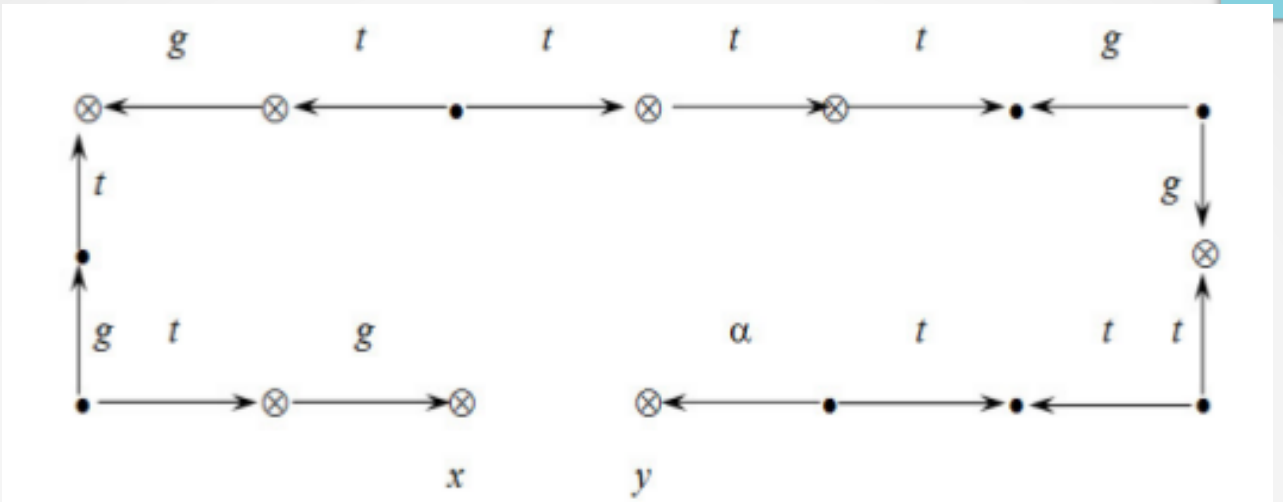
$$\vec{t}^{\,*}, \overleftarrow{t}^{\,*}, \vec{t}^{\,*}\vec{g}\overleftarrow{t}^{\,*}, \vec{t}^{\,*}\overleftarrow{g}\overleftarrow{t}^{\,*}$$

Рассмотрим все пути (с учетом симметрии) длины 2 между двумя субъектами, проходящие через вершины- объекты, по которым возможна передача прав доступа рис. а) и невозможна передача прав доступа рис. б). Любой путь, соединяющий двух субъектов, проходящий через объекты, по которому возможна передача прав доступа, не должен содержать фрагменты, приведенные на рис. б). В то же время очевидно, что любой такой путь, состоящий только из фрагментов, приведенных на рис. а), является мостом

