УДК 004.056.5

МЕТОД ПРОТИДІЇ АТАКАМ НА ТАБЛИЦІ МАРШРУТИЗАЦІЇ НА ОСНОВІ АРХІТЕКТУР БОТНЕТІВ ДЛЯ ОДНОРАНГОВОЇ ПІРИНГОВОЇ МЕРЕЖІ BITCOIN

П.І. СТЕЦЕНКО, Г.З. ХАЛІМОВ

Представлений метод протидії атакам на таблиці маршрутизації для однорангової пірингової мережі Bitcoin. Метод базується на архітектурах ботнетів. Клас атак на таблиці маршрутизації включає в себе атаки затемнення, ботнет-атаки та атаки інфраструктури. Метод включає в себе послідовність із десяти дій. Дії 1 – 4 рекомендується застосовувати в зв'язці, дія 5 є самостійною, дії 6 – 10 є опціональними. Розглянуті вразливості механізму адресації пірингової мережі Bitcoin. Розглянуто 4 сценарії зайняття зловмисником таблиці перевірених адрес вузла, розраховано порівняльні оцінки очікуваної кількості перезаписаних адрес. Метод дозволяє значно підвищити складність реалізації атак на таблиці маршрутизації, окремі дії підвищують необхідну кількість ресурсів для атаки вдвічі.

Ключові слова: криптовалюта Bitcoin, однорангова пірингова мережа, таблиця маршрутизації, ботнет-атака, одноранговий вузол, таблиця перевірених адрес, таблиця нових адрес.

ВСТУП

Сьогодні активно розвиваються різні додатки, в основі яких лежить концепція криптовалюти Bitcoin. Ключовими особливостями цих додатків є відкритість, децентралізація і концепція однорангової пірингової мережі. Останні кілька років проводилися дослідження безпеки протоколу Bitcoin, в той час як безпека однорангової пірингової мережі залишалася недостатньо дослідженою. Таким чином, дослідження безпеки даних мереж і розробка методу протидії, що підвищує складність реалізації атак на таблиці маршрутизації і не порушує принципи відкритості і децентралізації, є актуальною проблемою щодо широкого ряду додатків.

Аналіз публікацій

Криптовалюта Bitcoin використовує неструктуровану однорангову мережу. Замість аналізу специфічних особливостей існуючої мережі Bitcoin, ряд робіт присвячений розробці нових неструктурованих мереж, які є стійкими до візантійських атак [1, 2, 3, 4]. Внесення в чорний список некоректно функціонуючих тимчасових вузлів описується в роботі [4]. Централізоване рішення на основі інфраструктури відкритих ключів, яке не підходить для Bitcoin, представлено в роботі [2]. Проект Brahms є повністю децентралізованим, і обмежує швидкість обміну інформацією між одноранговими вузлами, що значно відрізняється від концепції Bitcoin [3]. Багато ботнетів, у тому числі і Bitcoin, використовують неструктуровані однорангові пірингові мережі і миттєвий обмін повідомленнями [5].

Визначення проблеми

У проаналізованих роботах не пропонувалися рішення щодо підвищення складності реалізації атак на таблиці маршрутизації в рамках однорангової пірингової мережі Bitcoin. Проблема розробки комплексного методу протидії такому класу атак є актуальним завданням не тільки для криптовалюти Bitcoin, а й для ряду додатків, побудованих на аналогічній платформі.

Мета роботи

Метою даної роботи є розробка методу протидії на основі архітектур ботнетів щодо класу атак на таблиці маршрутизації, який включає в себе атаки затемнення, ботнет-атаки і атаки інфраструктури.

Для досягнення поставленої мети необхідно вирішити такі задачі:

- провести аналіз механізму зберігання мережної інформації в одноранговій пірингової мережі Bitcoin;
- навести структурне подання ботнет-атаки на однорангову пірингову мережу Bitcoin;
- розробити метод протидії щодо атак на таблиці маршрутизації.

