Математические структуры и моделирование
2010, вып. 21, с. 119–127

УДК 65.012.810

ОБРАТНАЯ ЗАДАЧА ПОСТРОЕНИЯ МАНДАТНОЙ ПОЛИТИКИ БЕЗОПАСНОСТИ

С.В. Белим, Н.Ф. Богаченко, И.А. Фирдман

В данной работе проводится исследование возможности построения мандатной политики безопасности для компьютерных систем на основе известных правил разграничения доступа.

Введение

Разработка политики безопасности является одной из первых задач, которые приходится решать при построении защищенной системы обработки информации. При этом политика безопасности может строиться как неформально, в виде правил и инструкций, так и формально, в виде строгой математической модели. Несмотря на то, что неформальный подход проще для разработки и внедрения, он существенно уступает формальному в надежности, так как не допускает строгих доказательств гарантированной защищенности. Данное обстоятельство нашло отражение в различных документах, определяющих классы защищенности компьютерных систем. Так, например, в «Оранжевой книге» системы, имеющие неформальную политику безопасности, относятся к классам группы «С», а имеющие формальную политику безопасности — к высшему классу «А1» [8].

Выбор политики безопасности осуществляется исходя из задач, стоящих перед системой защиты. Выработка правил безопасности должна начинаться с четкого определения понятия «угроза безопасности» в контексте заданной системы. С другой стороны, знание злоумышленником политики безопасности системы позволяет выявить слабые места подсистемы защиты. Поэтому политика безопасности должна оставаться максимально закрытой информацией. Однако полностью скрыть правила безопасности невозможно в силу наличия утечки информации через внутренних сотрудников. Также злоумышленник может выявлять элементы политики безопасности экспериментальным путем, выделяя разрешенные и запрещенные потоки.

На сегодняшний день широкое распространение получили три вида формальных политик безопасности – дискреционная, мандатная и ролевая [3, 5]. В данной статье мы ограничимся рассмотрением мандатной политики безопасности и возможностями ее анализа злоумышленником. Основой мандатной поли-

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

тики безопасности является решетка ценностей. Напомним, что *решетка* – это частично упорядоченное множество, в котором каждое двухэлементное подмножество имеет как наименьшую верхнюю (sup), так и наибольшую нижнюю (inf) грани, принадлежащие этому множеству [4, с. 17].

1. Постановка задачи

Как уже было сказано во введении, мандатная политика безопасности строится на базе решетки ценностей, то есть обычной алгебраической решетки, элементы которой играют роль меток безопасности объектов и субъектов компьютерной системы [5]. Обычно задача построения мандатной политики безопасности формулируется следующим образом: задана решетка ценностей L и отображение C множества субъектов $\mathbf{S}$ и объектов $\mathbf{O}$ на решетку

$$C : \mathbf{S} \times \mathbf{O} \to L.$$

Окончательно политика безопасности формулируется определением правила P – сопоставления меток безопасности при обращении на доступ субъекта к объекту. После чего исследуются разрешенные и запрещенные каналы передачи информации. В дальнейшем такой подход будем называть ***прямой задачей*** построения мандатной политики безопасности.

Однако в практике защиты информации часто приходится решать задачу в другой постановке: для системы определены разрешенные и (или) запрещенные каналы передачи информации, требуется построить соответствующую мандатную политику безопасности, то есть решетку ценностей L, отображение C и правило P, приводящую к такому же разделению потоков информации. Такую постановку проблемы будем называть ***обратной задачей*** построения мандатной политики безопасности.

Необходимость решения обратной задачи построения мандатной политики безопасности возникает, как правило, в двух случаях. Во-первых, при формальном построении правил разграничения доступа для организации с уже сложившимися потоками информации, причем нарушение или изменение потоков существенно влияет на работоспособность организации. Во-вторых, при исследовании некоторой системы на наличие каналов утечки информации методом «черного ящика», когда с помощью проб и ошибок можно выявить разрешенные или запрещенные потоки информации и, на их основе, строить предположения о применяемой политике безопасности.

