УДК 004.056 : 004.424.47

В. А. Лужецький, д-р техн. наук, проф.;
Ю. В. Баришев, асп.

СИНТЕЗ ФУНКЦІЙ УЩІЛЬНЕННЯ ДЛЯ КЕРОВАНОГО ХЕШУВАННЯ В КОМП'ЮТЕРНІЙ КРИПТОГРАФІЇ

Проаналізовано підходи до побудови функцій ущільнення для реалізації керованого хешування. Сформульовано вимоги до методів генерування векторів керування та криптографічних примітивів для побудови керованого хешування та визначено підходи до генерування векторів керування та операції, що задовольняють відповідні вимоги. Запропоновано підходи до реалізації функцій ущільнення та наведено приклад методу, що їх реалізує.

Вступ

За останні роки суттєво збільшилась кількість атак на хеш-функції [1], зокрема суттєво збільшилась кількість загальних атак та розвинулися методи криптоаналізу. Якщо криптоаналіз базується на властивостях певних перетворень, що є складовими функції ущільнення, то загальні атаки використовують властивості конструкцій хешування, і в першу чергу — властивість ітеративності процесу хешування. Автори бачать вихід із цієї проблемної ситуації у переосмисленні принципів хешування. Однією з гілок у цьому напрямку є концепція керованого хешування — хешування, яке використовує набір параметрів перетворень на кожній ітерації, що залежить від значення вектора керування. Очікуваними перевагами хешування, що базується на цій концепції, порівняно з ітеративним хешуванням, є збільшення нелінійності перетворень і можливість розпаралелення обчислень, а також підвищення стійкості до загальних атак [2, 3].

На шляху до реалізації концепції керованого хешування необхідно розв'язати низку задач, зокрема визначити множину функцій ущільнення **F**. *Метою цього дослідження* є підвищення швидкості та стійкості хешування за рахунок реалізації концепції керованого хешування шляхом розробки методів генерування множин функцій ущільнення.

Для досягнення мети необхідно розв'язати такі задачі:

— проаналізувати відомі пропозиції щодо розробки функцій ущільнення для керованого хешування;

— визначити методи генерування векторів керування;

— визначити вимоги до криптографічних примітивів, що входитимуть до складу функцій ущільнення;

— визначити перетворення, що можуть використовуватись як криптографічні примітиви керованого хешування та параметри цих перетворень, що можуть бути використані як об'єкт керування;

— розробити підходи до формування наборів функцій ущільнення з визначених криптографічних примітивів.

Аналіз методів розробки функцій ущільнення

Більшість відомих підходів до проектування функцій ущільнення не можуть бути використані для керованого хешування, оскільки вони були розроблені для ітеративного хешування, а тому не передбачають наявності параметрів, якими можна керувати.

Одна з пропозицій щодо розробки керованих функцій ущільнення була реалізована у хеш-функціях Dynamic SHA та Dynamic SHA-2, які запропоновані для участі у конкурсі на новий стандарт хешування SHA-3 [4, 5]. Функції ущільнення, що використовуються у них, базуються на логічних функціях таких самих, як у функціях ущільнення стандарту хешування SHA-2. Як параметр перетворення, що керується, автором статті [4] запропоновано операцію простого зсуву. Керування відбувалося залежно від дещо модифікованих вхідних блоків даних. У роботі [5] була запропонована атака, яка використовувала властивість простого зсуву втрачати дані та здатність криптоаналітика впливати на керування, нав'язуючи певні блоки даних.

Значний розвиток у напрямку реалізації керованих криптографічних алгоритмів внесла робота [2], де наводиться аналіз класів керованих криптографічних примітивів. Не зважаючи на те, що такі примітиви першочергово розроблялися для шифрування, а тому вектор керування передбача-

