контурів регулювання потужності та довжини хвилі випромінювання а також інваріантності їх від параметрів навколишнього середовища. Для виконання цієї задачі потрібно: дослідити процес формування когерентного електромагнітного випромінювання, як об'єкту керування,особливу увагу при цьому слід приділити дослідженню динамічних властивостей; призвести ідентифікацію з метою ідентифікації основних каналів перетворення; побудувати імітаційну модель процесу, перевірити її на адекватність та провести синтез вдосконаленої системи керування побудованої на принципах автономності, інваріантності та каскадності.

Список використаних джерел

[1] Звелто О. З. 43 Принципы лазеров / Пер. под науч. ред. Т. А. Шмаонова. 4-е изд. – СПб.: Издательство «Лань», 2008. – 720 с: ил.
[2] Митрофанов А. С. Основные принципы работы лазеров. Учебное пособие по курсу "Лазерная физика, техника и технология". – СПб: СПбГИТМО(ТУ), 1999. – 74 с.
[3] Коэф Й., Фишер М., Легге М., Сейферт Й., Вернер Р. Лазеры с распределенными брэгговскими решетками на квантовых ямах, точках и с квантовыми каскадами.
[4] Жмудь В. А. Системы автоматического управления. Прецизионное управление лазерным излучением, 2018.
[5] Дураев В.П. Источники оптического излучения. – В кн.: Волоконно-оптическая техника: История, достижения, перспективы: Сб. статей под ред. Дмитриева С.А., Слепова Н.Н. – М.: Изд. Connect, 2000.
[6] Дураев В.П. Полупроводниковые лазеры с волоконой брэгговской решеткой и узким спектром генерации на длинах волн 1530-1560 нм. – Квантовая электроника, 2001.
[7] Ветров А. А., Данилов Д. А., Есипов С. С., Комиссаров С. С., Сергушичев А. Н. Сравнение температурных и электрических методов управления длиной волны излучения полупроводниковых лазеров. – 2009.

References

[1] O. Zvelto, *43 Principles of lasers*. SPb, Izdatelstvo «Lan», 2008. 720 p.
[2] A. S. Mitrofanov, *Basic principles of laser operation*. Textbook for the course "Laser Physics, Engineering and Technology". SPb, 1999, 74 p.
[3] J. Coeff, M. Fischer, M. Legge, J. Seifert, R. Werner, *Lasers with distributed Bragg gratings on quantum wells, points and with quantum cascades*.
[4] B. A. Zmud, "Automatic control systems. Precision control of laser radiation," 2018.
[5] V. P. Duraev, "Sources of optical radiation," in *Fiber-optic technology: History, achievements, prospects*, Moscow, Connect, 2000.
[6] V. P. Duraev, "*Semiconductor lasers with a Bragg lattice fiber and a narrow lasing spectrum at wavelengths of 1530–1560 nm*," Quantum Electronics, 2001.
[7] A. A. Vetrov *et al*. "Comparison of temperature and electrical methods for controlling the wavelength of radiation of semiconductor lasers," 2009.

УДК 004.91:004.056.55.347.135.224

АНАЛІЗ КОМПЛЕКСНОЇ МОДЕЛІ КРИПТОСИСТЕМИ ДЛЯ ЗАХИСТУ КОНФІДЕНЦІЙНИХ ДАНИХ В КОМП'ЮТЕРІ

Борцова Ю. В.[1], Плотніков В. М.[2]

[1,2] Одеська національна академія харчових технологій, Одеса, Україна
ORCID: [1] http://orcid.org/0000-0001-6712-8357, [2]http://orcid.org/0000-0001-9000-2568
E-mail: [1] bortsova.07@gmail.com

DOI: https://doi.org/10.15673/atbp.v12i2.1808

Анотація. *Для захисту конфіденційних даних від комп'ютерних злочинів користувач має подбати про безпеку своєї інформації власноруч, використовуючи існуючі сучасні програмні засоби. Одним з таких засобів є реалізація шифрування повідомлень за допомогою прикріплення цифрового підпису до даних. Для роботи криптосистем шифрування з відкритим ключем необхідно три алгоритми: алгоритм шифрування, алгоритм розшифрування та алгоритм генерації ключів. Одним з перспективних шляхів розвитку шифрування з відкритими ключами є використання моделі піднесення до великої степені дискретних логарифмів для генерування ключів, так званий алгоритм Діффі – Хеллмана. Аналізується спектр можливих кібератак, специфіку їх реалізації, та напрямки дії, надаються теоретичні аспекти протоколу обміну ключами Діффі-Хеллмана, проводиться їх реалізація, тестування протоколу і порівняльний аналіз реалізацій програмного продукту.*

Abstract. *For the hijacking of confidential data from computer users who want to give me some information about the safety of their own information, the vicious hand, and the vicious news program. One of these concerns є realizing encryption is more than necessary for the additional protection of digital signatures before a*

tribute. For cryptosystem robots, encryption with a secure key requires three algorithms: the encryption algorithm, the encryption algorithm, and the key generation algorithm. One of the promising ways to develop encryption with advanced keys is to choose a model for a high degree of discrete logarithms for generating keys, so the Diffie-Hellman algorithm ranks. The spectrum of possible cyber attacks is analyzed, the specifics of their implementation, those are straightforward, the theoretical aspects of the protocol are exchanged with the exchange of Diffie-Hellman keys, the implementation of the protocol is tested, and the protocol is analyzed.