Очевидно, что если две политики безопасности определяются одинаковыми параметрами (L, C, P), то они приводят к одинаковым множествам запрещенных и разрешенных потоков, в противном случае политика безопасности приводила бы к неоднозначным результатам функционирования, что недопустимо в практике защиты информации. Верно и обратное утверждение. Если две политики безопасности определяются двумя одинаковыми параметрами (L, C), или (L, P), или (C, P) и одинаковыми множествами запрещенных и разрешенных потоков, то и третий параметр политики безопасности будет одинаков. Верность этого утверждения также следует из однозначности политики безопасности.

Рассмотрим один из возможных путей решения обратной задачи построения мандатной политики безопасности в рамках субъектно-объектного подхода [5], сводящийся к исследованию различных алгебраических решеток и соответствующих им ориентированных графов. Мандатная политика безопасности определяется тремя параметрами. Будем варьировать два из них произвольным образом и определять третий. Согласно вышесказанному, такое решение будет единственным.

В качестве подбираемого параметра выберем решетку ценностей L. Сделаем предположения относительно отображения C. Пусть в системе каждому субъекту и каждому объекту сопоставляется ровно один элемент из решетки L, однако одному элементу из L может сопоставляться сколько угодно субъектов и объектов компьютерной системы. В качестве правил безопасности выберем разрешение потоков только снизу вверх: доступ $S \xrightarrow{p} O$ разрешен, если:

1) $p = read$ и $C(S) \geqslant C(O)$,
2) $p = write$ и $C(S) \leqslant C(O)$.

Для простоты изложения будем считать, что субъекты компьютерной системы также являются и объектами компьютерной системы. Применим следующий алгоритм построения ориентированного графа G:

1. Каждому объекту сопоставим вершину графа.
2. Если между двумя объектами системы возможны потоки информации в обе стороны, то соответствующие две вершины объединим в одну вершину.
3. Если между двумя объектами системы возможен поток информации только в одну сторону, то добавим в графе соответствующую ориентированную дугу.

Построенный граф задает отношение частичного нестрогого порядка на множестве объектов. Очевидно, что полученное множество не всегда будет являться алгебраической решеткой. В связи с чем можно сформулировать две разновидности задач построения решетки ценностей.

1. ***Полная задача***. Граф G известен полностью и необходимо определить соответствующую ему решетку. В этом случае предполагается, что в системе реализована мандатная политика безопасности и по разрешенным потокам информации решетка ценностей может быть полностью восстановлена.
2. ***Неполная задача***. Граф G известен не полностью, необходимо определить минимальную решетку, включающую его. Предполагается, что в системе реализована мандатная политика безопасности, однако по некоторым причинам известны не все разрешенные и запрещенные потоки.

В данной статье мы ограничимся решением только полной обратной задачи построения мандатной политики безопасности. Полная обратная задача построения мандатной политики безопасности сводится к построению решетки, диаграмма которой изоморфна графу G. В первую очередь необходимо проверить, является ли граф G решеточным [2].

Определение 1. *Решеточным графом* будем называть ориентированный граф, вершины которого образуют решетку, при этом:

- отношение порядка задается ориентированными путями: если в графе существует ориентированный путь $p(r_1, r_2)$, то $r_1 \geqslant r_2$;

- $r = \sup(r_1, r_2) \Longleftrightarrow$

  1. $\exists\ p(r, r_1)\ \&\ p(r, r_2)$, то есть r является верхней гранью.
  2. Если $\exists\ p(r', r_1)\ \&\ p(r', r_2)$, то $\exists\ p(r', r)$, то есть r минимальна среди всех верхних граней.