лося використовувати у більшій мірі як ключ для перемикання між режимами шифрування/розшифрування, ніж перемикач між різними наборами перетворень, у роботі [2] наводиться приклад того, як ці криптографічні примітиви можуть використовуватися для розробки хеш-функцій. Основою для розробки криптографічних примітивів у [2] стали ППМ (підстановочно-перестановочні мережі), а параметрами перетворення обрано тип підстановки та перестановки. Створюючи пари з цих перестановок, автори роботи [2] отримують керовані хеш-функції, причому вектором керування визначається, яка з перестановок буде використовуватись. Наприклад, для двох вхідних бітів x_1 та x_2 і вектора керування довжиною в один біт v криптографічні примітиви, записані у алгебраїчній нормальній формі, мають такий вигляд: $x_2 v \oplus x_1 \oplus 1$. Криптографічні примітиви, що мають два біти вектора керування, будуються шляхом об'єднання пари примітивів, запропонованих для вектора керування довжиною один біт [2]. Крім цих перестановок в роботі [2] розглядаються криптографічні примітиви на основі перестановок з трьох бітів. Хешування відбувається шляхом послідовного застосування до блока даних каскадів керованих перестановок двох та трьох сусідніх бітів, чергуючи їх з перестановками бітів у всьому блоці даних. За допомогою перестановок у всьому вхідному блоці досягається високий рівень лавинного ефекту та можливість впливу кожного вхідного біта на всі вихідні у разі достатньої кількості каскадів.

Недоліками підходу керованих ППМ є висока складність програмної реалізації та її низька швидкість хешування. Це пов'язано з тим, що ППМ першочергово були орієнтовані на апаратну реалізацію, а за програмної реалізації перестановки двох або трьох сусідніх бітів викликають труднощі, пов'язані з резервним копіюванням частини блока даних, накладанням маски та з'єднуванням результатів в один операнд. Аналогічні труднощі породжуються при спроби перемішування біт у всьому блоці даних, хоча для апаратної реалізації це виконується за допомогою простого розгалуження провідників. Крім того хеш-функції, запропоновані у [2] належать до класу хеш-функцій, що базуються на шифрах, а для найповнішої реалізації концепції керованого хешування доцільніше їх розробляти «з нуля». Так криптографічні примітиви, що входять до шифрів, повинні бути інволюціями або мати обернене перетворення, що у разі хешування несуттєво. Відповідно класи хешування, запропоновані у [2], можуть бути розширені для їх застосування для побудови хеш-функцій.

Підходи до розробки криптографічних примітивів у роботі [6] близькі до підходів, використаних у [2] — вони розроблялися, в першу чергу, для шифрування та орієнтовані на апаратну реалізацію, а також засновані на керованих перестановках. Однак крім перестановок керування пропонується виконувати за допомогою операції інвертування (виконання/невиконання), а також виконанням прямого чи оберненого перетворень, використанням складних чи простих функцій. Саме тому особливістю роботи [6] є те, що вектор керування може приймати більше значень, ніж кількість комбінацій вихідних бітів. Зокрема, у роботі [6] наводиться приклад, в якому для 2 бітів вхідних даних використовується 5 бітів керування. Останнє є причиною того, що атаки грубої сили можуть виявитись ефективнішими, ніж атаки, що використовують специфічні властивості операцій. Проте це твердження виконується для шифрування та задач ключового хешування. Для задач безключового хешування така «надмірність» вектора керування може стати причиною колізій, оскільки декілька різних наборів векторів керування призводять до однакових вихідних значень за однакових вхідних даних. Так, для прикладу з [6] — за допомогою 5 бітів векторів керування можна побудувати 32 різних набори перетворень, водночас існує 16 наборів різних відображень 2-х вхідних бітів у 2-х вихідних.

У зв'язку з тим, що відсутні методи керованого хешування, розроблені «з нуля» та першочергово орієнтовані на реалізацію керованості, виконаємо побудову функцій ущільнення для керованого хешування, не базуючись на відомих розв'язках цієї задачі. Для цього попередньо визначимо метод генерування вектора керування та криптографічні примітиви, з яких складатимуться функції ущільнення.

Аналіз методів генерування векторів керування та визначення криптографічних примітивів

Визначення джерела для формування вектора керування пропонується виконувати відповідно до таких критеріїв:

1. Значення векторів керування обираються з однаковою ймовірністю.
2. Вектори керування мають бути різними для різних каналів.

3. Вектори керування мають бути різними для різних повідомлень.

4. Бути невідомим зловмиснику (для ключового варіанта хешування).