1. МЕХАНІЗМ ЗБЕРІГАННЯ МЕРЕЖНОЇ ІНФОРМАЦІЇ

Зовнішні IP-адреси зберігаються в таблицях перевірених і нових адрес вузла. Таблиці знаходяться на диску і зберігаються, коли вузол перезавантажується.

Таблиця перевірених адрес складається з 64 блоків, кожен з яких може зберігати до 64 унікальних адрес для однорангових вузлів, з якими вузол успішно встановив вхідне чи вихідне з'єднання. Поряд з кожною збереженою адресою однорангового вузла, вузол зберігає мітку часу останнього успішного підключення до цього однорангового вузла.

Кожна адреса однорангового вузла відображається в блок у таблиці перевірених адрес, шляхом взяття геша від IP-адреси однорангового вузла і групи однорангового вузла, де група є /16 IPv4-префіксом, що містить IP-адресу однорангового вузла. Блок таблиці перевірених адрес вибирається у такий спосіб:

```
i = Hash( SK, IP ) % 4
Bucket = Hash( SK, Group, i ) % 64
return Bucket,
```

де SK – випадкове число, вибране з появою вузла;
IP – IP-адреса одноранговогo вузла і номер порту;
Group – група одноранговогo вузла.

Таким чином, кожна IP-адреса відображається в окремому блоці таблиці перевірених адрес і кожна група відображається в не більше ніж чотирьох блоках.

Адреса одноранговогo вузла вставляється у відповідний блок таблиці перевірених адрес, коли вузол успішно підключається до одноранговогo вузла. Механізм Bitcoin витискання застосовується, якщо блок заповнений (тобто містить 64 адреси). Даний механізм складається з таких етапів:

1) з блоку випадковим чином вибираються чотири адреси;

2) адреса, що має найстаршу мітку часу, замінюється адресою нового одноранговогo вузла в таблиці перевірених адрес;

3) вставляється до таблиці нових адрес.

Мітка часу, пов'язана з адресою одноранговогo вузла, оновлюється, якщо адреса одноранговогo вузла вже присутня в блоці. Мітка часу також оновлюється, коли активно підключений одноранговий вузол передає VERSION, ADDR, INVENTORY, GETDATA або PING повідомлення і пройшло більше 20 хвилин з моменту останнього оновлення.

Таблиця нових адрес складається з 256 блоків, кожен з яких може містити до 64 адрес для одноранговых вузлів, з якими вузол ще не ініціював успішного з'єднання. Вузол наповнює таблицю нових адрес інформацією, яка отримана від DNS сідерів, або з ADDR повідомлень.

Визначення 1. ADDR повідомлення – це повідомлення адресації для отримання мережної інформації від одноранговых вузлів, містить в собі до 1000 IP-адрес з їх мітками часу.

Визначення 2. DNS-сідер – це сервер, який відповідає на DNS-запити від Bitcoin-вузлів криптографічно неавтентифікованим списком IP-адрес для Bitcoin-вузлів.

Адреси в таблиці нових адрес також мають пов'язані з ними мітки часу. На адреси, отримані від DNS-сідерів, ставиться випадкова мітка часу давністю від 3 до 7 днів, у той час як на адреси, витягнуті з ADDR повідомлень, ставиться їх мітка часу з ADDR повідомлення плюс дві години.

Кожна адреса *a* вставляється в таблицю нових адрес, що відноситься до групи, і групи джерел, що містить IP-адреси підключеного одноранговогo вузла або DNS-сідера, від якого вузол дізнався адресу *a*. Блок вибирається в такий спосіб:

```
    i = Hash( SK, Src_Group, Group ) % 32
    Bucket = Hash( SK, Src_Group, i) % 256
    return Bucket,
```

де SK – випадкове число, вибране з появою вузла;

Group = / 16, що містить IP для вставки;

Src_Group = / 16, що містить IP передавального одноранговогo вузла.