- $r = \inf(r_1, r_2) \Longleftrightarrow$

  1. $\exists\ p(r_1, r)\ \&\ p(r_2, r)$, то есть r является нижней гранью.
  2. Если $\exists\ p(r_1, r')\ \&\ p(r_2, r')$, то $\exists\ p(r, r')$, то есть r максимальна среди всех нижних граней.

Задача проверки решеточности графа очевидно может быть решена прямым перебором.

Теорема 1. *Трудоемкость проверки решеточности графа прямым перебором не превышает* $O(n^4)$.

Доказательство. Пусть ориентированный граф задан матрицей смежности M размерности $n \times n$. Как будет показано ниже, для проверки решеточности графа следует вычислить *матрицу достижимости* M^*. Согласно [7, с. 176], элемент m^*_{ij} матрицы достижимости равен 1, если в орграфе существует ориентированный путь из вершины r_i в вершину r_j, и 0 – в противном случае. Трудоемкость *алгоритма Уоршелла*, вычисляющего матрицу достижимости ориентированного графа, равна $O(n^3)$ [6, с. 48].

Оценим трудоемкость поиска sup для заданной пары вершин r_i и r_j. Обозначим эту величину $T(\sup)$. Имея в распоряжении матрицу достижимости, за $O(n)$ шагов выбираются вершины, связанные одновременно ориентированными путями с вершинами r_i и r_j (это те вершины r_k, для которых $m^*_{ki} \cdot m^*_{kj} = 1$). Среди отобранных вершин выполняется проверка того, что существует одна и только одна вершина, в которую можно попасть из оставшихся. В худшем случае это можно осуществить за $O(n^2)$ шагов, используя все ту же матрицу достижимости. Тогда трудоемкость $T(\sup) = O(n) + O(n^2) = O(n^2)$.

Очевидно, что поиск inf имеет ту же трудоемкость, что и поиск sup, если в графе инвертировать ориентацию дуг. Для расчетов достаточно транспонировать матрицу достижимости.

Проверка того, что орграф является решеточным, сводится к проверке существования inf и sup для каждой пары вершин. Трудоемкость перебора всех пар равна $O(n^2)$.

В итоге искомая трудоемкость равна $O(n^3) + O(n^4) = O(n^4)$. ■

2. Некоторые решения полной задачи

На сегодняшний день в моделях безопасности компьютерных систем получили широкое распространение три решетки: линейная решетка, решетка подмножеств и MLS-решетка, представляющая собой декартово произведение первых двух [5, с. 14]. Рассмотрим подробно первые две решетки и алгоритмы проверки изоморфности им заданного решеточного графа.

При построении алгоритмов проверки изоморфности необходимо учитывать тот факт, что одной и той же решетке может соответствовать несколько решеточных графов. Как показано в [2], для произвольной решетки существует изоморфный ей решеточный граф, но он не единственен.

Определение 2. Решеточные графы, изоморфные одной и той же решетке, назовем *эквивалентными*.

Определение 3. *Эквивалентные преобразования* решеточного графа – это такие операции над графом, которые оставляют граф решеточным и не меняют изоморфную ему решетку.

Определение 4. Назовем решеточный граф *транзитивным*, если в нем дуга (r_1, r_2) существует тогда и только тогда, когда $r_1 \geqslant r_2$.

Заметим, что решеточный граф, в котором дуга (r_1, r_2) существует, тогда и только тогда, когда не существует ориентированного пути $p(r_1, r_2)$, такого, что $|p(r_1, r_2)| > 1$ (состоящего более чем из одной дуги) называется *диаграммой Хассе* [7, с. 80].

Определение 5. Дуга (r_1, r_2) решеточного графа называется *транзитивной*, если существует вершина r, отличная от r_1 и r_2, такая, что $r_1 \geqslant r$ (существует ориентированный путь $p(r_1, r)$) и $r \geqslant r_2$ (существует ориентированный путь $p(r, r_2)$).

С учетом данных определений можно сформулировать следующие очевидные утверждения.

