Математические структуры и моделирование
2010, вып. 21, с. 151–161

УДК 65.012.810

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД В ПОСТРОЕНИИ ПОЛИТИКИ БЕЗОПАСНОСТИ. СИСТЕМЫ С ЕСТЕСТВЕННОЙ ИЕРАРХИЕЙ

С.В. Усов

Проведена модификация модели дискреционного разделения доступа HRU для объектно-ориентированных компьютерных систем с учетом иерархии классов объектов.

Введение

Традиционно рассматриваемая модель системы безопасности компьютерных систем строится на основе субъектно-объектного подхода. Однако в последнее время, в связи с распространением объектно-ориентированного подхода в построении программного обеспечения компьютерных систем, складывается двойственная ситуация, когда один и тот же набор данных в одном случае интерпретируется как объект (пассивный), в другом случае как субъект (активный). В связи с этим, для более адекватного описания компьютерных систем, необходимо применение объектно-ориентированного подхода и соответствующая модификация моделей политик безопасности. В частности, объектно-ориентированная модель компьютерной системы построена в работе [1], где даны и необходимые определения. В последующей работе [2] определены элементарные операторы в рамках модели безопасности HRU [3] для объектно-ориентированных систем, а также установлена разрешимость проблемы безопасности системы для отдельных случаев, а именно для монооперационных HRU-систем и моноусловных монотонных HRU-систем.

В настоящей работе мы рассмотрим некоторые частные случаи моделей безопасности в объектно-ориентированных системах.

1. Классификация моделей безопасности объектно-ориентированных систем

Прежде всего определим HRU-модель, рассмотренную в работе [2], как *свободную*, т.к. в ней не накладывается никаких дополнительных ограничений на

Омский государственный университет им. Ф.М. Достоевского.
E-mail: raintower@mail.ru

отношения между уровнями доступа различных объектов различных классов. Однако наряду с HRU-подходом в сфере защиты информации распространен и иерархический подход [4], когда на множестве O объектов задан частичный порядок по уровню доступа.

Будем рассматривать компьютерную систему в виде множества объектов классов K, имеющих открытые поля f и скрытые поля p, а также методы обработки полей s. Пусть $O^k \subset O$ – множество объектов класса $k \in K$. В случае, если требуется уточнить класс объекта, поле f объекта $o^k \in O^k$ будем обозначать $o^k.f$, поле f класса k – $k.f$. Для скрытых полей и методов класса будем использовать аналогичные обозначения.

Все необходимые определения даны в работе [1], там же построена и модель системы безопасности для объектно-ориентированной компьютерной системы.

Матрица доступа M в данном случае - скрытое (private) поле объекта. От объекта к объекту даже одного класса (типа) содержание этого поля может отличаться. Строки матрицы объекта $o \in O$ соответствуют объектам компьютерной системы, столбцы - открытым (public) полям и методам объекта o, элементом матрицы является некоторое подмножество множества R видов доступа к полю, если столбец элемента соответствует public полю объекта и принимает значение 0 либо 1, если столбец элемента соответствует методу объекта, где «1» разрешает вызов метода, а «0» – запрещает.

Определение 1. HRU-модель системы безопасности называется иерархической, если на множестве объектов O задан частичный порядок-иерархия и в любой момент работы системы для любых двух объектов o, o' таких, что $o' \leq o$, для любого поля или метода $x \in X$, общего для объектов o и o', и для любого поля или метода $y \in X$ объекта o'' верно следующее: $o''.M[o, o''.x] \subset o''.M[o', o''.x]$ и $o'.M[o'', o'.x] \subset o.M[o'', o.x]$. Здесь и далее X - множество всевозможных полей и методов всех существующих в системе на данный момент времени объектов, " $\leq$ " – отношение частичного порядка.

Иерархическую систему можно представить в виде графа с вершинами, помеченными классами, и ребрами, задающими отношение частичного порядка.

Частным случаем является система на множестве объектов, на котором задано отношение полного порядка.

Однако в объектно-ориентированных системах классы (и, соответственно, объекты этих классов) уже связаны частичным отношением наследственности, поэтому в данном случае иерархия строится естественным образом.