Ключові слова: криптографія, стратегія безпеки, шифрування з відкритими ключами, алгоритм Діффі-Хеллманаб кібератака

Keywords: cryptography, security strategy, public key encryption, Diffie-Hellman algorithm, cyberattack

Вступ

Криптографія є кращою стратегією безпеки, яка розбудовувалася протягом десятиліть, особливо після впровадження й поширення комп'ютерів. Криптографія широко відома як наука про кодування даних для запобігання несанкціонованого доступу через незахищені канали зв'язку. Шифрування підрозділяється на три категорії: симетричне шифрування, асиметричне шифрування й геш-функції. Секретний ключ і симетричне шифрування розбудовувалися першими. Симетричне шифрування використовує один секретний або особистий ключ для шифрування й дешифрування даних. Процес розшифрування прямо протилежний процесу шифрування. Проте, однієї з основних проблем є передача ключа для розшифрування через інтернет. Це основна перешкода, тому що якщо ключ буде перехоплений зловмисниками, те зашифровані дані можуть бути легко розшифровані.

Ця проблема передачі ключа для симетричного шифрування була порушена в науково-дослідному проекті, проведеному Уітфілдом Діффі й Мартіном Хеллманом з Масачусетського технологічного інституту в 1976 році. Алгоритм, розроблений ними, широко відомий як алгоритм Діффі - Хеллмана, який використовується для безпечного обміну секретним ключем між двома сторонами в режимі реального часу в незахищеній мережі. Загальний секретний ключ дуже важливий для двох сторін, які, можливо, не спілкувалися раніше, тому вони можуть зашифрувати свої повідомлення. По суті це протокол узгодження ключа, який підтримує таємність між двома сторонами для обміну ключами. Узгодження ключів являє собою метод, у якому пристрою передачі даних у мережі встановлюють загальний секретний ключ між ними, без обміну якими-небудь секретними даними. У цьому методі пристроям необхідно обмінятися своїми відкритими ключами. Обоє пристрою після приймання відкритих ключів виконують операцію генерації ключа з використанням свого секретного ключа, щоб одержати загальний секретний ключ. Завдяки своїм чудовим атрибутам безпеки, у наш час алгоритм широко використовується й має безліч модифікацій. Деякі дослідники змінили алгоритм Діффі - Хеллмана й використовували його в таких протоколах безпеки, як «рівень захищених сокетов» (SSL), «протокол безпеки міжмережевої взаємодії» (IPsec) і т.д.

Проте, деякі недоліки алгоритму Діффі - Хеллмана усе ще існують. Одним з недоліків алгоритму Діффі - Хеллмана є обсяг обчислень, у зв'язку із цим збільшується тимчасова складність при генерації відкритих ключів.

1. Завдання аутентифікації даних

Внаслідок того, що інформація має нематеріальний характер, масиви даних не несуть на собі ніяких відбитків, по яких можна було б судити про їхнє минуле – про те, хто є автором, про час створення, про факти, час і авторах внесених змін. Модифікація інформаційного масиву не залишає відчутних слідів на ньому й не може бути виявлена звичайними методами. «Сліди модифікації» у тієї або іншій формі можуть бути присутнім тільки на матеріальних носіях інформації – так, спеціальна експертиза цілком здатна встановити, що сектор X на якійсь дискеті був записаний пізніше всіх інших секторів з даними на цій же доріжці дискети, і цей запис проводився на іншому дисководі. Зазначений факт, будучи встановленим, може, наприклад, означати, що в дані, збережені на дискеті, були внесені зміни. Але після того, як ці дані будуть переписані на інший носій, їх копії вже не будуть містити ніяких слідів модифікації. Реальні комп'ютерні дані за час свого життя багаторазово міняють фізичну основу вистави й постійно кочують із носія на носій, у силу чого їх не виявлене викривлення не представляє серйозних проблем.

Оскільки створення й використання інформаційних масивів практично завжди розділені в часі й/або в просторі, у споживача завжди можуть виникнути обґрунтовані сумніви в тому, що отриманий їм масив даних створений потрібним джерелом. Таким чином, у системах обробки інформації крім забезпечення її таємності важливо гарантувати наступні властивості для кожного масиву даних:

– *дійсність* – він прийшов до споживача саме таким, яким був створений джерелом і не перетерпів на своєму життєвому шляху несанкціонованих змін;

– *авторство* – він був створений саме тем джерелом, яким припускає споживач.

Забезпечення системою обробки цих двох якостей масивів інформації й становить завдання їх аутентифікації, а відповідна здатність системи забезпечити надійну аутентифікацію даних називається її автентичністю.

На перший погляд може здатися, що дане завдання вирішується простим шифруванням. Дійсно, якщо масив даних зашифрований з використанням стійкого шифру, такого, наприклад, як ДСТУ 3396.0-96 [6], то для нього практично завжди буде справедливо наступне:

– у нього важко внести зміни осмисленим образом, оскільки зі ступенем імовірності, що незначно відрізняється від одиниці, факти модифікації зашифрованих масивів даних стають очевидними після них розшифрування – ця очевидність виражається в тому, що такі дані перестають бути коректними для їхнього інтерпретатора: замість тексту українською мовою з'являється нісенітниця, архіватори повідомляють, що цілісність архіву порушена і т.д.;

– що тільки володіють секретним ключем шифрування користувачі системи можуть виготовити зашифроване повідомлення, таким чином, якщо до одержувача приходить повідомлення, зашифроване на його секретному ключі, він може бути впевненим у його авторстві, тому що крім нього самого тільки законний відправник міг виготовити це повідомлення.

Проте, використання шифрування в системах обробки даних саме по собі незручно забезпечити їхньої автентичності по наступних причинах:

1) Зміни, внесені в зашифровані дані, стають очевидними після розшифрування тільки у випадку великої надмірності вихідних даних. Ця надмірність має місце, наприклад, якщо масив інформації є текстом на якій–небудь людській мові. Однак у загальному випадку ця вимога може не виконуватися – якщо випадкова модифікація даних не робить їхнім неприпустимим для інтерпретації зі скільки–небудь значною часткою ймовірності, то шифрування масиву не забезпечує його дійсності. Говорячи мовою криптології, автентичність і таємність суть різні властивості криптосистем. Або, більш просто: властивості систем обробки інформації забезпечувати таємність і дійсність оброблюваних даних у загальному випадку можуть не збігатися.

