УДК 004.056.55; 004.75

П.В. Ковальський, П.О. Кравець,
Національний університет “Львівська політехніка”,
кафедра інформаційних систем та мереж

ІНФОРМАЦІЙНА МУЛЬТИАГЕНТНА СИСТЕМА ВИПРОБУВАННЯ СТІЙКОСТІ АЛГОРИТМІВ ШИФРУВАННЯ ДАНИХ

Розглянуто мультиагентну систему для випробування стійкості алгоритмів шифрування даних. Стійкість визначається часом, необхідним для дешифрування тексту, зашифрованого ключем заданої довжини. Результатом роботи мультиагентної системи є оцінювання часу підбору ключа для конкретного алгоритму шифрування та обсягів задіяних для цього ресурсів.

The multiagent system for testing firmness of cryptography algorithms is beeing viewed in this article. Firmness is determined, by the time which is necessary for decrypting text which is encrypted by the key of the set length. The result of multyagent system is an evaluation of time for search key to the concrete algorithm of cryptography and volumes which were involve for this purpose resource.

Вступ

Основними напрямами використання сучасних криптографічних методів є передавання конфіденційної інформації каналами зв'язку (наприклад, електронною поштою), встановлення достовірності повідомлень (електронний підпис), зберігання інформації (документів, баз даних) на носіях у зашифрованому вигляді [1].

Для запобігання промислового шпіонажу в конкурентних умовах роботи організацій та підприємств актуальною є проблема розроблення системи шифрування конфіденційної інформації на основі вибору надійного алгоритму шифрування даних. Для вирішення цієї проблеми виділяють два послідовні етапи [2]:

- доведення надійності алгоритму на основі математичних методів;
- перевірка алгоритму на практиці.

Другий етап передбачає умисне зламування алгоритму шифрування з використанням засобів комп’ютерної техніки. Тут можливе використання методу Brute Force (“груба сила”), що передбачає зламування шифру простим перебором всіх можливих варіантів ключів. Інші відомі методи дають змогу звузити діапазон пошуку ключів на основі наявної апріорної інформації, що в результаті призводить до пришвидшення криптоаналізу шифру.

Для перебору повного діапазону ключів шифру середньої довжини одному комп’ютеру знадобиться багато років. Вирішити цю проблему допомагає мережа Інтернет. Технології розподілених обчислень дають змогу об’єднати тисячі комп’ютерів в одну обчислювальну систему, яка може розв’язувати надоб’ємні задачі. Розв’язування задачі криптографічного аналізу можна досягти за допомогою програмної мультиагентної системи, яка дає змогу визначити затрати часу для зламування того чи іншого шифру, враховуючи темпи розвитку мережі Інтернет та потужності комп’ютерної техніки, які потенційно можна залучити для зламу шифру, вирахувати безпечний термін використання того чи іншого алгоритму шифрування та оптимальну довжину ключа.

З погляду побудови та реалізації мультиагентна система найпридатніша для вирішення проблеми випробування алгоритмів шифрування даних. Як відомо, агент – це апаратна або програмна сутність, здатна діяти в інтересах досягнення цілей, поставлених перед нею

користувачем. У рамках мультиагентних систем розглядаються агенти, як автономні компоненти, що діють за певним сценарієм. Виконана кожним агентом робота наближає систему до розв'язання поставленої задачі загалом [3].

Мультиагентні системи, як правило, будуються на основі інтелектуальних агентів, які, крім автономного виконання, підтримують взаємодії з іншими агентами, мають властивості адаптивного поводження, навчання на прецедентах і толерантність до помилок [4].

Метою цієї роботи є випробування стійкості криптографічних систем на основі мультиагентної системи зламання ключів шифрування даних. Як відомо, в більшості випадків надійність симетричної криптосистеми ґрунтується на довжині ключа шифрування. Пропонована мультиагентна система демонструє вплив довжини ключа на час зламання шифру. Метою роботи мультиагентної системи є виявлення слабких ключів, ключів середнього захисту та надійних ключів. Результат роботи системи залежить від виду алгоритму шифрування [1].