Предложение 1. *В диаграмме Хассе отсутствуют транзитивные дуги.*

Предложение 2. *Добавление или удаление транзитивной дуги является эквивалентным преобразованием решеточного графа.*

Предложение 3. *Эквивалентный транзитивный решеточный граф можно получить из диаграммы Хассе, пополнив ее всевозможными транзитивными дугами.*

Предложение 4. *Матрица достижимости транзитивного решеточного графа совпадает с его матрицей смежности.*

Предложение 5. *Пусть решеточный граф задан матрицей смежности. Тогда алгоритм Уоршелла строит матрицу смежности транзитивного решеточного графа, эквивалентного исходному.*

Доказательство. Алгоритм Уоршелла строит матрицу достижимости. Интерпретация этой матрицы как матрицы смежности заключается в добавлении к графу всех транзитивных дуг, то есть в построении транзитивного решеточного графа, эквивалентного исходному. ■

Далее будем считать, что ориентированный решеточный граф на n вершинах задан матрицей смежности M.

2.1. Линейная решетка

Под линейно упорядоченным множеством понимается множество (назовем его SL), для любых двух элементов которого определено отношение порядка. Как легко показать, такое множество образует решетку. Наименьшая верхняя и наибольшая нижняя грани для любых двух элементов определяются достаточно просто. Возьмем два элемента $a, b \in SL$, без потери общности будем считать, что $a \geqslant b$, тогда $\sup(a,b) = a$ и $\inf(a,b) = b$. Решетка данного вида является наиболее распространенной в системах защиты информации, она описывает уровни доступа к данным.

Теорема 2. *Трудоемкость проверки изоморфности решеточного графа некоторой линейной решетке не превосходит* $O(n^3)$.

Доказательство. Для проверки изоморфности решеточного графа линейной решетке достаточно показать, что любая пара вершин сравнима, то есть для любой пары вершин существует ориентированный путь, их соединяющий. Как и в теореме 1, можно воспользоваться матрицей достижимости M^*: r_1 и r_2 сравнимы тогда и только тогда, когда либо $(m_{ij} = 1)$, либо $(m_{ji} = 1)$ (одновременно равенство единице выполняться не может, так как решеточный граф не содержит ориентированных циклов [2]). Трудоемкость такой проверки $O(n^2)$. Но трудоемкость вычисления матрицы M^* равна $O(n^3)$. ■

Следствие 2.1. *Трудоемкость проверки изоморфности транзитивного решеточного графа некоторой линейной решетке равна* $O(n^2)$.

Доказательство. Пусть решеточный граф является транзитивным. Тогда согласно предложению 4 его матрица смежности совпадает с матрицей достижимости. Отсюда следует, что трудоемкость проверки изоморфности графа линейной решетке равна $O(n^2)$ (так как достаточно для каждой пары вершин выполнить проверку: $(m_{ij} = 1)$ или $(m_{ji} = 1)$). ■

2.2. Решетка подмножеств

Пусть задано множество X. Рассмотрим множество всех его подмножеств $SX = \{A | A \subseteq X\}$. Введем на множестве SX отношение порядка: $\forall A, B \in SX : \ A \geqslant B \Leftrightarrow A \supseteq B$. Легко доказать, что отношение $A \supseteq B$ есть отношение нестрогого порядка, а множество SX частично упорядоченно: так не для любой пары элементов определено отношение порядка. Наибольшую

нижнюю и наименьшую верхнюю грани определим на основе операций пересечения и объединения множеств: $\sup(A, B) = A \cup B$, $\inf(A, B) = A \cap B$. Также нетрудно доказать, что $(SX, \supseteq, \cup, \cap)$ – алгебраическая решетка.

Примерами использования данной решетки в реальных компьютерных системах может служить множество атрибутов файла и зависимость выходной величины от подмножества множества входных элементов.

Теорема 3. *Трудоемкость проверки изоморфности решеточного графа некоторой решетке подмножеств не превосходит* $O(n^3)$.