2) Факт успішного (у змісті попереднього пункту) розшифрування зашифрованих на секретному ключі даних може підтвердити їхнє авторство тільки в очах самого одержувача. Третя сторона не зможе зробити на підставі цього однозначного виводу про авторство масиву інформації, тому що його автором може бути кожний із власників секретного ключа, а їх як мінімум два – відправник і одержувач. Тому в цьому випадку спори про авторство повідомлення не можуть бути дозволені незалежним арбітражем. Це важливо для тих систем, де між учасниками немає взаємної довіри, що досить характерно для банківських систем, пов'язаних з керуванням значними цінностями.

Таким чином, існування проблеми підтвердження справжності й авторства масивів даних, окремої від завдання забезпечення їх таємності, не викликає сумніву. У наступних розділах справжньої роботи викладаються підходи до її розв'язку, що базуються на використанні класичних блокових шифрів. Для розв'язку зазначених завдань може бути використаний будь-який традиційний блоковий криптографічний алгоритм. В комп'ютерних кодах, прикладених до справжньої роботи, використовується найбільш знайомий шифр – криптоалгоритм ДСТУ 3396.0–96 [6].

2. Завдання імітозахисту даних

Під імітозахистом даних у системах їх обробки розуміють захист від нав'язування неправильних даних. Як ми вже з'ясували, практично завжди на деяких етапах свого життєвого циклу інформація виявляється поза зоною безпосереднього контролю над нею. Це трапляється, наприклад, при передачі даних по каналах зв'язку або при їхнім зберіганні на магнітних носіях ЕОМ, фізичний доступ до яких сторонніх осіб виключити майже ніколи не представляється можливим. Фізично запобігти внесенню несанкціонованих змін у дані в переважній більшості реальних систем їх обробки, передачі й зберігання не представляється можливим. Тому надто важливо вчасно виявити сам факт таких змін – якщо подібні випадкові або навмисні викривлення будуть вчасно виявлені, втрати користувачів системи будуть мінімальні й обмежаться лише вартістю «порожньої» передачі або зберігання неправильних даних, що, звичайно, у всіх реальних ситуаціях незмірно менше можливого збитку від їхнього використання. Метою зловмисника, що нав'язує системі неправильну інформацію, є видача її за справжню, а це можливо тільки в тому випадку, якщо сам факт такого нав'язування не буде вчасно виявлений, тому проста фіксація цього факту зводить нанівець усі зусилля зловмисника. Підіб'ємо підсумок – під захистом даних від несанкціонованих змін у криптографії розуміють не виключення самої можливості таких змін, а набір методів, що дозволяють надійно зафіксувати їхні факти, якщо вони мали місце [2,19].

Спробуємо знайти універсальні підходи до побудови такого захисту. Насамперед, у розпорядженні одержувача інформації повинна бути процедура перевірки або аутентифікації $A(T)$, що дозволяє перевірити дійсність отриманого масиву даних T. На виході зазначена процедура повинна видавати одне із двох можливих булівих значень – масив даних зорієнтується як справжній, або як неправильний: $A(T)\in\{0,1\}$ для будь–якого припустимого T. Умовимося,

що значення 1 відповідає справжньому масиву даних, а значення 0 – неправильному. Процедура аутентифікації повинна мати наступні властивості, що обмежують можливість зловмисника підібрати масив даних T_1, що відрізняється від справжнього масиву T ($T \neq T_1$), який би проте був би цією процедурою пізнаний як справжній ($A(T_1) = 1$):

- у зловмисника не повинне бути можливості знайти таке повідомлення інакше як шляхом перебору по безлічі припустимих повідомлень – остання можливість є в його розпорядженні завжди;
- імовірність успішно пройти перевірку на дійсність у випадково обраного повідомлення T^* не повинна перевищувати заздалегідь установленого значення p.

Тепер згадаємо про універсальність створюваної схеми захисту, який, зокрема, означає, що схема повинна бути придатної для захисту будь–якого масиву даних T з досить широкого класу. Однак, якщо реалізувати схему буквально, тобто використовувати для перевірки в точності те повідомлення, яке відправник повинен передати одержувачеві, принцип універсальності може прийти в протиріччя із другою вимогою до процедури перевірки. Дійсно, виходячи із цього принципу ми можемо зажадати, щоб усі можливі повідомлення T були припустимими, що зовсім явно порушить друга вимога до функції перевірки. Для того, щоб їх примирити, у схему необхідно ввести додаткові кроки – перетворення даних відправником і зворотне перетворення одержувачем. Відправник виконує перетворення даних з використанням деякого алгоритму F: $T' = F(T)$. Тоді, крім процедури аутентифікації, у розпорядженні одержувача повинна бути процедура G відновлення вихідних даних: $T = G(T')$. Увесь зміст цих перетворень полягає в тому, щоб безліч перетворених повідомлень $\{T'\}$, що взаємно однозначно відображається на безліч припустимих вихідних повідомлень $\{T\}$, було невідоме зловмисникові, і ймовірність випадково вгадати елемент із цієї безлічі була досить мала для того, щоб її можна було не брати до уваги [10,12].

Остання вимога в комбінацією із принципом універсальності однозначно приводить до необхідності внесення певної надмірності в повідомлення, що означає попросту той факт, що розмір перетвореного повідомлення повинен бути більше розміру вихідного повідомлення на деяку величину, саме й складову ступінь надмірності: $|T'| - |T| = \Delta$. Очевидно, що чим більше ця величина, тем менше ймовірність прийняти випадково взяте повідомлення за справжнє – ця ймовірність рівна $2^{-\Delta}$. Якби не вимога внесення надмірності, у якості функцій перетворення F і G даних могли б використовуватися функції зашифрування й розшифрування даних на деякому ключі K: $F(T) = E_K(T)$, $G(T') = D_K(T')$. Однак при їхнім використанні розмір масиву зашифрованих даних T' дорівнює розміру масиву вихідних даних T: $|T'| = |T|$, тому метод тут не підходить [13].