Криптографія на основі ключа

Протягом тривалого часу криптографія використовувалась винятково у військових цілях. Агентство національної безпеки США (National Security Agency, NSA) та його аналоги в колишньому Радянському Союзі, Англії, Франції, Ізраїлі та інших країнах витрачали мільярди доларів для забезпечення захисту власних ліній зв'язку, одночасно намагаючись зламати всі відомі системи шифрування. Протягом останніх 30 років значно зріс обсяг відкритих академічних досліджень у цьому напрямі. Сьогодні мистецтво криптографії вийшло за стіни військових відомств. Непрофесіонали отримали засоби, які дозволяють захистити себе від зловмисників надійними методами захисту інформації.

Криптографічний алгоритм, який також називається шифром, являє собою математичну функцію, яка використовується для шифрування та дешифрування даних.

Якщо безпека системи шифрування даних ґрунтується на збереженні в таємниці тільки алгоритму шифрування, то вона є обмеженою. Обмежені системи шифрування мають лише історичний інтерес і не зовсім відповідають сьогоднішнім стандартам. Незважаючи на це, обмежені алгоритми шифрування досить популярні для програм з низьким рівнем безпеки. Користувачі або не розуміють проблем, пов'язаних із захистом своїх систем, або не турбуються про них [5].

Сучасна криптографія вирішує проблему шифрування даних за допомогою ключа K. Такий ключ може бути довільним елементом великої множини можливих значень ключів. Шифрування та дешифрування залежить від значення та вигляду ключа. Математично це виглядає так:

$$E_K(M) = C,$$

$$D_K(C) = M,$$

де E_K – функція шифрування, D_K – функція дешифрування, M – звичайний відкритий текст, C – зашифрований текст.

При цьому виконується наступна рівність:

$$D_K(E_K(M)) = M\ .$$

Безпека таких алгоритмів повністю ґрунтується на ключах, а не на деталях реалізації алгоритмів. Це означає, що алгоритм може бути опублікований та проаналізований. Продукти, які використовують цей алгоритм, можуть тиражуватися. Немає значення, що зловмиснику відомий алгоритм шифрування. Якщо йому не відомий конкретний ключ, то він не зможе прочитати зашифроване повідомлення. Криптосистему з використанням ключа зображено на рис. 1.

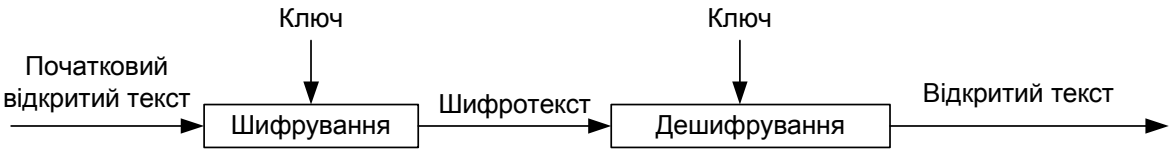

Рис. 1. Шифрування та дешифрування за допомогою ключа

Криптосистема, в якій для шифрування та дешифрування використовується один і той самий ключ, називається симетричною. Безпека симетричної криптосистеми є функцією двох факторів: надійності алгоритму та довжини ключа [6].

Припустимо, що надійність алгоритму є досконалою. Під цим розуміється відсутність кращого шляху зламання криптосистеми, ніж зламання “грубою силою” за допомогою перебору всіх можливих ключів.

Для виконання такого зламання криптоаналітику потрібний фрагмент шифротексту та відповідний фрагмент оригінального тексту.

Розрахувати складність зламання “грубою силою” не важко. Якщо використовується 8-бітний ключ, то існує 2^8, або 256 можливих ключів. Якщо довжина ключа дорівнює 56 бітам, то існує 2^{56} можливих ключів. Якщо комп’ютер може перевіряти 1000 000 ключів за секунду, то пошук ключа в середньому займе 2285 років [1].

Але потрібні потужності для зламання шифру важко передбачити. Найкращий спосіб перевірки шифру на стійкість – це розроблення системи для його зламання.

Мультиагентний криптоаналіз

Задача зламання "грубою силою" може бути ефективно розв'язана за допомогою мультипроцесорних систем. Кожен процесор перевіряє підмножину із загальної множини ключів. Процесорам не потрібно обмінюватись між собою великими об’ємами інформації – єдиним повідомленням у такому випадку є повідомлення про знайдений ключ [1].

Алгоритм роботи машини для зламання шифру „грубою силою” описано в [7, 8]. Вартість проекту сягає близько мільйона доларів. Машина може зламати 56-бітний DES ключ в середньому за 3.5 години (гарантовано за 7 годин) [1]. Скоротити процес обчислень можна за допомогою конвеєризації [9].