Доказательство. Алгоритм проверки изоморфности решеточного графа решетке подмножеств состоит из следующих этапов.

1. Проверяется, является ли число вершин графа степенью двойки (так как число узлов решетки подмножеств совпадает с мощностью булеана $\mathcal{B}(X)$ некоторого исходного множества $X = \{x_1, \ldots, x_k\}$ и равна 2^k, где $k = |X|$). Трудоемкость этого этапа не зависит от n.
2. Строится матрица достижимости M^*. Трудоемкость этого этапа $T_2 = O(n^3)$.
3. Каждой вершине графа сопоставляется некоторое подмножество множества X. Мощность каждого из этих подмножеств не превосходит k.

   Для формирования подмножеств на начальном шаге всем вершинам r_i приписывается множество $R_i = \varnothing$ (трудоемкость $O(n)$).

   Далее ищется сток (вершина без исходящих дуг) r_s. Согласно [2], такая вершина существует и единственна. Стоком является вершина, для которой строка в матрице смежности M состоит из одних нулей. Трудоемкость поиска стока $O(n^2)$.

   Затем выбираются вершины, имеющие одну и только одну исходящую дугу, причем эта дуга должна вести в сток (назовем их одноэлементными вершинами): для каждой такой вершины r_j в матрице смежности элемент $m_{js} = 1$, а остальные элементы этой строки – нулевые. Число одноэлементных вершин должно равняться $k = \log_2 n$. Трудоемкость поиска $O(n^2)$.

   Соответствующие одноэлементным вершинам множества пополняются: $R_{j_v} := R_{j_v} \cup \{x_v\}$ $(v = 1, \ldots, k)$. Трудоемкость пополнения $k \cdot T(\cup) = \log_2 n \cdot T(\cup)$, где $T(\cup)$ – трудоемкость операции объединения множеств.

   Окончательно множества, связанные с вершинами графа, формируются следующим образом: если элемент матрицы достижимости $m^*_{ij_v} = 1$ (существует путь из вершины r_i в одноэлементную вершину r_{j_v}), то $R_i := R_i \cup \{x_v\}$. Трудоемкость формирования множеств R_i для всех вершин равна $n \cdot k \cdot T(\cup) = n \cdot \log_2 n \cdot T(\cup)$.

   Будем считать, что множество реализовано при помощи двоичных векторов, тогда $T(\cup) = O(k) = O(\log_2 n)$ [1, с. 109]. Трудоемкость третьего этапа $T_3 = O(n) + O(n^2) + O(n^2) + O((\log_2 n)^2) + O(n \cdot (\log_2 n)^2) = O(n^2)$.
4. Проверяется, что $\{R_1, \ldots, R_n\} = \mathcal{B}(X)$. Очевидно, достаточно проверить, что все R_i $(i = 1, \ldots, n)$ различны. Трудоемкость этой процедуры $n^2 \cdot T(\lesseqgtr)$, где $T(\lesseqgtr)$ – трудоемкость операции сравнения множеств.

Если по-прежнему используется реализация множества при помощи двоичных векторов, то $T(\leqslant) = O(\log_2 n)$. Тогда трудоемкость четвертого этапа $T_4 = O(n^2 \cdot \log_2 n)$.

В итоге получаем, что результирующая трудоемкость представленного алгоритма проверки изоморфности решеточного графа решетке подмножеств равна $O(n^3) + O(n^2) + O(n^2 \cdot \log_2 n) = O(n^3)$. ■

Следствие 3.1. *Трудоемкость проверки изоморфности транзитивного решеточного графа некоторой решетке подмножеств равна* $O(n^2 \cdot \log_2 n)$.