Определение 2. HRU-модель объектно-ориентированной системы безопасности называется моделью с естественной иерархией, если в любой момент работы системы для любых двух объектов o^k, $o^{k'}$ классов k и k' таких, что k' является потомком k, для любого поля или метода x, общего для классов k и k', и для любого поля или метода $y \in X$ объекта $o^{k''}$ выполняются следующие соотношения: $o^{k''}.M[o^k, o^{k''}.x] \subset o^{k''}.M[o^{k'}, o^{k''}.x]$ и $o^{k'}.M[o^{k''}, o^{k'}.x] \subset o^k.M[o^{k''}, o^k.x]$. Здесь X - множество всевозможных полей и методов всех существующих в системе на данный момент времени объектов.

В графе системы с естественной иерархией ребра задают отношение наследственности на множестве классов и направлены от родителей к непосредственным потомкам, также имеется выделенная вершина-корень - это пустой класс-прародитель, которым прямо или опосредованно порождены все остальные классы. Класс будем называть финальным, если он не имеет наследников.

Определение 3. Класс объектно-ориентированной системы безопасности называется однородной, если матрицы доступа всех объектов этого класса в любой момент времени совпадают. В этом случае имеет смысл рассматривать матрицу доступа как атрибут не отдельного объекта класса, а класса целиком.

Определение 4. HRU-модель объектно-ориентированной системы безопасности называется однородной, все ее классы однородны.

Определение 5. HRU-модель объектно-ориентированной системы безопасности с естественной иерархией называется ациклической, если граф иерархии ацикличен.

Заметим, что модели с естественной иерархией либо однородны, либо все объекты системы являются представителями классов, не имеющих наследников. Во втором случае матрица доступов объекта o класса k подчиняется соотношениям из определения естественной иерархии как матрица наследника по отношению к матрице самого класса k.

Также отметим, что в работе [2] были рассмотрены свободные неоднородные HRU-модели объектно-ориентированных систем.

2. Однородные объектно-ориентированные системы с естественной иерархией

Определим следующие элементарные операции для работы с матрицей доступов:

1. $Create(o, k)$ – создает объект o класса $k \in K$, при этом $O := O \cup \{o\}$.

2. $Destroy(o, k)$ – уничтожает объект o, при этом $O := O \setminus \{o\}$.

Первые две операции никак не отражаются на матрице доступов.

3. $Enter(r, k, k'.f)$ – вносит право доступа r в $k'.M[k, k'.f]$, при этом матрица доступа изменяется по правилу: $k'.M[k, k'.f] := k'.M[k, k'.f] \cup \{r\}$. Данная элементарная операция может быть выполнена только при следующих условиях:

а) для любого потомка k_1 класса k, $r \in k'.M[k_1, k'.f]$;

б) для любого родителя k_2 класса k', обладающего полем f, $r \in k_2.M[k, k_2.f]$.

4. $Delete(r, k, k'.f)$ – удаляет право r доступа из $k'.M[k, k'.f]$, при этом матрица доступа изменяется по правилу: $k'.M[k, k'.f] := k'.M[k, k'.f] \setminus \{r\}$. Данная элементарная операция может быть выполнена только при следующих условиях:

а) для любого родителя k_1 класса k, обладающего полем f, $r \notin k'.M[k_1, k'.f]$;

б) для любого потомка k_2 класса k', $r \notin k_2.M[k, k_2.f]$.

5. $Grant(k, k'.s)$ – разрешает вызов объекту класса k метода $k'.s$, при этом матрица доступа изменяется по правилу $k'.M[k, k'.s] := 1$. Данная элементарная операция может быть выполнена только при следующих условиях:

а) для любого потомка k_1 класса k $k'.M[k_1, k'.s] = 1$;

б) для любого родителя k_2 класса k', обладающего методом s, $k_2.M[k, k_2.s] = 1$.

6. $Deprive(k, k'.s)$ – запрещает вызов объекту класса k метода $k'.s$, при этом матрица доступа изменяется по правилу $k'.M[k, k'.s] := 0$. Данная элементарная операция может быть выполнена, только при выполнении следующих условий:

а) для любого родителя k_1 класса k, обладающего методом s; $k'.M[k_1, k'.s] = 0$

б) для любого потомка k_2 класса k' $k_2.M[k, k_2.s] = 0$.

Условия а) и б) выполнения операций $Enter$, $Delete$, $Grant$, $Deprive$ будем называть условиями целостности.