Зараз для зламання шифру не потрібно конструювати дорогу надпотужну машину. Використовуючи мережу Інтернет, можна об'єднати мільйони комп'ютерів у розподілену систему для зламання шифру. Як показують проекти розподілених обчислень, багато користувачів мережі Інтернет надають процесорний час своїх комп'ютерів безкоштовно заради цікавості взяти участь у таких проектах [10]. Розподілена обчислювальна система є прототипом тієї, яку зловмисник може реально спроектувати та реалізувати. Тільки при спробі самостійного зламання шифру можна говорити про надійність або ненадійність алгоритму шифрування, про необхідну довжину ключа та про безпечні терміни їх використання.

Розподілені обчислення найкраще реалізувати за допомогою мультиагентної системи. Така система передбачає створення та розповсюдження слабозв'язаних між собою програмних агентів. У такому випадку програмному агенту потрібно отримати від користувача мультиагентної системи діапазон ключів, а після завершення роботи агент повертає повідомлення про результат пошуку ключа. На основі мультиагентної системи розроблено декілька проектів зламання шифрів.

Проект RC5 "Bovine" був розпочатий з метою "грубої атаки" на шифр RC5 [1]. Основою проекту є координаційні RC5 сервери, які розподіляють для перевірки блоки ключів між тими учасниками проекту, у яких працює спеціальна програма-клієнт. Час тривалості проекту дуже залежить від кількості учасників, оскільки дешифрування ведеться методом простої перевірки всіх можливих ключів.

Правильний 56-бітовий ключ проекту RC5-32/12/7 було знайдено за 250 днів у 1997 році. У 2002 році було знайдено 64-бітний ключ RSA Labs за проектом RC5-32/12/8. На його пошук знадобилося 1757 днів.

3 грудня 2002 року стартував проект RC5-32/12/9, який передбачає пошук 72-бітного ключа RSA Labs [11]. За розв’язання цієї задачі компанія RSA Labs оголосила призовий фонд у розмірі $10,000 (US). Голосування здійснюється через спеціальне розширення сервера статистики, де кожен виконаний блок вважається за один голос [11].

Для залучення якомога більшої кількості учасників корпорація distributed.net реалізувала проект RC5-32/12/9 у формі гри. Бажаючі взяти участь формують окремі команди, які змагаються за

пошук правильного ключа. На сайті distributed.net постійно ведеться статистика щодо кількості ключів, опрацьованих кожною командою.

Статистика проекту відображає також обсяги робіт, виконаних кожним учасником проекту: блоки завдань, процесорний час та ін. Для досягнення високого місця в рейтингах окремі учасники постійно модернізують та нарощують потужності своїх комп'ютерів. Великі команди часто розпочинають свої власні міні-змагання, наприклад, хто набере більшу кількість балів за тиждень. Завдяки цьому розподілені обчислення стають цікавішими для дійсних та потенційних учасників проекту.

Система, яка використовується в distributed.net, працює над випробуванням одного алгоритму шифрування з невеликим ключем (56, 64, 72 біти) [12]. Пропонована нами мультиагентна система може працювати за різними алгоритмами. Кожен алгоритм шифрування та дешифрування оформляється в окремому модулі. Це дає змогу заміняти алгоритми шифрування без заміни програмних агентів та без зміни структури бази даних. Основна відмінність пропонованої системи полягає в тому, що підбір починається з однобайтних ключів і у міру вичерпання множини ключів послідовно збільшується до довжини, яка є максимальною для конкретного алгоритму шифрування. Тобто, коли множина однобайтних ключів вичерпується, починають використовувати двобайтні ключі і так далі. Крім того, після перебору множини ключів фіксованої довжини формується проміжний висновок. Цей висновок містить рекомендації щодо використання алгоритму та ключа відповідної довжини, а саме, скільки часу можна безпечно зберігати інформацію, зашифровану ключем такої довжини.

Модель мультиагентної системи випробування стійкості шифрування даних

Для моделювання мультиагентної системи доцільно скористатися мовою моделювання UML [13]. Діаграма в UML – це графічне подання набору елементів, зображуване у вигляді зв'язаного графу з вершинами (сутностями) і ребрами (відношеннями). Діаграми зображають для візуалізації структурно-функціональних особливостей системи.

