ФГБОУ ВПО «Воронежский государственный
технический университет»

Кафедра систем информационной безопасности

336-2014

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к практическим занятиям № 1–4 по дисциплине
«Управление информационной безопасностью»
для студентов специальности
090303 «Информационная безопасность
автоматизированных систем»
очной формы обучения

Воронеж 2014

Составитель д-р техн. наук К. А. Разинкин

УДК 004.056.5

Методические указания к практическим занятиям № 1–4 по дисциплине «Управление информационной безопасностью» для студентов специальности 090303 «Информационная безопасность автоматизированных систем» очной формы обучения» / ФГБОУ ВПО «Воронежский государственный технический университет»; сост. К. А. Разинкин. Воронеж, 2014. 57 с.

Методические указания нацелены на привитие практических навыков управления информационной безопасностью на основе моделей дискреционного, мандатного и ролевого управления доступом, безопасности информационных потоков. В указаниях приведены основные теоретические положения и примеры решения типовых задач.

Методические указания подготовлены в электронном виде в текстовом редакторе и содержатся в файле Разинкин_ПЗ_УИБ_№1-4.pdf.

Табл. 4. Ил. 33. Библиогр.: 13 назв.

Рецензент д-р техн. наук, проф. А. Г. Остапенко

Ответственный за выпуск зав. кафедрой д-р техн. наук, проф. А. Г. Остапенко

Издается по решению редакционно-издательского совета Воронежского государственного технического университета

ВВЕДЕНИЕ

Управление информационной безопасностью (ИБ) - неотъемлемая часть управления любой современной организацией в целом, независимо от ее размера и сферы деятельности.

Управление ИБ - сложный непрерывный процесс, перед которым стоит множество целей и задач, являющихся обеспечивающими, вспомогательными по отношению к основным бизнес-целям и задачам организации. Они формулируются в различных документах организации: концепциях, стратегиях, политиках, стандартах, инструкциях и т. д.

Процесс управления ИБ распадается на тесно взаимосвязанные подпроцессы, каждый из которых вносит существенный вклад в достижение общих целей управления ИБ. Объектами управления в рамках этих подпроцессов являются активы, риски ИБ, инциденты ИБ, непрерывность бизнеса, изменения, усовершенствования и многое другое. От эффективности и результативности каждого из этих подпроцессов зависят общая эффективность и результативность всей деятельности по управлению ИБ в организации [5].

В методических указаниях рассмотрены формальные модели управления доступом и информационными потоками и их практические реализации в компьютерных системах (КС), создающие предпосылки для развития теории компьютерной безопасности и разработки новых эффективных методов анализа защищенности современных или перспективных КС, таких как операционных систем, СУБД, систем электронного документооборота и т.д. [2].

Практическое занятие № 1
Модель решётки

Теоретические положения

Пусть X — конечное множество.

Определение 1. Бинарное отношение «<» на множестве X назовем отношением строгого порядка, когда для любых $a,b,c \in X$ выполняются три свойства:

- антирефлексивность: не выполняется $a < a$;
- транзитивность: $(a < b, b < c) \Rightarrow (a < c)$;
- антисимметричность: не одновременно $a < b$ и $b < a$.

Определение 2. Бинарное отношение «≤» на множестве X назовем отношением частичного порядка, когда для любых $a,b,c \in X$ выполняются три свойства:

- рефлексивность: $a \le a$;
- транзитивность: $(a \le b, b \le x) \Rightarrow (a \le c)$;
- антисимметричность: $(a \le b, b \le a) \Rightarrow (a = b)$.

Определение 3. Для a, $b \in X$ элемент $c = a \oplus b \in X$ называется наименьшей верхней границей, когда выполняются условия:

- $a \le c, b \le c$; ;
- для $d \in X$ истинно $(a \le d, b \le d) \Rightarrow (c \le d)$.

Определение 4. Для a, $b \in X$ элемент $c = a \oplus b \in X$ называется наибольшей нижней границей, когда выполняются условия:

- $c \le a, c \le b$;
- для $d \in X$ истинно $(d \le a, d \le b) \Rightarrow (d \le c)$.

Для пары элементов частично упорядоченного множества X не обязательно существует наименьшая верхняя

(наибольшая нижняя) граница, но, если она существует, то из антисимметричности следует ее единственность.

Определение 5. Пусть X — частично упорядоченное множество. $\langle X, \leq \rangle$ называется решеткой, когда для любых $a, b \in X$ существуют $a \oplus b \in X$ и $a \otimes b \in X$.

Лемма 1. Для любого набора $S = \{a_1, a_2, ..., a_n\}$ элементов решетки $\langle X, \leq \rangle$ существуют единственные элементы:

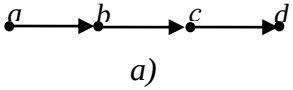
$\oplus S = a_1 \oplus a_2 \oplus ... \oplus a_n$ — наименьшая верхняя граница *S;*

$\otimes S = a_1 \otimes a_2 \otimes ... \otimes a_n$ — наибольшая нижняя граница *S.*

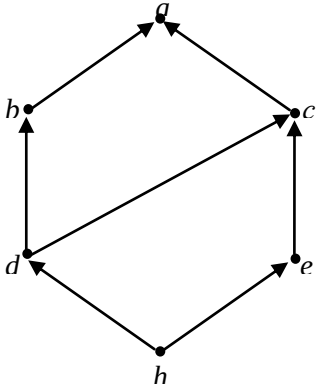
Для решетки $\langle X, \leq \rangle$ существует максимальный элемент *high*= $\oplus X$ и минимальный элемент *low* = $\otimes X$.

Определение 6. Линейная решетка (линейная шкала) из n элементов — это линейное упорядоченное множество; можно всегда считать $X = \{1, 2, ... n\}$.

Как правило, решетки представляют с помощью ориентированных графов (рис. 1). При этом вершинами графа являются элементы множества *X,* и для $a_1, a_2 \in X$ справедливо неравенство $a_1 \leq a_2$, когда в графе существует путь из a_1 в a_2.



а)



б)

Рис. 1. Пример решёток

Частным важным случаем решеток является решетка подмножеств некоторого конечного множества *U*.

Определение 7. Пусть U — конечное множество, $X = 2^U$ — множество всех подмножеств множества U. Определим решетку $(X, \leq)$ с бинарным отношением частичного порядка "$\leq$", где для $a, b \subseteq U, a, b \in X$ выполняется условие $a \leq b$ тогда и только тогда, когда $a \subseteq b$.

При этом $a \oplus b = a \cup b, \quad a \otimes b = a \cap b.$

Другим важным случаем решеток является решетка многоуровневой безопасности *(Multi Level Security — MLS)*. Данная решетка строится как прямое произведение линейной решетки *L* и решетки *X* подмножеств множества *U*.

Определение 8. Пусть $(L, \leq)$ — линейная решетка, $(X, \leq)$ — решетка подмножеств U. Определим решетку многоуровневой безопасности $(X \times L, \leq)$ с бинарным отношением частичного порядка «$\leq$», где для $(a, \alpha), (b, \beta) \in X \times L$ выполняется условие $(a, \alpha) \leq (b, \beta)$ тогда и только тогда, когда $a \subseteq b, \alpha \leq \beta$.

При этом

$$(a, \alpha) \oplus (b, \beta) = (a \cup b, \max(\alpha, \beta));$$
$$(a, \alpha) \otimes (b, \beta) = (a \cap b, \min(\alpha, \beta));$$

На практике при использовании решеток многоуровневой безопасности решетка $(L, \leq)$ является линейной шкалой уровней конфиденциальности, а $(X, \leq)$ — решеткой подмножеств множества неиерархических категорий информации.

Типовые задачи

Задание 1. Задает ли решетку граф на рис. 2?

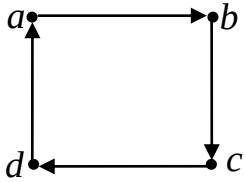


Рис. 2. Граф к заданию 1

Решение. Так как выполняются условия $a \le b, b \le c$,и $a \ne b$, то в соответствии с *определением 2* не выполняется свойство антисимметричности отношения частичного порядка «≤» на множестве {*a,b,c,d*}. Следовательно, по *определению 5* граф не задает решетку.

Задание 2. Задаёт ли решётку граф на рис. 3?

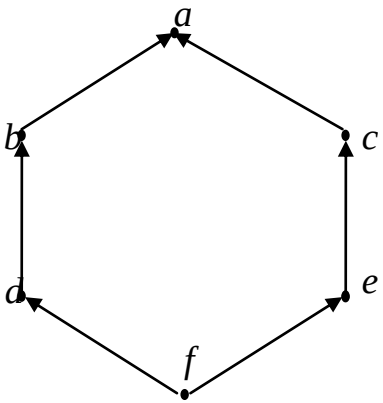


Рис. 3. Граф к заданию 2

Решение. В соответствии с *определением 2* выполнены все свойства отношения частичного порядка «≤» на множестве *{a, b,* c, *d,* e, *f}*. Для каждой пары вершин, соединенных в графе путем, существует наименьшая верхняя и наибольшая нижняя границы. Например, справедливы

равенства $f \oplus b = b$ и $f \otimes b = f$. Для каждой пары, не соединенных в графе путем, приведем значения наименьших верхних и наибольших нижних границ:

$$d \oplus e = a, d \otimes e = f;$$

$$b \oplus e = a, b \otimes e = f;$$

$$d \oplus c = a, d \otimes c = f;$$

$$b \oplus c = a, b \otimes c = f.$$

Следовательно, по определению 5 граф задает решетку.

Задание 3. Нарисуйте граф, соответствующий решетке многоуровневой безопасности $(X \times L, \leq)$, для решетки $(L, \leq)$ = *{Low, Middle,High}* и $(X, \leq)$ — решетки подмножеств множества *U = = {Political, Military}.*

Решение. Построим граф, задающий решетку $(X \times L, \leq)$ (рис. 5). При этом используем сокращения: *P {Political), M (Military).*