Кожна пара (група, група джерел) гешується в один блок таблиці нових адрес. Кожен блок містить унікальні адреси. Якщо блок сповнений, то над усіма 64 адресами в блоці виконується механізм Bitcoin витискання. Якщо який-небудь з адрес має термін більше 30 днів або має занадто багато невдалих спроб підключення, то така адреса витискується на користь нової адреси. В іншому випадку Bitcoin витискання використовується з невеликою зміною – адреса, що витискається, відкидається. Окрему адресу можна відображати в кілька блоків, якщо вона оголошується декількома одноранговими вузлами.

2. БОТНЕТ-АТАКА

Визначення 3. Ботнет – взаємопов'язана мережа комп'ютерів, заражених шкідливою програмою без відома користувача і контрольованих зловмисником. Як засіб передачі керуючих команд комп'ютерам-учасникам ботнету використовується протокол прикладного рівня IRC.

Bitcoin гарантує, що вузол не зберігає занадто багато IP-адрес з однієї і тієї ж групи (тобто /16 адрес IPv4 в блоці адрес) [6].

Ботнет, атакуючий, утримує *t* адрес в різних групах. У роботі кожна адреса береться як гешування у рівномірно випадковий блок у таблиці перевірених адрес, тому кількість адрес, що геруються, для кожного блоку біноміально розподіляється як $B\left(t,\frac{1}{64}\right)$, де $B(n,p)$ – біноміальний розподіл, що підраховує успіхи в послідовності з n незалежних так/ні спроб, кожна з яких приносить "так" з ймовірністю p.

Сценарії, які відображають скільки з 64 × 64 записів в таблиці перевірених адрес може бути зайнято зловмисником, наведені на рис. 1.

1) Від початку порожній. У кращому випадку для атакуючого всі 64 блоки з самого початку порожні. Очікувана кількість злочинних адрес, збережених у таблиці перевірених адрес, дорівнює:

$$64E\left[\min\left(64, B\left(t,\frac{1}{64}\right)\right)\right]. \qquad (1)$$

2) Bitcoin витискання. Розглянемо найгірший випадок для зловмисника, коли кожен блок *i* заповнений 64 законними адресами. Ці адреси матимуть більш старі мітки часу, ніж всі адреси зловмисника A_i, які він намагається вставити в блок *i*. Слід зазначити, що адреси вузлів, які активно підключені до атакованого вузла, не обов'язково мають більш старі мітки часу.

Механізм Bitcoin витискання вимагає, щоб для кожної адреси, яка знову додається, вибиралися чоти-

ри випадкових адреси, що зберігаються в блоці, і витискувалась адреса з найстаршою міткою часу. Якщо одна з чотирьох вибраних адрес є легітимною (і вона буде старше всіх адрес зловмисника), тоді легітимна адреса буде перезаписана адресою атакуючого.

Нехай Y_a для $a = 0$ .... A_i – кількість злочинних адрес, які зберігаються в блоці i, враховуючи, що зловмисник вставив a унікальних адрес у блок i. Нехай $X_a = 1$, якщо a-й вставлена адреса успішно перезаписала легітимну адресу та $X_a = 0$ в іншому випадку. Тоді

$$E\left[X_a \mid Y_{a-1}\right] = 1 - \left(\frac{Y_{a-1}}{64}\right)^4$$

З цього маємо:

$$E\left[X_a \mid Y_{a-1}\right] = Y_{a-1} + 1 - \left(\frac{Y_{a-1}}{64}\right)^4 \quad (2)$$

$$E\left[Y_1\right] = 1 \quad (3)$$

де вираз (3) випливає з того, що блок від початку заповнений легітимними адресами. Тепер маємо рекурентне співвідношення для $E\left[Y_a\right]$, яке можна вирішити чисельно. Можна знайти, що $E\left[Y_a\right] > 63$ при $a \geq 101$. Таким чином, зловмисник може перезаписувати 63 з 64 легітимних адрес в блоці після вставки 101 унікальної адреси. Таким чином, очікувана кількість злочинних адрес у всіх блоках обчислюється таким чином:

$$64 \sum_{a=1}^{t} E\left[Y_a\right] \Pr\left[B\left(t, \frac{1}{64}\right) = a\right]. \quad (4)$$

3) Випадкове витискання. Розглядається найгірший випадок для атакуючого, коли кожен блок заповнений легітимними адресами, але тепер передбачається, що кожна вставлена адреса витискає випадково вибрану адресу. Слід зазначити, що механізм, закладений у Bitcoin, працює за іншим принципом, а даний варіант аналізується для порівняння.