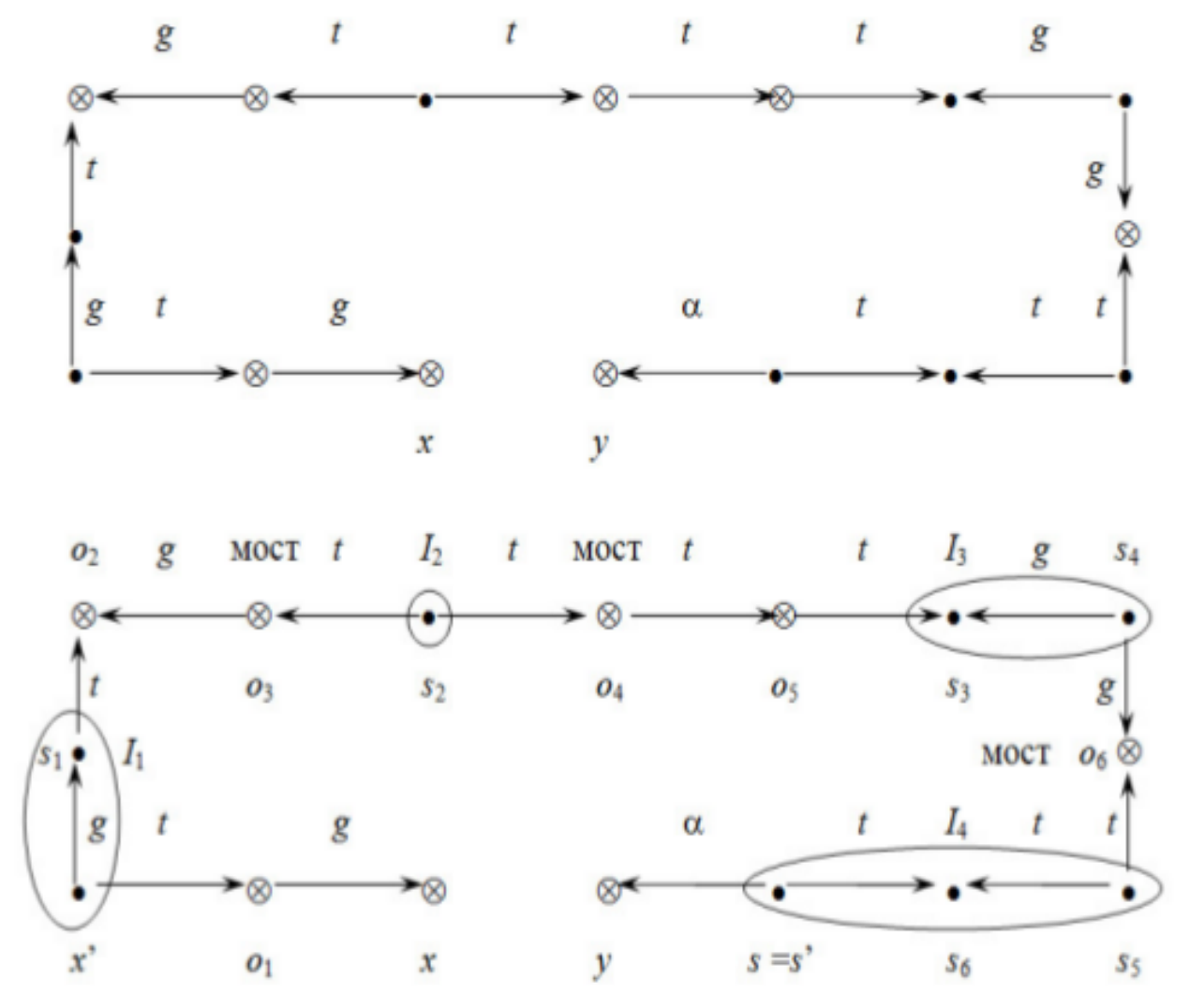


Виды путей длины 2
(∗ — или t, или g)

Мосты



Мосты


g t t t t g
t
g
g t g
α t t t
x
y
o2 g МОСТ t I2 t МОСТ t t I3 g s4
t
o3 s2 o4 o5 s3 g
s1 I1
МОСТ o6
g t g
α t I4 t t
x'
o1 x
y
s =s'
s6 s5

Теоретико-графовая модель TAKE-GRANT

Похищение прав доступа

Определение. *Для исходного состояния системы* $\Gamma_0 (O_0, S_0, E_0)$ *и прав доступа* $\alpha \subseteq R$ *предикат "***возможно похищение**(α, x, y, Γ_0)*" является истинным тогда и только тогда, когда существуют графы доступов системы* $\Gamma_1 (O_1, S_1, E_1)$, $\Gamma_2 (O_2, S_2, E_2)$, …, $\Gamma_N (O_N, S_N, E_N)$ *такие, что:*

$$\Gamma_0 (O_0, S_0, E_0) \vdash_{c_1} \Gamma_1 (O_1, S_1, E_1) \vdash_{c_2} \dots \vdash_{c_N} \Gamma_N (O_N, S_N, E_N) \text{ и } (x, y, \alpha) \in E_N$$

где $c_1, c_2, \dots, c_N$ *– команды переходов*;
при этом, если $\exists (s, y, \alpha) \in E_0$,
то $\forall z \in S_j, j=0,1,\dots, N$ *выполняется*: $c_1 \neq$ **grant**(α, s, z, y).

Теоретико-графовая модель TAKE-GRANT

Теорема.

В произвольном графе доступов $\Gamma_0 (O_0, S_0, E_0)$ *предикат* ***"возможно похищение(*** α***, x, y,*** Γ_0 ***)"*** *истинен тогда и только тогда, когда выполняются условия*:

- $(x, y, \alpha) \notin E_0$.
- *существуют субъекты* $s_1, \ldots, s_m$ *такие, что* $(s_i, y, \gamma_i) \in E_0$ *для* $i=1, \ldots, m$ *и* $\alpha = \gamma_1 \cup \ldots \cup \gamma_m$
- *являются истинными предикаты* ***"возможен доступ(t, x,*** s_i***,*** Γ_0***)"*** *для* $i=1, \ldots, m$.