Задачи для самостоятельного решения

1. Задают ли графы на рис. 4 решётку

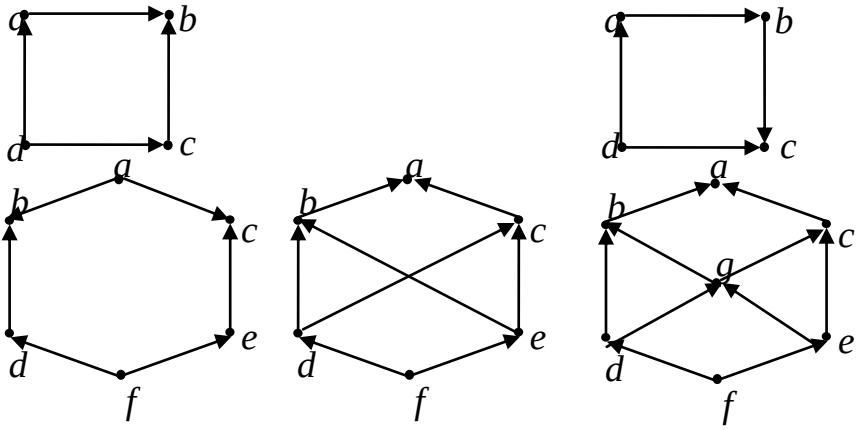


Рис. 4. Графы к задаче для самостоятельного решения

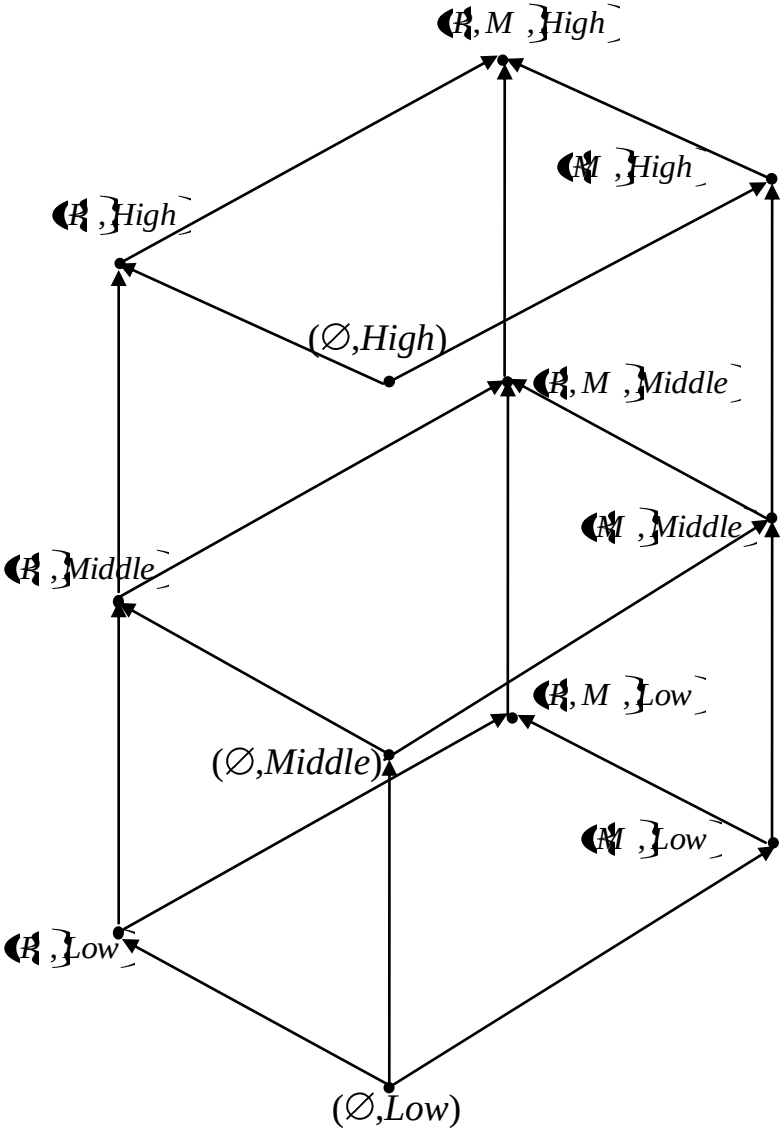


Рис. 5. Граф, илюстрирующий решение задания 3

Практическое занятие № 2
Дискреционное управлением доступом (модели Харрисона-Руззо-Ульмана и типизированная матрицы доступов)

Теоретические положения

Модель Харрисона-Руззо-Ульмана (ХРУ) используется для анализа систем защиты, реализующих дискреционную политику управления доступом.

В модели ХРУ используются следующие обозначения:

O — множество объектов системы (сущности-контейнеры в модели ХРУ не рассматриваются);

S — множество субъектов системы $S \subseteq O$;

R, — множество видов прав доступа субъектов к объектам, например права на чтение *(read),* на запись *(write),* владения *(own);*

M — матрица доступов, строки которой соответствуют субъектам, а столбцы соответствуют объектам. $M[s,o] \in R$ — права доступа субъекта *s* к объекту *o.*

Определение 1. Автомат, построенный согласно описанию модели ХРУ, назовем системой ХРУ.

Функционирование системы рассматривается только с точки зрения изменений в матрице доступа. Возможные изменения определяются шестью видами примитивных операторов, представленных в табл. 1.

В результате выполнения примитивного оператора *a* осуществляется переход из состояния $q = (S, O, M)$ в результирующее состояние *q' = (S',0',M').* Данный переход обозначим через $q \vdash_{\alpha} q'$.

Из примитивных операторов составляется конечное число команд системы ХРУ. Каждая команда включает две

части: 1) условия, при которых выполняется команда; 2) последовательность примитивных операторов.

Таблица 1

Примитивные модели ХРУ

Примитивный оператор	Исходное состояние $q = (S,O,M)$	Результирующее состояние $q' = (S',O',M')$
«внести» право r в $M[s,o]$	$s \in S;$ $o \in O;$ $r \in R$	$S' = S, O' = o; M'[s,o] = M[s,o] \cup r;$ для $(s',o') \neq (s,o)$ выполняется равенство $M'[s',o'] = M[s',o']$
«удалить» право r из $M[s,o]$	$s \in S;$ $o \in O;$ $r \in R$	$S' = S, O' = o; M'[s,o] = M[s,o] \setminus r;$ для $(s',o') \neq (s,o)$ выполняется равенство $M'[s',o'] = M[s',o']$
«создать» субъект s'	$s' \notin O$	$S' = S \cup \{s'\}; O' = O \cup \{s'\}$ для $(s,o) \in S \times O$ выполняется равенство $M'[s,o] = M[s,o];$ для $o \in O'$ выполняется равенство $M'[s',o] = \varnothing;$ для $s \in S'$ выполняется равенство M'[s, s'] = $\varnothing$
«создать» объект o'	$o' \in O$	$S' = S; O' = O \cup \{o'\}$ для $(s,o) \in S \times O$ выполняется равенство $M'[s,o] = M[s,o];$ для $s \in S'$ выполняется равенство $M'[s,o'] = \varnothing$
«уничтожить» субъект s'	$s' \in S$	$S' = S \setminus \{s'\}; O' = O \setminus \{s'\};$ для $(s,o) \in S' \times O'$ выполняется равенство $M'[s,o] = M[s,o]$
«уничтожить» объект o'	$o' \in O$ $o' \in S$	$S' = S; O' = O \setminus \{o'\}$ для (s, o) $\in S' \times O'$ выполняется равенство $M'[s,o] = M[s,o]$

Таким образом, запись команды имеет следующий вид:

$$command\ c(x_1,...,x_k)$$

$$if\ (r_1 \in M[s_1, x_{0_1}])\ and...and\ (r_m \in M[s_m, x_{o_m}])\ then$$

$$\alpha_1;$$

$$.....$$

$$\alpha_n$$

$$endif$$

$$end$$

где $r_1,...,r_m \in R$ — права доступа; $\alpha_1,...,\alpha_n$ — последовательность примитивных операторов, параметрами которых, а также параметрами условий, являются параметры команды $x_1,...,x_k$. Следует отметить, что наличие условия в теле команды не является обязательным. При выполнении команды $c(x_1,...,x_k)$ система осуществляет переход из состояния q в новое состояние q'. Данный переход обозначим как

$$q \vdash_{c(x_1,...,x_k)} q',$$

при этом $q' = q$, когда одно из условий команды $c(x_1,...,x_k)$ не выполнено, $q'=q_n$, когда условие команды выполнено и существуют состояния $q_1,...,q_n$ такие, что

$$q = q_0 \vdash_{\alpha_1} q_1 \vdash_{\alpha_2} \ldots \vdash_{\alpha_n} q_n.$$

Пример 1. Команда создания субъектом *s* личного файла *f*:

command CreateFile(s, f)
«создать» объект *f*;
«внести» право владения *own* в $M[s, f]$;
«внести» право на чтение *read* в $M[s, f]$
«внести» право на запись *write* в $M[s, f]$
end.

Пример 2. Команда передачи субъекту S' права *read* к файлу *f* его владельцем субъектом *s:*

command GrantRead(s, s', f)
if (own $\in M[s, f]$*) then*
«внести» право *read в* $M[s', f]$;
endif
end.

Другая дискретиционная модель, получившая название типизированной матрицы доступов (ТМД), представляет собой развитие модели ХРУ, дополненной концепцией типов, что позволяет несколько смягчить те условия, для которых возможно доказательство безопасности системы.

Формальное описание модели ТМД включает в себя следующие элементы:

O — множество объектов системы;

S — множество субъектов системы $S \subseteq O$;

R — множество прав доступа субъектов к объектам;

M — матрица доступов;

C — множество команд;

T — множество типов объектов;

$t: O \to T$ - функция, ставящая в соответствие каждому объекту его тип;

$q = (S, O, t, M)$ — состояние системы;

Q — множество состояний системы.

Состояния системы изменяются в результате применения к ним команд из множества *C*. Команды в модели ТМД имеют тот же формат что и в модели ХРУ, при этом для всех параметров команд указывается их тип:

$$command\ c(x_1 : t_1, ..., x_k : t_k)$$
$$if\ (r_1 \in M[s_1, x_{o_1}])\ and\ ...\ and\ (r_m \in M[s_m, x_{o_m}])\ then$$

α_1;

.....

α_n

endif

end

Перед выполнением команды происходит проверка типов фактических параметров, и, если они не совпадают с указанными в определении команды, то команда не выполняется. В модели ТМД используются шесть видов примитивных операторов, отличающихся от аналогичных операторов модели ХРУ только использованием типизированных параметров (табл. 2).

Таким образом, ТМД является обобщением модели ХРУ, которую можно рассматривать как частный случай ТМД с одним единственным типом для всех объектов и субъектов. С другой стороны, любую систему ТМД можно выразить через систему ХРУ, введя для обозначения типов специальные права доступа, а проверку типов в командах заменив проверкой наличия соответствующих прав доступа.

Определение 2. Пусть $c(x_1 : t_1,...,t_k : t_k)$ — некоторая команда системы ТМД. Будем говорить, что x_i является дочерним параметром, а t_i является дочерним типом в $c(x_1 : t_1,...,t_k : t_k)$, где $1 \le i \le k$, в случае, когда в ней имеется один из следующих примитивных операторов:

- «создать» субъект x_i с типом t_i;
- «создать» объект x_i *с* типом x_i.

В противном случае будем говорить, что x_i является родительским параметром, а t_i является родительским типом в команде $c(x_1 : t_1,...,t_k : t_k)$.

Заметим, что в одной команде тип может быть одновременно и родительским, и дочерним. Например:

command $foo(s_1 : u, s_2 : u, s_3 : v, o_1 : w, o_2; b)$

«создать» субъект s_2 с типом *u;*

«создать» субъект s_3 с типом *v;*

end

Здесь *u* является родительским типом относительно s_1 дочерним типом относительно s_2. Кроме того, *w* и *b* являются родительскими типами, а *v* — дочерним типом.

Появление в каждой команде дополнительных неявных условий, ограничивающих область применения команды только объектами соответствующих типов, позволяет несколько смягчить жесткие условия модели ХРУ, при которых критерий безопасности является разрешимым.

Определение 3. Система монотонной ТМД (МТМД) — система ТМД, в командах которой отсутствуют немонотонные примитивные операторы вида «удалить»... и «уничтожить»...

Определение 4. Каноническая форма системы МТМД (КФ МТМД) — система МТМД, в которой команды, содержащие примитивные операторы вида «создать»..., не содержат условий и примитивных операторов вида «внести»…

Определение 5. Граф создания системы МТМД — ориентированный граф с множеством вершин *T*, в котором ребро от вершины *u* к вершине *v* существует тогда и только тогда, когда в системе имеется команда, в которой *u* является родительским типом, а υ — дочерним типом.

Граф создания для каждого типа позволяет определить:

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

Таблица 2

Примитивные операторы модели ТМД

Примитивный оператор	Исходное состояние $q = (S,0,t,M)$	Результирующее состояние $q' = (S', O', t', M')$
«внести» право r в $M[s,o]$	$s \in S$ $o \in O$ $r \in R$	$S' = S; O' = O; t' = t$; $M'[s,o] = M[s,o] \cup r$; для (s', o') выполняется равенство $M'[s',o'] = [s',o']$
«удалить» право r из $M[s,o]$	$s \in S$ $o \in O$ $r \in R$	$S' = S; O' = O; t' = t$; $M'[s,o] = M[s,o]\backslash\{r\}$; для $(s', o') \neq (s,o)$ выполняется равенство $M'[s',o'] = M[s',o']$
«создать» субъект s' с типом t_s.	$s' \notin O$	$S' = S \cup \{s'\}; O' = O \cup \{s'\}$; для $o \in O$ выполняется равенство $t'(o) = t(o)$; $t'(s') = t_s$ для $(s,o) \in S \times O$ выполняется равенство $M'[s',o'] = [s,o]$ для $o \in O'$ выполняется равенство $M'[s',o] = \varnothing$; для $s \in S'$ выполняется равенство $M'[s, s'] = \varnothing$
«создать» объект o' с типом t_o	$o' \notin O$	$S' = S; O' = O \cup \{o'\}$; для $o \in O$ выполняется равенство $t'(o) = t(o)$; $t'(o') = t_0$; для $(s, o) \in S \times O$ выполняется равенство $M'[s,o] = M[s,o]$; для $s \in S'$ выполняется равенство $M'[s,o'] = \varnothing$
« уничтожить » субъект s'	$s' \in S$	$S' = S \setminus \{s'\}; O' = O \setminus \{s'\}$; для $o \in O'$ выполняется равенство $t'(o) = t(o)$; для (s, o) $S' \times O'$ выполняется равенство $M'[s,o] = M[s,o]$

Продолжение табл. 2

Примитивный оператор	Исходное состояние q = (S,0,t,M)	Результирующее состояние q' = (S', O' ,t', M')
«уничтожить» объект o'	$o' \notin O$; $s' \in S$	$S' = S; O' = O \setminus \{o'\}$; для $o \in O'$ выполняется равенство $t'(o) = t(o)$; для (s, o) $S' \times O'$ выполняется равенство $M'[s,o] = M[s,o]$

Определение 6. Система МТМД (КФ МТМД) называется ациклической (АМТМД или, соответственно, АКФМТМД) тогда и только тогда, когда ее граф создания не содержит циклов; в противном случае говорят, что система является циклической.