Вектори керування можуть формуватися на основі: — псевдовипадкових чисел; — блоків даних; — проміжних хеш-значень.

Методи генерування на основі псевдовипадкових чисел не відповідають вимозі 3, а на основі блоків даних — вимозі 4. Хоча блоки даних як джерела вектора керування недосконалі, вони викликають цікавість з точки зору їх застосованості для безключового хешування та з метою збільшення нелінійності перетворень. Методи генерування вектора керування на основі проміжних хеш-значень відповідають визначеним вище вимогам щодо методів генерування, тому їм необхідно надати перевагу в процесі реалізації концепції керованого хешування.

Криптографічні примітиви, що будуть використовуватись для побудови функцій ущільнення, повинні відповідати таким вимогам:

— бути природними для універсальних мікропроцесорів, а тому швидко виконуватися на них; — забезпечувати рівномірність та збалансованість результату перетворення; — забезпечувати вплив усіх вхідних біт на вихідне значення.

Виходячи з вимоги до хешування, яка обумовлена його прикладним застосуванням, як криптографічні примітиви бажано брати операції, які є природними для універсальних процесорів, тобто виконуються за один такт. До таких операцій належать: додавання за модулем 2^n (+); виключне або $(\oplus)$; інвертування (~); логічне множення $(\wedge)$; логічне додавання $(\vee)$; циклічний зсув на u біт праворуч (>>> u); простий зсув на u біт праворуч/ліворуч (>> u/ << u). З цих криптографічних примітивів для синтезу функцій ущільнення можуть бути використані +, $\oplus$, ~, $\wedge$, $\vee$ та >>> u, оскільки >> u не забезпечує впливу всіх вхідних біт на вихідне значення. Природними параметрами перетворень, якими зручно керувати, є параметр u для операції циклічного зсуву. Крім того, керування може здійснюватись над виконанням та невиконанням операції інвертування.

Розробка методів генерування функцій ущільнення

Функції ущільнення повинні мати різні параметри перетворень та обиратися під час кожної ітерації залежно від значення вектора керування на цій ітерації v_i. Операція $\oplus$ дозволяє забезпечити рівномірний вплив кожного біта на вихідне хеш-значення, відповідно доцільно її використати для домішування результату обробки блоків даних до хеш-значення, отриманого на попередній ітерації:

$$h_i = h_{i-1} \oplus m_i, \tag{1}$$

де h_i — проміжне хеш-значення, отримане після i-ї ітерації хешування; m_i — i-й блок даних, що входить до складу повідомлення $\mathbf{M} = m_1 \| m_2 \| \ldots \| m_l$.

Оскільки сучасне хешування повинно бути адаптованим для багатоядерних апаратних платформ [7], тобто бути багатоканальним, методи підвищення стійкості такого хешування до мультиколізій варто використати ще на етапі розробки функцій ущільнення. Використаємо метод підвищення стійкості хешування без зв'язку між каналами, запропонований у статті [7], оскільки він не потребує введення додаткових операндів та дозволяє досягти більшої швидкості хешування. Для зв'язку блоків даних використаємо нелінійну операцію

$$h_i = h_{i-1} \oplus \left(m_i \wedge m_{i-1-r_i}\right), \tag{2}$$

де r_i — ціле число з проміжку $[0;\ l-1)$, яке визначається за формулою

$$r_i = rand\left(m_i\right), \tag{3}$$

де $rand(\cdot)$ — деяка функція, значення якої мають рівномірний закон розподілу.

Наявність у перетворенні (2) вибору одного блока залежно від значення іншого дозволить досягти лавинного ефекту без раундових повторень, оскільки в загальному випадку біти у блоках даних розподіляються рівномірно. Оскільки операція $\wedge$ сприяє збільшенню кількості нулів, тому доцільно використати у функції ущільнення і симетричну до неї $\vee$. У зв'язку з тим, що збільшення кількості пов'язаних один з одним блоків даних сприятиме збільшенню його стійкості, пропонується виконувати пов'язування за допомогою операції $\vee$ іншого блока даних з цим

$$h_i = h_{i-1} \oplus \left(m_i \wedge m_{i-1-r_{2i}}\right) \oplus \left(m_i \vee m_{i-1-r_{2i+1i}}\right). \quad (4)$$