Лема. Якщо k елементів випадковим чином та незалежно один від одного вставлені в n блоків таблиці, а X є випадковою змінною, що підраховує кількість непустих блоків, то маємо:

$$E\left[X\right] = n\left(1 - \left(\frac{n-1}{n}\right)^k\right) \approx n\left(1 - e^{-\frac{k}{n}}\right) \quad (5)$$

Таким чином, вираз (5) випливає з $\left(n-1\right)/n \approx e^{-1/n}$ для $n \gg 1$.

Застосовуючи лему, знаходимо очікувану кількість злочинних адрес в усіх блоках:

$$4096\left(1 - \left(\frac{4095}{4096}\right)^t\right), \quad (6)$$

де t – кількість адрес, що утримує атакуючий в різних групах;

4096 – це кількість всіх записів у таблиці перевірених адрес 64 блоки × 64 адреси в кожному.

4) Використання декількох раундів. Атака на таблицю маршрутизації проходить в кілька раундів; в кожному раунді атакуючий неодноразово вставляє кожну зі своїх t адрес в таблицю перевірених адрес. Водночас як кожна адреса завжди відображається в тому ж блоці в таблиці перевірених адрес у кожному раунді, Bitcoin витискання відображає кожну адресу в інший слот у цьому ж блоці в кожному раунді. Таким чином, зловмисна адреса, яка не зберігається в таблиці перевірених адрес в кінці одного раунду, все ще може бути успішно збережена в цьому блоці в наступному раунді.

До сих пір розглядався тільки один раунд. Однак проведення атаки в кілька раундів дозволить зберегти в таблицю перевірених адрес більшу кількість адрес. Після достатньої кількості раундів, очікуване число адрес задається виразом (1), тобто атака виконується як при кращому випадку для атакуючого.

Аналіз ресурсів для запуску ботнет-атаки. Ботнет-атака, проведена в кілька раундів, може дозволити зловмиснику повністю заповнити таблицю перевірених адрес, що складається з близько 6000 адрес («від початку порожня» лінія на рис. 1).

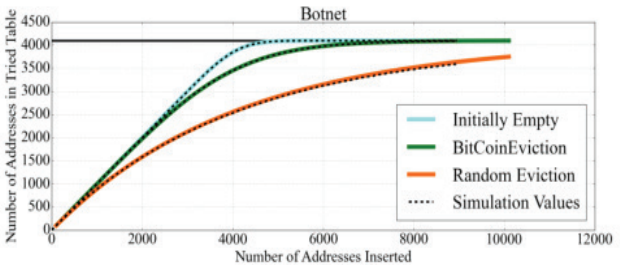

Рис. 1. Графік відношення очікуваної кількості адрес, збережених в таблиці перевірених адрес, для різних сценаріїв до кількості адрес (ботів) t

В ході побудови графіка значення обчислювалися з виразів (1), (4), (6). З графіка видно, що з кількістю ботів до 1000, ступінь заповнення таблиці перевірених адрес однакова для всіх трьох сценаріїв. Далі у випадку сценарію, при якому таблиця перевірених адрес від початку не заповнена, в ході використання менше 5000 ботів зловмисник повністю заповнює таблицю. Очевидно, що це найбільш сприятливий сценарій для зловмисника. При сценарії Bitcoin витискання повне заповнення таблиці перевірених адрес досягається зловмисником в ході використання близько 7000 ботів, а застосування випадкового витискання не дозволяє повністю заповнити таблиці в ході використання 10000 ботів. Таким чином, використання випадкового витискання підвищує кількість ресурсів, необхідних для перезапису таблиці перевірених адрес зловмисними адресами. Це підвищує складність реалізації атаки в цілому.