Теоретико-графовая модель TAKE-GRANT

Если политика разграничения доступа в КС запрещает субъектам, имеющим в исходном состоянии права доступа к определенным объектам, непосредственно предоставлять эти права другим субъектам, которые изначально такими правами не обладают, то, тем не менее, такие первоначально "обделенные" субъекты могут получить данные права при наличии в графе доступов возможностей получения доступа с правом *t* к первым субъектам

Расширенная модель Take–Grant

В классической модели Take–Grant по существу рассматриваются два права доступа: t и g, а так же четыре правила (правила де-юре) преобразования графа доступов.

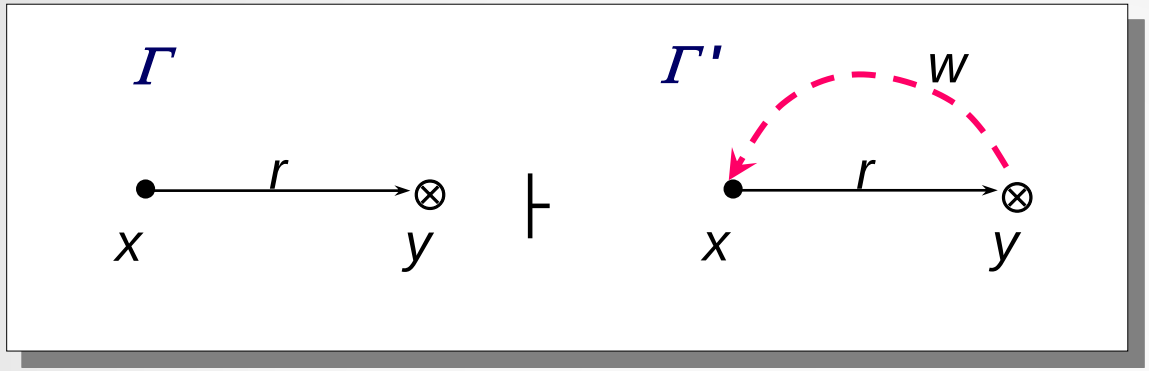
В расширенной модели дополнительно рассматриваются два права: на чтение r (read) и на запись w (write), а так же шесть правил (правила де-факто) преобразования графа доступов: pose, spy, find, pass и два правила без названия.

В результате применения к графу доступов правил де–факто в него добавляются мнимые дуги, помеченные r или w (и изображаемые пунктиром).

Вместе с дугами графа, соответствующими правам r и w (реальными дугами), мнимые дуги указывают на направления информационных каналов в системе.

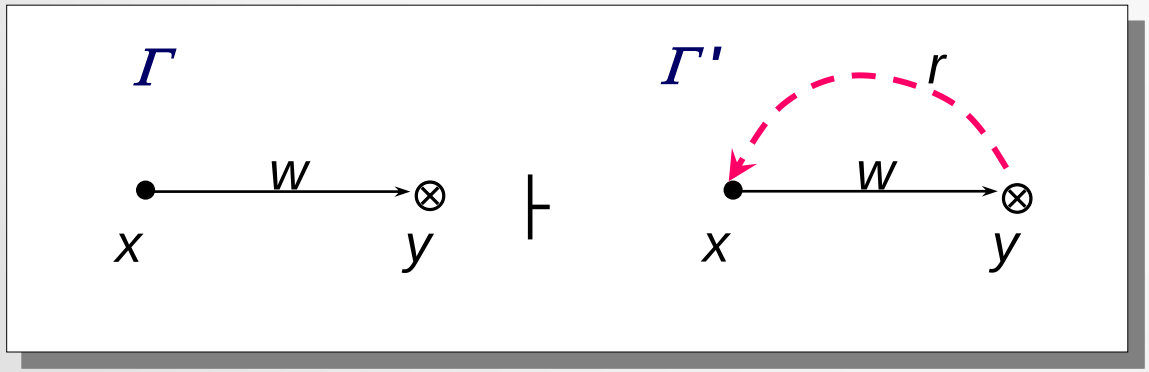
Расширенная модель Take–Grant

Команда без названия $\alpha_1(x, y)$



имеется неявная возможность передачи (записи) [конфиденциальной] информации из объекта ***y*** субъекту ***x***, когда тот осуществляет доступ ***r*** к объекту ***y***