Типовые задачи

Задание 1. Докажите, что для общего случая систем ХРУ не существует алгоритма проверки возможности утечки права доступа *r* для заданной пары субъект *s* и объект *o*.

Решение. Пусть задана система ХРУ, в которой определены множества *R,Q,C.* Построим эквивалентную ей систему ХРУ, определив множества R^*, Q^*, C^*.

Пусть в каждом состоянии $q^* = (S^*, O^*, M^*)$ соответствующем состоянию $q = (S,0,M)$, справедливы равенства:

$$S^* = S \cup \{s_{own}\};$$

$$O^* = O \cup \{s_{own}\};$$

$$R^* = R \cup \{own, noown\}.$$

Кроме того, для $s \in S, o \in O$ справедливы равенства $M^*[s, o] = M[s, o]$, $M^*[s_{own}, o] = \{noown\}$, $M^*[s_{own}, s_{own}] = \{own\}$.

Каждую команду $c(x_1,...,x_k) \in C$ исходной системы ХРУ заменим командой $c(x_1,...,x_k,x)$, которая удовлетворяет следующим условиям:

- условие команды $c(x_1,...,x_k,x)$ содержит все условия команды $c(x_1,...,x_k)$ и дополнительные условия: $noown \in M^*[s,x_i], i=1,...,k$ и $own \in M^*[s,x]$;
- команда $c(x_1,...,x_k,x)$ содержит все примитивные операторы команды $c(x_1,...,x_k)$
- если команда $c(x_1,...,x_k)$ содержит примитивный оператор вида «создать» субъекта x_i или «создать» объект x_i, где $1 \le i \le k$, то после него в команду $c(x_1,...,x_k,x)$ добавляется примитивный оператор «внести» право *noown в* $M^*[s,x_i]$;
- если команда $c(x_1,...,x_k)$ содержит примитивный оператор вида «внести» право r в $M^*[s,x_j]$, где $1 \le i \le k$, $1 \le j \le k$, то после него в команду $c^*(x_1,...,x_k,x)$ добавляется примитивный оператор «внести» право r в $M^*[s,x]$.

Таким образом, утечка права доступа r в исходной системе происходит тогда и только тогда, когда в ячейке $M^*[s_{own},s_{own}]$ эквивалентной ей системы появляется данное право доступа. Следовательно, если бы существовал алгоритм проверки возможности утечки права доступа r для заданной пары субъект s и объект o (в данном случае для пары s_{own},s_{own}), то существовал бы алгоритм проверки безопасности произвольных систем ХРУ.

Задание 2. Представьте произвольную систему ТМД системой ХРУ.

Решение. Пусть задана система ТМД, в которой определены множества *R,Q,T,C*. Построим эквивалентную ей

систему ХРУ, определив множества R^*, Q^*, C^*.

Пусть в каждом состоянии $q^* = (S^*, O^*, M^*)$ системы ХРУ, соответствующем состоянию $q = (S,0,t,M)$, справедливы равенства:

$$S^* = S \cup \{s_{own}\};$$

$$O^* = O \cup \{s_{own}\};$$

$$R^* = R \cup T \cup \{own\}.$$

Кроме того, для $s \in S$ справедливы равенства

$$M^*(s_{own}, o) = \{t\}, M^*(s_{own}, s_{own}) = \{own\}.$$

Для каждой команды $c(x_1 : t_1, ..., x_k : t_k) \in C$ системы ТМД в систему ХРУ добавим команду $c^*(x_1, ..., x_k, x)$, которая удовлетворяет следующим условиям:

- условие команды $c^*(x_1, ..., x_k, x)$ содержит все условия команды $c(x_1 : t_1, ..., x_k : t_k)$ и дополнительные условия: $t_i \in M^*(x, x_i)$, где $i = 1,2,...k$ и $own \in M^*(x, x)$;
- команда $c^*(x_1, ..., x_k, x)$ содержит все примитивные операторы вида «внести»..., «удалить»..., «уничтожить»... команды $c(x_1 : t_1, ..., x_k : t_k)$;
- если команда $c(x_1 : t_1, ..., x_k : t_k)$ содержит примитивный., оператор вида «создать» субъект x_i с типом t_i или «создать» объект x_i с типом t_i, где $1 \le i \le k$, то в команду $c^*(x_1, ..., x_k, x)$ добавляются примитивные операторы «создать» субъект x_i или «создать» объект x_i соответственно и примитивный оператор «внести» право t_i в $M^*(x, x_i)$.

Таким образом, строится система ХРУ, эквивалентная системе ТМД.

Задание 3. Постройте граф создания для системы МТМД со следующим набором команд:

command $a1(x:\alpha, y:\beta, z:\beta)$

«создать» субъект *x* с типом *a*

end;

command $a2(x:\alpha, y:\gamma, z:\beta, s:\delta)$

«создать» объект *y* с типом γ;

«создать» субъект *s* с типом δ;

end;

command $a3(x:\varepsilon, y:\delta, z:\beta, s:\gamma, o:\delta)$

«создать» субъект *o* с типом δ;

«создать» объект *x* с типом ε;

end.

Является ли данная система ациклической?

Решение. В соответствии с определением 5 строим граф создания. Вершинами графа являются типы $T = \{\alpha, \beta, \gamma, \delta, \varepsilon\}$.

В команде $a1(x:\alpha, y:\beta, z:\beta)$ родительским типом является β, дочерним — α. Следовательно, добавляем в граф ребро (β, α).

В команде $a2(x:\alpha, y:\gamma, z:\beta, s:\delta)$ родительскими типами являются α и β, дочерними — γ и δ. Следовательно, добавляем в граф ребра (α, γ), (α, δ), (β, γ) и (β, δ).

В команде $a3(x:\varepsilon, y:\delta, z:\beta, s:\gamma, o:\delta)$ родительскими типами являются β, γ и δ, дочерними — δ и ε. Следовательно, добавляем в граф ребра (β, δ), (β, ε), (γ, δ), (γ, ε), (δ, δ) и (δ, ε).

Получаем граф создания, приведенный на рис. 6.

Система не является ациклической, так как граф создания содержит цикл (δ, δ).

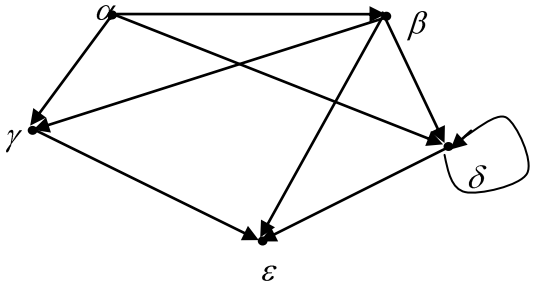


Рис. 6. Граф, илюстрирующий решение задания 3

Задачи для самостоятельного решения

1. Для команды $C(s_{i_0}, a_{i_0}) = (s_{i_1}, a_{i_1}, l)$ машины Тьюринга выпишите две представляющие ее команды модели ХРУ.

2. Постройте графы создания для систем МТМД со следующими наборами команд.

command $a1(x:\alpha, y:\beta, z:\gamma)$
«создать» субъект *y* с типом β;
«создать» субъект *x* с типом α;
end
command $a2(x:\beta, y:\delta, z:\delta)$
«создать» субъект *z* с типом δ;
end
command $a3(x:\varepsilon, y:\alpha, z:\delta)$
«создать» объект *x* с типом ε;
end.

Практическое занятие № 3
Управление распространением прав доступа на основе классической модели Take-Grant

Теоретические положения

Классическая модель *Take- Grant* ориентирована на анализ путей распространения прав доступа в системах дискреционного управления доступом. Классическую модель *Take-Grant* будем рассматривать на основе [3].

Основными элементами модели *Take- Grant* являются: O — множество объектов; $S \subseteq O$ — множество субъектов;

$R = \{r_1, r_2, ..., r_m\}$ — множество видов прав доступа, где *t(take)* — право брать права доступа, *g(grant)* — право давать права доступа;

$G = (S, O, E)$ — конечный помеченный ориентированный без петель граф доступов, описывающий состояние системы. Элементы множеств S и O являются вершинами графа, которые будем обозначать ⊗ — объекты (элементы множества $O \backslash S$) и • — субъекты (элементы множества S) соответственно. Элементы множества $E \subseteq O \times O \times R$ являются ребрами графа. Каждое ребро помечено непустым подмножеством множества видов прав доступа *R*.

Состояние системы описывается соответствующим ему графом доступов. В отличие от модели ХРУ в модели *Take-Grant* возможно наличие прав доступа не только у субъектов к объектам, но и у объектов к объектам.

Основная цель классической модели *Take- Grant* — определение и обоснование алгоритмически проверяемых условий проверки возможности утечки права доступа по исходному графу доступов, соответствующего некоторому состоянию системы.

Порядок перехода системы модели *Take-Grant* из состояния в состояние определяется правилами

преобразования графа доступов, которые в классической модели носят название де-юре правил. Преобразование графа G в граф G' в результате выполнения правила op обозначим через $G \vdash_{op} G'$.

В классической модели *Take-Grant* рассматриваются четыре де-юре правила преобразования графа, выполнение каждого из которых может быть инициировано только субъектом, являющимся активной компонентой системы (рис. 7-10): *take* — брать права доступа; *grant* — давать права доступа.

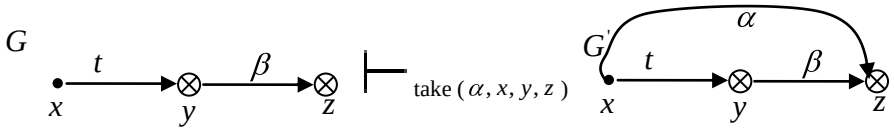


Рис. 7. Применение правила *take* (α, x, y, z)

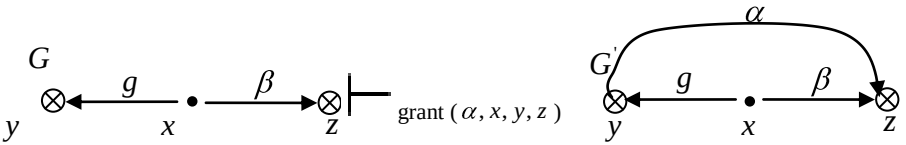


Рис. 8. Применение правила *grant* (α, x, y, z)

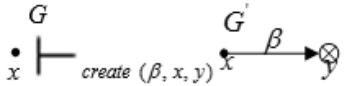


Рис. 9. Применение правила create(β, x, y)

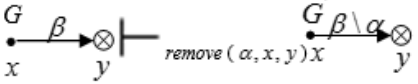


Рис. 10. Применение правила remove(α, x, y)

create — создавать новый объект или субъект, при этом субъект создатель может взять на созданный субъект любые права доступа (по умолчанию предполагается, что при выполнение правила *create* создается объект, случаи, когда создается субъект, оговариваются особо); *remove* — удалять права доступа.

Условия применения де-юре правил в исходном состоянии $G = (S, O, E)$ и результаты их применения в результирующем состоянии $G' = (S', O', E')$ приведены в табл. 3.