Така атака не може бути з легкістю запущена з законного хмарного сервісу. Як правило, такі сервіси виділяють <20 адрес на орендаря [7, 8, 9]. Проте,

Bitcoin був атакований ботнетами такого і навіть більшого розмірів [10, 11, 12]. Наприклад, ботнет Miner володів 29000 хостами з зовнішніми IP-адресами [13]. Водночас як деякі ботнет-напади концентруються в кількох діапазонах IP-адрес [14], важливо пам'ятати, що ботнет-атака, описувана в цій роботі, вимагає не більше ≈ 6000 груп. Багато інших ботнетів вимагають набагато більше ресурсів [5]. Наприклад, ботнет Walowdac був в основному в діапазонах 58.x-100.x і 188.x-233.x [14], що створює $42 \times 2^8 + 55 \times 2^8 = 24832$ груп.

Перезапис таблиці нових адрес. Для повного розуміння дій зловмисника слід розглянути як він повинен відправляти ADDR повідомлення, які перезапишуть таблицю нових адрес "сміттям" із IP-адрес. Нехай використовуватимуть "сміття" з нерозподіленого блоку адрес класу A IPv4 252.0.0.0/8, призначеного IANA як "зарезервовані для майбутнього використання" [15]; будь-які підключення до цих адрес терпітимуть невдачу, змушуючи атакований вузол вибирати адресу з таблиці перевірених адрес. Слід нагадати, що пара (група, група джерел) визначає блок, у якому зберігається адреса в ADDR повідомленні. Таким чином, якщо атакуючий контролює вузли в s різних групах, то s – це кількість груп джерел. Передбачається, що вузли в кожній групі джерел можуть висувати ADDR повідомлення, що містять адреси з g різних груп. "Сміття", тобто 252.0.0.0/8 блок адрес, задає верхню межу g на рівні $2^8 = 256$. Кожна група містить a різних адрес.

Оцінка ботнет-атаки. У ботнет-атаці кожен з вузлів атакуючого t знаходиться в окремій групі джерел. Для $s = t > 200$, що має місце для всіх атак ботнетів, застосовуючи лему, отримаємо, що очікувана кількість заповнених блоків таблиці за допомогою s груп джерел становить:

$$E[N] = 256\left(2 - \left(\frac{255}{256}\right)^{32s}\right). \qquad (7)$$

Вираз (7) показує, що кількість груп джерел $s = t$ необмежена. Таким чином, потрібно, щоб кожен одноранговий вузол відправляв одне ADDR повідомлення, що містить 1000 адрес з 250 окремими групами з чотирьох адрес кожна. Оскільки $s = t$, таке велике, можна змоделювати це, припускаючи, що кожна пара (група, група джерел) вибирає блок у таблиці нових адрес рівномірно випадковим чином і вставляє 4 адреси в той блок. Таким чином, очікувана кількість адрес, введених у блок, приблизно складатиме:

$$4 \times E\left[B\left(250t, \frac{1}{256}\right)\right] = 3.9t.$$

При $t > 200$ очікується, принаймні, 780 адрес, вставлених у кожен блок. З виразів (2) і (3), отримуємо $E[Y_{780}] \approx 64$, таким чином, кожен блок таблиці нових адрес, ймовірно, буде повним.

3. МЕТОД ПРОТИДІЇ НА ОСНОВІ АРХІТЕКТУР БОТНЕТІВ

Узагальнюючи вище сказане, атакуючий з достатньою кількістю IP-адрес і часу може успішно провести атаку на таблиці маршрутизації будь-якого цільового вузла, незалежно від стану таблиць перевірених і нових адрес вузла, що атакується. В даному розділі представлено метод протидії, який робить атаки на таблиці маршрутизації більш складними в реалізації. Даний метод протидії заснований на архітектурах ботнетів, і розроблений так, щоб не конфліктувати з архітектурою мережі Bitcoin і не порушувати принципи відкритості та децентралізації.

Метод включає в себе наступну послідовність дій. Дії 1 – 4 рекомендується застосовувати в зв'язці, дія 5 є самостійною, дії 6 – 10 є опціональними.