Для генерування псевдовипадкових чисел r_{2i} та r_{2i+1} пропонується використовувати один з двох методів. Перший — за допомогою використання двох функцій, аналогічних функції (3). Цей метод доцільно використовувати, коли не існує значних обмежень на обчислювальні ресурси, що використовуються під час хешування. Для задач, де ці обмеження суттєві, пропонується використовувати інший метод — генерування r_{2i+1} залежно від іншого блока даних:

$$\begin{cases} r_{2i} = rand\left(m_i\right); \\ r_{2i+1} = rand\left(m_{i+1}\right). \end{cases} \quad (5)$$

Використання другого методу має особливість: пара блоків m_i та m_{i+1} завжди пов'язані з парою $m_{i-1-r_{2i+1}}$ та $m_{i-r_{2i+1}}$. Завдяки цьому збільшується вплив позицій блоків, оскільки враховується не лише їх позиція у повідомленні, а також сусідство блоків. Для реалізації керованості хешування пропонується використовувати операції циклічного зсуву

$$h_i = h_{i-1} \oplus \left(m_i >>> u_{(i)(1)} \wedge m_{i-1-r_{2i}} >>> u_{(i)(2)}\right) \oplus \left(m_i >>> u_{(i)(3)} \vee m_{i-1-r_{2i+1}} >>> u_{(i)(4)}\right), \quad (6)$$

де $u_{(1)(i)}$, $u_{(2)(i)}$, $u_{(3)(i)}$, $u_{(4)(i)}$ — частини вектора керування v_i.

Використовуючи підходи, описані вище, можна побудувати й інші функції ущільнення, наприклад, додавши до формули (6) операції інвертування, які будуть виконуватись чи не виконуватись залежно від значення вектора керування. Крім того замість операції $\oplus$ використовувати +, а також їх комбінації.

У низці задач швидкість хешування важливіша, ніж його стійкість. Тому для прискорення пропонується змінювати в таких випадках формулу (6) таким чином:

$$h_i = h_{i-1} \oplus \left(m_i >>> u_{(i)(1)} \wedge m_{i-1-r_{2i}} >>> u_{(i)(2)}\right) \oplus \left(m_{i+1} >>> u_{(i)(3)} \vee m_{i-r_{2i+1}} >>> u_{(i)(4)}\right). \quad (7)$$

У разі хешування за формулою (7) пропонується використовувати подвійний приріст лічильника блоків даних $\left(i = i + 2\right)$, що рівнозначно використанню блоків даних подвійної довжини з їх розбиттям на дві частини під час обробки. Для прикладу розглянемо багатоканальне кероване хешування, що реалізується на 32-розрядній платформі. Для того, щоб за допомогою хешування отримати 256-розрядне вихідне хеш-значення, необхідно використати 8 каналів. Нехай для формування вектора керування використовується проміжне хеш-значення, отримане на попередній ітерації у $\left(j-1\right)$-му каналі:

$$v_i^{(j)} = g\left(h_{i-1}^{(j-1)}\right), \quad (8)$$

де $g(\cdot)$ — деяка функція, що використовується для формування вектора керування з проміжного хеш-значення.

Для того, щоб за різних значень вектора керування результат перетворення істотно змінювався, пропонується задати такі обмеження на параметри зсуву: $u_{(1)(i)}$, $u_{(4)(i)}$ — парні числа, $u_{(2)(i)}$, $u_{(3)(i)}$ — непарні числа. Так, для хешування достатньо вектора керування довжиною 16 бітів, який можна отримати з 32-бітового проміжного хеш-значення за допомогою порозрядного додавання старших 16 бітів до молодших 16. Подамо вектор керування як конкатенацію чотирьох 4-бітових підблоків $v_i^{(j)} = v_{(i)(1)}^{(j)} \parallel v_{(i)(2)}^{(j)} \parallel v_{(i)(3)}^{(j)} \parallel v_{(i)(4)}^{(j)}$. Параметри хешування, що керуються, на кожній ітерації для j-го каналу визначатимуться так:

$$\begin{cases} u_{(i)(1)}^{(j)} = 2v_{(i)(1)}^{(j)}; \\ u_{(i)(2)}^{(j)} = 2v_{(i)(2)}^{(j)} + 1; \\ u_{(i)(3)}^{(j)} = 2v_{(i)(3)}^{(j)} + 1; \\ u_{(i)(4)}^{(j)} = 2v_{(i)(4).}^{(j)} \end{cases} \quad (9)$$

Ітерація хешування у j-му каналі матиме вигляд формули (6).