Без ограничения общности можно считать, что не обладающий полем (или методом) $x \in X$ класс все же таковым обладает, но это поле пусто (или метод ничего не выполняет), а доступом к такому полю (методу) обладают все классы.

Определение 6. Под состоянием компьютерной системы в момент времени t будем понимать пару $Q(t) = (O(t), M(t))$, где $O(t) = \{o_j\}$ - множество всех объектов системы в момент времени t, а $M(t) = \{k.M[](t), k \in K\}$ - семейство всех матриц доступа классов системы в состоянии на момент t. Начальное состояние системы будем обозначать $Q(0)$.

Состояния компьютерной системы в модели HRU изменяются под воздействием запросов на модификацию матрицы доступа в виде команд $\gamma(x_1 : k_1, \ldots, x_m : k_m)$ следующего формата:

if – условная часть (представляет собой конъюнкцию значений логических выражений вида $r \in k'.M[k, k'.f]$ или $k'.M[k, k'.s] = 1$);

$then$ – последовательность элементарных операций.

Здесь аргументы x_i, $i = 1 \ldots m$ представляют собой поля или функции классов $k_i \in K$, участвующие в качестве аргументов в условной части либо в элементарных операциях. Классы аргументами команды не являются: они определяют саму команду.

Отметим, что помимо уже входящих в команду условий каждая операция $Enter$, $Delete$, $Grant$, $Deprive$ по умолчанию добавляет в if-часть команды свои условия целостности, соединенные операцией конъюнкции. Таким образом, если хотя бы одно условие целостности не будет соблюдено, вся команда не будет выполнена.

Кроме того, не должно существовать команд, содержащих одновременно операции $Enter(r, k, k''.f)$ и $Delete(r, k, k'.f)$, либо $Grant(k, k''.s)$ и $Deprive(k, k'.s)$, либо $Enter(r, k', k.f)$ и $Delete(r, k'', k.f)$, либо $Grant(k', k.s)$ и $Deprive(k'', k.s)$, где k'' - потомок k'. Несоблюдение этого условия приведет к нарушению естественной иерархии. Действительно, пусть, к примеру, $Enter(r, k', k.f)$ и $Delete(r, k'', k.f)$ присутствуют в одной команде, и в момент, когда $r \in k.M[k'', k.f]$, но $r \notin k.M[k', k.f]$, команда выполняется, в результате

чего класс k'' (потомок) теряет право r на поле $k.f$, а родительский класс k' его приобретает.

Все прочие сочетания элементарных операций являются допустимыми, т.к. не могут привести к нарушению естественной иерархии.

Наконец, обратимся к вопросу безопасности однородных систем с естественной иерархией.

Факт перехода системы под действием команды γ из состояния $Q(t)$, в котором она находилась в момент времени t, в состояние $Q(t+1)$, в котором она находилась в следующий момент времени $t+1$, будем записывать в сокращенном виде: $Q(t) \to_\gamma Q(t+1)$.

Определение 7. Будем говорить, что состояние однородной системы $Q(t)$ допускает утечку права доступа $r \in R$, если существуют такие команда γ, класс k и поле $k'.f$ класса k', что $r \notin k'.M[k, k'.f](t)$, но $r \in k'.M[k, k'.f](t+1)$, где $Q(t) \to_\gamma Q(t+1)$.

Определение 8. Будем говорить, что состояние однородной системы $Q(t)$ допускает утечку права вызова, если существуют такие команда γ, класс k и метод $k'.s$ класса k', что $k'.M[k, k'.s](t) = 0$, но $k'.M[k, k's](t+1) = 1$, где $Q(t) \to_\gamma Q(t+1)$.

Определение 9. Будем говорить, что система r-безопасна, если не существует такой последовательность команд $\gamma_1 \dots gamma_T$, что $Q(t-1) \to_{\gamma_t} Q(t)$, $t = 1, 2 \dots T$, и $Q(T)$ допускает утечку права r либо утечку права вызова.

Оказывается, проблема безопасности однородных систем решается очень просто.

Теорема 1. *Проблема r-безопасности системы является NP-разрешимой для однородной объектно-ориентированной модели (в том числе, и для модели с естественной иерархией).*

Доказательство. Мы исходим из предположения, что рассматриваемая система не является динамической относительно своей классовой структуры, в частности, количество классов постоянно и их структура определена заранее. В таком случае конечным является и множество состояний всех матриц доступа этих классов (т.к. множество прав доступа R конечно, и множество классов K конечно). А значит, конечно и количество операций, изменяющих состояния ячеек матрицы доступа. Наконец, отсюда следует, что конечно количество различных команд, которые можно составить из этих операций.