Перші п'ять дій гарантують, що:

- якщо атакований вузол має h легітимних адрес у таблиці перевірених адрес до початку атаки, та p-частка з них приймає вхідні з'єднання під час атаки, коли атакований вузол перезавантажується, то навіть зловмисник з необмеженою кількістю адрес не зможе успішно провести атаку на таблиці маршрутизації по відношенню до такого вузла з ймовірністю, що перевищує:

$$\Pr = f^8 < \left(1 - \frac{p \times h}{64 \times 64}\right)^8; \qquad (8)$$

- якщо вихідне з'єднання атакованого вузла з найстаршою міткою часу спрямоване до легітимного одноранговго вузла до початку проведення атаки, то вона завершується невдачею, якщо той одноранговий вузол приймає вхідні з'єднання, коли атакований вузол перезавантажується.

1) Детерміноване випадкове витискання. Замінити Bitcoin витискання таким чином: аналогічно тому, як кожна адреса детерміновано гешується в окремий блок у таблицях перевірених і нових адрес, додатково детерміновано гешувати адресу в окремий слот в цьому блоці. Таким чином, зловмисник не зможе збільшити кількість адрес, що зберігаються, шляхом повторної вставки однієї і тієї самої адреси в декількох раундах (п. 4, розд. 2). Завдяки цьому зменшується кількість адрес для атаки, що зберігаються в таблиці перевірених адрес.

2) Випадковий вибір адрес. Розглянутий клас атак також використовує значний ухил у бік формування вихідних з'єднань до адрес зі свіжими мітками часу. Таким чином зловмисник, якому належить лише невелика частка адрес f = 30% в таблиці перевірених адрес атакованого вузла, може збільшити ймовірність успіху за рахунок збільшення τ_ℓ (часу, інвестованого в атаку). Така перевага

атакуючого може бути усунена, якщо адреси вибиратимуться випадковим чином з таблиць перевірених і нових адрес. Таким чином, показник успіху 50% завжди вимагає, щоб зловмисник заповнював $\sqrt[8]{0.5} = 91.7\%$ таблиці перевірених адрес, що вимагає близько 3680 тимчасових вузлів у ботнет-атаці. Поєднуючи це з детермінованим випадковим витісненням, цифра підвищиться до 10194 ботів для ймовірності успіху 50%.

3) Тестування адреси перед витисканням. Перед збереженням адреси в його детерміновано підібраному слоті в блоці в таблиці перевірених адрес, спочатку пропонується перевіряти наявність адреси з більш старою міткою часу, що зберігається в цьому ж слоті. Якщо така адреса є, то короткочасно спробувати підключитися до неї, і якщо з'єднання виявилось успішним, то не витискати стару адресу з таблиці перевірених адрес, і зберігати нову адресу в таблицю перевірених адрес тільки у разі збою з'єднання.

Припустимо, що існує h легітимних адрес у таблиці перевірених адрес до початку проведення атаки. Також передбачається, що кожна з h легітимних адрес у таблиці перевірених адрес діюча, тобто приймає вхідні з'єднання незалежно один від одного з ймовірністю p. За допомогою тестування перед витисканням зловмисник не зможе витиснути p x h легітимних адрес (в очікуванні) з таблиці перевірених адрес, незалежно від кількості різних адрес, які він контролює. Таким чином, навіть якщо решта таблиці перевірених адрес заповнена зловмисними адресами, ймовірність успішного проведення атак на таблиці маршрутизації по відношенню до цільового вузла обмежена виразом (8).

Це значно відрізняється від функціонуючого на даний момент протоколу, в якому зловмисники з достатньою кількістю адрес мають необмежену можливість успіху, навіть якщо таблиця перевірених адрес заповнена легітимними адресами.

4) Використання короткочасних вихідних тестових підключень. Дана дія передбачає додавання вихідного з'єднання, яке б встановлювало короткочасне тестове з'єднання з випадково обраними адресами в таблиці нових адрес. Якщо з'єднання є успішним, то адреса витискається з таблиці нових адрес і вставляється в таблицю перевірених, а в іншому випадку, адреса витискається з таблиці нових адрес.