Висновок

З аналізу відомих підходів до розробки функцій ущільнення, що використовуються для вектора керування, випливає, що у відомій літературі відсутні підходи до проектування керованих функцій ущільнення «з нуля».

У зв'язку з цим запропоновано розпочати побудову хеш-функцій, не базуючись на відомих підходах. Першим кроком на цьому шляху стало визначення методу генерування векторів керування. Аналіз можливих джерел для генерування вектора керування, а саме блока даних, псевдовипадкових послідовностей чисел, проміжних хеш-значень, продемонстрував необхідність надання переваги останньому.

Згідно зі сформульованими вимогами до криптографічних примітивів для побудови функцій ущільнення обрано операції $+$, $\oplus$, $\sim$, $\wedge$, $\vee$ та $>>> u$.

Наведено підходи до розробки функцій ущільнення, які дозволяють досягти покращення лавинного ефекту за рахунок обробки декількох блоків даних під час однієї ітерації. Причому, номер одного блока пропонується визначати залежно від значення іншого блока. Як параметри керування пропонується використовувати кількість бітів, на яку виконується циклічний зсув, а також виконання чи не виконання операції інвертування. Також запропоновано підходи до підвищення швидкості хешування шляхом розпаралелення обчислень, яке є стійким до атак.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. V. Klima. Generic collision attacks on narrow-pipe hash functions faster than birthday paradox, applicable to MDx, SHA-1, SHA-2 and SHA-3 narrow-pipe candidates [Електронний ресурс] / Vlastimil Klima, Danilo Gligoroski. — 2010. — 4 с. — Режим доступу : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.104.5370&rep=rep1&type=pdf.

2. Молдовян Н. А. Криптография: от примитовов к синтезу алгоритмов / Н. А. Молдовян, А. А. Молдовян, М. А. Еремеев. — СПб. : БХВ-Петербург, 2004. — 448 с.

3. Баришев Ю. В. Підхід до хешування, що стійке до аналізу зловмисника / Ю. В. Баришев // Системи обробки інформації. — № 3. — 2010. — С. 99—100.

4. Zijie Xu Dynamic SHA [Електронний ресурс] / Zijie Xu // Cryptology ePrint Archive. — 2007. — 34 с. — Режим доступу : http://eprint.iacr.org/2007/476.pdf.

5. Cryptanalysis of Dynamic SHA(2) [Електронний ресурс] / [J-P. Aumasson, O. Dunkelman, S. Indesteege and B. Preneel] // COSIC publications, 2009. — 18 с. — Режим доступу : https://www.cosic.esat.kuleuven.be/publications/article-1277.pdf.

6. Рудницький В. М. Модель уніфікованого пристрою криптографічного перетворення інформації / В. М. Рудницький, В. Г. Бабенко // Системи обробки інформації. — № 3. — 2009. — С. 91—95.

7. Лужецький В. А. Конструкції хешування стійкі до мультиколізій [Електронний ресурс] / В. А. Лужецький, Ю. В. Баришев // Наукові праці Вісник Вінницького політехнічного інституту. — № 1. — 2010. — 8 с. — Режим доступу : http://www.nbuv.gov.ua/e-journals/vntu/2010_1/2010-1.files/uk/10valsam_ua.pdf.

Рекомендована кафедрою захисту інформації

Стаття надійшла до редакції 11.02.11
Рекомендована до друку 1.03.11

Лужецький Володимир Андрійович — завідувач кафедри, ***Баришев Юрій Володимирович*** — аспірант.

Кафедра захисту інформації, Вінницький національний технічний університет, Вінниця