УДК 681.3.06

В.В. Кручинин, А.А. Шелупанов

Подходы к созданию защищенного архива на основе разделения секрета

Рассматриваются подходы к созданию защищенных баз данных, основанные на разделении реляционных таблиц на две части: кодирования отношений и множеств доменов, заданных деревьями И/ИЛИ. Приводится обобщенная структура системы, основанная на таком подходе. Даются приблизительные оценки размеров доменов.

Введение

Принцип разделения секрета предполагает разделение информации на части между участниками таким образом, что только данная группа участников может восстановить секрет, но ни одна другая группа не может восстановить этот секрет [1]. Применительно к реляционным базам данных предлагается разделить таблицы отношений на две части: таблицу кодов и множество доменов. Таблица кодов содержит коды кортежей, полученных из номеров значений атрибутов, хранящихся в доменах. Рассмотрим эту идею более подробно. Каждому кортежу декартового произведения множеств степенью n можно поставить в соответствие число и вместо кортежа в базе данных хранить это число. Для этого зададим отображение

$$F : A_1 \times A_2 \times \ldots \times A_n \to N_n ,$$

где $A_1 \times A_2 \times \ldots \times A_n$ — декартово произведение множеств; N_n — множество номеров $\overline{0,n}$.

Если F биективно, то можно задать обратное отображение $F^{-1} : N_n \to A_1 \times A_2 \times \ldots \times A_n$.

Таким образом, биективное отображение F задает алгоритм идентификации кортежа декартового произведения:

$$\text{num} = \text{Rank}(D, a) ,$$

где $a \in A_1 \times A_2 \times \ldots \times A_n$, $\text{num} \in N_n$, D — описание множеств декартового произведения $A_1 \times A_2 \times \ldots \times A_n$. Отображение F^{-1} задает алгоритм генерации значения кортежа по номеру

$$a = \text{Generate}(D, \text{num}) ,$$

где $a \in A_1 \times A_2 \times \ldots \times A_n$, $\text{num} \in N_n$, D — описание множеств. Тогда отношение $R \subset A_1 \times A_2 \times \ldots \times A_n$ можно однозначно представить подмножеством целых чисел $\text{num} \subset N_n$.

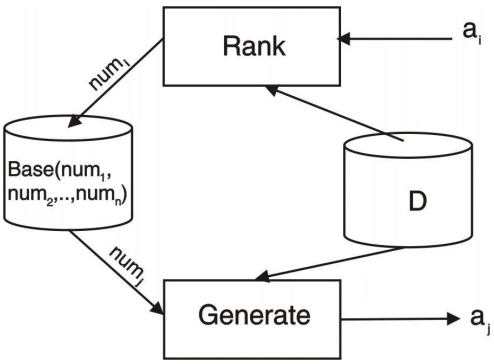

Рис. 1. Описание структуры базы данных

Используя алгоритмы Rank и Generate, можно предложить следующую структуру базы данных (рис. 1). При записи кортежа в базу данных работает алгоритм Rank, который присваивает номер данному кортежу. Далее этот номер хранится в базе данных. При выборке данных из базы работает алгоритм Generate, который по заданному номеру получает кортеж. Важным элементом является описание множеств декартового произведения D. Рассмотрим подробнее способы организации D, Rank, Generate.

1. Механизм реализации

Рассмотрим способ построения описаний множеств значений доменов D, алгоритмов идентификации Rank и генерации Generate. В качестве такого инструмента предлагается использовать деревья И-ИЛИ [2]. Правила построения дерева И-ИЛИ следующие:

1. Если некоторое множество разбивается на n множеств $\{A_i\}_{i=1}^{n}$, то это разбиение можно представить ИЛИ-узлом. При этом должно быть выполнено следующее условие:

$$A_i \cap A_j = \varnothing, \; i \neq j . \qquad (1)$$

2. Если искомое множество является комбинацией элементов из n множеств, то данное преобразование представляется И-узлом. В этом случае условие (1) не требуется, необходимо, чтобы комбинация была уникальной.

Листьями такого дерева являются элементы или множества, разбиение которых не производится. Используя два этих правила, можно строить деревья И-ИЛИ для описания различных классов множеств.

