3. Берников В.О. Сравнительный анализ криптостойкости симметричных алгоритмов шифрования // Труды БГТУ. Сер. 3, Физико-математические науки и информатика. – Минск: БГТУ, 2020. – № 1 (230). – С. 69–73.

4. Берников В. О. Метод на основе атрибутов текста в многоключевой стеганографической системе // Информационные технологии в образовании, науке и производстве: VII Международная научно-техническая интернет-конференция, 16-17 ноября 2019 г. (в печати).

УДК 004.89

А. В. Олеферович, асп.; В.В. Смелов, доц., канд. техн. наук
(БГТУ, г. Минск)

МЕТОД АВТОРИЗАЦИИ НА ОСНОВЕ РАЗЛОЖЕНИЯ ЧИСЕЛ НА ПРОСТЫЕ СОМНОЖИТЕЛИ

Построение системы информационной безопасности начинается с выявления (идентификации) субъектов и объектов информационной безопасности. В информационных системах под субъектами подразумеваются активные компоненты (пользователи, программные средства), а под объектами – пассивные компоненты системы (аппаратное обеспечение, информационные ресурсы, каналы связи, программные коды) [1].

В общем случае формальное описание системы S авторизации произвольной информационной системы может быть описано в виде тройки:

$$S = < R, U, A >,$$

где $R=\{r_1, r_2, r_3, \ldots, r_N\}$ – конечное множество объектов (ресурсов) информационной системы, доступ к которым регулируется системой авторизации S; $U=\{u_1, u_2, u_3, \ldots, u_L\}$ – конечное множество субъектов информационной системы, доступ которых к объектам R регулируется системой авторизации S; $A \subset U \times R$ – бинарное отношение, определенное на декартовом произведении $U \times R$ и такое, что если $<u,r> \in A$, то субъекту $u \in U$ разрешен доступ (привилегия) к объекту $r \in R$, бинарное отношение A может быть представлено в виде графа $G_A=<U \cup R, A>$, где $U \cup R$ – множество вершин графа; A – множество дуг. По всей видимости, построение бинарного отношения A над множеством $U \times R$ можно свести к построению характеристической функции

$$H(u,r)=\begin{cases} 1, & <u,r> \in A \\ 0, & <u,r> \notin A \end{cases},$$

т. е. субъекту $u \in U$ разрешен доступ к объекту $r \in R$, если $H(u,r)=1$. Тогда бинарное отношение A может быть записано в виде $A = \{<u,r> \in A \mid H(u,r)=1\}$.

Пусть $P=\{p_1,p_2,p_3,\ldots,p_N\}$ – конечное множество простых чисел [2]. Представим множество R объектов авторизации в виде множества пар $R=\{r_i = <\dot{r}_i,\ddot{r}_i>, i=\overline{1,N}\}$, где $\dot{r}_i$ – идентификатор объекта, а $\ddot{r}_i$ – его дескриптор. Значение идентификатора может быть выбрано произвольно при условии его уникальности, а значение дескриптора $\ddot{r}_i = p_i,\ p_i \in P,\ i = \overline{1,N}$ – простое число.

Множество U субъектов информационной системы тоже представим в виде множества пар $U=\{ u_j = <\dot{u}_j,\ddot{u}_j>, j=\overline{1,L}\}$, где $\dot{u}_j$ – идентификатор субъекта, а $\ddot{u}_j$ – дескриптор. Как и прежде, значение идентификатора может быть выбрано произвольно при условии его уникальности, а дескриптор субъекта зададим в виде произведения:

$$\ddot{u}_j = x_1\, x_2 x_3 \ldots x_N,$$

где $x_i = \begin{cases} 1, & <u_j,r_i> \notin A \\ \ddot{r}_i, & <u_j,r_i> \in A \end{cases}$.

Другими словами, $\ddot{u}_j$ является произведением дескрипторов $\ddot{r}_i$, тех объектов системы S, доступ к которым разрешен субъекту u_j.

Характеристическая функция H проверки принадлежности пары $<u_j,r_j>, j=\overline{1,L}$, $i=\overline{1,N}$ множеству A может быть записана в следующем виде:

$$H(u_j,r_i) = \begin{cases} 1, & \ddot{u}_j \bmod \ddot{r}_i = 0 \\ 0, & \ddot{u}_j \bmod \ddot{r}_i \neq 0 \end{cases}$$

В некоторых случаях объекты системы авторизации могут иметь более сложную схему доступа. Например, пусть r_1, r_2, r_3 – три объекта системы авторизации S. При этом в системе должен быть обеспечен независимый авторизированный доступ к каждому из объектов r_1 и r_2, а к объекту r_3 разрешен доступ только тем субъектам, которым разрешен доступ и к r_1 и к r_2. Такая зависимость авторизированного доступа может быть разрешена, если выбрать в качестве дескриптора объекта r_3 приведенное произведение дескрипторов объектов r_1 и r_2: $\ddot{r}_3 = \ddot{r}_1\ddot{r}_2/\gcd(\ddot{r}_1,\ddot{r}_2)$, где $\gcd$ – функция, вычисляющая наибольший общий делитель для двух натуральных чисел. Очевидно, что верно следующее общее утверждение:

$$\forall u_j \in U,\ \{r_{i_1}, r_{i_2}, r_{i_3}=\langle \ddot{r}_{i_3}, \ddot{r}_{i_1}\ddot{r}_{i_2}/\gcd(\ddot{r}_{i_1},\ddot{r}_{i_2})\rangle\} \subset R \mid H(u_j, r_{i_1})=H(u_j, r_{i_2})=1: H(u_j, r_{i_3})=1$$