Найбільше природно реалізувати алгоритм перетворення із внесенням надмірності простим додаванням до вихідних даних контрольної комбінації фіксованого розміру, що обчислюється як деяка функція від цих даних: $T' = F(T) = (T,C)$, $C = f(T)$, $|C| = \Delta$. У цьому випадку виділення вихідних даних з перетвореного масиву полягає в простому відкиданні доданої контрольної комбінації C: $T = G(T') = G(T,C) = T$. Перевірка на дійсність полягає в обчисленні для змістовної частини T отриманого масиву даних T' значення контрольної комбінації $C' = f(T)$ і порівнянні його з переданим значенням контрольною комбінацією C. Якщо вони збігаються, повідомлення вважається справжнім, інакше – неправильним:

$$A(T') = \begin{cases} 1, C = f(T) \\ 0, C \neq f(T) \end{cases}.$$

Тепер розглянемо властивості, яким повинна задовольняти функція створення контрольної комбінації f:

1. Ця функція повинна бути обчислювально необоротної, тобто не повинне існувати способу підібрати масив даних T під задану контрольну комбінацію C інакше як перебором по просторі можливих значень T.

2. Ця функція не повинна бути відома зловмисникові – у нього не повинне бути способу обчислити контрольну комбінацію C ні для якого масиву даних T. Ця вимога по суті означає, що функція f повинна бути секретної, розглянемо його докладніше:

- по-перше, відповідно до загальновизнаного в криптографії принципу Кірхгоффа вимога таємності функції створення контрольної комбінації слід замінити на застосування відкритої функції, що використовує вектор секретних параметрів (ключ) – точно так само, як це робиться при побудові шифрів: $C = f(T) = f_K(T)$.
- по–друге, виявляється, що в окремих випадках ця вимога можна істотне послабити. Справа в тому, що дійсна мета цього пункту – виключити для зловмисника можливість відправити неправильне повідомлення T_1, постачивши його коректне обчисленою контрольною комбінацією $C_1 = f(T_1)$. Цього можна досягнути двома такими способами:
  a) за допомогою використаного вище вимоги таємності функції обчислення контрольної комбінації або залежності її від вектора секретних параметрів (ключа);
  b) за допомогою організації такого протоколу використання засобів захисту, який би виключав можливість подібного нав'язування неправильних даних.

Очевидно, що можливість (b) може бути реалізована тільки якщо контрольна комбінація передається або зберігається окремо від даних, що захищаються. Незважаючи на гадану екзотичність, така можливість зустрічається досить часто, мова про неї спереду [1-5].

Розглянемо деякі добре відомі способи обчислення контрольної комбінації й оцінимо можливість їх використання в розглянутій системі імітозахисту даних.

Найпростішим прикладом такої комбінації є контрольна сума блоків повідомлення, узята по модулю деякого числа, звичайно беруть два в ступені розміру блоку:

якщо $T = (T_1, T_2, ..., T_m)$, тоді

$$C = f(T) = (T_1 + T_2 + ... + T_m) \bmod 2^N,$$

де $N = |T_1| = |T_2| = ... = |T_m|$ – розмір блоків повідомлення.

Однак таке перетворення не відповідає обом вищевикладеним вимогам до функції обчислення контрольної комбінації й тому непридатне для використання в схемі імітозахисту:

– По-перше, і це найголовніше – воно не виключає можливість добору даних під задану контрольну комбінацію. Дійсно, нехай відправник інформації передав по ненадійному каналу повідомлення T і контрольну суму C для нього, обчислену по наведеній вище формулі. У цьому випадку все, що буде потрібно зловмисникові для нав'язування одержувачеві довільно взятого неправильного масиву даних $T' = (T'_1, T'_2, ..., T'_{m'})$ – це доповнити його ще одним блоком, обчисленим по наступній формулі:

$$T'_{m'+1} = C - (T'_1 + T'_2 + ... + T'_{m'}) \bmod 2^N.$$

Усі блоки неправильного повідомлення, крім одного, не обов'язково останнього, зловмисник може встановити довільними.

– По-друге, розглянуте перетворення не є криптографічним, і для зловмисника не важко буде виготовити контрольну комбінацію для довільного обраного їм повідомлення, що дозволяє йому успішно видати його за справжнє – якщо контрольна комбінація зберігається або передається разом з масивом, що захищається, даних.

У цей час відомі два підходи до розв'язку завдання захисту даних від несанкціонованої зміни, що базуються на дві викладені вище підходах до створення контрольної комбінації:

1. Створення MAC – *Message Authentification Code – коду аутентифікації повідомлень*. Цей підхід полягає в тому, що контрольна комбінація обчислюється з використанням секретного ключа за допомогою деякого блокового шифру. Важливо, що на основі будь–якого такого шифру можна створити алгоритм обчислення MAC для масивів даних довільного розміру. У літературі MAC іноді не цілком коректно називається криптографічною контрольною сумою, або, що більш точно, криптографічною контрольною комбінацією. Даний підхід до аутентифікації даних загальновизнаний і закріплений практично у всіх криптографічних стандартах – імітовставка, формована відповідно до ДСТУ 3396.0–96 є типовим зразком MAC.

2. Створення MDC – *Manipulation Detection Code – коду виявлення маніпуляцій* (з даними). Для обчислення MDC для блоку даних використовується так звана необоротна функція стиску інформації, у літературі також називана однобічною функцією, функцією однобічного стиску (перетворення) інформації, криптографічної геш–функцією, або просто геш–функцією. Зрозуміло, що її необоротність повинна носити обчислювальний характер:

– обчислення прямої функції $Y = f(X)$ легко здійсненне обчислювально;
– обчислення зворотної функції $X = f^{-1}(Y)$ нездійсненне обчислювально, тобто не може бути виконане більш ефективним шляхом, чому перебором по безлічі можливих значень X.