Таблица 3

Де-юре правила классической модели *Take-Grant*

Правила	Исходное состояние $G=(S,O,E)$	Результирующее состояние $G' = (S',0',E')$
take(α ,x,y,z)	$x \in S;\ y, z \in O;\ (x, y, t) \subseteq E,$ $(y, z, \beta) \subseteq E; x \neq z; \alpha \subseteq \beta$	$S' = S; O' = O;$ $E' = E \cup (x, z, \alpha)$
grant(α ,x,y,z)	$x \in S; y, z \in O; (x, y, g) \subseteq E;$ $(x, z, \beta) \subseteq E; y \neq z; \alpha \subseteq \beta$	$S' = S; O' = O;$ $E' = E \cup (y, z, \alpha)$
create(β ,x,y)	$x \in S; y \notin o; \beta \neq \varnothing$	$O' = O \cup \{y\};$ если y субъект, то $S' = S \cup \{y\},$ иначе $S' = S,$ $E' = E \cup (x, y, \beta)$
remove(α ,x,y)	$x \in S; y \in O, (x, y, \beta) \subseteq E;\ \alpha \subseteq \beta$	$S' = S; O' = O;$ $E' = E \setminus (x, y, \alpha)$

В модели *Take- Grant* основное внимание уделяется определению условий, при которых в системе возможно распространение прав доступа для случая, когда не рассматриваются ограничения на кооперацию субъектов при передаче прав доступа, и случая, когда на кооперацию субъектов наложены ограничения.

Данный случай характеризуется тем, что что при передаче прав доступа не накладывается ограничений на кооперацию субъектов системы, участвующих в этом процессе.

Определение 1. Пусть $x, y \in O_0, x \neq y$ — различные объекты графа доступов $G_0 = (S_0, O_0, E_0, \alpha \subseteq R$.. Определим предикат *can_share(*α*,x,y,*G_0*)*, который будет истинным тогда и только тогда, когда существуют графы $G_1 = (S_1, O_1, E_1, \alpha), .., G_N = (S_N, O_N, E_N, \alpha)$ и правила $op_1, ..., op_N$, где $N \geq 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ и $(x, y, \alpha) \subseteq E_N$.

Определение истинности предиката *can_share(*α*,x,y,*G_0*)* непосредственно по определению является в общем случае алгоритмически неразрешимой задачей, так как требует проверки всех траекторий функционирования системы. По этой причине для проверки истинности предиката *can_share(*α*,x,y,*G_0*)* следует определить необходимые и достаточные условия, проверка которых возможна. Решение данной задачи будет выполнено в два этапа. На первом этапе будут определены и обоснованы условия истинности предиката *can_share(*α*,x,y,*G_0*)* для графов, все вершины которых являются субъектами, на втором этапе условия истинности предиката *can_share(*α*,x,y,*G_0*)*будут определены и обоснованы для произвольных графов.

Определение 2. Пусть *G = (S,S,E)* — граф доступов, все вершины которого являются субъектами. Говорят, что вершины графа доступов являются *tg*-связными или что они соединены *tg* -путем, когда, без учета направления ребер, в графе между ними существует путь такой, что каждое ребро этого пути помечено *t* или *g.*

Теорема 1. Пусть $G_0 = \langle S_0, S_0, E_0 \rangle$ — граф доступов, содержащий только вершины субъекты, $x, y \in S_0, x \neq y$. Тогда предикат *can_share(*α*,x,y,*G_0*)* истинен тогда и только тогда, когда выполняются *условия 1* и *2*.

Условие 1. Существуют субъекты $s_1,...,s_m \in S_0 : \langle s_i, y, \gamma_i \rangle \subseteq E_0$, где $i = 1,...,m$ и $\alpha = \gamma_1 \cup ... \cup \gamma_m$.

Условие 2. Субъекты x и s_i являются *tg*-связными в графе G_0, где $i = 1,...,m$.

Пусть N = 1. Тогда существует $\langle s, y, \alpha \rangle \subseteq E_0$ и x и s соединены ребром t или g в графе G_0. Возможны четыре случая такого соединения x и s, для каждого из которых, указана последовательность преобразований графа, требуемая для передачи прав доступа (рис. 11-14).

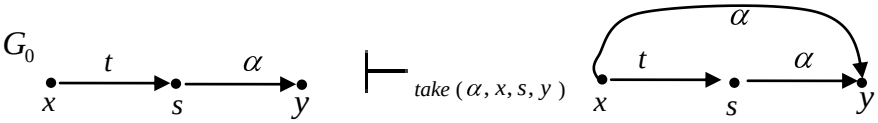


Рис. 11. Первый случай

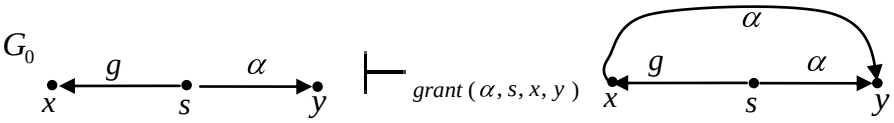


Рис. 12. Второй случай

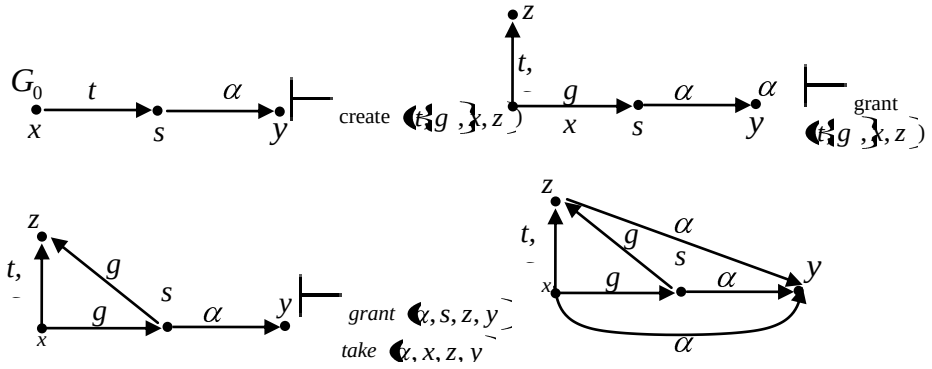


Рис. 13. Третий случай

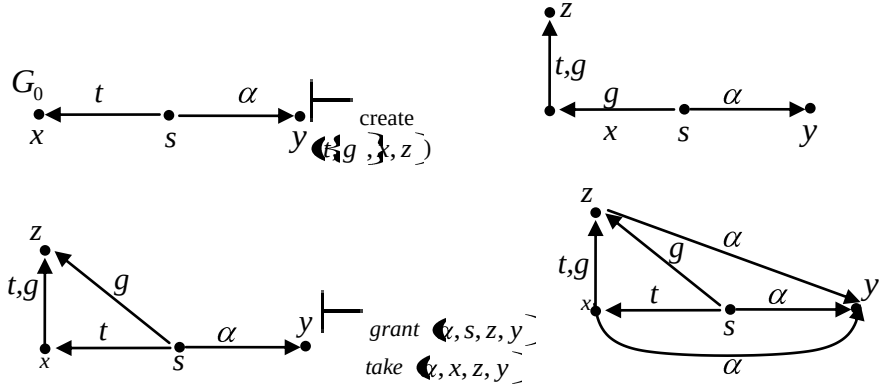


Рис. 14. Четвёртый случай

Для определения истинности предиката *can_share(α ,x,y, G_0)*в произвольном графе дадим определения.

Определение 3. Островом в произвольном графе доступов Go называется его максимальный *tg*-связный подграф, состоящий только из вершин субъектов.

Определение 4. Мостом в графе доступов G_0 называется *tg*-путь, концами которого являются вершины субъекты, проходящий через вершины объекты, словарная

запись которого имеет вид $\vec{t}^{*}$, $\overleftarrow{t}^{*}$, $\vec{t}^{*}\vec{g}\overleftarrow{t}^{*}$, $\vec{t}^{*}\vec{g}\overleftarrow{t}^{*}$, где символ «*» означает многократное (в том числе нулевое) повторение.

Определение 5. Начальным пролетом моста в графе доступов Go называется *tg*-путь, началом которого является вершина субъект, концом — объект, проходящий через вершины объекты, словарная запись которого имеет вид $\vec{t}^{*}\vec{g}$.

Определение 6. Конечным пролетом моста в графе доступов Go называется ig-путь, началом которого является вершина субъект, концом — объект, проходящий через вершины объекты, словарная запись которого имеет вид $\vec{t}^{*}$.

Теорема 2. Пусть $G_0 = (S_0, O_0, E_0)$ — произвольный граф доступов, $x, y \in O_0, x \neq y$. Предикат *can_share(a,x,y,G₀)* истинен тогда и только тогда, когда или $(x, y, \alpha) \subseteq E_0$, или выполняются *условия 1-3*.

Условие 1. Существуют объекты $s_1,....,s_m \in O_0$:
$(s_i, y, \gamma_i) \subseteq E_0$ для $i = 1,....,m$ и $\alpha = \gamma_1 \cup ... \cup \gamma_m$.

Условие 2. Существуют субъекты $x_1',...,x_m', s_1',....,s_m' \in S_0$:

а) $x = x_i'$ или x_i' соединен с *x* начальным пролетом моста в графе G_0, где $i = 1,...m$;

б) $s_i = s_i'$ или s_i' соединен с s_i конечным пролетом моста в графе G_0, где $i = 1,...,m$.

Условие 3. В графе G_0 для каждой пары (x_i', s_i'), $i = 1,...m,$, существуют острова $I_{i,1},...,I_{i,u_i}$, где $u_i \geq 1$, такие, что $x_i' \in I_{i,1}, s_i' \in I_{i,u_i}$ и существуют мосты между островами $I_{i,j}$ и $I_{i,j+1,} j = 1,...,u_i - 1$.

Доказательство. Проведем доказательство теоремы для $m = 1$, так как схему доказательства для этого случая легко продолжить на случай $m > 1$.

При m=1 условия 1-3 формулируются следующим образом (рис. 15).

Условие 1. Существует объект $s \in O_0 : (s, y, \alpha) \subset E_0$.

Условие 2. Существуют субъекты $x', s' \in S_0$:

а) x=x' или x' соединен с x начальным пролетом моста в графе G_0;

б) s=s' или s' соединен с s конечным пролетом моста в графе G_0.

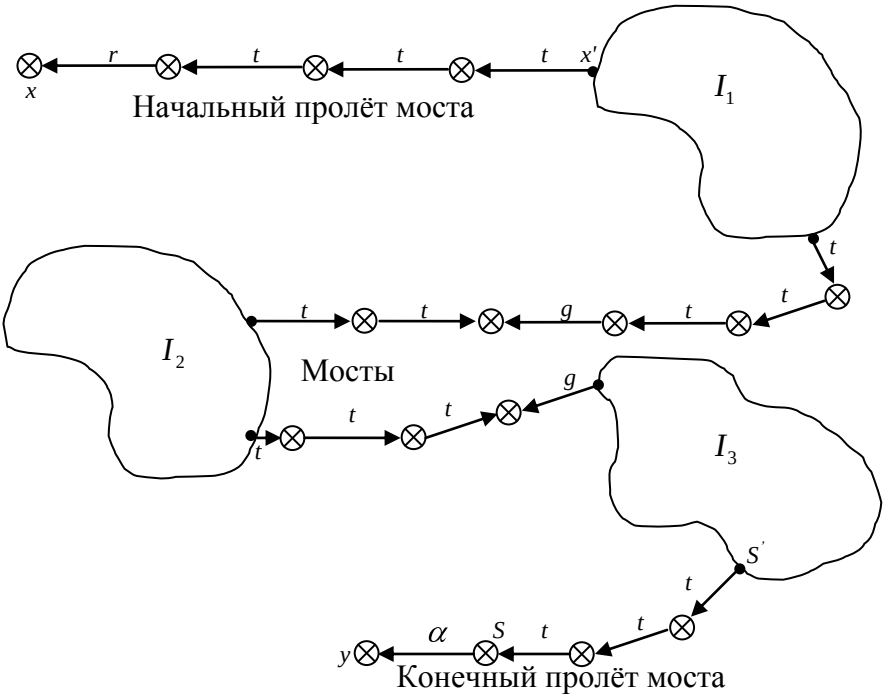


Рис. 15. Пример пути передачи объекту x прав доступа α к объекту y

Условие 3. В графе G_0 существуют острова $I_1,...,I_u$, $u \geq 1$, такие, что $x' \in I_1, s' \in I_u$, в существуют мосты между островами I_j и I_{j+1}, $j = 1,...,u-1$.

Достаточность. Если $\langle x, y, \alpha \rangle \subset E_0$, то предикат $can\_share(\alpha, x, y, G_0)$ истинен.

Типовые задачи

Задание 1. Проверьте, является ли мостом граф доступов на рис. 16, а.