Другими словами, для рассматриваемого примера, если дескриптор объекта r_3 равен приведенному произведению дескрипторов объектов r_1 и r_2 ($\ddot{r}_3 = \ddot{r}_1\ddot{r}_2/\gcd(\ddot{r}_1,\ddot{r}_2)$), то субъект авторизации, имеющий доступ объектам r_1 и r_2, имеет доступ и к объекту r_3. Далее, объекты, доступ к которым обусловлен доступом к другим объектам, будем называть составными объектами, а объекты, доступ к которым не зависит от доступа к другим объектам, – элементарными объектами авторизации. При этом будем говорить, что составные объекты составлены из других объектов.

В системах авторизации информационных систем используется понятие роли. Роль может быть назначена субъектам авторизации, в результате чего субъекты приобретают (наследуют) привилегии (возможности доступа), которыми обладала эта роль. При этом одному субъекту может быть назначено несколько ролей, а одна роль может быть назначена нескольким субъектам.

Пусть $T=\{t_1,t_2,t_3,\ldots,t_M\}$ – множество ролей, заданных в системе авторизации S. По аналогии с субъектами, роль можно представить как множество пар $T=\{\langle \dot{t}_k,\ddot{t}_k\rangle,\ k=\overline{1, M}\}$, где $\dot{t}_k$ – идентификатор роли, выбранный из условия уникальности, а $\ddot{t}_k$ – дескриптор.

Назначение роли t_k к субъекту u_j будем обозначать с помощью оператора grant и записывать в следующей форме: $t_k \text{grant } u_j$. Формально определим оператор grant следующим образом:

$$\forall\ u_j=\langle \dot{u}_j,\ddot{u}_j\rangle \in U,$$
$$t_k \in T:\ t_k \text{ grant } u_j \Leftrightarrow u_j=\langle \dot{u}_j,\ddot{u}_j\ \ddot{t}_k/\gcd(\ddot{u}_j,\ddot{t}_k)\rangle.$$

Дескриптор роли зададим в виде произведения:

$$\ddot{t}_k = x_1\ x_2 x_3 \ldots x_N,$$

где $x_i = \begin{cases} 1, & \forall u_j \in U \mid t_k \text{ grant } u_j : \langle u_j, r_i\rangle \notin A \\ \ddot{r}_i, & \forall u_j \in U \mid t_k \text{ grant } u_j : \langle u_j, r_i\rangle \in A \end{cases}$.

Другими словами, $\ddot{t}_k$ является произведением дескрипторов $\ddot{r}_i$, тех объектов системы S, доступ к которым может обеспечиваться назначением роли t_k любому субъекту.

Основным достоинством предложенной системы авторизации является простота ее реализации, так как в основе лежат простейшие арифметические операции. Применение предложенного метода авто-

ризации для такого рода программных приложений, по мнению авторов, является целесообразным: программная реализация является простой, проверка правомочности доступа не требует больших вычислительных ресурсов, предлагаемая процедура авторизации просто встраивается в существующие сетевые протоколы (например, протокол RFC 7519 для создания токенов доступа, основанных на формате JSON [3]).

ЛИТЕРАТУРА

1. Хоффман Л. Современные методы защиты информации. М.: Сов.Радио,1980.

2. Айерлэнд К., Роузен М. Классическое введение в современную теорию чисел . – М.: Мир, 1987.

3. Предлагаемый стандарт RFC 7519 [Электронный ресурс] URL: https://tools.ietf.org/html/rfc7519 дата доступа:10.10.2020.

УДК 004.051

Долговечный В.Н. маг.; П. П. Урбанович, проф., д-р техн. наук
(БГТУ, г. Минск)

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ СБОРЩИКА МУСОРА И JIT КОМПИЛЯТОРА НА ПЛАТФОРМАХ .NET 5.0, .NET FRAMEWORK 4.8 И .NET CORE 3.1

10 ноября 2020 г. вышел релиз фреймворка .NET 5, который является развитием .NET Core в долгой эволюции фреймворка .NET. .NET 5.0 - это первый релиз на пути к унификации платформы .NET, который позволяет более плавно мигрировать с .NET Framework и использовать преимущества и функции .NET Core. .NET 5 – это объединение его двух предшественников .NET framework 4.8 и .NET Core 3.1.

В данном исследование рассмотрим, улучшилась ли производительность сборщика мусора (garbage collector GC) и компилятора Just-In-Time (JIT) на реальных примерах.

Сборщик мусора .NET управляет выделением и освобождением памяти для приложения. При каждом создании объекта среда CLR выделяет память для объекта из управляемой кучи. Пока в управляемой куче есть доступное адресное пространство, среда выполнения продолжает выделять пространство для новых объектов. Механизм оптимизации сборщика мусора определяет наилучшее время для выполнения сбора, основываясь на выполненных операциях выделения памяти. Когда сборщик мусора выполняет сборку, он проверяет нали-