Обидва способи обчислення контрольної комбінації – MDC і MAC – обирають у якості аргументу блок даних довільного розміру й видають у якості результату блок даних фіксованого розміру [18,19].

Прокоментуємо відмінності: підхід на основі MAC вимагає для обчислення контрольної комбінації секретного ключа, для другого це не потрібно. Потенційний зловмисник не зможе обчислити MAC для довільного сфабрикованого їм повідомлення, але зможе обчислити MDC, тому що для цього не потрібно ніяких секретних даних, тому MAC може передаватися від джерела до приймача по відкритому каналу, тоді як для передачі MDC потрібен захищений канал.

Вдалося б, переваги першого підходу настільки очевидні, що другий підхід не зможе знайти собі застосування. Однак це не так – використання MAC вимагає, щоб попередньо між учасниками інформаційного обміну були розподілені ключі. Якщо ж цього не відбулося, для його реалізації необхідний спеціальний канал, що забезпечує *таємність* і *дійсність* переданої інформації, по якому паралельно з передачею даних по незахищеному каналу будуть передаватися ключі. Для передачі ж MDC потрібен канал, що забезпечує тільки *дійсність* переданих даних, вимога таємності відсутня, і це робить даний метод кращим при одноразовій передачі даних: основна інформація передається по звичайному незахищеному каналу, а MDC повідомляється відправником одержувачеві по каналу, який може прослуховуватися але не може бути використаний для нав'язування неправильних даних – наприклад, голосом по телефону – якщо учасники обміну особисто знайомі й добре знають голоси один одного [11].

У таблиці 1 наведені порівняльні характеристики обох підходів.

Крім того, підхід на основі створення MDC більш простий і зручний для систем, де створення й використання інформаційних масивів розділені в часі, але не в просторі, тобто для контролю цілісності *збереженої*, а не *переданої* інформації – наприклад, для контролю незмінності програм і даних у комп'ютерних системах. При цьому контрольна комбінація (MDC) повинна зберігатися в системі таким чином, щоб виключити можливість її модифікації зловмисником.

Обоє підходу допускають можливість реалізації на основі будь–якого класичного блокового шифру. При цьому надійність отриманої системи імітозахисту, звичайно за умови її коректної реалізації, буде визначатися стійкістю використаного блокового шифру – це твердження вираження легко доводиться. У двох наступних розділах будуть розглянуто обоє підходу до контролю незмінності масивів даних

Таблиця 1 – Порівняльні характеристики підходів до розв'язку завдання контролю незмінності масивів даних

№	Параметр порівняння	Підхід	
		обчислення MAC	обчислення MDC
1.	Використовуване перетворення даних	Криптографічне перетворення (функція зашифрування)	Однобічна функція, функція необоротного стиску інформації
2.	Використовувана секретна інформація	Секретний ключ	Не використовується
3.	Можливість для третьої сторони обчислити контрольну комбінацію	Зловмисник не може обчислити контрольну комбінацію, якщо йому не відомий секретний ключ	Зловмисник може обчислити контрольну комбінацію для довільного блоку даних
4.	Зберігання й передача контрольної комбінації	Контрольна комбінація може зберігатися й передаватися разом з масивом, що захищається, даних	Контрольна комбінація повинна зберігатися й передаватися окремо від масиву, що захищається, даних
5.	Додаткові умови	Вимагає попереднього розподілу ключів між учасниками інформаційного обміну	Не вимагає попередніх дій
6.	Області, у яких підхід має перевага	Захист від несанкціонованих змін даних при їхній передачі	Разова передача масивів даних, контроль незмінності файлів даних і програм

.

Створення коду аутентифікації повідомлень

Створення коду аутентифікації повідомлень із використанням процедури криптографічного перетворення даних офіційно або напівофіційно закріплена в багатьох стандартах на алгоритми шифрування. Так, наприклад, у різних коментарях до стандарту шифрування США рекомендується використовувати DES для створення контрольної комбінації [5]. Український стандарт шифрування ДСТУ 3396.0–96 [6] явно передбачає режим створення імітовставки, яка є не чому іншим, як зразком MAC.

Схема використання криптографічного перетворення EK для створення коду аутентифікації досить проста: усе вихідне повідомлення розбивається на блоки, потім послідовно для кожного блоку перебуває результат перетворення по алгоритму EK побітової суми блоку по модулю 2 з результатом виконання попереднього кроку. Таким чином, одержуємо наступне рівняння для створення контрольної комбінації:

$$C = C_K(T) = E_K(T_1 \oplus E_K(T_2 \oplus E_K(\ldots \oplus E_K(T_m)))).$$

Схема алгоритму створення MAC наведено на рисунку 1.

– Вхідні дані – масив даних T, розбитий на m блоків фіксованого розміру, рівного розміру блоку даних використаного шифру (для більшості найбільш відомих шифрів – 64 біта): $T = (T_1, T_2, ..., T_m)$. Останній блок даних T_m яким–небудь способом доповнюється до повного блоку даних, якщо має менший розмір.

– MAC одержує нульове початкове значення.

Наступний крок алгоритму 2 виконуються послідовного для кожного блоку вихідних даних у порядку їх проходження.

– Побітова сума по модулю 2 чергового блоку вихідних даних T_i с поточним значенням MAC S зазнає перетворенню по алгоритму зашифрування, результат стає новим поточним значенням MAC.

– Результатом роботи алгоритму – MAC для вхідного масиву даних – є останнє поточне значення MAC, отримане на кроці 2.