Решение. Используем следующие обозначения для вершин графа доступов: s_1, s_2 – субъекты, o_1, o_2 – объекты (рис.16, б)

Так как в *определении 4* нет ограничения на повтор объектов на мосту, то граф доступов задает мост со словарной записью $\vec{t}, \vec{t}, \overleftarrow{g}, \overleftarrow{t}$, С концами в вершинах-субъектах s_1, s_2 и проходящий через объекты o_2, o_1, o_2 (рис. 16, в).

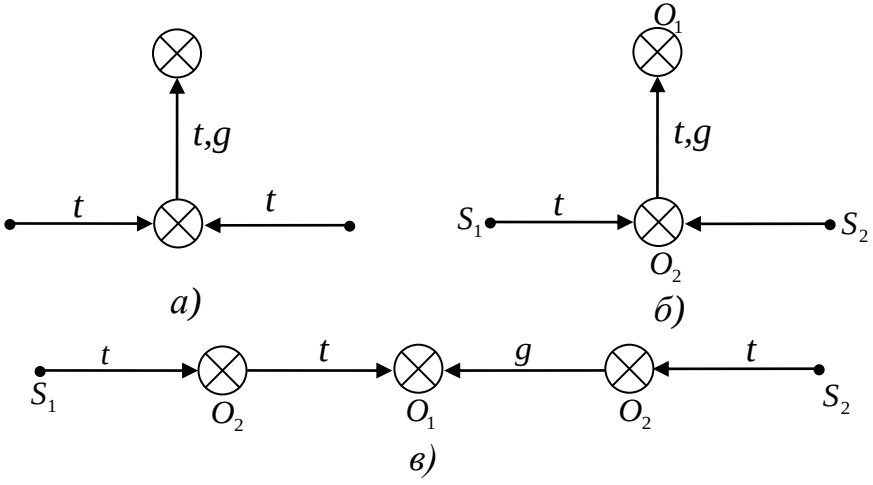


Рис. 16. Графы, илюстрирующие решение задания 1

Задание 2. Проверьте, истинен ли предикат *can_share* (α, x, y, G_0) для графа доступов G_0 на рис. 17. Решение задачи должно быть получено путем проверки выполнения условий *теоремы 2*.

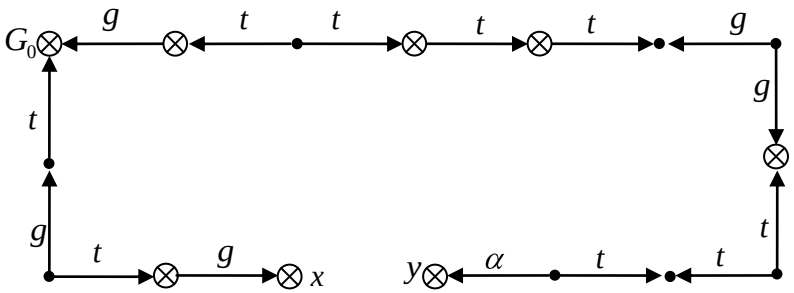


Рис. 17. Граф к заданию 2

Решение. Введем обозначения для объектов и субъектов графа доступов $G_0 = (S_0, O_0, E_0)$.

Существует субъект $s = s' \in S_0$ такой, что верно условие $(s_i, y, \alpha) \in E_0$, следовательно, *условие 1 теоремы 2* выполнено. Так как s является субъектом, и существует субъект $x' \in S_0$ такой, что он соединен с объектом x начальным пролетом моста, то *условие 2 теоремы 2* выполнено.

Выделим в графе G_0 острова $I_1 = \{x, s_1\}$, $I_2 = \{s_1\}$, $I_3 = \{s_4, s_6, s'\}$. Каждая пара соседних островов соединена мостом: I_1 и I_2 соединены мостом, проходящим через вершины s_1, o_2, o_3, s_2; I_2 и I_3 соединены мостом, проходящим через вершины s_2, o_4, o_5, s_3; I_1 и I_2 соединены мостом, проходящим через вершины s_4, o_6, s_5 (рис. 18). Следовательно, *условие 3 теоремы 2 выполнено*.

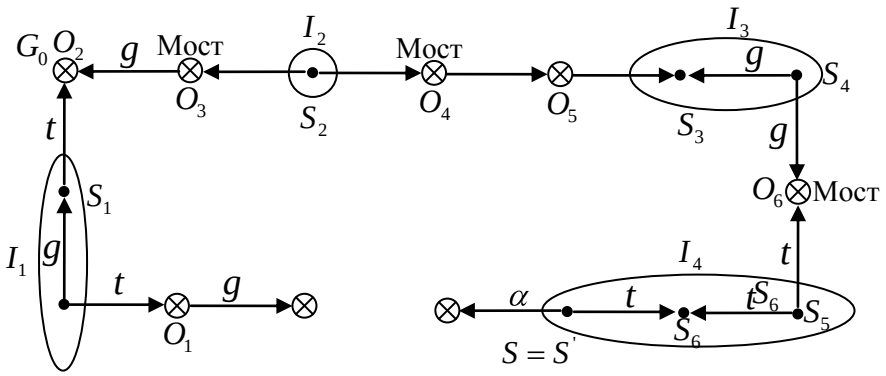


Рис. 18. Граф, илюстрирующий решение задания 2

Таким образом, выполнены все условия *теоремы 2*, и предикат *can_share* (α, x, y, G_0) является истинным.

Задание 3. Пусть в графе доступов G_0 субъекты s_1 и s_2 соединены некоторым путем (рис. 19) и известно, что существует последовательность правил преобразования графа доступов G_0, в результате применения которой с использованием рассматриваемого пути субъект s_i получает право доступа β к объекту o_2. Можно ли доказать, что тогда существует последовательность правил преобразования графа доступов G_0, в результате применения которой субъект s_2 может получить право доступа α к объекту o_1? При решении задачи не следует использовать утверждения *теорем 1* и *2*.

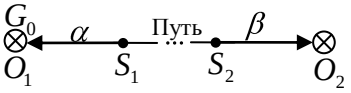


Рис. 19. Граф к заданию 3

Решение. Пусть существуют графы $G_1,...,G_N = (S_N, O_N, E_N)$ и правила $op_1,...,op_N$ (использующие ребра пути, соединяющего субъектов s_1 и s_2) такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ и $(s, y, \alpha) \subseteq E_N$, где $N \geq 1$. Пусть $op = create(t, g, s_2, o_3)$, заменим в последовательности правил преобразования состояний $op_1,...,op_N$ объект o_2 на объект o_3, право доступа β на право доступа g (рис. 20), получим последовательности правил преобразования состояний $op_1',...,op_N'$. Тогда положим $G_0 \vdash_{op} G_1' \vdash_{op_1'} G_1' \vdash_{op_2'} \ldots \vdash_{op_N'} G_{N+1}' = (S_{N+1}', O_{N+1}', E_{N+1}')$.

При этом выполняется условие $(s_1, o_3, g) \in E_{N+1}'$.

...

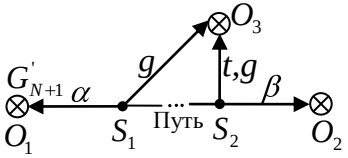


Рис. 20. Граф, илюстрирующий решение задания 3

Следовательно, существует последовательность правил преобразования графа доступов G_0, в результате применения которой субъект s_2 может получить право доступа α к объекту o_1.

Задачи для самостоятельного решения

1. Являются ли мостами графы доступов на рис. 21.

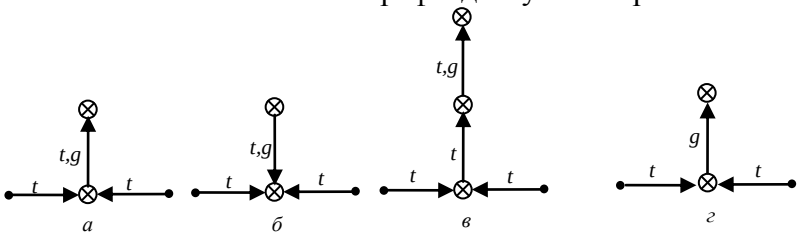


Рис. 21. Граф к задаче для самостоятельного решения

Практическое занятие № 4
Управление распространением прав доступа на основе расширенной модели Take-Grant

Теоретические положения

Направления развития модели Take-Grant Рассмотренные в классической модели *Take-Grant* способы анализа путей распространения прав доступа в системах с дискреционным управлением доступом имеют в большей степени теоретическое значение, так как, как: правило, в реальных КС не реализуются столь сложные графы доступов, для анализа которых необходимо использовать *теоремы 1* и *2* предыдущего занятия. В то же время на основе классической модели были разработаны ее расширения [9], которые развивают идеи классической модели, предлагая новые механизмы анализа, в большей степени применимые к современным системам защиты информации. Рассмотрим два расширения модели.

1. Де-факто правила, предназначенные для поиска и анализа информационных потоков.
2. Алгоритм построения замыкания графа доступов и информационных потоков.

Де-факто правила расширенной модели Take-Grant

Вместо прав доступа *take* и *grant* в расширенной модели в первую очередь рассматриваются права доступа *read* и *write,* наличие которых у субъектов системы является причиной возникновения информационных потоков.

Расширенная модель *Take-Grant* строится на основе классической модели. Ее основными элементами являются: O — множество объектов; $S \subseteq O$ — множество субъектов; $R = \{r_1, r_2, ..., r_m\} \cup \{t, g\} \cup \{r, w\}$ —множество видов прав доступа и видов информационных потоков, где *r (read)* — право на

чтение или информационный поток на чтение, *w (write)* — право на запись или информационный поток на запись; $G = \{S, O, E \cup F\}$ — конечный помеченный ориентированный без петель граф доступов и информационных потоков, описывающий состояние системы. Элементы множеств *S, O* являются вершинами графа. Элементы множества $E \subseteq O \times O \times R$ являются «реальными» ребрами графа, соответствующими правам доступа, и в графе доступов обозначаются сплошными линиями. Элементы множества $E \subseteq O \times O \times \{r, w\}$ являются «мнимыми» ребрами, соответствующими информационным потокам, и в графе доступов обозначаются пунктирными линиями. Каждое «реальное» ребро помечено непустым подмножеством множества видов прав доступа *R*, каждое «мнимое» Ребро помечено непустым подмножеством множества *{r,w}.*

Порядок перехода системы расширенной модели *Take-Grant* из состояния в состояние определяется де-юре и де-факто правилами преобразования графа доступов и информационных потоков. Преобразование графа *G* в граф *G'* в результате выполнения правила обозначим следующим образом: $G \vdash_{op} G'$.

Определение де-юре правил *take, grant, create, remove* совпадает с определением этих правил в классической модели *Take-Grant.* Де-юре правила применяются только к «реальным» ребрам (элементам множества *E).*

Де-факто правила применяются к «реальным» или «мнимым» ребрам (элементам множества $E \cup F$), помеченным *r* или *w.* Результатом применения де-факто правил является добавление новых «мнимых» ребер во множество *F.* Рассматриваются шесть де-факто правил: два вспомогательных и четыре основных.

Рассмотрим порядок применения де-юре правил преобразования графа доступов. В отличие от де-юре правил, для применения трех из шести де-факто правил требуется участие двух субъектов (рис. 22-27).

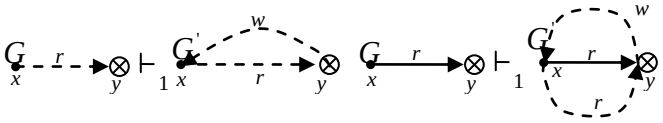


Рис. 22. Применение первого де-факто правила

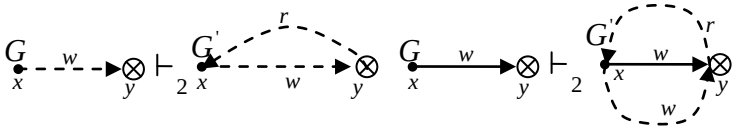


Рис. 23. Применение второго де-факто правила

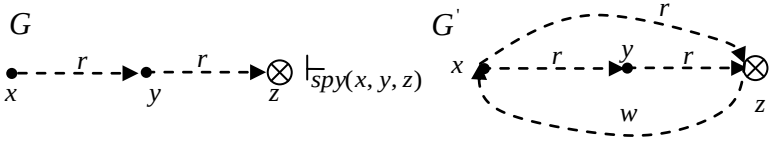


Рис. 24. Применение правила $spy(x, y, z)$

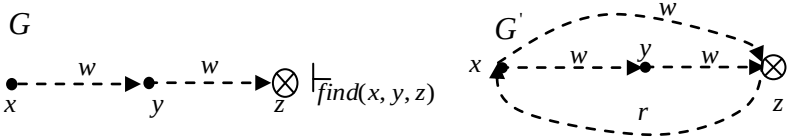


Рис. 25. Применение правила $find(x, y, z)$

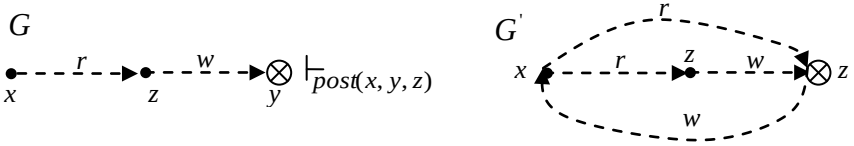


Рис. 26. Применение правила $post(x, y, z)$

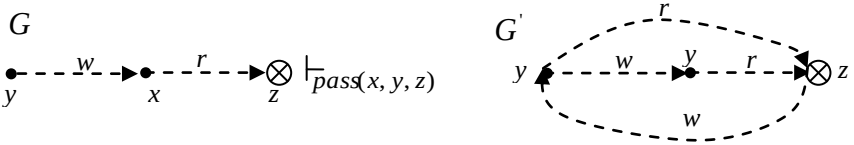


Рис. 27. Применение правила $pass(x, y, z)$

Условия применения де-факто правил в исходном состоянии $G = (S, O, E \cup F)$ и результаты их применения в результирующем состоянии $G' = (S, O, E \cup F')$ приведены в табл. 4.

Из определения де-факто правил следует, что для анализа информационных потоков достаточно рассматривать потоки одного вида: либо на чтение, либо на запись. В дальнейшем будем рассматривать только информационные потоки на запись. Будем также предполагать, что при возникновении информационного потока не накладывается ограничений на кооперацию субъектов системы, участвующих в этом процессе.

Таблица 4

Де-факто правила расширенной модели *Take-Grant*

Правило	Исходное состояние $G = (S, O, E \cup F)$	Результирующее состояние $G' = (S, O, E \cup F')$
Первое правило	$x \in S; y \in O; (x, y, r) \in E \cup F$	$F' = F \cup (y, x, w), (x, y, r)$
Второе правило	$x \in S; y \in O; (x, y, w) \in E \cup F$	$F' = F \cup (y, x, r), (x, y, w)$
spy(x,y,z)	$x, y \in S; x \neq x; (x, y, r), (y, z, r) \subset E \cup F$	$F' = F \cup (x, z, r), (x, x, w)$
find(x,y,z)	$x, y \in S; z \in O; x \neq z; (x, y, w), (y, z, w) \subset E \cup F$	$F' = F \cup (x, z, w), (x, x, r)$
post(x,y,z)	$x, y \in S; z \in O; x \neq y; (x, y, r), (y, z, w) \subset E \cup F$	$F' = F \cup (x, z, r), (x, x, w)$
pass(x,y,z)	$x, y \in S; z \in O; y \neq z;$ $(x, y, w), (x, z, r) \subset E \cup F$	$F' = F \cup (y, z, r), (x, y, w)$

Определение 1. Пусть $x, y \in O_0, x \neq y, x,$ — различные объекты графа доступов и информационных потоков $G_0 = (S_0, O_0, E_0 \cup F_0)$. Определим предикат $can\_write(x, y, G_0)$,

который будет истинным тогда и только тогда, когда существуют графы $G_1 = (S_1, O_1, E_1 \cup F_1), ..., G_N = (S_N, O_N, E_N \cup F_N)$, и де-юре или де-факто правила $op_1, ..., op_N$, где $N \geq 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ и $(x, y, w) \in F_N$.

Для проверки истинности предиката $can\_write(x, y, G_0)$, также следует определить необходимые и достаточные условия, задача проверки которых алгоритмически разрешима.

Теорема 1. Пусть $G_0 = (S_0, O_0, E_0 \cup F_0)$ граф доступов и информационных потоков, $x, y \in O_0, x \neq y$. Тогда предикат $can\_write(x, y, G_0)$ истинен тогда и только тогда, когда существуют объекты $o_1, ..., o_m \in O_0$, где $o_1 = x, o_m = y$, такие, что или $m = 2$ и $(x, y, w) \in F_0$, или для $i = 1, ..., m-1$ выполняется одно из условий:

- $o_i \in S_0$ и или истинен предикат $can\_share(w, o_i, o_{i+1}, G_0)$, или $(o_i, o_{i+1}, w) \in E_0 \cup F_0$;
- $o_{i+1} \in S_0$ и или истинен предикат $can\_share(r, o_{i+1}, o_i, G_0)$, или $(o_i, o_{i+1}, r) \in E_0 \cup F_0$;
- $o_i, o_{i+1} \in S_0$ и или истинен предикат $can\_share(\alpha, o_i, o_{i+1}, G_0)$, или истинен предикат $can\_share(\alpha, o_{i+1}, o_i, G_0)$, где $\alpha \in \{t, g\}$, или существует объект $o_i' \in O_0$ такой, что либо истинны предикаты $can\_share(t, o_i, o_i', G_0)$, $can\_share(g, o_{i+1}, o_i', G_0)$, либо истинны предикаты $can\_share(g, o_i, o_i', G_0)$, $can\_share(t, o_{i+1}, o_i', G_0)$.

Построение замыкания графа доступов и информационных потоков

Для проверки истинности предиката *can_share(α ,x,y,Go)* или *can_write(,x,y,Go)* для многих пар вершин неэффективно использовать алгоритмы проверки условий *теорем 1, 2* (предыдущее занятие). Эффективнее применять алгоритмы, позволяющие осуществить проверку истинности данных предикатов сразу для всех пар вершин. Такие алгоритмы реализуют преобразование графа доступов и информационных потоков в его замыкание.

Определение 2. Пусть $G = (S, O, E \cup F)$ — граф доступов и информационных потоков такой, что для каждого субъекта $s \in S$ существует объект $o \in O$ такой, что выполняется условие $(s, o, \{t, g, r, w\}) \subseteq E$. Тогда замыканием (или де-факто-замыканием) графа *G* называется граф доступов и информационных потоков $G^* = (S, O, E^* \cup F^*)$, полученный из *G* применением последовательности правил *take, grant* и де-факто правил. При этом применение к графу *G** указанных правил не приводит к появлению в нем новых ребер.

Алгоритм построения замыкания графа доступов состоит из трех этапов: построение *tg*-замыкания; построение де-юре-замыкания; построение замыкания.

Определение 3. Пусть $G = (S, O, E \cup F)$ — граф доступов и информационных потоков такой, что для каждого субъекта $s \in S$ существует объект $o \in O$ такой, что выполняется условие $(s, o, \{t, g, r, w\}) \subseteq E$. Тогда *tg*-замыканием графа *G* называется граф доступов и информационных потоков $G^{tg} = (S, O, E^{tg} \cup F)$, полученный из *G* применением последовательности правил *take* или *grant.* При этом каждое ребро $(o_1, o_2, \alpha) \in E^{tg} \setminus E$ имеет вид (o_1, o_2, t) или (o_1, o_2, g), и применение к графу G^{tg} правил *take* или

grant не приводит к появлению в нем новых ребер указанного вида.

Определение 4. Пусть $G = \langle S, O, E \cup F \rangle$ — граф доступов и информационных потоков такой, что для каждого субъекта $s \in S$ существует объект $o \in O$ такой, что выполняется условие $\langle s, o, \{t, g\}, r, w \rangle \subseteq E$. Тогда де-юре-замыканием графа G называется граф доступов и информационных потоков $G^{\text{де-юре}} = \langle S, O, E^{\text{де-юре}} \cup F \rangle$, полученный из G применением последовательности правил *take* или *grant.* При этом применение к графу $G^{\text{де-юре}}$ правил *take* или *grant* не приводит к появлению в нем новых ребер.

Алгоритм 1.

Алгоритм построения *tg*-замыкания графа доступов и информационных потоков $G = \langle S, O, E \cup F \rangle$ состоит из пяти шагов.

Шаг 1. Для каждого $s \in S$ выполнить правило $create\langle \{t, g\}, r, w \rangle$; при этом создаваемые объекты занести во множество *O,* создаваемые ребра занести во множество *E.*

Шаг 2. Инициализировать: $L = \{\langle x, y, \alpha \rangle \in E, \alpha \in \{t, g\}\}$ — список ребер графа доступов и информационных потоков и $N = \varnothing$ — множество вершин.

Шаг 3. Выбрать из списка *L* первое ребро $\langle x, y, \alpha \rangle$. Занести *x* и *y* во множество *N.* Удалить ребро $\langle x, y, \alpha \rangle$ из списка *L.*

Шаг 4. Для всех вершин $z \in N$ проверить возможность применения правил *take* или *grant* на тройке вершин x, y, z с использованием ребра $\langle x, y, \alpha \rangle$, выбранного на шаге 3. Если в результате применения правил *take* или *grant* появляются новые ребра вида $\langle a, b, \beta \rangle$, где $\{a, b\} \subseteq \{x, y, z\}$ и $\beta \in \{t, g\}$, занести их в конец списка *L* и множество *E*

Шаг 5. Пока список *L* не пуст, перейти на шаг 3.

Представление систем Take-Grant системами ХРУ

Решение задачи представления систем классической модели *Take-Grant* системами модели ХРУ позволяет лучше изучить основные свойства двух моделей систем дискреционного управления доступом.

Построим гомоморфизм системы *Take-Grant* и системы ХРУ.

Пусть состояние системы *Take-Grant* описывается графом $G = (S_{tg}, O_{tg}, E)$, а R_{tg} — множество прав доступа системы *Take-Grant.*

Положим для системы ХРУ:

$R = R_{tg} \cup \{own\}$ — множество прав доступа;

$S = O = O_{tg}$ — множество субъектов и объектов системы ХРУ;

$M_{|S|\times|S|}$ — матрица доступов, где для $x, y \in O_{tg}$, если $(x, y, r) \in E$, то $r \in M[x, y]$ и для $s \in S_{tg}$ выполняется условие $own \in M[s, s]$; $q = (S, O, M)$ — состояние системы.

Переход системы ХРУ в соответствии с правилами модели *Take-Grant* осуществляется в результате применения команд пяти видов для каждого $\alpha = \{r_1, ..., r_k\} \subseteq R_{tg}$.

command take $\_\alpha(x, y, z)$

if $(own \in M[x, x])$ *and* $(t \in M[x, y])$ *and* $(r_1 \in M[y, z])$ *and*...
and $(r_k \in M[y, z])$ *then*
«внести» право r_1 в $M[x, z]$;
……
«внести» право r_k в $M[x, z]$;
endif
end.

command grant_α(x, y, z)

if (*own* ∈ *M*[*x*, *x*]*and*(*g* ∈ *M*[*x*, *y*]*and*(*r*1 ∈ *M*[*y*, *z*]*and*...
and(*rk* ∈ *M*[*x*, *z*])*then*
«внести» право *r*1 в *M*[*y*, *z*];
……
«внести» право *rk* в *M*[*y*, *z*];
endif
end.

command create_object_α(x, y)
if (*own* ∈ *M*[*x*, *x*])*then*
«создать» субъект *y*;
«внести» право *r*1 в *M*[*x*, *y*];
……
«внести» право *rk* в *M*[*x*, *y*];
endif
end.

и т.д. с другими субъектами.

В приведенных командах, реализующих правила *take* и *grant*, не учитывается случай, когда выполняется условие *x*=*z*. В модели *Take-Grant* не допускается появление петель в графе доступов. Таким образом, в командах *take_α (x, y, z)* и *grant_α (x, y, z)* системы ХРУ следует учесть этот случай, например, путем включения в команды немонотонных примитивных операторов вида «удалить» право *r* из *M*[*x*,*x*]. Однако, если исключить из определения модели *Take-Grant* требование отсутствия петель в графе доступов, то рассмотренные в модели условия передачи прав доступа и реализации информационных потоков существенно не изменятся.

Дискреционные ДП-модели. Базовая ДП-модель

Для анализа условий передачи прав доступа и реализации информационных потоков между сущностями в [1] построено семейство формальных моделей КС, названных ДП-моделями. Основой всех ДП-моделей является базовая ДП-модель КС с дискреционным управлением доступом.