Будем искать цепочку команд наименьшей длины, переводящую систему в состояние, допускающее утечку некоторого права r. Заметим, что такая цепочка не должна содержать команд, не изменяющих ни одну матрицу доступа (как, например, команда, добавляющая право r в ячейку матрицы доступа, в которой право r уже имеется).

Допустим,

$$Q(0) \to_{\gamma_1} Q(1) \to \dots \to_{\gamma_{T-1}} Q(T-1) \to_{\gamma_T} Q(T),$$

где $Q(T)$ допускает утечку права $r \in R$ в ячейку $k.M[k', k.f](T)$.

В такой цепочке одна и та же команда не может выполняться после одинаковых состояний системы: действительно, пусть в цепочке присутствует фрагмент

$$Q(t_1) \to_\gamma Q(t_2)$$

и фрагмент

$$Q(t_3) \to_\gamma Q(t_4),$$

причем

$$Q(t_1) = Q(t_3),$$

но тогда

$$Q(t_2) = Q(t_4),$$

и цепочку можно сократить:

$$Q(t_1) \to_\gamma Q(t_4).$$

Таким образом, ни одна команда не встречается в цепочке большее число раз, чем количество всевозможных состояний системы.

Итак, длина минимальной цепочки, допускающей утечку права, конечна, и количество всевозможных команд в цепочке конечно. Получаем, что нам достаточно перебрать конечное множество цепочек. ■

3. Неоднородные объектно-ориентированные системы с естественной иерархией

Во-первых, отметим, что такие системы довольно сложны в реализации. Действительно, все объекты произвольного класса k обладают (по включению) не меньшим набором прав, чем любой объект каждого класса-родителя k, и не большим, чем любой объект каждого класса-наследника k. С другой стороны, на произвольный метод или поле x каждого объекта класса k предъявляется (по включению) прав не меньше, чем на соответствующий метод любого объекта класса-наследника k, и не больше, чем на соответствующий метод любого объекта класса-родителя k. Причем набор прав доступа любого объекта к любому полю должен изменяться в процессе работы системы без нарушения вышеназванных ограничений.

Во-вторых, в реальных системах, как правило, происходит работа только с объектами финальных классов, а вспомогательные (не являющиеся финальными) классы из цепи, соединяющей прародителя и финальный класс (т.е. все остальные классы этой цепи), не используются, либо объекты этих классов неотличимы в смысле набора прав доступа, т.е. классы являются однородными. В таких системах матрицы доступа объектов нефинального (а значит, однородного) класса будем отождествлять с матрицей доступа самого класса, а набор прав доступа финального класса будем искать как нижнюю грань (по включению) по всем наборам прав объектов этого класса. Таким образом, матрицы доступа на классах можно считать изначально заданными.

Определение 10. Неоднородные объектно-ориентированные системы с естественной иерархией будем называть финально-неоднородными, если все ее нефинальные классы однородны, и для любого финального класса k, его объекта o^k, его поля или метода x и произвольного объекта o', содержащего поле или метод x', выполнено: $o^k.M[o', o^k.x] \subset k.M[o', k.x]$ и $o'.M[k, o'.x'] \subset o'.M[o^k, o'.x']$.

Заметим, что неоднородная система сводится к финально-неоднородной добавлением к каждому неоднородному классу k наследника k', совпадающего со своим родителем во всем, вплоть до прав доступа. Такой наследник будет являться финальным классом, и можно работать с его объектами вместо объектов класса-родителя k, который теперь можно рассматривать как однородный. Данный подход обладает следующим недостатком: пусть k'' - неоднородный (возможно, финальный) класс, для которого k является предком (не обязательно непосредственным), тогда любой объект o'' класса k'' и любой объект o класса k находятся в естественном отношении порядка, однако объект o' класса k' уже находиться с объектом o'' в отношении порядка не будет (и, в частности, может обладать большим по включению набором прав доступа, чем o''), т.е. описанное нами сведение не является (в контекстном смысле) эквивалентным.