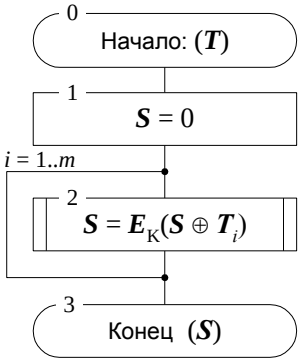

Рис. 1 – Алгоритм створення коду аутентифікації для масиву даних

Розглянемо властивості криптографічних перетворень E_K, використовуваних для шифрування даних, і визначимо ті з них, які необхідні при створення MAC:

1. Перетворення даних повинне використовувати в якості параметра секретний ключ K. Його таємність визначає таємність зашифрованих даних.

2. Перетворення даних повинне бути криптографічне стійким, тобто не повинне існувати іншої можливості визначити вхідний блок алгоритму при відомому вихідному й невідомому ключі, або визначити ключ при відомих вхідному й вихідному блоках інакше як перебором за можливими значенням вхідного блоку й ключа в першому й у другому випадках відповідно.

3. Перетворення даних повинне бути оборотним – для того, щоб була здійсненна процедура розшифрування.

Якщо перетворення, що шифрує E_K передбачається використовувати для створення коду аутентифікації, виконання третьої властивості не потрібно, тому що при цьому перетворення завжди виконується в одну сторону. Крім того криптостійкість алгоритму перетворення може бути трохи нижче, чим при шифруванні, і це не приведе до зниження надійності всієї схеми. Дійсно,

при створення MAC у розпорядженні криптоаналітіка є тільки один блок даних – MAC, який є функцією відразу всіх блоків вихідного тексту, а при зашифруванні в його розпорядженні є набір блоків шифротексту, кожний з яких залежить тільки від одного блоку вихідного тексту. Очевидно, у першому випадку його завдання суттєво складніше. Саме із цієї причини в ДСТУ 3396.0–96 для створення імітовставки використовується спрощений 16–раундовий цикл перетворення, тоді як для шифрування – повний 32–раундовий.

Створення коду виявлення маніпуляцій

Підхід до створення контрольної комбінації масиву даних за допомогою обчислювально необоротних функцій одержав розвиток тільки останнім часом у зв'язку з появою практичних схем цифровому підпису, тому що по своїй суті він є способом обчислення геш–функції, яка використовується у всіх схемах цифрового підпису [9].

Існує велика кількість можливих підходів до побудови обчислювально необоротних функцій, практично завжди самим важким є обґрунтування властивості необоротності запропонованої функції. Однак є клас способів, де така властивість не має потреби в доказі, воно просто випливає з характеристик застосованого методу – це побудова функцій однобічного перетворення на основі класичних блокових шифрів. Даний підхід відомий досить давно й викладений у ряді робіт, з україньскоськомовних відзначимо [7-12], у його основі лежить той факт, що рівняння зашифрування блоку даних по циклу простої заміни $Y = EK(X)$ обчислювально нерозв'язне щодо ключа K – це є невід'ємною властивістю будь–якого дійсно стійкого шифру. Навіть при відомих відкритому (X) і зашифрованому (Y) блоках ключ K не може бути визначений інакше як перебором по безлічі можливих значень. Алгоритм створення контрольної комбінації для масиву даних T наступний:

- масив даних T розбивається на блоки фіксованого розміру, рівного розміру ключа використовуваного шифру:

$$T = (T_1, T_2, ..., T_m);$$
$$|T_1| = |T_2| = ... = |T_{m-1}| = |K|, \ 0 < |T_m| \le |K|.$$

- при необхідності останній (неповний) блок доповнюється яким–небудь образом до блоку повного розміру;
- MDC або геш – повідомлення обчислюється по наступній формулі:

$$C = H(T) = E_{T_m}(E_{T_{m-1}}(...E_{T_1}(S))),$$

де S – початкове заповнення алгоритму – може вибиратися довільно, звичайно вважають, що $S = 0$.

Нескладно довести, що завдання добору масиву даних $T' = (T'_1, T'_2, ..., T'_{m'})$ під задану контрольну комбінацію C еквівалентна наступній системі рівнянь добору ключа для заданих вхідного й вихідного блоків даних криптоалгоритма:

$$E_{T'_1}(S) = S_1,$$
$$E_{T'_2}(S_1) = S_2,$$
$$...$$
$$E_{T'_{m'}}(S_{m'-1}) = C.$$

Немає необхідності вирішувати відразу всі ці рівняння щодо ключа T'_i – усі блоки масиву даних T', крім одного, можуть бути обрані довільними – це визначить, усі значення S_i, і лише один, будь–який з них, повинен бути визначений розв'язком відповідного рівняння $E_{T'_i}(S_{i-1}) = S_i$ відносно T'_i. Тому що дане завдання обчислювально нерозв'язне в силу використання криптостійкого алгоритму шифрування, запропонована схема обчислення MDC має гарантовану стійкість, рівної стійкості використовуваного шифру [15,19].

Однак дана схема не враховує проблему побічних ключів шифру, яка полягає в наступному: може існувати кілька ключів, з використанням яких при зашифруванні однакові блоки відкритого тексту переводяться в однакові блоки шифротексту:

$$E_{K_1}(X) = E_{K_2}(X)$$

при деяких X і $K_1 \ne K_2$.

Один із цих ключів – той, на якому проводилося зашифрування – «дійсний», а інший – «побічний». Таким чином, побічним ключем для деякого блоку даних X і деякого дійсного ключа K називається ключ K', який дає точно такий же результат зашифрування блоку X, що й дійсний ключ K: $E_{K'}(X) = E_K(X)$. Ясно, що для різних блоків вихідного масиву даних побічні ключі також у загальному випадку різні – імовірність зустріти пари ключів, що переводять одночасно трохи пара однакових блоків відкритих текстів у пари однакових блоків шифротекстів стрімко убуває з ростом числа цих пар. Тому виявлення побічного ключа криптоаналітіком при дешифруванні повідомлення не є його особливим успіхом, тому що з імовірністю, що незначно відрізняється від 1, на цьому знайденому ключі він не зможе правильно розшифрувати ніяких інших блоків шифротексту. Зовсім інша справа в алгоритмі створення MDC – тут виявлення побічного ключа означає, що зловмисник підібрав такий неправильний, тобто відсутній у повідомленні блок даних, використання якого приводить до дійсного MDC вихідного масиву даних.