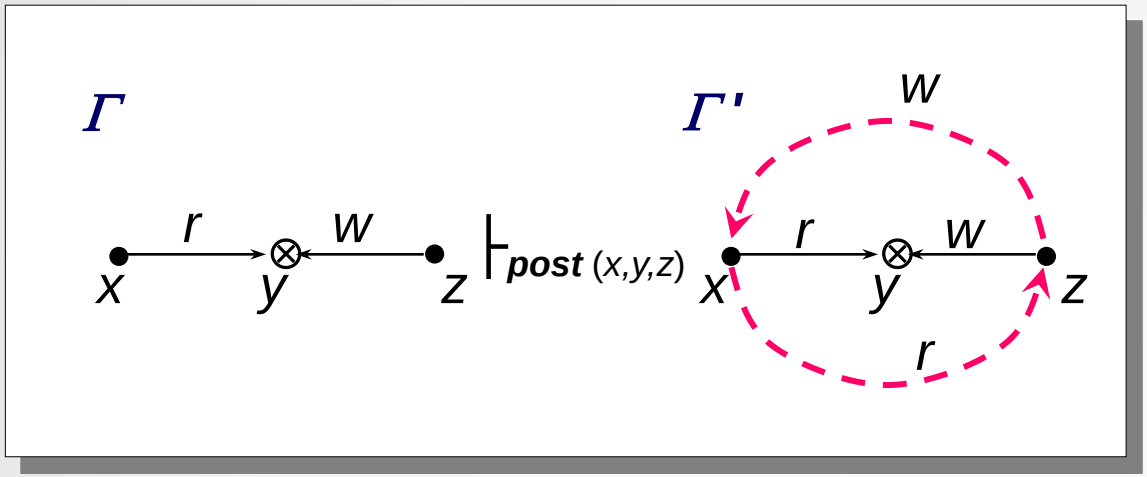
Команда без названия $\alpha_2(x, y)$



имеется неявная возможность получения (чтения) объектом ***y*** [конфиденциальной] информации от субъекта ***x***], когда тот осуществляет доступ ***w*** к объекту ***y***

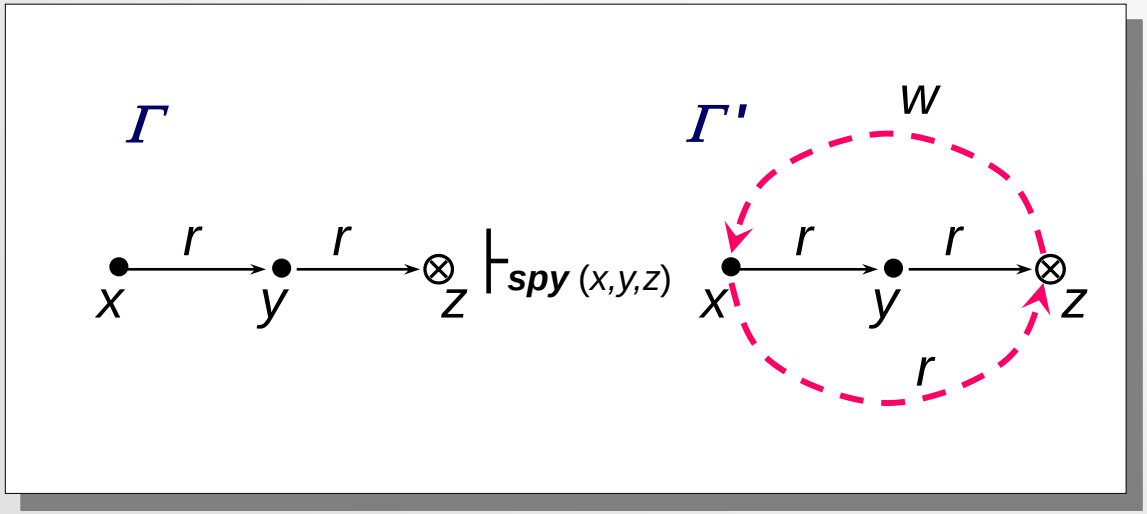
Расширенная модель Take–Grant

Команда *post*(*x, y, z*)



субъект ***x*** получает возможность чтения информации от (из) другого субъекта ***z***, осуществляя доступ ***r*** к объекту ***y***, к которому субъект ***z*** осуществляет доступ ***w***, а субъект ***z***, в свою очередь, получает возможность записи своей информации в субъект ***x***