Доказательство. Пусть решеточный граф является транзитивным. Тогда в алгоритме, представленном в доказательстве теоремы 3, этап 2 можно опустить, так как согласно предложению 4 в этом случае матрица смежности будет совпадать с матрицей достижимости. Отсюда следует, что трудоемкость проверки изоморфности графа решетке подмножеств равна $O(n^2 \cdot \log_2 n)$. ■

Таким образом, если есть дополнительная информация о типе используемой решетки, то задача восстановления политики безопасности упрощается.

Решетка подмножеств играет существенную роль при построении мандатной политики безопасности. Следующая теорема показывает, что любую мандатную политику безопасности можно свести к построенной на решетке подмножеств при условии, что будут использоваться не все элементы решетки. Такая ситуация возможна на практике, когда в качестве меток безопасности используется тематический классификатор, но при этом задействованы не все тематики.

Теорема 4. *Любая решетка изоморфна некоторой подрешетке решетки подмножеств.*

Доказательство. Пусть дана конечная решетка S с вершинами $\{a_0, a_1, \ldots, a_n\}$. Будем считать, что a_0 – нижняя грань всей решетки, a_n – верхняя грань всей решетки. В силу антисимметричности отношения порядка, в решетке S отсутствуют различные вершины a_i и a_j такие, что $a_i \leqslant a_j$ и $a_j \leqslant a_i$. Покажем, что в этом случае S изоморфна некоторой подрешетке решетки всех подмножеств множества $A = \{1, 2, \ldots, n\}$. Для этого сопоставим каждой вершине a_i множество $A_i = \{k \in A | a_k \leqslant a_i\}$ ($A_0 = \varnothing$, $A_n = A$). При этом очевидно $A_i \subseteq A_j$, если $a_i \leqslant a_j$. Обратно, если $A_i \subseteq A_j$, то из $i \in A_i$ следует $i \in A_j$ и $a_i \leqslant a_j$. Таким образом отображение $\varphi : S \to A$, определенное как $\varphi(a_i) = A_i$ $(i = 0, \ldots, n)$, сохраняет отношение порядка. Кроме того φ инъективно, поскольку из $A_i = A_j$ следует $A_i \subseteq A_j$ и $A_j \subseteq A_i$, затем $a_i \leqslant a_j$ и $a_j \leqslant a_i$, откуда следует $a_i = a_j$. Поэтому φ является вложением решетки S в решетку подмножеств множества A. ■

Таким образом, решение обратной задачи можно начинать с построения решетки подмножеств, а затем уже пытаться оптимизировать решение.

3. Заключение

Обратная задача построения мандатной политики безопасности разрешима в общем случае за полиномиальное время. Данный результат существенен при построении соответствующих программных комплексов, так как гарантирует получение результата за приемлемое время. Наличие дополнительной информации о типе используемой решетки существенно уменьшает время решения задачи.

Литература

1. Ахо А.В., Хопкрофт Д.Э., Ульман Д.Д. Структуры данных и алгоритмы. М.: Издательский дом «Вильямс», 2000. 384 с.
2. Белим С.В., Богаченко Н.Ф., Ракицкий Ю.С. Совместная реализация мандатного и ролевого разграничения доступа к информации в компьютерных системах // Математические структуры и моделирование. 2009. Омск: ООО «УниПак». Вып. 20. С. 141-152.
3. Гайдамакин Н.А. Разграничение доступа к информации в компьютерных системах. Екатеринбург: Изд-во Урал. ун-та, 2003. 328 с.
4. Гретцер Г. Общая теория решеток / Под ред. Д.М. Смирнова. М.: Мир, 1981. 456 с.
5. Девянин, П.Н. Модели безопасности компьютерных систем. М.: Издательский центр «Академия», 2005. 144 с.
6. Новиков Ф.А. Дискретная математика для программистов. СПб.: Питер, 2001. 304 с.
7. Хаггарти Р. Дискретная математика для программистов / Пер. с англ. М.: Техносфера, 2005. 400 с.
8. URL: http://www.securitylab.ru/informer/240650.php (дата обращения: 22.03.2010).