Подібні з'єднання допоможуть вичистити сміття з таблиці нових адрес при збільшенні кількості адрес з новими мітками часу в таблиці перевірених адрес, які найімовірніше будуть онлайн, коли вузол перезавантажиться.

5) Резервні з'єднання при перезавантаженні атакованого вузла. Дана дія є самостійною по відношенню до розглянутих вище. Ґрунтуючись на результатах роботи [16], додаються два з'єднання, які зберігаються між перезавантаженнями. Таким чином, додається резервна таблиця, що записує адреси поточних вихідних з'єднань і час першого підключення до кожної адреси. Після перезавантаження, вузол присвячує два додаткових вихідних з'єднання найстарішим резервним адресам, які приймають вхідні з'єднання. Тепер, на додаток до подолання інших дій методу, успішний зловмисник повинен також розірвати резервні з'єднання. Атаки на таблиці маршрутизації зазнають невдачі, якщо атакований вузол підключається до резервної адреси, яка не контролюється атакуючим.

Крім цих п'яти дій, можна використовувати додаткові механізми збільшення складності реалізації атак на таблиці маршрутизації.

6) Збільшення кількості блоків у таблицях адрес. Одним з найбільш очевидних способів збільшення складності реалізації атак на таблиці маршрутизації є збільшення розміру таблиць перевірених і нових адрес. Припустимо, що кількість блоків у таблиці перевірених адрес подвоєно. Отже, застосовуючи лему, можемо обчислити очікувану кількість заповнених блоків під час використання заданої кількості груп в атаці:

$$E[\Gamma] = 64\left(1-\left(\frac{63}{64}\right)^{4s}\right) \approx \left(1-e^{-\frac{4s}{64}}\right), \qquad (9)$$

де s – кількість груп, які використовуються зловмисником в атаці;

Γ – кількість непустих блоків у таблиці перевірених адрес, $\Gamma \in [0;64]$.

Якщо розглядати атаки інфраструктури і ботнету, то з виразу (9) випливає, що блоки, заповнені s групами змінюються від $\left(1-e^{-\frac{4s}{64}}\right)$ до $\left(1-e^{-\frac{4s}{128}}\right)$. Значення ступенів заповнення таблиці перевірених адрес для різної кількості непустих блоків і кількості груп, які використовуються зловмисником в атаці, наведені в табл. 1.

Таблиця 1

Ступінь заповнення таблиці перевірених адрес зловмисними адресами залежно від кількості груп в атаці і кількості непустих блоків

Γ / S	64	128	256
1	0,061	0,031	0,016
5	0,268	0,145	0,075
10	0,465	0,268	0,145
20	0,713	0,465	0,268
50	0,956	0,790	0,542
100	0,998	0,956	0,790
500	1	1	0,999
1000	1	1	1

Таким чином, зловмиснику для атаки інфраструктури необхідно подвоїти кількість груп для того, щоб очікувано заповнити ту ж частину таблиці перевірених адрес. Аналогічно і для ботнет-атаки необхідно подвоїти кількість ботів. Важливо відзначити, однак, що цей захід протидії корисний тільки тоді, коли таблиця перевірених адрес вже містить достатню кількість легітимних адрес, щоб зловмисник мав меншу частину адрес в цій таблиці. Однак, якщо таблиця перевірених адрес у цілому порожня (або містить в більшості своїй застарілі адреси для вузлів, які більше не є діючими), то зловмисник як і раніше матиме більшу частину адрес в таблиці перевірених адрес, навіть якщо кількість блоків у таблиці перевірених адрес збільшилася. Таким чином, ця дія має супроводжуватися іншою дією (наприклад, використанням короткочасних вихідних тестових підключень), що збільшує кількість легітимних адрес, що зберігаються в таблиці перевірених адрес.