Для построения HRU-модели финально-неоднородной объектно-ориентированной системы с естественной иерархией определим следующие элементарные операции:

1. $Create(o^k, k)$ – создает объект o^k класса $k \in K$, при этом $O := O \cup \{o^k\}$, $o^k.M[] := k.M[]$, и в матрицу доступа каждого объекта $o'^{k'}$, $k \in K$ системы добавляется строка, соответствующая объекту o^k, причем $o'^{k'}.M[o^k, o'^{k'}.x] = o'^{k'}.M[k, o'^{k'}.x]$ для всех полей и открытых методов x объекта $o'^{k'}$.

2. $Destroy(o^k)$ – уничтожает объект o^k, при этом $O := O \setminus \{o^k\}$.

3.1 $Enter(r, o^k, o'^{k'}.f)$ – вносит право доступа r в $o'^{k'}.M[o^k, o'^{k'}.f]$, где o^k - объект класса k, $o'^{k'}$ - объект класса k'. При этом матрица доступа изменяется по правилу $o'^{k'}.M[o^k, o'^{k'}.f] := o'^{k'}.M[o^k, o'^{k'}.f] \cup \{r\}$. Данная элементарная операция может быть выполнена, только если k и k' - финальные классы, и выполнены следующие условия целостности:

а) $r \in k'.M[o^k, k'.f]$;

б) $r \in o'^{k'}.M[k, o'^{k'}.f]$.

3.2 $Enter(r, o^k, k'.f)$ – вносит право доступа r в $k'.M[o^k, k'.f]$, где o^k - объект класса k. При этом матрицы доступа изменяются по правилу: $k'.M[o^k, k'.f] := k'.M[o^k, k'.f] \cup \{r\}$, и $o'^{k'}.M[o^k, o'^{k'}.f] := o'^{k'}.M[o^k, o'^{k'}.f] \cup \{r\}$ для всех объектов $o'^{k'}$ класса k'. Данная элементарная операция может быть выполнена, только если k - финальный класс, а k' -однородный, и выполнены следующие условия целостности:

а) $r \in k'.M[k, k'.f]$;

б) для любого родителя k_2 класса k', обладающего полем f, $r \in k_2.M[o^k, k_2.f]$.

3.3 $Enter(r, k, o'^{k'}.f)$ – вносит право доступа r в $o'^{k'}.M[k, o'^{k'}.f]$, где $o'^{k'}$ – объект класса k'. При этом матрицы доступа изменяются по правилу $o'^{k'}.M[k, o'^{k'}.f] := o'^{k'}.M[k, o'^{k'}.f] \cup \{r\}$ и $o'^{k'}.M[o^k, o'^{k'}.f] := o'^{k'}.M[o^k, o'^{k'}.f] \cup \{r\}$

для всех объектов o^k класса k. Данная элементарная операция может быть выполнена, только если k' – финальный класс, а k – однородный, и выполнены следующие условия целостности:

а) для любого потомка k_1 класса k, $r \in o'^{k'}.M[k_1, o'^{k'}.f]$;

б) $r \in k'.M[k, k'.f]$.

3.4 $Enter(r, k, k'.f)$ – вносит право доступа r в $k'.M[k, k'.f]$, при этом матрицы доступа изменяются по правилу $k'.M[k, k'.f] := k'.M[k, k'.f] \cup \{r\}$ и $o'^{k'}.M[o^k, o'^{k'}.f] := o'^{k'}.M[o^k, o'^{k'}.f] \cup \{r\}$ для всех объектов $o'^{k'}$ класса k' и всех объектов o^k класса k. Данная элементарная операция может быть выполнена, только если k и k' – однородные классы и выполнены следующие условия целостности:

а) для любого потомка k_1 класса k, $r \in k'.M[k_1, k'.f]$;

б) для любого родителя k_2 класса k', обладающего полем f, $r \in k_2.M[k, k_2.f]$.

Операции *Delete*, *Grant* и *Deprive* для классов и объектов определяются аналогично операции *Enter* с учетом соответствующих условий целостности.

Определение 11. Под состоянием компьютерной системы в момент времени t будем понимать пару $Q(t) = (O(t), M(t))$, где $O(t) = \{o_j\}$ - множество всех объектов системы в момент времени t, а $M(t) = \{y.M[](t), y \in K \cup O\}$ - семейство всех матриц доступа классов и объектов системы в состоянии на момент t. Начальное состояние системы будем обозначать $Q(0)$.