Для того, щоб зменшити ймовірність нав'язування неправильних даних через знаходження побічних ключів, у кроках криптографічного перетворення застосовуються не самі блоки вихідного повідомлення, а результат їх розширення за деякою схемою. Під схемою розширення тут розуміється процедура побудови блоків даних більшого розміру із блоків даних меншого розміру. Прикладом може служити, наприклад, функція розширення, у якій вихідний блок будується з байтів (або 2–,4–,... і т.д. –байтових слів) вихідного блоку, що перелічуються в різному

порядку. Зазначене розширення варто застосовувати, якщо розмір ключа використаного шифру в кілька раз перевищує розмір його блоку даних. Так, для алгоритму DES, з розміром блоку даних 64 біта й ключа 56 біт у розширенні немає необхідності. Якщо в схемі використовується алгоритм [1] з розміром блоку 64 біта й розміром ключа 256 біт, варто використовувати 64– або 128–бітні блоки вихідного тексту й розширювати їх до розмірів 256 біт. Приклад функції розширення 128–бітового блоку в 256–бітовий може бути, наприклад, що випливають:

Вихідний блок:

$$T = (B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8, B_9, B_{10}, B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{16}),$$

Після розширення:

$$P(T) = (B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8, B_9, B_{10}, B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{16}, B_1, B_4, B_7, B_{10}, B_{13}, B_{16}, B_3, B_6, B_9, B_{12}, B_{15}, B_2, B_5, B_8, B_{11}, B_{14}),$$

де Bi – байти блоку даних, $|B_i| = 8$.

Схема алгоритму створення геш–коду з використанням класичного блокового шифру наведено на рис. 2 [1-4].

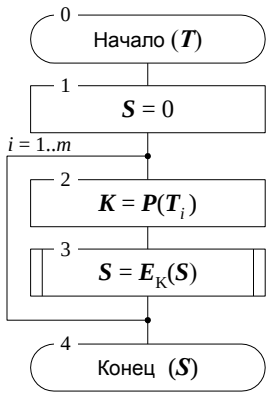

Рис. 2 – Алгоритм створення коду виявлення маніпуляцій для масиву даних

Вхідні дані – масив даних T, розбитий на m блоків фіксованого розміру, що не перевищує розмір ключа використаного криптоалгоритма й, як правило, що ділить його націло: $T = (T_1, T_2, ..., T_m)$. Останній блок даних T_m яким–небудь способом доповнюється до повного блоку даних, якщо має менший розмір.

MDC одержує нульове початкове значення (це значення може бути, у принципі, кожним).

Наступні кроки 2 і 3 алгоритму виконуються послідовного для кожного блоку вихідних даних у порядку їх проходження.

Виконується розширення чергового блоку Ti даних за допомогою функції розширення P до розміру ключа шифру.

Виконується зашифрування поточного значення MDC на ключі, отриманому на кроці 2, результат стає новим поточним значенням MDC.

Результатом роботи алгоритму (тобто MDC для всього вхідного масиву даних) є останнє поточне значення MDC, отримане на кроці 3.

Розглянутий алгоритм також може бути використаний для створення геш–коду в схемах цифрового підпису.

Висновки

Незважаючи на велику кількість методик захисту, програмних рішень та шифрів, на сьогодні не існує надійної системи для комплексного захисту мережі підприємства. Практично будь-яку сучасну систему захисту можна обійти, та будь-який шифр можна зламати. Для цього необхідний лише час та відповідні необхідні ресурси.

Для зламу того чи іншого шифру необхідна різна кількість часу та ті чи інші допоміжні дані, чи то шифротекст, чи то відомості про роботи алгоритму шифрування, або його вхідні параметри. Кожен алгоритм має одну із основних своїх характеристик, яка має назву криптографічна стійкість. Криптографічна стійкість – це властивість алгоритму протидіяти криптографічному аналізу. На сьогоднішній день розрізняються два види систем за криптографічною стійкістю – це абсолютно стійкі та достатньо стійкі системи. Оскільки абсолютно стійкі системи дуже важко реалізувати, тому на практиці їх не використовують. Натомість використовуються достатньо стійкі системи, стійкість яких залежить виключно від того, які можливості у зловмисника. Чим більшими та потужнішими можливостями володіє зловмисник, тим легше йому вдається зламати той чи інший шифр.

Окрім, власне, криптографічних характеристик алгоритму як таких, можуть бути специфічні проблемні ділянки, якими дуже часто користуються зловмисники. До таких ділянок можна віднести слабкі ключі для симетричних алгоритмів, або ж S-блоки, які значно знижуються криптостійкість алгоритму. Знаючи про такі проблемні місця зловмисники можуть будувати на їх основі різні види атак.

Як ми бачимо добитися абсолютно стійкого алгоритму шифрування на практиці дуже важно, а криптографічні алгоритмів з достатньою стійкістю не дають достатнього захисту для того, щоб необхідні дані залишилися в секреті.

Тому тематика даної роботи є досить актуальною на сьогоднішній день. На сьогодні постійно і безупинно ведуться кібератаки напади на основні світові компанії, на основні комунікаційні центри та канали зв'язку. На фоні цього постійно ведеться пошук нових алгоритмів шифрування та методів захисту інформації, Також постійно вдосконалюються відомі алгоритми шифрування від тих чи інших видів атак, підвищується загальна криптографічна стійкість алгоритму.