7) Збільшення кількості вихідних з'єднань. 80% одноранговых вузлів Bitcoin дозволяють як мінімум 40 вхідних з'єднань [17]. Таким чином, можна зробити запит вузлів, щоб зробити кілька додаткових вихідних з'єднань не ризикуючи, що мережа витратить ємність з'єднання. Деякі вузли (наприклад, шлюзи об'єднань майнінгу), як показують виміри, вже це роблять [18]. Наприклад, використовуючи дванадцять вихідних з'єднань замість восьми (на додаток до короткочасних вихідних і двом резервним з'єднанням), можна зменшити ймовірність успішного проведення атаки з f^8 до f^{12}. Таким чином, для досягнення ймовірності успіху в 50% атакуючому для атаки інфраструктури тепер потрібно 46 груп, а для ботнет-атаки – 11796 ботів. Хоча це поліпшення не таке значне як дії 1 – 5, це простий спосіб підвищити складність реалізації атак на таблиці маршрутизації.

8) Заборона незапрошених ADDR повідомлень. Вузол може не приймати великі незапрошені ADDR повідомлення (з кількістю адрес більше заданої) від вхідних одноранговых вузлів, і запитувати ADDR повідомлення від вихідних з'єднань тільки коли його таблиця нових адрес містить занадто мало адрес. Це запобігає флудингу таблиці нових адрес атакованого вузла сміттям шляхом злочинних вхідних з'єднань. Ця зміна не є шкідливою, оскільки навіть в поточній мережі достатньо адрес в таблиці нових адрес [18]. Слід звернути увагу на те, що вузол запитує ADDR повідомлення при встановленні вихідного з'єднання. Одноранговий вузол відповідає n випадково вибраними адресами зі своїх таблиць перевірених і нових адрес, де $n \in [x; 2500]$, а x – це 23% адрес, що зберігаються одноранговим вузлом. Якщо кожен одноранговий вузол посилає порядку n = 1700 адрес, то таблиця нових адрес вже на $8n/16384 = 83\%$ заповнена до моменту, коли Bitcoin вузол завершує установку вихідних з'єднань.

9) Підвищення різноманітності вхідних з'єднань. На даний момент вузол Bitcoin може мати всі вхідні з'єднання від однієї IP-адреси, що робить занадто легким процес монополізації вхідних з'єднань атакованого вузла під час атаки затемнення або атак нестачі сполук [19]. Дана дія передбачає обмеження прийняття з'єднань від однієї IP-адреси.

10) Моніторинг та виявлення аномалій. Розглянутий в роботі клас атак на таблиці маршрутизації має кілька специфічних "відбитків", які роблять його помітним. Такі "відбитки" включають в себе:

- сплеск короткочасних вхідних TCP-з'єднань від різних IP-адрес;
- великі ADDR повідомлення, що посилаються по таким з'єднанням;
- вміст у ADDR повідомленнях IP-адрес, що є сміттям.

Зловмисник, який раптово підключає велику кількість вузлів до мережі Bitcoin, також може бути виявлений. Таким чином, системи моніторингу та виявлення аномалій, що вишукують таку поведінку, також можуть бути корисні. Такі системи змушуватимуть зловмисника проводити атаки на низькій швидкості або витрачати ресурси на перезапис таблиці нових адрес (замість того, щоб використовувати "сміття").

ЗАКЛЮЧЕННЯ

У роботі проведений аналіз і описаний механізм зберігання мережної інформації одноранговоі піринговоі мережі Bitcoin. У ході проведеного аналізу встановлено, що найбільш уразливим місцем у механізмі адресації є зберігання мережної інформації, а саме таблиці перевірених адрес і механізм Bitcoin витискання. Таблиця перевірених адрес складається з 64 блоків, кожен з яких може зберігати до 64 унікальних адрес для одноранговых вузлів, з якими вузол успішно встановив вхідне чи вихідне з'єднання. Поряд з кожною збереженою адресою одноранговоі вузла, вузол зберігає мітку часу останнього успішного підключення до цього одноранговоі вузлу.

Коли вузол успішно підключається до одноранговоі вузлу, адреса одноранговоі вузла вставляється у відповідний блок таблиці перевірених адрес. