УДК 004.056.5

ПРИМЕНЕНИЕ АСИММЕТРИЧНЫХ АЛГОРИТМОВ ШИФРОВАНИЯ ДЛЯ УПРАВЛЕНИЯ ДОСТУПОМ К ИЕРАРХИЧЕСКОЙ ИНФОРМАЦИИ

Сидоров Дмитрий Петрович
ФГБОУ ВПО «Мордовский государственный университет им. Н.П.Огарева»
E-mail: sidorovd@mail.ru

Аннотация. В большинстве компьютерных систем вся обрабатываемая информация разделяется в зависимости от своей важности на так называемые «классы безопасности», которые образуют иерархию. Кроме того, характерной является ситуация, когда информация внутри каждого «класса безопасности» дополнительно зависит от времени ее размещения в данном классе. В статье рассматривается методика управления доступом пользователей к информационным ресурсам, основанная на применении асимметричных алгоритмов шифрования.
Ключевые слова: асимметричный алгоритм; шифрование; управление доступом; иерархическая информация.

Одной из непременных составляющих современных информационных технологий является объединение вычислительных ресурсов организации в единую корпоративную сеть. Корпоративная сеть, как правило, является территориально распределенной, то есть объединяющей подразделения и структуры, находящиеся на значительном удалении друг от друга. Часто узлы корпоративной сети оказываются расположенными в различных городах, а иногда и странах. Внутри корпоративной сети определена специальная политика, описывающая используемые аппаратные и программные средства, правила получения доступа пользователей к сетевым ресурсам, правила управления сетью, контроль использования ресурсов и дальнейшее развитие сети.

При реализации корпоративных информационных систем на базе технологий Internet/intranet важнейшими вопросами являются: организация защиты информации, централизованное управление информационными ресурсами, разграничение доступа к ресурсам. Особенно это важно при организации доступа пользователей из внешних сетей к ресурсам корпоративной информационной системы, так называемая extranet-технология.

В большинстве компьютерных систем вся обрабатываемая информация разделяется в зависимости от своей важности на так называемые «классы безопасности», которые образуют иерархию. Причем эта иерархия, как правило, является древовидной, например, как на рисунке 1.

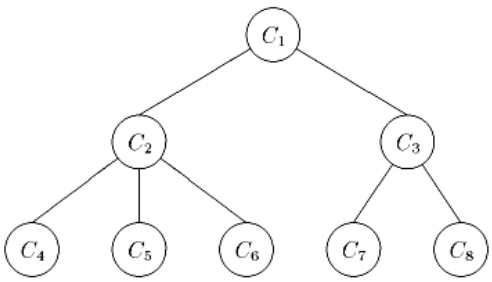

Рис. 1. Пример иерархии классов.

Кроме того, характерной является ситуация, когда информация внутри каждого класса безопасности дополнительно упорядочена по временным (ударение на букву «ы») интерва-

лам.

Рассмотрим задачу управления доступом к иерархической информации. Пусть C_i, $i = \overline{1,m}$ – классы информации. Предположим, что они упорядочены при помощи некоторого бинарного отношения «≤». При этом очевидным является следующее требование: если пользователь имеет доступ в класс C_i, то он должен иметь доступ и в любой класс C_j такой, что $C_j \le C_i$.

Использование шифрования для управления доступом означает, что для каждого класса C_i назначается ключ K_i и вся информация в этом классе шифруется при помощи данного ключа. В этом случае, пользователь, получая доступ к классу C_i вместе с ключом K_i должен хранить и ключи от всех нижестоящих классов. Данный подход неэффективен если иерархия содержит большое количество классов и пользователь имеет доступ к одному из классов, находящихся в вершине иерархии.

Более эффективным является метод, основанный на использовании иерархических ключей шифрования. Суть его заключается в следующем: ключ шифрования K_i для класса C_i назначается таким образом, что из него можно вычислить K_j, но только для классов, удовлетворяющих условию $C_j \le C_i$. Таким образом, пользователь, получив доступ в класс C_i, может расшифровать данные в классе $C_j \le C_i$.