Список використанних джерел

[1] Брюс Шнайер. Прикладная криптография. – 2-е изд. – М.: Триумф, 2012. – 816 с.
[2] Schneider Electric. Защита систем от кибератак.– Выпуск №36. – 2011.–277с.
[3] Молдовян Н. А. Практикум по криптографии с открытым ключем. – СПб.: БХВ-Петербург, 2013. – 304 с.: ил.
[4] Н. Смарт. Криптография. – Москва: Техносфера, 2005 – 528 с.
[5] Н. Коблиц. Курс теории чисел и криптографии. – М: Науч. изд-во ТВП, 2011. – 254 с.

[6] ДЕРЖАВНИЙ СТАНДАРТ УКРАЇНИ. Захист інформації. Технічний захист інформації. Основні положення. ДСТУ 3396.0–96.
[7] ДЕРЖАВНИЙ СТАНДАРТ УКРАЇНИ 3396.2–97 . Захист інформації, технічний захист інформації, терміни та визначення.
[8] Закон України "Про електронні документи та електронний документообіг" від 22.05. 2003 р. №851–IV.
[9] Закон України "Про електронний підпис" від 22.05. 2003 р. № 852–IV.
[10] Антонюк А.О. Основи захисту інформації в автоматизованих системах/ А.О. Антонюк. – К.: КМ Академія, 2006. – 244 с.
[11] Вербіцький О.В. Вступ до криптології/ О. В. Вербіцький. – Львів: Вид–во НТЛ, 2008. – 248 с.
[12] Герасименко В. А.Основы защиты информации/ В. А. Герасименко. – М.: Инкомбук, 1997. – 537 с.
[13] Домарев В. В. Безопасность информационных технологий. Методы создания систем защиты/ В. Домарев. – К.: ТИД ДС, 2001. – 688 с.
[14] Закон України «Про інформацію». – К.:Відомості Верховної Ради України, 1992. – N 48. – Ст.650 .
[15] Медведовский И.Д. Атака на Internet/ И.Д. Медведовский, П.В., Семьянов, Д. Г. Леонов. – М.: ДМК, 1999. – 336 с.
[16] W. Diffie,M.E.Hellman. New Directions in cryptography// IEEE Trans. Inform. Theory, IT-22, vol 6 (Nov. 1976), pp. 644-654.
[17] У. Диффи. Первые десять лет криптографии с открытым ключом /пер. с англ./ М., Мир, ТИИЭР.–1988.–т.76.–N5.
[18] Касперски К. Атака на Windows NT. // LAN / Журнал сетевыхрешений. 2015, декабрь, С. 88 - 95.

References

[1] B. Schneier. *Prikladnaia kriptografia*. 2-nd ed. Moscow, Triumf, 2012, 816 p.
[2] Schneider Electric. *Zashchita sistemy ot kiberatak*, no. 36, 2011, 277 p.
[3] N. A. Moldovyan, "*Workshop on public key cryptosystems*," SPb. BHV-Petersburg, 2007, 304 p.
[4] N. Smart, *Cryptography*, Moscow, 2005, 528 p.
[5] Koblitz N. *The course of number theory and cryptography*. Moskow, 2011, 254 p.
[6] Derzhavnyi standart Ukrainy. zahist informatsiyi. Tehnichniy zahist informatsiyi. Osnovni polozhennya. DSTU 3396.0–96.
[7] Derzhavnyi standart Ukrainy. Zahist informatsiyi 3396.2–97 . Zahist informatsiyi, tehnichniy zahist informatsiyi, termini ta viznachennya.
[8] Zakon Ukrainy "Pro elektronni dokumenti ta elektronniy dokumentoobig" 22 May, 2003, no. 851–IV.
[9] Zakon Ukrainy "Pro elektronniy pidpis" , 22 May, 2003, no. 852–IV.
[10] A.O. Antoniuk, "Osnovi zahistu InformatsIYi v avtomatizovanih sistemah," Kyiv, KM AkademIya, 2006, 244 p.
[11] O.V. Verbitskiy, "Vstup do kriptologiyi," LvIv, NTL, 2008, 248 p.
[12] V. A. Gerasimenko, "Osnovyi zaschityi informatsii," Moscow,Inkombuk, 1997, 537 p.
[13] V. V. Domarev, "Bezopasnost informatsionnyih tehnologiy. Metodyi sozdaniya sistem zaschityi," Kyiv, TID DS, 2001, 688 p.
[14] Zakon Ukrainy "Pro InformatsIyu," Kyiv,vidomosti Verhovnoyi Rady Ukrainy, 1992, no. 48, 650 p.
[15] I. D. Medvedovskiy, P.V., Semyanov, D. G. Leonov "Ataka na Internet," Moscow, DMK, 1999, 336 p.
[16] W. Diffie, M.E.Hellman. New Directions in cryptography// IEEE Trans. Inform. Theory, IT-22, vol 6 (Nov. 1976), pp. 644-654.
[17] W. Diffie. First 10 years of cryptography with open key. Trans. eng., Moscow, Mir,1988, vol.76, no.5.
[18] K. Kasperski. Ataka na Windows NT. LAN. Zhurnal setevykh reshenii, Dec. 2015, pp. 88-95.

УДК 681.5.033.3

ПРИКЛАДИ АНАЛІЗУ СТАЛИХ ПРОЦЕСІВ В САР ЗАСОБАМИ MATLAB

Левінський М.В.[1], Левінський В.М.[2]

[1] Національний університет «Одеська морська академія», м. Одеса, Україна
[2] Одеська національна академія харчових технологій, м. Одеса, Україна
ORCID: [1] 0000-0002-6544-5110, [2] 0000-0002-3563-528X
E-mail: [1] MaxLevinskyi@gmail.com, [2] ValeryLevinskyi@gmail.com

DOI: https://doi.org/10.15673/atbp.v12i2.1810