Вариантом дерева И/ИЛИ назовем дерево, которое получается из заданного путем отсечения дуг, кроме одной, у всех ИЛИ-узлов. Корнем варианта будет являться корень дерева И-ИЛИ. На рис. 2 показан пример дерева И-ИЛИ и всех его вариантов.

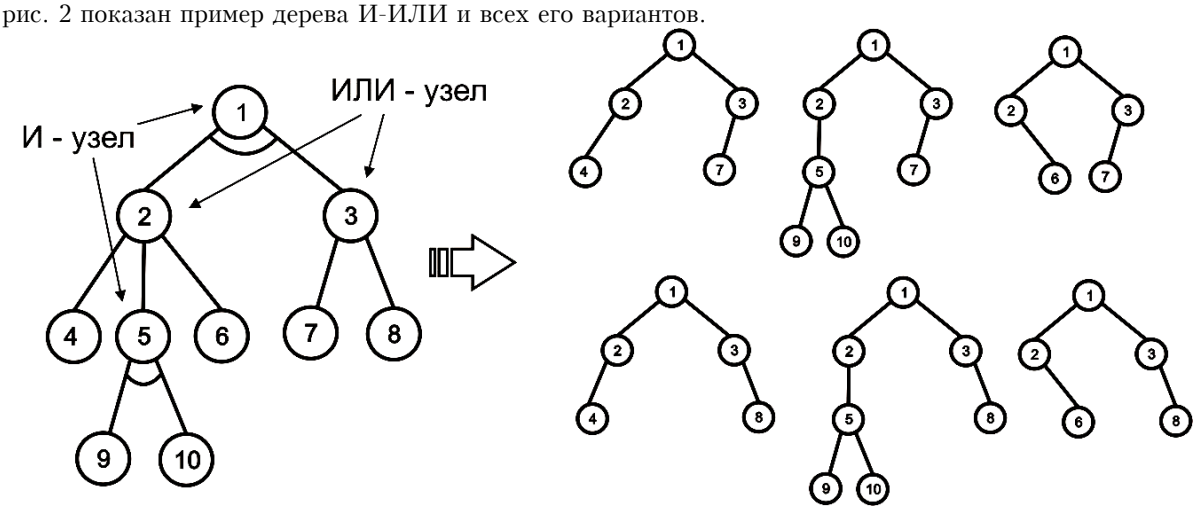

Рис. 2. Дерево И-ИЛИ и все его варианты

Если дерево описывает некоторое множество, то вариант описывает один элемент множества. Общее число вариантов в дереве (или мощность множества) можно вычислить по формуле:

$$\omega(z) = \begin{cases} \sum_{i=1}^{n} \omega(s_i^z) & \text{для ИЛИ-узла,} \\ \prod_{i=1}^{n} \omega(s_i^z) & \text{для И-узла,} \\ 1, & \end{cases} \qquad (2)$$

где z — рассматриваемый узел дерева; $\{s_i^z\}$ — множество сыновей узла z; n — число сыновей.

Тогда, зная $\omega(z)$ для каждого узла, можно предложить следующий алгоритм генерации варианта (Generate):

1. Корень дерева записывается в вариант и заносится в стек $Stack \overset{push}{\leftarrow} < s_{root}, L >$.

2. Из стека вынимается пара $< z, l_z > \overset{pull}{\leftarrow} Stack$. Если стек пуст, то завершить работу.

3. Определяется тип текущего узла. Если это И-узел, то переход на шаг 4, иначе переход на шаг 5.

4. Все сыновья $\{s_j^z\}_{j=1}^m$ рассматриваемого узла z записываются в данный вариант V, вычисляется $l_A(s_i^z)$, используя выражение

$$l_A(s_i^z) = \begin{cases} \dfrac{l_A(z)}{\prod_{j=1}^{i-1} \omega(s_j^z)} \bmod \omega(s_i^z) & i > 1, \\ l_A(z) \bmod \omega(s_i^z) & i = 1, \end{cases} \qquad (3)$$

и пары $< s_j^z, l_A(s_j^z) >$ заносятся в стек.