Состояния компьютерной системы в модели HRU изменяются под воздействием запросов на модификацию матрицы доступа в виде команд $\gamma(x_1 : k_1, \ldots, x_m : k_m)$ следующего формата:

if – условная часть (представляет собой конъюнкцию значений логических выражений вида $r \in y'.M[y, y'.f]$ или $y'.M[y, y'.s] = 1$);

$then$ – последовательность элементарных операций.

Здесь $y, y' \in K \cup O\}$, а каждый аргумент x_i, $i = 1 \ldots m$ представляет собой либо объект, либо поле или функцию класса, либо поле или функцию объекта определенного класса $k_i \in K$, участвующую в качестве аргументов в условной части либо в элементарных операциях. Классы не являются аргументами команды, они определяют саму команду (как содержащиеся в теле команды операторы либо условия): так, например, если x_i - это объект, то он может быть только предзаданного класса k_i. Создаваемый операцией *Create* объект также не является аргументом команды, т.к. важен только его класс.

Напомним, что помимо уже входящих в команду условий каждая операция *Enter*, *Delete*, *Grant*, *Deprive* по умолчанию добавляет в if-часть команды свои условия целостности, соединенные операцией конъюнкции. Таким образом, если хотя бы одно условие целостности не будет соблюдено, вся команда не будет выполнена.

К тому же, не должно существовать команд, содержащих одновременно операции $Enter(r, y, y''.f)$ и $Delete(r, y, y'.f)$, либо $Grant(y, y''.s)$ и $Deprive(y, y'.s)$, либо $Enter(r, y', y.f)$ и $Delete(r, y'', y.f)$, либо $Grant(y', y.s)$ и $Deprive(y'', y.s)$, где y'' - потомок класса y', если $y'' \in K$, и y'' - представитель класса y', если $y'' \in O^{y''}$. Несоблюдение этого условия приведет к нарушению естественной

иерархии. Все прочие сочетания элементарных операций являются допустимыми, т.к. не могут привести к нарушению естественной иерархии.

Наиболее естественный способ избавиться от команд, способных нарушить естественную иерархию, – запретить таким операциям, как *Enter* и *Delete*, находиться в одной команде вообще. Однако такой шаг заметно уменьшит количество систем, попадающих под рассмотрение.

Определение 12. Будем говорить, что HRU-модель объектно-ориентированной системы с естественной иерархией почти монотонна, если ни одна из операций *Create*, *Enter*, *Grant* не может находиться в одной команде с одной из операций *Destroy*, *Delete*, *Deprive*.

Прежде чем приступить к обсуждению проблемы безопасности системы, перепишем пару определений для случая финально-неоднородных систем.

Определение 13. Будем говорить, что состояние финально-неоднородной системы $Q(t)$ допускает утечку права доступа $r \in R$, если существуют такие команда γ, класс либо объект y и поле $y'.f$ класса, либо объекта y', что $r \notin y'.M[y, y'.f](t)$, но $r \in y'.M[y, y'.f](t+1)$, где $Q(t) \to_\gamma Q(t+1)$.

Определение 14. Будем говорить, что состояние финально-неоднородной системы $Q(t)$ допускает утечку права вызова, если существуют такие команда γ, класс, либо объект y и метод $y'.s$ класса либо объекта y', что $y'.M[y, y'.s](t) = 0$, но $y'.M[y, y's](t+1) = 1$, где $Q(t) \to_\gamma Q(t+1)$.

Теорема 2. *Проблема r-безопасности системы является NP-разрешимой для почти монотонной финально-неоднородной объектно-ориентированной модели с естественной иерархией.*

Доказательство. Будем считать, что все объекты одного класса обладают полным доступом друг к другу. В противном случае наша задача не сильно усложнится.

Будем искать цепочку команд наименьшей длины, переводящую систему в состояние, допускающее утечку некоторого права r. По-прежнему считаем, что такая цепочка не должна содержать команд, не изменяющих ни одну матрицу доступа.

Допустим,

$$Q(0) \to_{\gamma_1} Q(1) \to \cdots \to_{\gamma_{T-1}} Q(T-1) \to_{\gamma_T} Q(T),$$

где $Q(T)$ допускает утечку права $r \in R$ в ячейку $M[](T)$.