Для построения базовой ДП-модели в расширенную модель *Take-Grant* внесены следующие основные изменения:

- вместо множества объектов рассмотрено множество сущностей (объектов и контейнеров) с заданной на нем иерархической структурой;
- кроме прав доступа к сущностям, рассмотрены доступы к сущностям;
- предполагается, что только субъекты могут иметь права доступа к сущностям или доступы к сущностям (данное свойство соответствует используемым в большинстве современных КС правилам управления доступом);
- при анализе условий передачи прав доступа вместо двух прав доступа *take* и *grant* использовано одно право доступа на владение сущностью *own* (как правило, в КС если субъект имеет к сущности одно из прав доступа *take* или *grant,* то он имеет оба этих права доступа);
- предполагается, что если субъект имеет к сущности право доступа *own,* то он может получить любое право доступа к данной сущности новый субъект порождается (создается) другим субъектом из сущности;
- рассматриваются только информационные потоки на запись в сущность (информационному потоку на чтение всегда соответствует информационный поток на запись, направленный в противоположную сторону, и наоборот).

В основе базовой ДП-модели использован классический подход, состоящий в том, что каждая моделируемая КС представляется абстрактной системой, каждое состояние которой представляется графом доступов, каждый переход системы из состояния в состояние осуществляется в результате применения одного из правил преобразования графов доступа.

В рамках рассматриваемых ДП-моделей использованы следующие предположения.

Предположение 1. В начальном состоянии КС отсутствуют доступы субъектов к сущностям и информационные потоки.

Предположение 2. При любых запросе субъекта на доступ к сущности или доступе субъекта к сущности реализуется информационный поток по времени.

Предположение 3. При реализации информационного потока по памяти от сущности-источника к сущности-приемнику в том же направлении реализуется информационный поток по времени.

В базовой ДП-модели используются следующие обозначения и определения.

$E = O \cup C$ — множество сущностей, где O — множество объектов, C — множество контейнеров и $O \cap C = \varnothing$; $S \subseteq E$ — множество субъектов;

$R_r = \{read_r, write_r, append_r, execute_r, own_r\}$ — множество видов прав доступа, где $read_r$ — право доступа на чтение из сущности; $write_r$ — право доступа на запись в сущность; $append_r$ — право доступа на запись в конец сущности; $execute_r$ — право доступа на выполнение (активизацию) сущности; own_r — право доступа на владение сущностью, позволяющее субъекту-владельцу передавать права доступа к сущности другим субъектам или удалить сущность;

$R_a = \{read_a, write_a, append_a\}$ — множество видов доступа, где $read_a$ — доступ на чтение из сущности; $write_a$ — доступ на

запись в сущность; $append_a$ — доступ на запись в конец слова, содержащегося в сущности;

$R_f = \{write_m, write_t\}$ — множество видов информационных потоков, где $write_m$ — информационный поток по памяти на запись в сущность; $write_t$ — информационный поток по времени на запись в сущность;

$R_{raf} = R_r \cup R_a \cup R_f$ — множество видов прав доступа, видов доступа и видов информационных потоков, при этом множества R_r, R_a, R_f попарно не пересекаются.

Перечисленные элементы множества R_{raf} являются наиболее распространенными в современных КС, с их использованием, как правило, можно представить любой вид права доступа, вид доступа или вид информационного потока в КС.

Определение 5. Иерархией сущностей называется заданное на множестве сущностей E отношение частичного порядка «$\leq$», удовлетворяющее условию: если для сущности $e \in E$ существуют сущности $e_1, e_2 \in E$ такие, что $e \leq e_1, e \leq e_2$, то $e_1 \leq e_2$ или $e_2 \leq e_1$.

В случае, когда для двух сущностей $e_1, e_2 \in E$ выполняются условия $e_1 \leq e_2$ и $e_1 \neq e_2$, будем говорить, что сущность e_1 содержится в сущности-контейнере e_2, и будем использовать обозначение $e_1 < e_2$.

Определение 6. Определим $H: E \rightarrow 2^E$ — функцию иерархии сущностей, сопоставляющую каждой сущности $c \in E$ множество сущностей $H(c) \subseteq E$ и удовлетворяющую следующим условиям.

Условие 1. Если сущность $e \in H(c)$, то $e < c$ и не существует сущности-контейнера $d \in C$ такой, что $e < d, d < c$.

Условие 2. Для любых сущностей $e_1, e_2 \in E, e_1 \neq e_2$, по определению выполняются равенство $H(e_1) \cap H(e_2) = \varnothing$ и условия:

- если $o \in O$, то выполняется равенство $H(o) = \varnothing$;
- если $e_1 < e_2$, то или $e_1, e_2 \in E \setminus S$, или $e_1, e_2 \in S$;
- если $e \in E \setminus S$, то $H(e) \subseteq E \setminus S$;
- если $s \in S$, то $H(s) \subseteq S$.

Определение 7. Пусть определены множества $S, E, R \subseteq S \times E \times R_r$, $A \subseteq S \times E \times R_a$, $F \subseteq E \times E \times R_f$ функция иерархии сущностей H.

Определим $G = (S, E, R \cup A \cup F, H)$ — конечный помеченный ориентированный граф, без петель, где элементы множеств *S, E* являются вершинами графа, элементы множества $R \cup A \cup F$ — ребрами графа. Назовем $G = (S, E, R \cup A \cup F, H)$ графом прав доступа, доступов и информационных потоков или, сокращенно, графом доступов. При этом в графе доступов будем использовать следующие обозначения:

- вершины из множества *S* (соответствующие субъектам) в графе доступов будут обозначаться «•»;
- вершины из множества *E**S* (соответствующие сущностям, не являющимся субъектами) в графе доступов будут обозначаться «⊗»;
- каждое ребро графа доступов помечено одним из элементов множества R_{raf};
- каждое ребро из множества *R* будет обозначаться стрелкой вида, представленного на рис. 28, а (элементы множества *R* являются ребрами графа доступов, соответствующими правам доступа субъектов к сущностям);
- каждое ребро из множества *A* будет обозначаться стрелкой вида, представленного на рис. 28, б (элементы множества *A* являются ребрами графа доступов, соответствующими доступам субъектов к сущностям);

- каждое ребро из множества *F*, помеченное $write_m$, будет обозначаться стрелкой вида, представленного на рис. 28, в;

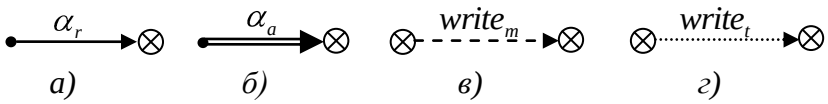


Рис. 28. Обозначения ребер графа доступов:
а — ребро из множества *R*, помеченное $\alpha_r \in R_r$;
б — ребро из множества *A*, помеченное $\alpha_a \in R_a$;
в — ребро из множества *F*, помеченное $write_m$;
г — ребро из множества F, помеченное $write_t$

- каждое ребро из множества F, помеченное $write_t$, будет обозначаться стрелкой вида, представленного на рис. 27,г (элементы множества *F* являются ребрами графа доступов, соответствующими информационным потокам между сущностями). Используем также обозначения:

  $\sum(G^*, OP)$ — система, при этом:

  - каждое состояние системы представляется графом доступов;
  - G^* — множество всех возможных состояний;
  - *OP* — множество правил преобразования состояний;
  - $G \vdash_{op} G'$ — переход системы $\sum(G^*, OP)$ из состояния *G* в состояние *G'* с использованием правила преобразования состояний $op \in OP$.

Если для системы $\sum(G^*, OP)$ определено начальное состояние, то будем использовать обозначение:

$\sum(G^*, OP, G_0)$ — система $\sum(G^*, OP)$ с начальным состоянием *Go*.

В начальном состоянии $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ системы $\sum(G^*, OP, G_0)$ в соответствии с предположением 1 выполняются равенства $A_0 = \varnothing$, $F_0 = \varnothing$.

Типовые задачи

Задание 1. Проверьте, истинен ли предикат $can\_write(x, y, G_0)$ для графа доступов G_o на рис. 29. Решение задачи должно быть получено путем непосредственного применения де-юре и де-факто правил.

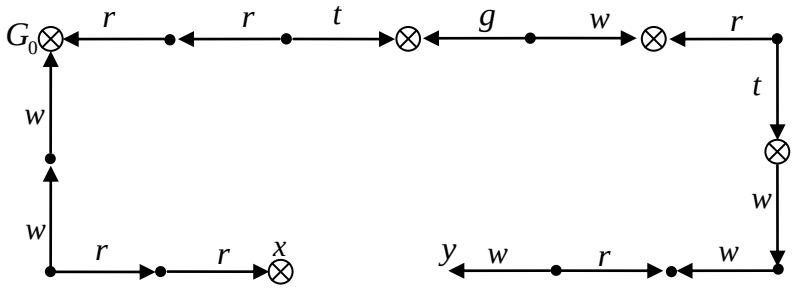


Рис. 29. Граф к заданию 1

Решение. Введем обозначения для объектов и субъектов графа доступов G_0. Применим к графу G_0 следующие де-юре правила:

$$op_1 = grant(w, s_6, o_2, o_3);$$
$$op_2 = take(w, s_5, o_2, o_3);$$
$$op_1 = take(w, s_7, o_4, s_8);$$
$$op_1 = grant(w, s_6, o_2, o_3);$$

Получим граф G_3 (рис. 30) такой, что выполняется условие

$$G_0 \vdash_{op_1} G_1 \vdash_{op_2} G_2 \vdash_{op_3} G_3$$

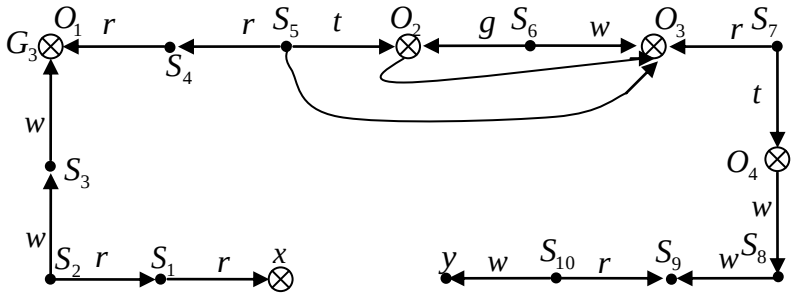


Рис. 30. Граф, илюстрирующий решение задания 1

Применим к графу G_3 следующие де-факто правила

$$op_4 = spy(s_2, s_1, x);\ op_5 = find(s_2, s_3, o_1);\ op_6 = spy(s_5, s_4, o_1);$$
$$op_7 = post(s_5, s_2, o_1);\ op_8 = post(s_7, s_5, o_3);\ op_9 = find(s_5, s_7, s_8);$$
$$op_{10} = post(s_{10}, s_8, s_9);\ op_{11} = find(s_5, s_8, s_{10});\ op_{12} = find(s_5, s_{10}, y);$$
$$op_{13} = find(s_2, s_5, y);\ op_{14} = pass(s_2, y, x);$$

Получим граф G_{14} (рис. 31) такой, что выполняются условия

$$G_0 \vdash_{op_1} G_1 \vdash_{op_2} G_2 \vdash_{op_3} \ldots \vdash_{op_{14}} G_{14} = (V_{14}, O_{14}, E_{14} \cup F_{14}^{'}) \text{ и } (x, y, w) \in F_{14}$$

Следовательно, предикат $can\_write(x, y, G_0)$ является истинным.

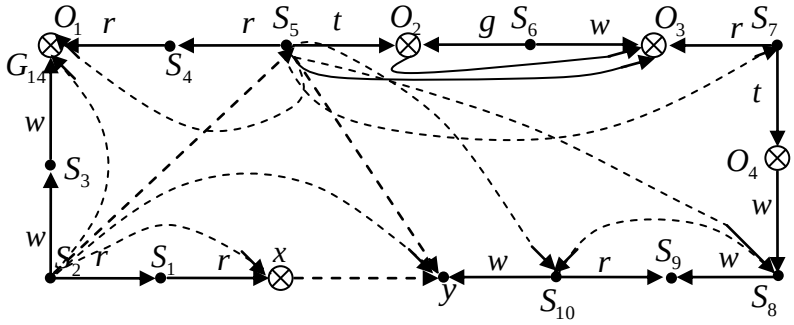


Рис. 31. Граф, илюстрирующий решение задания 1

Задание 2. Реализуйте информационный поток на запись от субъекта *x* к субъекту *y* и от субъекта *y* к субъекту *x* для системы с графом доступов на рис. 32.

