О множествах наборов прав в системах безопасности с матрицей доступа

С.В. Усов
raintower@mail.ru

Омский государственный университет им. Ф.М. Достоевского, Омск, Россия

Аннотация

Статья посвящена изучению свойств наборов прав доступа в системах с дискреционным разграничением доступа. Рассматривается модель Харрисона-Руззо-Ульмана. Показано, что множества наборов прав формируют систему независимости. Приведен пример строения подсистемы безопасности, когда упомянутая система независимости является матроидом.

Данная работа посвящена изучению некоторых свойств множеств наборов прав доступа в дискреционных моделях систем безопасности.

Рассмотрим компьютерную систему с дискреционной политикой безопасности Харрисона-Руззо-Ульмана[1].

В рамках модели HRU рассматриваются следующие множества:

- множество объектов доступа $\mathbf{O}$;
- множество субъектов доступа $\mathbf{S}$, причем $\mathbf{S} \subseteq \mathbf{O}$;
- множество прав доступов R;

Возможность получения доступа субъекта к объекту определяется матрицей доступов – таблицей M, строки которой подписаны субъектами, а столбцы – объектами. Каждая ячейка содержит некоторый набор прав $\alpha \in R$, которыми обладает субъект строки по отношению к объекту столбца матрицы доступов. Содержание ячейки матрицы доступов, находящейся на пересечении строки $s \in \mathbf{S}$ и столбца $o \in \mathbf{S}$, будем обозначать $M[s, o]$.

Для внесения изменений в матрицу доступов используются так называемые элементарные операторы – это системные команды определенного вида, всего их 6:

1. $CreateSubject(s)$ – создает субъект $s \in \mathbf{S}$;
2. $CreateObject(o)$ – создает объект $o \in \mathbf{O}$;
3. $DestroySubject(s)$ – уничтожает субъект s;
4. $DestroyObject(o)$ – уничтожает объект o;
5. $Enter(r, s, o)$ – вносит право доступа $r \in R$ в $M[s, o]$, при этом матрица доступов изменяется по правилу: $M[s, o] := M[s, o] \cup \{r\}$;

In: Sergey V. Belim, Nadezda F. Bogachenko (eds.): Proceedings of the Workshop on Data Analysis and Modelling (DAM 2016), Omsk, Russia, October 2016, published at http://ceur-ws.org

6. $Delete(r, s, o)$ – удаляет право доступа $r \in R$ из $M[s, o]$, при этом матрица доступов изменяется по правилу: $M[s, o] := M[s, o] \setminus \{r\}$.

Элементарные операторы HRU группируются в команды, которые можно рассматривать как запросы на изменение матрицы доступов. Каждая команда состоит из двух частей: условия выполнения команды и последовательности элементарных операторов. Команда HRU имеет следующий вид:

$Command\gamma(x_1, x_2, \ldots, x_k)$
$if\ r_1 \in M[s_1, o_1]\ and$
$r_2 \in M[s_2, o_2]\ and$
.
.
.
$r_l \in M[s_l, o_l]$
$<$ условия выполнения команды $>$
$then$
$op_1, op_2, \ldots, op_n$
$<$ операторы, составляющие команду $>$.

Здесь каждый из аргументов команды $x_1 \ldots x_k$ представляет собой либо объект из, либо субъект, участвующий в условной части либо в элементарных операторах; $o_1 \ldots o_l \in \mathbf{O}$, $s_1 \ldots s_l \in \mathbf{S}$, $r_1 \ldots r_l \in R$.

При выполнении каждой команды система переходит из одного состояния в другое. Пусть система в момент времени t находится в состоянии $Q(t)$, тогда после применения некоторой команды γ система перейдет в новое состояние $Q(t+1)$ в следующий момент времени $t+1$.

Отметим, что в условной части команды проверяется только наличие права в определенной ячейке матрицы доступов, но не его отсутствие.

Итак, пусть – общая матрица доступа, и $M(t)$ – состояние матрицы доступа в момент времени t. Тогда для $M(t)$ можно рассмотреть множество $A(t)$ как неструктурированный набор прав:

$$A(t) = \{a \in \mathbf{S} \times \mathbf{O} \times R | a = (s, o, r), r \in M(t)[s, o]\}.$$

Здесь в случае объектно-субъектной модели $\mathbf{S}$ – множество всех субъектов, допускающих активацию в данной системе, $\mathbf{O}$ – множество всех объектов, которые могут быть созданы в системе, R – множество всевозможных прав, которыми может обладать субъект на объект. В случае объектно-ориентированной модели $\mathbf{S}$ – множество объектов, которые могут быть созданы в системе, $\mathbf{O}$ – множество полей и методов объектов системы, R – множество всевозможных прав, которыми может обладать объект на поле или метод.