5. Если это ИЛИ-узел, то используя выражение

$$l_O(s_k^z) = \begin{cases} l_O(z) & \text{при } l_O(z) < \omega(s_k^z),\ k = 1, \\ \min\limits_k [l_O(z) - \sum_{j=1}^{k} \omega(s_j^z)] & \text{при } l_O(s_k^z) \geq 0,\ k > 1, \end{cases} \qquad (4)$$

определяется единственный сын s_k^z и $l_o(s_k^z)$. Сын записывается в вариант V, а пара $< s_k^z, l_o(s_k^z) >$ заносится в стек.

6. Переход на шаг 3.

Анализ данного алгоритма показывает, что временная сложность пропорциональна количеству узлов, которые заносятся в стек, следовательно, пропорциональна числу узлов в варианте. При этом количество делений равно числу сыновей всех И-узлов варианта плюс число сложений и сравнений для ИЛИ-узлов [см. выражения (3) и (4)].

Построим алгоритм нумерации варианта для данного дерева И-ИЛИ. Для этого необходимо найти сопоставление варианта V в дереве D и нахождение соответствующего номера i. Сопоставление производится следующим образом:

Первоначально в стек M_1 заносится корень варианта V, в стек M_2 – корень дерева D.

Если стек M_1 пуст, то завершить работу алгоритма.

Из стека M_1 извлекается узел варианта dv, а из стека M_2 извлекается узел d.

Если это узлы И, то все сыновья dv заносятся в стек M_1, а сыновья d заносятся в M_2. Переход на шаг 2.

Если это узлы ИЛИ, то сын dv ищется в множестве сыновей узла d. Если найдено совпадение, то сыновья заносятся в стек. Переход на шаг 2.

Если dv и d листья, то они удаляются из стека.

Вычисление номера начинаем производить с рассмотрения листьев варианта V. Все листья варианта имеют значения $\omega(z) = 1$.

После того как сопоставление найдено, выполняем следующие действия:

1. Для каждого И-узла z вычисляем

$$l_z = l_1 + \omega(s_1)(l_2 + \omega(s_2)(\ldots(l_n)\omega(s_{n-1}))\ldots)),$$

где $\{s_i\}_{i=1}^n$ – сыновья узла z, а $\{l_i\}_{i=1}^n$ – соответствующие номера, полученные для сыновей.

2. Для каждого ИЛИ-узла вычисляем $l_z = \sum_{i=1}^{k-1} \omega(i) + l_1$, где k – номер соответствия для узла ИЛИ в дереве D, l_1 – номер варианта для этого сына. Рекурсивно производим вычисления номера, пока не достигнем корня дерева. Полученное число l_z для корня варианта будет номером варианта, т.е. $V = R(l_z)$. Очевидно, что $l_z \le \omega(z)$. Таким образом, для множества, представленного деревом И-ИЛИ, можно создать алгоритмы Rank и Generate.

2. Преобразование таблицы атрибутов в дерево И-ИЛИ

Рассмотрим построение дерева И-ИЛИ для таблицы атрибутов. Поскольку значение $a \in A_1 \times A_2 \times \ldots \times A_n$ является комбинацией элементов из множеств $\{A_i\}_{i=1}^n$, то корень дерева будет И-узлом, имеющим n сыновей, каждый i-й сын соответствует множеству A_i, графическое изображение такого соответствия показано на рис. 3.

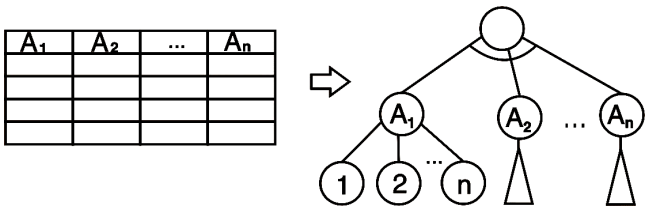

Рис. 3. Соответствие между таблицей и деревом И-ИЛИ

Общее число множества значений вычисляется по формуле

$$\omega(T) = \prod_{i=1}^{n} \omega(A_i).$$

Далее для каждого множества A_i строится свое дерево И-ИЛИ. В общем случае можно выделить следующие типы:

1. Множество значений A_i представлено справочником.
2. Множество значений A_i представлено числовым интервалом.
3. Множество значений A_i представлено деревом И-ИЛИ.