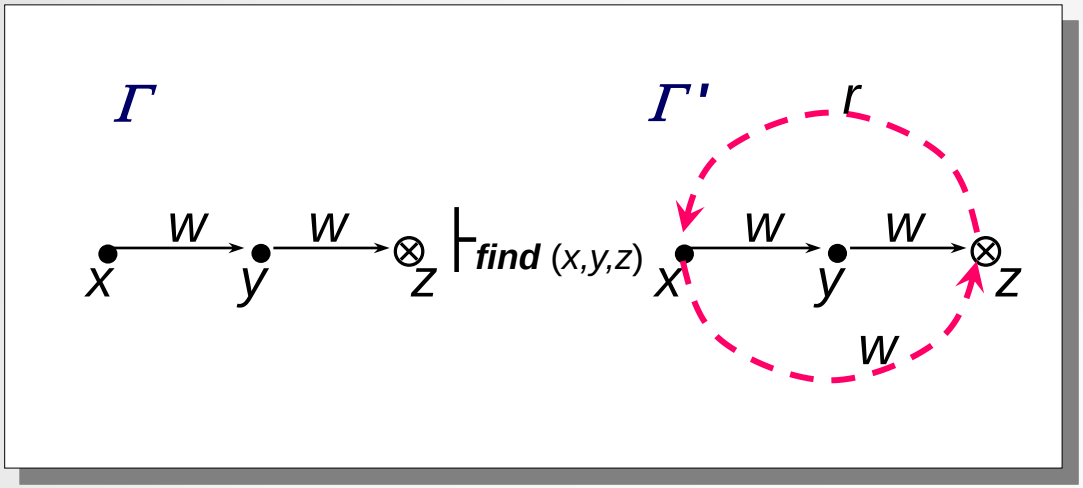
Команда *spy*(*x, y, z*)



субъект ***x*** получает возможность чтения информации из объекта ***z***, осуществляя доступ ***r*** к субъекту ***y***, который, в свою очередь, осуществляет доступ ***r*** к объекту ***z***, при этом также у субъекта ***x*** возникает возможность записи к себе информации из объекта ***z***