Рассмотрим семейство $\mathbf{A}$ всевозможных подмножеств декартова произведения $\mathbf{S} \times \mathbf{O} \times R$. Понятно, что $A(t) \in \mathbf{A}$. Семейство $\mathbf{A}$ конечно, если множества $\mathbf{S}$ и $\mathbf{O}$ конечны. Такие компьютерные системы будем называть конечными. Далее, если не оговорено противного, будем рассматривать именно конечные компьютерные системы.

Безопасность системы будет определена в соответствии с работой [2]:

Определение 1. Будем говорить, что состояние системы допускает утечку права доступа $r \in R$, если существуют такие команда γ, объект o и субъект s, что $r \notin o.M[s, o]$ в текущем состоянии системы, но $r \in o.M[s, o]$ в состоянии, в которое система перейдет из текущего по выполнении команды γ.

Определение 2. Будем говорить, что система r-безопасна, если не существует последовательности команд, переводящих систему из начального состояния в состояние, допускающее утечку права доступа r.

Определение 3. Будем говорить, что набор прав доступа $A(t) \in \mathbf{A}$ допускает утечку права доступа $r \in R$, если состояние системы $Q(t)$ с соответствующей набору прав $A(t)$ матрицей доступа $M(t)$ допускает утечку права r.

Определение 4. Будем говорить, что набор A r-безопасен, если система с соответствующей набору прав A начальной матрицей доступа $M(0)$ r-безопасна.

Кроме того, напомним, что системой независимости на конечном множестве X называется пара $(X, \mathbf{I})$, $\mathbf{I} \subseteq 2^X$, такая, что $\mathbf{I}$ содержит пустое множество и любое подмножество любого элемента из $\mathbf{I}$ само лежит в $\mathbf{I}$. Множества, являющиеся элементами $\mathbf{I}$, называются независимыми, а множества, принадлежащие $2^X \setminus \mathbf{I}$ – зависимыми. Максимальные по включение независимые множества называются базами, минимальные по включению зависимые множества – циклами. Наконец, система независимости называется матроидом, если все ее базы равномощны [3].

Теорема 1. *Семейство $\mathbf{I} \subseteq \mathbf{A}$ всех r-безопасных наборов прав доступа образует систему независимости.*

Доказательство. Заметим, что условием для выполнения какой-либо команды служит наличие некоторых прав, в то время как отсутствие прав в условной части команды никогда не проверяется.

Пусть для начального состояния системы $Q(0)$ с общим набором прав $A(0)$ существует цепочка команд $\gamma_1, \gamma_2, \ldots, \gamma_T$, переводящая система в состояние $Q(T)$, допускающее утечку права доступа r. Без ограничения доступа считаем цепочку $\gamma_1, \gamma_2, \ldots, \gamma_T$ несократимой, в частности, при последовательном выполнении команд этой цепочки условия каждой команды оказываются выполнены (в противном случае, если условия не выполнены и команды не выполняется, ее можно выкинуть из цепочки). Пусть $A(0) \subseteq A'(0)$, и $Q'(0)$ – состояние той же системы, соответствующее набору $A'(0)$. Применим к этому состоянию системы цепочку команд $\gamma_1, \gamma_2, \ldots, \gamma_T$: по-прежнему условия выполнения всех команд будут выполнены. Итак, надмножество набора прав доступа, не являющегося r-безопасным, само не r-безопасно. Наоборот, подмножество r-безопасного набора будет r-безопасным. ■

Рассмотрим на множестве наборов всевозможных прав $\mathbf{A}$ аддитивную функцию $\phi : \mathbf{A} \to \mathbb{R}$. Для этого достаточно задать значение функции на элементах множества $\mathbf{S} \times \mathbf{O} \times R$. Смыслом этой функции может быть, например, оценка эффективности права доступа. По теореме Радо-Эдмондса [3] жадный алгоритм точно решает задачу нахождения максимального значения аддитивной функции на r-безопасном наборе, если семейство $\mathbf{I}$ всех r-безопасных наборов прав доступа является матроидом.

Теорема 2. *Существует дискреционная модель безопасности компьютерной системы, семейство всех r-безопасных наборов прав доступа которой является матроидом.*

В качестве доказательства приведем пример такой компьютерной системы для объектно-ориентированной модели HRU [4]. Напомним основные отличия объектно-ориентированной модели HRU от классического субъектно-объектного аналога.