Проблема назначения криптографического ключа для работы с иерархическими данными впервые была изучена Аклем (Akl) и Тэйлором (Taylor) в 1983 году [1], а затем и другими исследователями [2].

Большинство исследователей данной проблемы концентрировали свое внимание на разработке алгоритмов, которые имели или более эффективную реализацию, или позволяли добавлять и удалять классы в иерархии.

В данной статье рассматривается ситуация когда пользователь приписывается к некоторому классу только на определенный интервал времени. Рассмотрим алгоритм назначения криптографического ключа, зависящего от времени, то есть каждый класс C_i имеет несколько ключей $K_{i,t}$, где $K_{i,t}$ есть ключ класса C_i в момент времени *t*. Обычно общее время жизни системы разделяется на отрезки и отсчет начинается с 0. Допустим, что данные были помещены в класс C_j в момент времени *t* и зашифрованы, используя ключ $K_{j,t}$. Пользователь из класса C_i в промежуток времени от t_1 до t_2 обладает информацией $I(i, t_1, t_2)$, где $1 \le i \le m$, $t_1 \le t_2$. С помощью этой информации он может вычислить ключ $K_{j,t}$ класса C_j для момента времени *t*, но только в том случае, если $C_j \le C_i$ и $t_1 \le t \le t_2$. Таким образом пользователь может расшифровать данные, сохраненные в классе C_j в момент времени *t*.

Для реализации данного подхода можно использовать асимметричный алгоритм шифрования, известный под названием RSA.

Предположим, что максимальное количество периодов времени есть $T + 1$. Для простоты будем предполагать, что T является целым числом, то есть система начинает работать в момент 0 и заканчивает в момент T. Пусть H хэш-функция, которая не является секретной и известна всем пользователям.

1. Инициализация. Выбираем два случайных больших простых в сильном смысле числа p и q. Вычисляем $n = pq$ и $\varphi(n) = (p - 1)(q - 1)$. Затем выбираем случайное целое число a, такое, что $1 < a < n$.

Для каждого класса C_i, $i = \overline{1,m}$ выбираем случайное целое число e_i, $1 < e_i < \varphi(n)$ такое, что $\gcd(e_i, \varphi(n)) = 1$, где gcd – наибольший общий делитель. Затем, используя обобщенный алгоритм Евклида, находим целые числа d_i, $i = \overline{1,m}$, $1 < d_i < \varphi(n)$ такие, что

$e_i d_i \equiv 1 \pmod{\varphi(n)}$.

Выбираем случайные целые числа g_1 и g_2 взаимно простые с $\varphi(n)$ и вычисляем h_1 и h_2 такие, что $g_1 h_1 \equiv g_2 h_2 \equiv 1 \pmod{\varphi(n)}$. Затем вычисляем

$$K_i = a^{\prod_{C_k \le C_i} d_k} \bmod n.$$

2. Назначение ключа. Ключ, назначаемый классу C_i в момент времени *t*, есть

$K_{i,t} = H(K_i^{h_1^t h_2^{T-t}} \bmod n)$.

3. Открытые и секретные параметры. Открытыми параметрами являются $g_1, g_2, e_1, e_2, ..., e_m, n$. Они должны быть доступны всем пользователям системы. Все остальные параметры являются секретными и к ним пользователи не должны иметь доступа.

4. Информация $I(i, t_1, t_2)$. Когда пользователь приписывается к классу C_i и получает свой ключ от этого класса для периода времени от t_1 до t_2, он получает информацию

$I(i, t_1, t_2) = K_i^{h_1^{t_2} h_2^{T-t_1}} \bmod n$.

Так как размер (количество бит) $K_i^{h_1^{t_2} h_2^{T-t_1}} \bmod n$ равен размеру *n*, то размер $I(i, t_1, t_2)$ также равен *n* и не зависит от числа классов в иерархии и периода времени от t_1 до t_2.