Діаграми прецедентів мають велике значення для візуалізації та документування поводження елемента. Вони полегшують розуміння системи, підсистем та класів, подаючи погляд ззовні на те, як такі елементи можуть використовуватись у відповідному контексті. Крім того, такі діаграми важливі для тестування системи та розуміння її внутрішньої будови при зворотному проектуванні [13]. Діаграму прецедентів мультиагентної системи зображено на рис. 2.

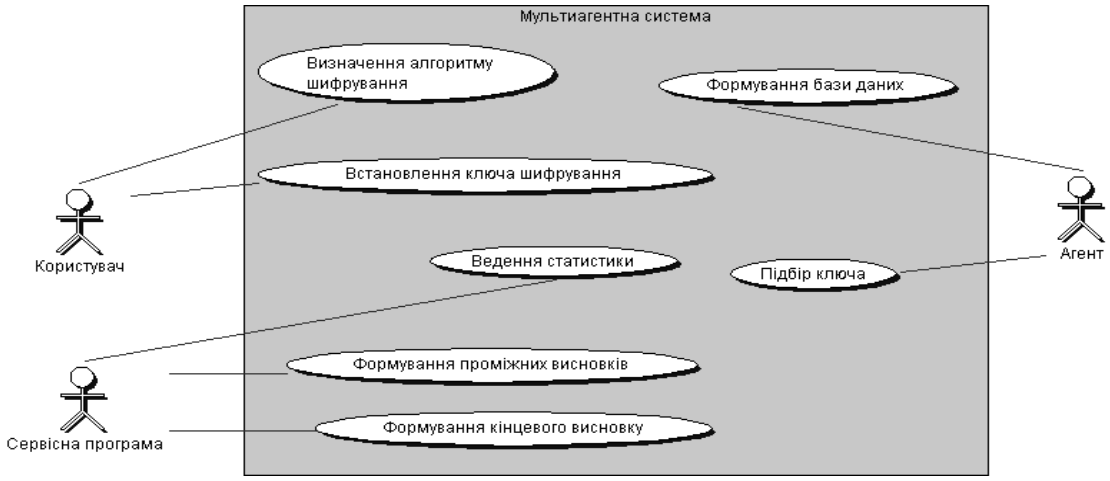

Рис. 2. Діаграма прецедентів

На діаграмі прецедентів зображені актори – сутності, які є дійовими особами системи. У нашому випадку акторами є користувачі системи, сервісні програми та агенти. Кожен актор асоціюється з певними прецедентами.

До завдань користувача або організатора мультиагентної мережі належать: вибір алгоритму шифрування, створення невеликого тексту для подальшого його шифрування та формування ключа шифрування.

Основна робота сервісної програми – це генерація діапазонів ключів для програмних агентів. На відміну від системи підбору ключів в корпорації distributed.net, в якій відбувається підбирання всього діапазону можливих ключів, сервісна програма пропонованої мультиагентної системи генерує тільки ті ключі, які мають ASCII коди символів з графічним позначенням, оскільки ця система кодування є найпоширенішою. Також вибір ключів здійснюється не послідовно, а методом ділення діапазону ключів навпіл.

Основною роботою програмного агента є визначення поточної кількості необхідних для перевірки ключів та безпосередній перебір отриманих ключів за допомогою сервісної програми. Сервісна програма отримує від агента інформацію про комп'ютер учасника, кількість потрібних ключів та інформацію про те, чи знайдено ключ. Натомість агент отримує діапазон ключів та команду про припинення чи продовження роботи.

Детальніше розкрити мету такої мультиагентної системи можна за допомогою дерева розв'язків. Сукупність листків дерева відображає набір дій, виконання яких призводить до головної мети – досягнення верхівки дерева розв'язків. Дерево розв'язків пропонованої мультиагентної системи зображено на рис. 3.