Система рассматривается в виде множества объектов $\mathbf{O}$, имеющих открытые поля и скрытые поля, а также методы обработки полей. Каждый из объектов принадлежит какому-то классу k из множества всех классов системы $\mathbf{K}$. Класс задает структуру объекта: определяет, какими общими свойствами будут обладать поля и методы всех объектов этого класса. Объекты одного класса отличаются друг от друга только содержанием своих полей.

Полем f называется некоторая область памяти фиксированной длины, которая может содержать произвольные данные и изменяться в процессе функционирования системы. Методом s называется некоторое отображение, использующее в качестве аргументов поля, и не изменяющееся в процессе функционирования компьютерной системы.

Классом объектов называется произвольный список полей и методов, каждому из которых присвоено одно значение из множества $\{private, public\}$. Поля и методы, которым присвоено значение *private*, называются скрытыми. Поля и методы, которым присвоено значение *public*, называются открытыми.

Совокупность полей и методов построенных по списку, задаваемому классом, называется объектом класса. Объект класса, для которого не существенна либо очевидна в данной задаче принадлежность к конкретному классу, будем называть объектом. Для каждого объекта $o \in \mathbf{O}$ опредеяются множество скрытых полей $o.P$, множество открытых полей $o.F$ и множество методов $o.S$.

Доступом метода s к полю f называется активизация метода s таким образом, что объект o является аргументом соответствующего отображения. Доступ к скрытым полям объекта имеют только методы этого объекта. Кроме того, методы могут обращаться к открытым полям других объектов. Возможность доступа к скрытым полям объекта только методами этого же объекта называется инкапсуляцией. Для открытых полей (*public*) будем считать верным следующее предположение: открытые поля всех объектов имеют одно и то же множество возможных типов доступа. Множество доступов к открытым полям обозначим через R.

Для построения системы дискреционного разделения доступа для каждого объекта $o \in \mathbf{O}$ вводится дополнительное *private* поле M, содержащее локальную матрицу доступов: $o.M : \mathbf{O} \times (o.F \cup o.S) \to 2^R \cup \{0,1\}$, причем $o'.M[o,f] \to 2^R$, где $f \in o'.F$; $o, o' \in \mathbf{O}$, то есть для открытых полей в явном виде задается множество разрешенных доступов, и $o'.M[o,s] \to \{0,1\}(o, o'?O)$, где $s \in o'.S$; $o, o' \in \mathbf{O}$ то есть для методов определяем разрешение (1) или запрет (0) вызова.

По аналогии с субъектно-объектной моделью HRU, для объектно-ориентированных моделей вводятся следующие элементарные операторы:

1. $Create(o,k)$ – создает объект o класса $k \in \mathbf{K}$, если $o \in \mathbf{O}$.
2. $Destroy(o)$ – уничтожает объект o, если $o \in \mathbf{O}$.
3. $Enter(r,o,o'.f)$ – вносит право доступа r в $o'.M[o,f]$, если $o, o' \in \mathbf{O}$.
4. $Delete(r,o,o'.f)$ – удаляет право r доступа из $o'.M[o,f]$, если $o, o' \in \mathbf{O}$.
5. $Grant(o,o'.s)$ – разрешает вызов объекту o метода $o'.s$, если $o, o' \in \mathbf{O}$.
6. $Deprive(o,o'.s)$ – запрещает вызов объекту o метода $o'.s$, если $o, o' \in \mathbf{O}$.

Под состоянием компьютерной системы в каждый момент времени понимается совокупность множества всех объектов системы в этот момент времени и состояния всех матриц доступов $o.M$ объектов системы в этот момент времени. Состояния компьютерной системы в модели HRU изменяются под воздействием запросов на модификацию матрицы доступа в виде команд следующего формата:

$Command\gamma(x_1, ..., x_g)$:

if <конъюнкция логических выражений вида $r \in o'.M[o,f]$ или $o'.M[o,s] = 1$>

$then$ <последовательность элементарных операторов>.

Здесь аргументы x_i представляют собой объекты, участвующие в качестве аргументов в условной части либо в элементарных операторах.

Доказательство. Пусть множество классов системы $K = \{k_1, k_2, \ldots, k_m\}$. Для простоты примем, что все объекты каждого класса устроены довольно просто: они обладают одним методом s и одним открытым полем f, к которому существует два вида доступа: r (безопасный) и w (опасный). Будем рассматривать множество независимости, совпадающее со множеством всех w-безопасных наборов прав доступа. Для этого необходимо определить, какие команды присутствуют в системе.