Во-первых, в такой цепочке не может быть более одной «деструктивной» команды, содержащей операции *Destroy*, *Delete*, *Deprive* (одна такая команда может потребоваться в том случае, если изначально в матрице $M[](0)$ право r содержалось в ячейке, куда впоследствии произойдет его утечка). Также в ней содержится не более $|K|$ команд, содержащих только операции *Create* – по одному новому объекту o^k для каждого класса $k \in K$.

Под o^k будем понимать первый созданный объект класса k (возможно, он присутствовал в состоянии $Q(0)$ системы, но позже был удален единственной командой, содержащей операцию $Destroy$. Пусть теперь o'^k - новый объект класса k, созданный позже o^k. В аргументах всех последующих команд заменяем o'^k на o^k (сделать это возможно, т.к. они одного типа), при этом матрица доступов $o^k.M[](t)$ в любой момент времени будет содержать в точности те права доступа, которые содержали до замены $o^k.M[](t)$ и $o'^k.M[](t)$ в совокупности, поскольку в цепочке больше не встретятся «деструктивные» команды. Если утечка права связана с одним из новых объектов класса k, то она произойдет в новой цепочке не позже, чем в исходной цепочке, т.к. условием выполнения команды является только наличие права, а не его отсутствие. Конечно, при движении по цепочке новые объекты продолжат создаваться, но их можно игнорировать, поскольку в дальнейшем в цепочке никаких операций с ними не производится.

Таким образом, за время работы системы будут изменяться матрицы доступов не более чем $|O(0)| + |K|$ объектов, не более чем $|O(0)| + |K|$ строчек в каждой и не более, чем N столбцов, максимум R прав в каждой ячейке таблицы. Здесь $O(0)$ - множество объектов в состоянии $Q(0)$, а N - наибольшее количество полей и методов в классе. Поскольку каждая команда добавляет хотя бы одно право в какую-то ячейку какой-то матрицы (кроме единственной «деструктивной» команды), то, очевидно, не более чем за $2(|O(0)|+|K|)^2NR+1$ команд произойдет утечка права r. Итак, длина цепочки конечна.

Назовем две команды γ_1 и γ_2 условно-эквивалентными, если при замене в них всех аргументов каждого из классов k на объект o_k, определенный выше, команды будут совпадать по изменению матриц доступа во всех возможных состояниях Q системы. Условная эквивалентность является отношением эквивалентности и, таким образом, разбивает множество команд (вообще говоря, бесконечное) на классы эквивалентности. Таких классов эквивалентности – конечное число, т.к. каждый класс эквивалентности e может представлять команда γ_e, аргументами которой являются лишь поля или методы конечного числа классов объектно-ориентированной системы, а объекты, как однозначно определяемые своим классом, аргументами не являются. Количество классов же в объектно-ориентированной системе мы считаем величиной конечной и постоянной.

На самом деле команды γ_e не обязаны принадлежать системе. Но для цепочки команд представителем каждого класса эквивалентности можно всегда выбирать команду, наименьшую по максимальному числу объектов одного класса в строке аргументов. Таким образом, в допускающей утечку права r цепочке конечной длины на каждом месте может стоять команда из некоторого конечного множества. Получаем, что нам достаточно перебрать конечное множество цепочек.

Заметим, что если объекты одного класса обладают лишь частичным динамично меняющимся доступом к друг другу, в нашем доказательстве достаточно увеличить максимальное количество рассматриваемых новых объектов каждого класса до двух, т.к. возможна ситуация, в которой происходит утечка права доступа объекта класса k к полю другого объекта того же класса, и оба объекта

отсутствовали в $Q(0)$. Но это не повлияет на конечность задачи (придется лишь слегка изменить понятие условной эквивалентности).

■

Литература

1. Белим С.В., Белим С.Ю. Объектно-ориентированная модель компьютерной системы // Математические структуры и моделирование. 2008. Вып. 18. С. 98-101.
2. Белим С.В., Усов С.В. Объектно-ориентированный подход в построении политики безопасности // Математические структуры и моделирование. 2009. Вып. 20. С. 153-159.
3. Michael A. Harrison, Walter L. Ruzzo, and Jeffrey D. Ullman. Protection in operating systems. Communications of the ACM, 19(8):461–471, August 1976.
4. Гайдамакин Н.А. Разграничение доступа к информации в компьютерных системах. Уральский университет, 2003.