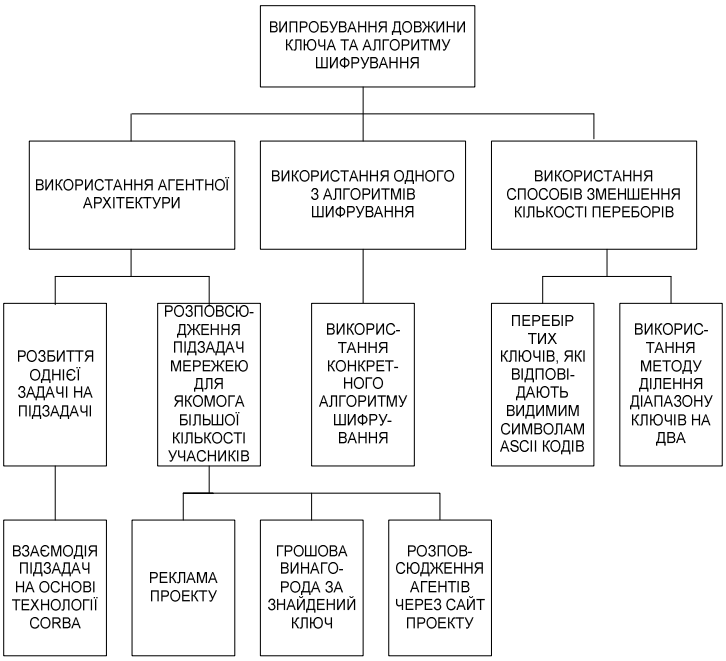

Рис. 3. Дерево розв'язків

Робота будь-якої інформаційної системи полягає у збиранні, збереженні та опрацюванні даних. У пропонованій мультиагентній системі напрацьовані дані є основою для формування результату її роботи. Під час роботи системи буде запам'ятовуватись інформація про комп'ютери учасників та агентів, які на них працюють, час підбору діапазонів ключів та інша статистична інформація. Отже, системі потрібно два відношення (рис. 4). Перше відношення – для збереження інформації про комп'ютери, де розміщуються агенти, а друге відношення – для статистичної інформації.

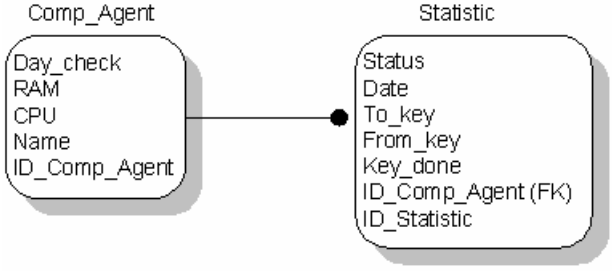

Рис. 4. ER – діаграма бази даних

Перше відношення Comp_Agent містить такі атрибути:

- ID_Comp_Agent – первинний ключ відношення;
- Name – ім’я комп’ютера, на якому виконується агент;
- CPU – тактова частота процесора певного комп’ютера (потрібно для обчислення швидкості підбирання ключа, ключ/сек.);
- RAM – оперативна пам’ять комп’ютера, де виконується агент;
- Day_check – середня кількість ключів, які комп’ютер може опрацювати за добу.

Друге відношення Statistic містить такі атрибути:

- ID_Statistic – первинний ключ відношення;
- ID_Comp_Agent – вторинний ключ відношення;
- From_key – початкове значення ключа шифрування;
- To_key – кінцеве значення ключа шифрування;
- Date – дата запису цієї інформації;
- Status – статус про те, чи знайдено ключ (так/ні).

Атрибути From_Key та To_Key означають діапазон ключів, які агент з ідентифікатором ID_Comp_Agent планує опрацювати за одну добу.

Для формування проміжних висновків в такій мультиагентній системі використовується відношення Recomendations (рис. 5). Підпрограма, яка використовує це відношення, спочатку аналізує відношення Comp_Agent та Statistic, а потім, на основі часу перебору тестованого діапазону ключів, формує проміжний висновок.

У відношенні Recomendations містяться такі атрибути:

- ID – ідентифікатор запису, який однозначно ідентифікує запис у відношенні;
- Recomendation – рекомендація щодо використання ключа певної довжини;
- days_from – нижня межа часу підбору ключа шифрування (в днях);
- days_to – верхня межа часу підбору ключа шифрування (в днях).

Рис. 5. ER – діаграма бази правил

Реалізацію такої мультиагентної системи пропонується здійснити засобами мови розподіленого програмування Java. Оскільки написані мовою Java програми працюють на багатьох популярних операційних системах, це дасть змогу залучити до проекту щонайбільше комп’ютерів. Відповідно, імовірність знайти ключ шифрування збільшиться.

В організаційному плані для реалізації проекту потрібно створити інтернет-сайт, через який учасники проекту могли б завантажити програму-агент. Також треба розрекламувати проект для залучення якомога більшої кількості учасників. Наприклад, можна запропонувати певну грошову винагороду тому учаснику, комп’ютер якого знайде правильний ключ.

Алгоритм роботи мультиагентної системи

Алгоритм роботи мультиагентної системи з підбирання ключа шифрування складається з таких кроків.