5. Вычисление ключей. С помощью $I(i, t_1, t_2)$ и открытых параметров пользователь может вычислить $K_{j,t}$, если $C_j \le C_i$ и $t_1 \le t \le t_2$. Действительно

$$I^{g_1^{t_2-t} g_2^{t-t_1} \prod_{C_k \le C_i, C_k \not\le C_j} e_k} \bmod n =$$

$$= a^{g_1^{t_2-t} h_1^{t_2} g_2^{t-t_1} h_2^{T-t_1} \prod_{C_l \le C_i} d_l \prod_{C_k \le C_i, C_k \not\le C_j} e_k} \bmod n =$$

$$= a^{g_1^{-t} g_2^{t} h_2^{T} \prod_{C_l \le C_j} d_l \prod_{C_k \le C_i, C_k \not\le C_j} e_k d_k} \bmod n =$$

$$= a^{h_1^{t} h_2^{T-t} \prod_{C_l \le C_j} d_l} \bmod n = K_j^{h_1^t h_2^{T-t}} \bmod n.$$

Здесь "!≤" обозначает отрицание "≤".

Рассмотрим перспективные направления использования данного алгоритма.

Первое применение данной схемы связано с защищенной передачей данных множеству пользователей по каналам связи, которым может быть локальная сеть, Internet и др. При такой передаче данные не защищены, т. к. любой подключенный к каналу связи может перехватить их. Для защиты от подслушивания данные могут быть зашифрованы и затем переданы по каналу связи. Таким образом, только получатель, имеющий необходимый ключ сможет их расшифровать и прочитать. Однако в некоторых случаях требуется более гибкая система защиты передаваемых данных. Рассмотрим это на примере системы электронной подписки на рассылку каких-либо данных.

Некоторые компании обеспечивают рассылку данных в электронной форме своим подписчикам (для определенности будем предполагать, что это электронный вариант некоторой газеты). Мы рассмотрим ситуацию, когда некоторые подписчики интересуются только определенной частью газеты и хотят подписаться только на эту часть, чтобы уменьшить плату за подписку. Кроме того подписка осуществляется на определенный период времени, после истечения которого подписчик не должен иметь возможности расшифровать сообщения старым ключом. Предположим, что имеется *m* допустимых частей подписки. Пусть S_i – данные

из *i*-й части и U_i – множество пользователей кто подписался на S_i. Одним из простейших решений данной проблемы является шифрование каждого S_i с помощью ключа K_i и распространение этого ключа среди всех подписчиков из U_i. Если у пользователя из U_i истек срок подписки, то компания изменяет ключ K_i на K_i' и отправляет его оставшимся подписчикам из U_i. Нетрудно заметить, что при таком решении проблемы вскоре у компании возникнут сложности с распространением ключей.

С помощью вышеуказанного алгоритма возможно следующее решение данной проблемы. Пусть подписка на каждый день разделяется на части P_i, $1 \le i \le m$, которые формируют иерархию. Далее, *i*-я часть подписки S_i состоит из данных отнесенных к P_i, а также данных из всех нижестоящих классов P_j, $P_j \le P_i$. Таким образом, пользователь, который подписался на P_i может получать данные из всех классов, стоящих ниже P_i. Система работает следующим образом:

1. Для каждой даты *t* компания шифрует данные в P_i с помощью ключа $K_{i,t}$, $1 \le i \le m$ и затем рассылает их. Заметим, что только данные в P_i (не все данные из S_i) шифруются с помощью $K_{i,t}$.
2. Когда подписчик подписывается на *i*-ю часть, т. е. S_i, на период времени от t_1 до t_2, он получает информацию $I(i, t_1, t_2)$.
3. Когда подписчик получает данные (j, t, D) и $P_j \le P_i$, $t_1 \le t \le t_2$, он использует $I(i, t_1, t_2)$ для получения ключа $K_{j,t}$ и расшифровывает данные *D*.