Для представления множества уникальных объектов, которые используются в базе данных некоторого домена, используется справочник. Справочник имеет две части, первая часть содержит пронумерованные уникальные объекты, вторая часть резервная, предназначена для внесения новых объектов. Соответствие между справочником и деревом И-ИЛИ показано на рис.4. Справочник представляется ИЛИ-узлом, а все сыновья являются элементами справочника. Тогда общее число вариантов дерева (или элементов множества) равно

$$\omega(A_i) = n + m.$$

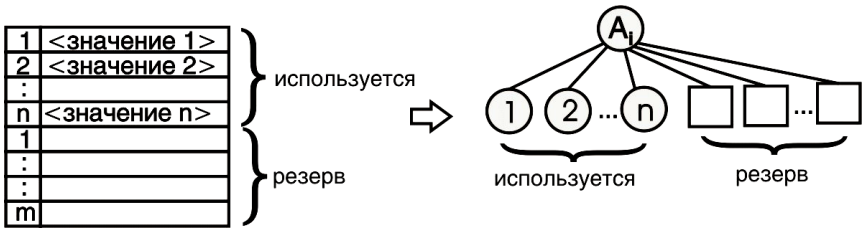

Рис. 4. Соответствие между справочником и деревом И-ИЛИ

Для представления числового интервала задаются границы и шаг, тогда данное множество можно представить деревом И-ИЛИ, которое имеет ИЛИ-узел в качестве корня, а сыновья – конкретные значения чисел из этого интервала. Графическое изображение такого дерева показано на рис. 5.

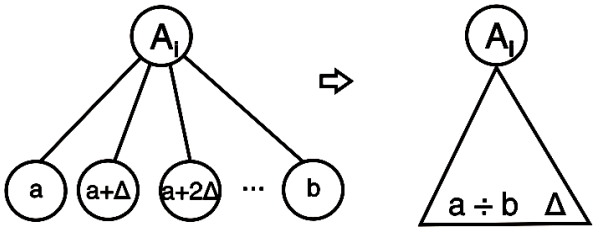

Рис. 5. Дерево для представления числа

Тогда общее число вариантов (элементов множества) будет

$$\omega(A_i) = \frac{b-a}{\Delta}.$$

Множество значений A_i может быть представлено деревом И-ИЛИ. Рассмотрим несколько наиболее распространенных примеров. Если A_i – это дата, то ее можно представить деревом И-ИЛИ (рис. 6)

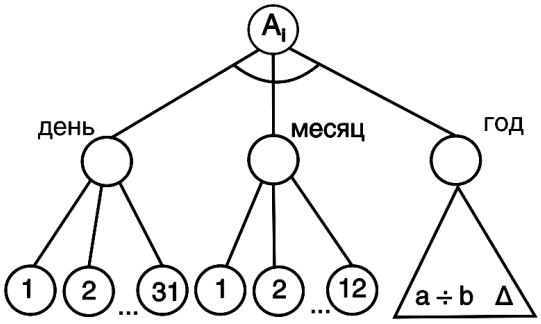

Рис. 6. Дерево И-ИЛИ для представления даты

Здесь при описании даты год представлен некоторым числовым интервалом. Например, 1950–2050, Δ=1. Тогда общее число вариантов может быть представлено формулой

$$\omega(\text{Дата}) = \omega(\text{день}) \cdot \omega(\text{месяц}) \cdot \omega(\text{год}).$$

Аналогично может быть представлен атрибут «время».

3. Пример организации защищенного архива

Рассмотрим применение данного подхода для организации архива удостоверяющего центра с историями сертификатов открытого ключа [3]. Структура такого архива определяется атрибутами полей stdat стандарта сертификата x.509. Перечислим их:

C (страна),
L (размещение),
ST (штат или провинция),
O (организация),
OU (подразделение),
CN (фамилия, имя, отчество),
STREET (адрес),
E (электронный адрес).

На основе данного стандарта каждый удостоверяющий центр формирует свои собственные атрибуты полей. Например, в поле CN могут быть занесены паспортные данные, номер свидетельства ИНН, номер пенсионного свидетельства и т.д. Для разделения архива на две части необходимо стандартные атрибуты представить в виде дерева И/ИЛИ (рис. 7).