1. *Підготовка*. Організатор проекту (організація, яка формує мультиагентну мережу) формує текст та ключ шифрування. Визначається алгоритм шифрування. Відкритий текст, ключ

шифрування, та модуль алгоритму шифрування надається сервісній програмі. Сервісна програма зашифровує відкритий текст заданим алгоритмом шифрування за допомогою заданого ключа.

2. *Розповсюдження.* Програмні агенти розповсюджуються через сайт проекту та безпосередньо між учасниками проекту.

3. *Ініціалізація агента.* Учасник проекту запускає отриману програму-агент; агент, своєю чергою, встановлює зв'язок із сервісною програмою. Програма-агент аналізує апаратні можливості комп'ютера та обчислює кількість ключів, необхідну для першого перебору тривалістю 7 днів. Сервісна програма реєструє комп'ютер, на якому працює агент, надає оригінальний та зашифрований тексти, а також перший діапазон ключів. Попрацювавши 7 днів, агент обчислює середню кількість ключів, які він може опрацювати за добу, та періодичність з'єднання з мережею Інтернет.

4. *Відсилання та отримання даних.* Опрацювавши дані, програмний агент повідомляє сервісну програму про те, чи знайдено ключ шифрування. На основі аналізу комп'ютера учасника програмний агент запитує потрібну йому кількість ключів.

5. *Робота агента.* Агент послідовно починає підбирати ключ за ключем. Для цього зашифрований текст він розшифровує першим ключем з заданого діапазону, потім порівнює результат з оригінальним текстом. Якщо тексти однакові, то ключ знайдено, і виконується перехід на крок 4, інакше агент випробовує наступний ключ з діапазону. Якщо діапазон ключів вичерпався, відбувається перехід на крок 4.

6. *Знайдено ключ.* Якщо ключ знайдено, сервісна програма робить певну позначку в базі даних. Агенти, які звертаються за новим діапазоном ключів, перевіряють це значення. Якщо воно істинне – агенти припиняють роботу, а сервісна програма на основі записів з бази даних формує висновок щодо використання ключа певної довжини разом із заданим алгоритмом шифрування.

Наведений вище алгоритм роботи мультиагентної системи доцільно зобразити засобами UML у вигляді діаграми дій, наведеної на рис. 6.