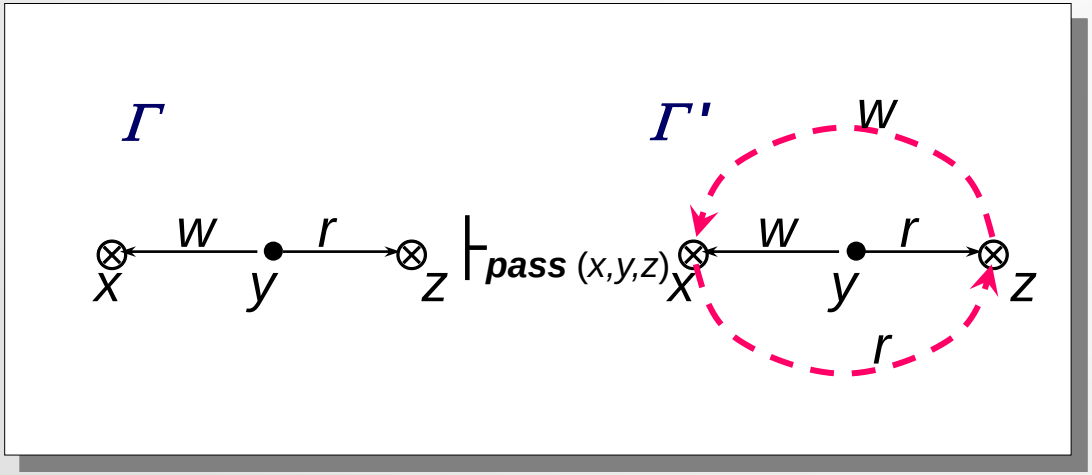
Расширенная модель Take–Grant

Команда ***find*(*x, y, z*)**



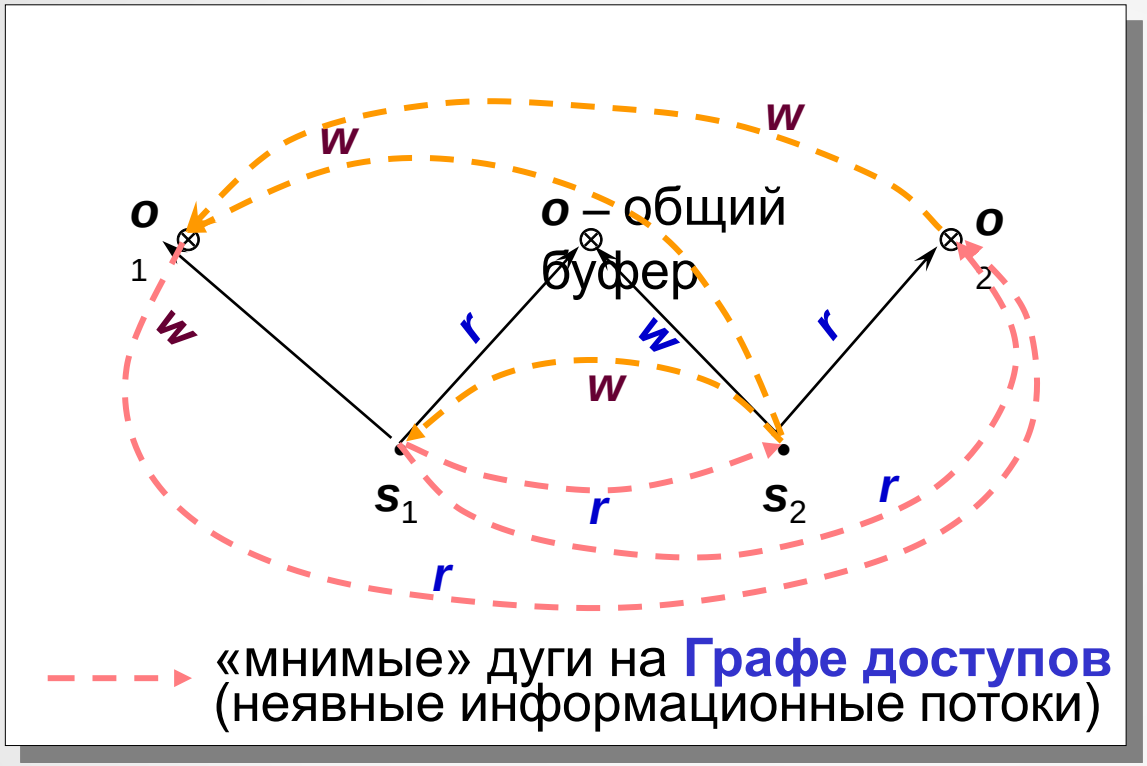
субъект ***x*** получает неявн. возможность передачи (записи) конф. информации в объект ***z***, осуществляя доступ ***w*** к субъекту ***y***, который, в свою очередь, осуществляет доступ ***w*** к объекту ***z***, при этом также у объекта ***z*** возникает неявн. возможность чтения конф. информации из субъекта ***x***

Команда ***pass*(*x, y, z*)**



при осуществлении субъектом ***y*** доступа ***r*** к объекту ***z*** возникает неявная возможность внесения из него конф. информации в другой объект ***x***, к которому субъект ***y*** осуществляет доступ ***w***, и, кроме того, возникает возможность получения информации (чтения) объектом ***x*** из объекта ***z***

Расширенная модель Take–Grant



Анализ возможности возникновения неявного информационного канала (потока) между двумя произвольными объектами (субъектами) ***x*** и ***y*** системы осуществляется на основе поиска и построения в графе доступов **пути** между ***x*** и ***y***, образованного **мнимыми дугами**, порождаемыми применением команд ***де-факто*** к различным фрагментам исходного Графа доступов

Расширенная модель Take–Grant

Расширенная модель TAKE-GRANT позволяет анализировать специфические проблемы в дискреционных системах разграничения доступа:

- - при допущении возможности или при наличии достоверных фактов о состоявшемся неявном информационном потоке от одного объекта(субъекта) к другому объекту(субъекту), анализировать и выявлять **круг возможных субъектов-"заговорщиков"** несанкционированного информационного потока

- для какой-либо пары объектов (субъектов) осуществлять анализ не только возможности неявного информационного потока, но и **количественных** характеристик по тому или иному маршруту:
- возможно взвешивание мнимых дуг на Графе доступов посредством оценки вероятности их возникновения
- возможны количественные сравнения различных вариантов возникновения неявного потока по длине пути на Графе доступов

- **оптимизировать** систему назначений доступа по критериям минимизации возможных неявных информационных потоков

Достоинства и недостатки дискреционных моделей

Достоинства дискреционных моделей

Хорошая гранулированность защиты (позволяют управлять доступом с точностью до отдельной операции над отдельным объектом)

Простота реализации

Недостатки дискреционных моделей

Слабые защитные характеристики из-за невозможности для реальных систем выполнять все ограничения безопасности

Проблема "троянских коней"

Сложности в управлении доступом из-за большого количества назначений прав доступа