Такая система электронной подписки иммет ряд преимуществ.

1. Компания использует оптимальный способ передачи данных, т. к. каждая часть P_i передается только один раз, а общий объем передаваемых данных равен размеру подписки.
2. Каждый подписчик хранит у себя только ключевую информацию, размер которой не зависит от числа частей подписки и от периода на который производится подписка.
3. Компания освобождается от необходимости формировать новые ключи и рассылать их подписчикам в случае истечения срока некоторых подписок.
4. Компания может использовать старые номера подписки. Для этого она сохраняет их в зашифрованной форме в базе данных. Когда пользователь хочет получить доступ к S_i от даты t_1 до t_2, он просто получает $I(i, t_1, t_2)$ и работает с базой данных.

Вторым приложением вышеуказанной схемы является система дублирования криптографических ключей. Суть ее заключается в следующем. Предположим, что служащие некоторой компании используют криптографические алгоритмы для шифрования своих файлов в компьютерной системе. С одной стороны это обеспечивает конфиденциальность информации, но с другой имеет ряд особенностей.

1. Служащий может потерять свой ключ. В результате уже никто не сможет расшифровать файлы и важная информация будет потеряна.
2. Возможно, что служащий отсутствует в данный момент, а его файлы срочно требуются. В этом случае вышестоящий начальник может разрешить другому служащему расшифровать файлы.
3. Возможно, что служащий специально зашифровал файлы с целью нанесения вреда в знак протеста против каких либо действий начальства.

Существует несколько способов решения данных проблем, в частности возможен следующий.

1. Все служащие компании упорядочены по классам согласно их должности. Начальник из вышестоящего класса имеет право разрешить любому доступ к данным, хранящимся в низших классах.

2. Когда служащий устраивается в компанию (или подтверждает свою должность, например один раз в год), он получает информацию $I(i, t_1, t_2)$, где C_i – класс, соответствующий его должности, t_1 – время ранее зашифрованных файлов, которые ему разрешается расшифровать, t_2 – следующая дата его подтверждения должности.
3. Любой служащий из класса C_i, шифруя свой файл с помощью ключа K в момент времени t должен присоединить к файлу специальное поле $E(K_{j,t}, K)$, где E – используемый алгоритм шифрования с секретным ключом, например DES.
4. Если вышестоящему начальнику необходимо разрешить другому служащему доступ к файлу (зашифрованному в момент t) служащего из класса C_j, то он использует свою информацию $I(i, t_1, t_2)$ для получения ключа $K_{j,t}$, а затем использует $K_{j,t}$ для расшифрования специального поля $E(K_{j,t}, K)$, получая ключ K от файла. Затем он дает ключ K служащему для расшифрования файла. Таким образом ни один служащий не может хранить зашифрованные данные как единоличный владелец, т. к. вышестоящий начальник может расшифровать их.

Литература

1. Akl, S.G. Cryptographic Solution to a Problem of Access Control in a Hierarchy/ S.G. Akl, P.D. Taylor // ACM Trans. Computer Systems. – 1983. − V. 1, No. 3. − P. 239 − 248.
2. Harn, L. A Cryptographic Key Generation Scheme for Multilevel Data Security/ L. Harn, H.Y. Lin // Computers and Security. – 1990. − V. 9, No. 6. − P. 539 − 546.

APPLICATION OF ASYMMETRIC ENCRYPTION ALGORITHMS FOR CONTROLLING ACCESS TO HIERARCHICALLY-ORGANIZED INFORMATION

Sidorov Dmitry Petrovich

Abstract. In the most of computer systems all stored information is divided into so-called «security classes» based on the importance. These classes form a hierarchy. More than that, it is common when the stored data itself depends on the moment of time it was included in a particular «security class». This article presents a method of controlling user access to informational resources based on asymmetric encryption algorithms.

Keywords. Asymmetric encryption algorithm; encryption; access control; hierarchical data.