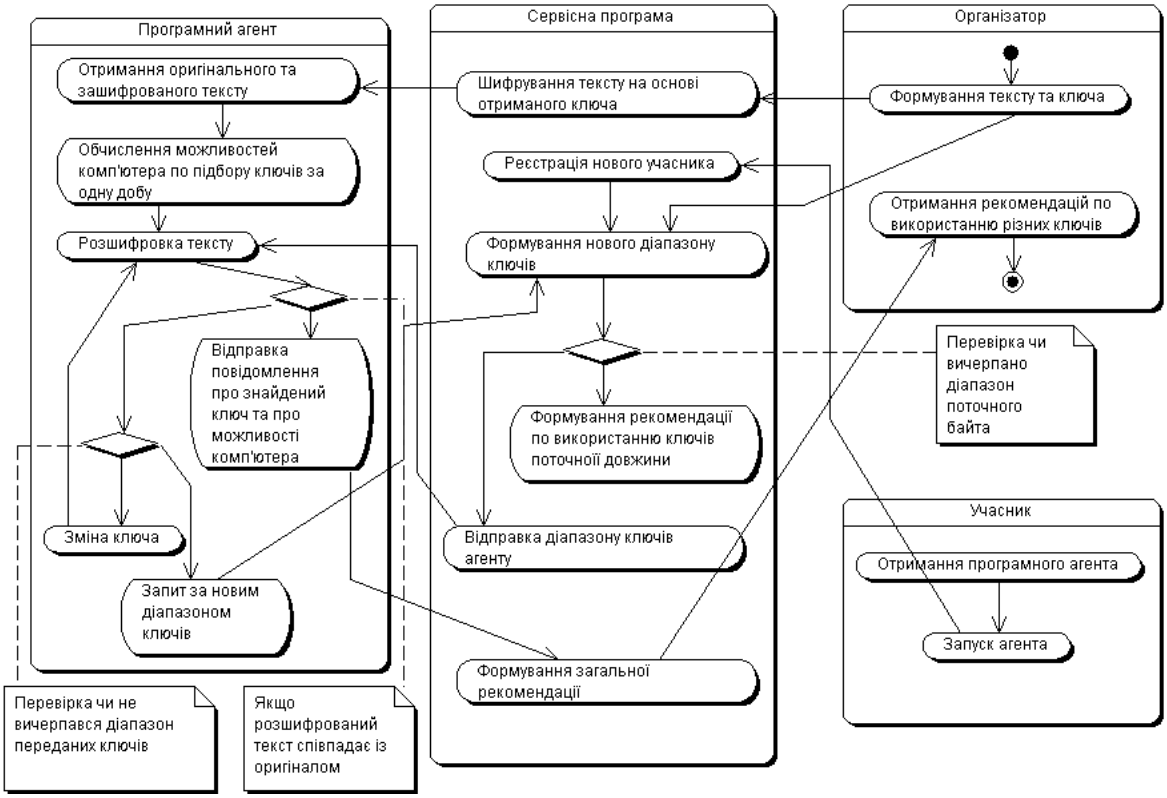

Рис. 6. Діаграма дій

Ця діаграма є важливою не тільки для моделювання динамічних аспектів поводження системи, але й для побудови виконавчих систем за допомогою прямого та зворотного проектування [13].

Висновки

У зв'язку із постійним збільшенням потужностей комп'ютерів у зловмисників з'являються шанси зламати шифр методом повного перебору діапазону ключів. Час зламання шифру визначається добутком кількості можливих ключів шифру на середній час перевірки одного ключа та на кількість задіяних комп'ютерів. Для захисту від зламання шифру необхідно знати достовірні часові терміни, протягом яких зловмисник не зможе зламати цей шифр. Таку інформацію можна отримати за допомогою запропонованої мультиагентної системи випробування стійкості алгоритму шифрування даних, що ґрунтується на підбиранні ключа шифрування змінної довжини. Система моделює реально можливу ситуацію зламування шрифтів та призначена для випробування стійкості алгоритмів шифрування даних і надійності вибраних довжин ключів.

Пропонована мультиагентна система ґрунтується на аналізі проміжних результатів для ключів змінної довжини, тому її можливості підбору шрифтів є кращими, ніж у відомих системах з фіксованою довжиною ключа.

Недоліком пропонованої мультиагентної системи є те, що результати роботи можуть бути отримані тільки через тривалий час (можливо, 1 рік і більше, залежно від кількості агентів та потужності обчислювальних засобів). Реалізація проекту вимагає значних процесорних ресурсів, часу та реклами для залучення потенційних учасників для перебору ключів.

Незважаючи на це, перспективи використання мультиагентної системи є добрими, оскільки користувачам потрібна достовірна інформація про безпечність використання їх ключів.

1. Шнаэр Б. Прикладная криптография. – СПб: Питер, 2000. 2. S.R. White, "Covert Distributed Processing with Computer Viruses", Advances in Cryptology CRYPTO '89 Proceedings, Springer-Verlag, 1990, pp. 616-619. 3. Katia P. Sycara: Multiagent systems, AI Magazine, Vol. 10, No. 2, 1998, pp. 79-93. 4. Mike Wooldridge and Nick Jennings: Intelligent Agents: Theory and Practice, Knowledge Engineering Review, v10n2, June 1995 5. Attack Against the DES, 04/11/2002 http://logic.stanford.edu/software/magenta/magentacpp.html. 6. J. R. Searle: Speech Act, Cambridge University Press, 1969 7. M.J. Wiener, "Efficient DES Key Search." presented at the rump session of CRYPTO '93, Aug 1993 8. M.J. Wiener, "Efficient DES Key Search," TR-244, School of Computer Science, Car leton University, May 1994. 9. Borue, S. U., Ermolayev, V. A. Tolok V. A. : "Application of diakoptical MAS framework to planning process modelling" Submitted to UkrPROG'2000, May 23-26, 2000, Kiev, Ukraine 10. E Hendessi and M.R. Arcf, "A Successful Attack Against the DES, " Third Canadian Workshop on Information Theory and Applications, Springer-Verlag, 1994, pp. 78–90. 11. RC5-72 Clients Available!, 24/01/2005 http://www.distributed.net/ 12. Distributed Computing: Distributed Communities by Howard Feldman 05/22/2003. http://www.onlamp.com/pub/a/onlamp/2003/05/22/distributed.html. 13. Буч Г., Рамбо Дж., Якобсон. Язык UML. Руководство пользователя. – 2004.