Далее рассматриваем структуру каждого атрибута. Если это атрибут C (страна), то соответствующий домен можно представить простым справочником. Если это атрибут CN, то для него можно задать некоторую структуру. Например, CN = {фамилия, имя, отчество, номер свидетельства ИНН}.

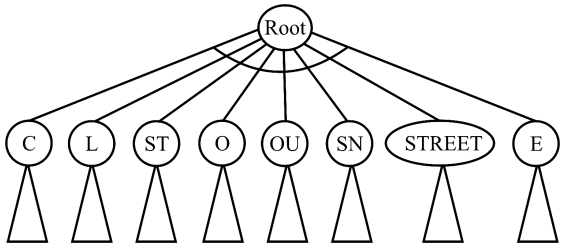

Рис. 7. Дерево И-ИЛИ для атрибутов сертификата

Рассмотрим подробнее домены для построения CN. Уже имеются достаточно большие справочники по именам и фамилиям в Интернете. Так, например, одна из самых объемных баз имен СССР насчитывает около 8000, база фамилий – около 170000 [4]. Очевидно, что число отчеств не будет превышать числа имен. Это говорит, что размеры доменов не такие большие. Предположим, что мощность доменов будет прирастать 5% в год, тогда для доменов имен, отчеств и фамилий необходимо зарезервировать 8000, 8000, 170000 записей дополнительно. Тогда общее число кортежей будет:

$$\omega(\text{ФИО}) = \omega(\text{имя}) \cdot \omega(\text{отчество}) \cdot \omega(\text{фамилия}) = 16000 \times 16000 \times 340000 = 8704 \cdot 10^{10}.$$

Аналогичная ситуация будет для адресов: город, улица, номер дома, почтовый индекс.

Мощности доменов будут измеряться $10^4 - 10^6$. Тогда размеры кортежей возрастут:

$$\omega(root) = \sum_{k=1}^{n} \omega_k ,$$

где n – общее число доменов; ω_k – мощность k-го домена. Предположим, что $n = 15$.

С учетов вышеизложенного

$$\omega(root) \approx 10^{100}.$$

Тогда для представления кода кортежа полученного алгоритмом Rank, необходимо 100 байт при байтовом представлении десятичных цифр, 50 байт при двоично-десятичной системе кодирования, или 333-битовое число.

Таким образом, если база данных насчитывает 10^8 кортежей, то объем ее не будет превышать 10^{10} байт.

4. Обобщенная структура архива

Обобщенная структура архива показана на рис. 8.

Основные модули и подсистемы:

1) подсистема ввода сертификата – обеспечивает ввод полей сертификата;
2) подсистема поиска – обеспечивает контекстный поиск в архиве;
3) подсистема управления доменами – обеспечивает поиск, занесение заданных значений полей;
4) модуль Rank – обеспечивает формирование номера (кода) для данного сертификата;
5) модуль Generate – обеспечивает получение номеру (коду) в базе данных формирование соответствующего кортежа сертификата;
6) Idx_k – индексный файл для организации поиска;
7) Base – база данных, хранящая коды кортежей;
8) D(K) – домен, хранящий значения атрибута K (C,L,ST,O,OU,CN);
9) S_i – кортеж сертификата;
10) num_i - код сертификата.

Рассмотрим работу системы по обобщенной схеме, представленной на рис. 8. Внесение сертификата в архив производится следующим образом: значения полей сертификата заполняются в системе ввода. Далее вызывается модуль Rank, который в соответствии с алгоритмом нумерации варианта, строит вариант, находит значения соответствующих полей в доменах. Если значение в домене найдено, то соответствующий номер возвращается в Rank, если нет, то данное значение заносится в домен и его номер возвращается в Rank. Из полученных номеров полей сертификата формируется код сертификата. Если такого кода не существует, то код записывается в базу данных Base и в соответствии с ключами производятся записи в индексные файлы.

Подсистема поиска организована следующим образом. Из множества атрибутов сертификата выделяются подмножества, по которым формируются индексные файлы, в которых, в частности, хранятся ссылки на коды сертификатов в базе Base. При необходимости просмотреть значения полей сертификата передается код сертификата numi в модуль Generate, который производит получение сертификата на основе алгоритма генерации варианта в дереве И-ИЛИ.