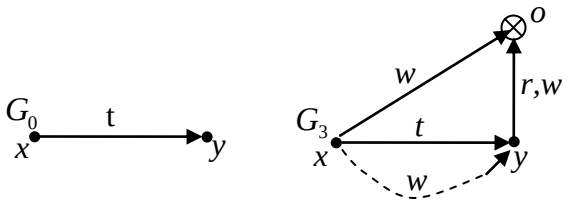


Рис. 32. Граф доступа и граф, илюстрирующий решение задания 2

Решение. Реализуем информационный поток на запись от субъекта *x* к субъекту *y*. Применим к графу G_0 следующие де-юре и де-факто правила:

$$op_1 = create\langle x, \{w\}, \{y, o\}\rangle;\ op_2 = take\langle v, x, y, o\rangle;\ op_3 = post\langle y, x, o\rangle.$$

Получим граф G_3 (рис. 32) такой, что выполняются условия

$$G_0 \vdash_{op_1} G_1 \vdash_{op_2} G_2 \vdash_{op_3} G_3 = \langle S_3, O_3, E_3 \cup F_3\rangle \text{ и } \langle x, y, w\rangle \in F_3.$$

Реализация информационного потока на запись от субъекта *y* к субъекту *x* осуществляется аналогично.

Задание 3. Постройте tg – замыкания графа доступов G_0 на рис. 32.

Применим к графу G_0 алгоритм 1. В результате выполнения шага 1 алгоритма получим граф доступов G_1 (рис. 33, а).

В результате выполнения шага 2 алгоритма инициализируем список $L = \langle x, o_1, t\rangle, \langle x, o_1, g\rangle, \langle x, y, t\rangle, \langle y, o_2, t\rangle, \langle y, o_2, g\rangle$ и множество $N = \varnothing$.

Выполнение шагов 3-5 с ребрами (x, o_1, t) и (x, o_1, g) не приводит к изменениям в графе доступов.

При текущих ребрах (x, y, t), (y, o_2, t) и (y, o_2, g) в результате выполнения шагов 3-5 в граф доступов и в конец списка L добавляются ребра $(o_1, y, t), (x, o_2 t)$ и (x, o_2, g) соответственно. Таким образом, получим граф доступов G_2 (рис. 33, б). При этом справедливы равенства $L = (o_1, y, t), (x, o_2, t), (x, o_2, g)$ и $N = (x, y, o_1, o_2)$.

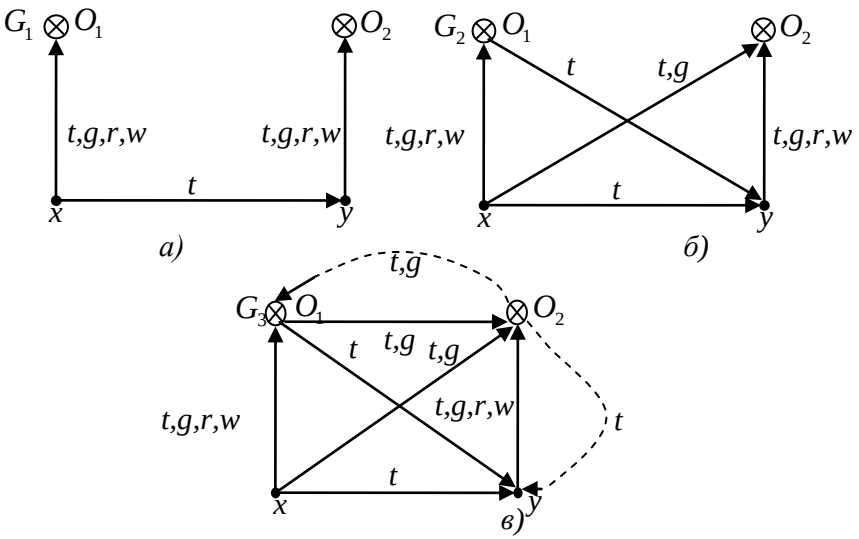


Рис. 33. Граф доступа и граф, илюстрирующий решение задания 3

Выполнение шагов 3-5 с ребром (o_1, y, t) не приводит к изменениям в графе доступов. При текущих ребрах (x, o_2, t) и (x, o_2, g) в граф доступов и в конец списка L добавляются ребра (o_1, o_2, t), (o_1, o_2, g), (o_2, o_1, t), (o_2, o_1, g) и (o_2, y, t). Выполнение шагов 3-5 с ребрами (o_1, o_2, t) и (o_1, o_2, g) не приводит к изменениям в графе доступов. При текущих ребрах

(o_2, o_1, t) и (o_1, o_2, g) в граф доступов и в конец списка L добавляются ребра (y, o_1, t) и (y, o_1, g). Выполнение шагов 3-5 с ребрами $(o_2, y, t), (y, o_1, t), (y, o_1, g)$ не приводит к изменениям в графе доступов.

Алгоритм закончил работу. В результате получен граф доступов G_3 (рис. 32, в), являющийся *tg*-замыканием графа доступов G_0.

Задачи для самостоятельного решения

1. Выразите правило *spy* расширенной модели *Take-Grant* через ее другие де-факто правила.

2. Как по аналогии с определением безопасного начального состояния в модели ХРУ и с использованием определения предиката $can\_share(\alpha, x, y, G_0)$ определить безопасное начальное состояние в модели *Take-Grant?*

3. Как представить систему, построенную на основе классической модели *Take-Grant,* системой ТМД? Постройте граф создания и каноническую форму системы МТМД, представляющей систему *Take-Grant.* При каких условиях система *Take-Grant* может быть представлена системой АМТМД?

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определения управления доступом (логическим), правил разграничения доступа, объекта и субъекта доступа.
2. Каковы назначение и основные достоинства унифицированных систем управления идентификацией?
3. Рассмотрите основное содержание политики в отношении логического доступа.
4. На основе каких документов, процедур и средств осуществляется управление доступом пользователей к активам организации?
5. Как должны назначаться и использоваться привилегированные права доступа?
6. Руководствуясь какими рекомендациями пользователи должны выбирать и изменять свои пароли?
7. Как осуществляется управление паролями?
8. В чем заключается политика чистого стола/экрана?
9. Каковы особенности сетевой аутентификации (по сравнению с аутентификацией при доступе к отдельному компьютеру)? Какие виды такой аутентификации рекомендуется использовать?
10. Какие средства защиты применяются в современных сетях - интранетах и экстранетах?
11. Какими защитными мерами осуществляется управление доступом к прикладным системам (приложениям)?
12. Как и на основе чего обеспечить ИБ при работе пользователей с переносными устройствами и в дистанционном режиме?
13. Какие процедуры при управлении защищенной передачей данных и операционной деятельности должны быть регламентированы?
14. В чем заключается принцип разделений полномочий пользователей?
15. Какие мероприятия по управлению ИБ обеспечивают разделение сред разработки и промышленной эксплуатации систем?

16. Какие специальные вопросы требуется решить при управлении СОИ сторонними лицами и организациями?

17. Что важно учитывать при планировании нагрузки и приемке систем?

18. На что необходимо обратить внимание при защите ПО и СОИ от вредоносных программ?

19. На основе чего осуществляется управление сетевыми ресурсами?

20. Как организуется защита носителей информации?

21. Каков порядок обмена информацией и ПО?

22. Что дает резервирование информации? Как правильно его организовать?

23. Подробно рассмотрите процесс выработки требований к ИБ систем. Какие виды требований ИБ обычно должны быть определены?

24. Каковы области формулирования требований для эшелонированной защиты вычислительной среды организации?

25. Как обеспечивается ИБ системных файлов?

26. Кратко перечислите основные средства крипто-графической информации, используемые в сетевой среде.

27. Каковы основные цели и функции процессов управления конфигурациями, изменениями и обновлениями? Что между ними общего и в чем различия?

28. Опишите жизненный цикл процесса управления обновлениями ИБ.

29. Как осуществляется физическая защита и защита от воздействия окружающей среды? В чем разница понятий логического и физического доступа?

30. Как в организации создаются охраняемые зоны? Что такое периметр безопасности?

31. Как защитить оборудование организации от различных видов угроз?

32. Каковы основные функции монитора безопасности объектов (МБО) и монитора безопасности субъектов (МБС) в изолированной программное среде (ИПС), в чём их отличие друг от друга.

33. Почему для реализации ИПС необходимо тебовать наличия в КС контроля порождения объектов?

ЗАКЛЮЧЕНИЕ

Основное внимание в методических указаниях уделено формальному описанию основных политик управления доступом. Рассмотрены так же технические аспекты управления ИБ к которым отнесены управление логическим доступом пользователей к активам организации, управление защищенной передачей данных и операционной деятельностью, разработкой и обслуживанием информационных систем с учетом требований к их ИБ, управление конфигурациями, изменениями и обновлениями в активах организации.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Девянин, П. Н. Анализ безопасности управления доступом и информационными потоками в компьютерных системах [Текст] / П. Н. Девянин. — М.: Радио и связь, 2006. — 176 с. .

2. Девянин, П. Н. Модели безопасности компьютерных систем. Управление доступом и информационными потоками [Текст]: учеб. пособие для вузов; 2-е изд., испр. и доп. / П. Н. Девянин. – М.: Горячая линия – Телеком, 2013. – 338 с.

3. Девянин, П. Н. Теоретические основы компьютерной безопасности [Текст]: учеб. пособие для вузов / П. Н. Девянин, О. О. Михальский, Д. И. Правиков, А. Ю. Щербаков. – М.: Радио и связь, 2000. – 192 с. .

4. Зегжда, Д. П. Основы безопасности информационных систем [Текст] / Д. П. Зегжда, А. М. Ивашко. – М.: Горячая линия – Телеком, 2000. – 452 с.

5. Милославская, Н. Г. Технические, организационные и кадровые аспекты управления информационной безопасностью [Текст]: учеб. пособие для ВУЗов; 2-е изд., испр. / Н. Г. Милославская, М. Ю. Сенаторов, А. И. Толстой. – М.: Горячая линия – Телеком, 2014. – 214 с.

6. Носов, В. А. Основы теории алгоритмов и анализа их сложности. Курс лекций [Текст] / В. А. Носов. – М: МГУ, 1992. – 140 с.

7. Фороузан, Б. А. Криптография и безопасность сетей [Текст] / Б. А. Фороузан. – М.: БИНОМ. Лаборатория знаний, 2010. – 784 с.

8. Bell, D. E. Secure Computer Systems: Unified Exposition and Multics Interpretation [Text] / D. E. Bell, L. J. LaPadula. – Bedford, Mass.: MITRE Corp., 1976. – MTR-2997 Rev. 1.

9. Frank, J. Extending The Take-Grant Protection System [Text] / J. Frank, M. Bishop // Department of Computer Science. — University of California at Davis, 1984.

10. McLean, J. The Specification and Modeling of

Computer Security [Text] / J. McLean //Computer. 1990. Vol. 23, № 1. Sandhu R. Rationale for the RBAC96 family of access control models// Proceeding of the 1st ACM Workshop on Role-Based Access Control. - ACM, 1997.

11. Sandhu, R. Role-Based Access Control, Advanced in Computers [Text] / R. Sandhu // Academic Press. 1998. Vol. 46.

12. Sandhu, R. The typed access matrix model [Text] / R. Sandhu // In Proceeding of the IEEE Symposium on Research in Security and Privace, Oakland, CA, May 1992. — P. 122-136.

13. Управление ключами шифрования и безопасность сети: Информация [Электронный ресурс] – Режим доступа: http://www.intuit.ru/studies/courses/553/409/info

СОДЕРЖАНИЕ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к практическим занятиям № 1–4 по дисциплине
«Управление информационной безопасностью»
для студентов специальности
090303 «Информационная безопасность
автоматизированных систем»
очной формы обучения

Составитель
Разинкин Константин Александрович

В авторской редакции

Подписано к изданию 28.11.2014
Уч.-изд. л. 3,5

ФГБОУ ВПО «Воронежский государственный
технический университет»
394026 Воронеж, Московский просп., 14