Рассмотрим компьютерную систему как двудольный граф G, в котором каждому объекту o соответствует две вершины: $o.s$ и $o.f$, очевидно, представляющие метод и поле этого объекта. Вершины $o.s$ и $o'.f$ соединены ребром в момент времени t, если $a = (o, o'.f, r) \in A(t)$, где $A(t)$ – текущий набор прав доступа системы. При этом, возможно, $o = o'$. Мы будем отождествлять право $a = (o, o'.f, r)$ с ребром графа G. Таким образом, $G(t) = G(\mathbf{O}, A(t))$. Построим такую систему команд, которая допускает добавление ребра в этом графе (то есть права r некоторого объекта на поле f этого или другого объекта), только если оно не приводит к образованию цикла. Если же цикл в графе уже существует, то соответствующее состояние системы допускает утечку права w. Для реализации требуемых команд введем дополнительное (фиктивное) право доступа r', обозначающее отсутствие права r.

Вот команда, добавляющее ребро, не приводящее к образованию цикла:

$Command1(o_1 : k_1, o_2 : k_2, \ldots, o_m : k_m)$

if < конъюнкция условий вида $r' \in o^*.M[o^{**}, f]$ таких, что в графе G отсутствие ребер $(o^{**}, o^*.f, r)$, соответствующих проверяемым правам, достаточно для его ацикличности, то есть G без указанных ребер является лесом >

$then$

$Enter(r,o,o'.f)$ (важно, что условной части не проверялось отсутствие ребра $a = (o, o'.f, r)$),

$Delete(r', o, o'.f)$.

Команд такого вида может быть довольно много, так как количество лесов из условия команды, равно как и множество вариантов добавляемого ребра из тела команды, довольно велико. Впрочем, таких команд может и не быть вообще.

Команды, приводящая к утечке права w в случае наличия в графе системы цикла, выглядят гораздо проще:

$Command2(o_1 : k_1, o_2 : k_2, , o_m : k_m)$

$if <$ конъюнкция условий вида $r \in o^*.M[o^{**}, f]$ таких, что в графе G ребра $(o^{**}, o^*.f, r)$, соответствующих проверяемым правам, образуют цикл, например:

$r \in o^*.M[o^{**}, f] \& r \in o^*.M[o^*, f] \& r \in o^{**}.M[o^{**}, f] \& r \in o^{**}.M[o^*, f] >$

$then$

$Enter(w, o, o'.f)$.

Команда, создающая объект, может внести либо право доступа r этого объекта к полям других объектов, либо право доступа r других объектов к полю созданного объекта, но не то и другое одновременно. Такие действия не нарушат ацикличность графа системы. Также в данной системе допустимы произвольные команды, не содержащие операции $Enter$.

Понятно, что система независимости, образованная w-безопасными множествами прав доступа построенной компьютерной системы, будет является двудольным матроидом, а именно – циклическим матроидом графа системы G. ■

Замечание 1. Пример, построенный в теореме, работает только в случае однородной системы безопасности. Его можно распространить на произвольную конечную систему, если ввести отдельный подкласс для каждого объекта (из конечного множества).

Замечание 2. Существует и более простой пример матроида: когда для утечки права w достаточно набрать определенное (скажем, l) количество прав r. При этом если $|A| < l$, то можно добавить еще одно право l. Однако и в этом случае требуется однородность системы и невозможность подставить в команду один и тот же объект несколько раз.

Замечание 3. Система, построенная в теореме, допускает проверку r-безопасности, причем довольно простую: если в начальном состоянии системы элементы множества $A(0)$ образуют цикл, то система допускает утечку права w, в противном случае она w-безопасна.

Список литературы

[1] M. A. Harrison, W. L. Ruzzo, J. D. Ulman. Protection in Operating Systems. *Communications of the ACM*, 14–25, 1975.

[2] M. V. Tripunitara, N. Li. The Foundational work of Harrison-Ruzzo-Ullman Revisited. *Center for Education and Research in Information Assurance and Security, Purdue University, West Lafayette, IN 47907-2086*, 2006.

[3] J. Edmonds. Matroids and the Greedy Algorithms. *Math Programming*, 127–136, 1971.

[4] S. V. Belim, S. Yu. Belim, S. V. Usov. Ob"ektno-orientirovannaya modifikatsiya modeli bezopasnosti HRU. *Problemy informatsionnoy bezopasnosti. Komp'yuternye sistemy*, 39(1):6–14, 2010 (in russian).

Some Properties of Sets of Rights in Access Matrix Models of Security Systems

Sergey V. Usov

The article is devoted to studying the properties of sets of access rights in the models of security systems with discretionary access control. The Harrison-Ruzzo-Ulman model of access control is considered. It is shown that the sets of rights create an independence system. An example of the security system with the sets of rights forming matroid is shown.