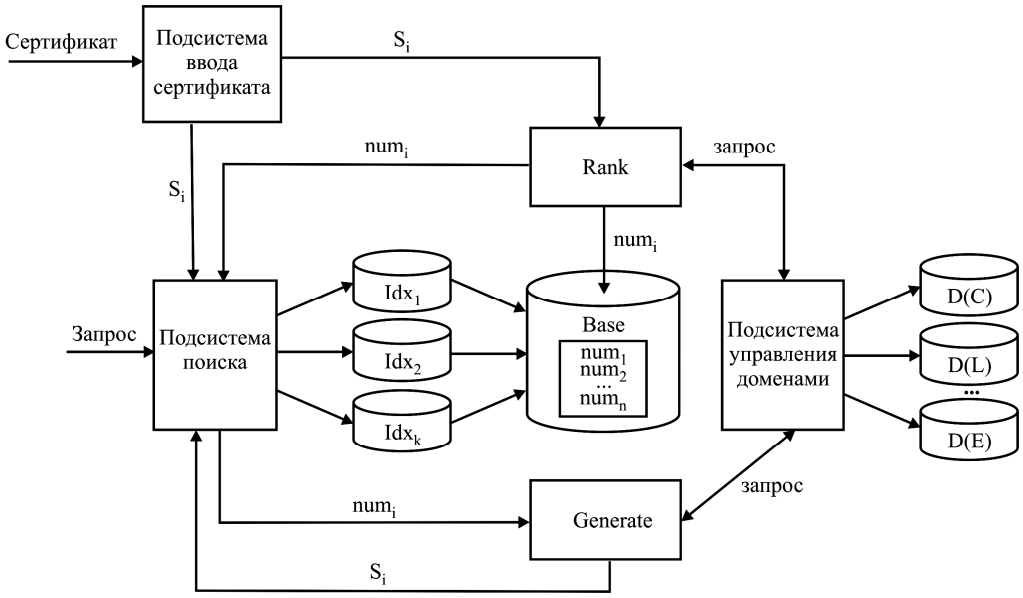

Рис.8. Структура архива

Подсистема управления доменами обеспечивает управление доменами, представленными в виде справочников. Например: справочник имен, справочник фамилий, справочник организаций, справочник городов, справочник названий улиц. Поскольку для формирования кода сертификата необходимо знать фиксированный размер справочника, то размер следует установить в соответствии с выражением

<Размер справочника>=<текущий размер>+k* <число новых за год>,

где k — количество лет, на которые рассчитан срок активной работы архива.

Подсистема управления доменами может быть распределенна и реализована на федеральном уровне. За каждый домен может отвечать отдельная организация, которая обеспечивает эффективное управление данным доменом.

Основные выводы

Такой подход к созданию архивов обеспечивает:

1. Создание условий повышения уровня защищенности, поскольку информация сертификата разделена на части, каждая из которых не дает возможность получить доступ к сертификату.
2. Экономический эффект, поскольку размер базы данных уменьшается за счет того, что основная информация хранится в доменах.

Литература

1. Блейкли Г.Р., Кабатянский Г.А. Обобщенные идеальные схемы, разделяющие секрет, и матроиды // Проблемы передачи информации. 1997. – Т. 33, вып. 3. – С. 102–110.

2. Кручинин В.В. Методы построения алгоритмов генерации и нумерации комбинаторных объектов на основе деревьев И/ИЛИ. – Томск: «В-Спектр», 2007. – 200 с.

3. Шелупанов А.А. и др. Основы информационной безопасности. – М.: Горячая линия – Телеком, 2006. – 544 с.

4. http://familii.jino-net.ru/

Кручинин Владимир Викторович
К.т.н, зав. лабораторией инструментальных систем моделирования и обучения, ТУСУР
Тел.: 45 10 00
Эл. почта: kru@tcde.ru

Шелупанов Александр Александрович
Д.т.н., проф., зав. каф. КИБЭВС, ТУСУР
Тел.: 41 34 26

V.V. Kruchinin, A.A. Shelupanov
Approaches to designing protected archives based on secret division

Approaches to creation of the protected databases, based on division of relational tables into two parts are considered: coding of relations and domains set by OR/AND trees. The generalized structure of system based on such approach is resulted. Rough estimates of the sizes of domains are given.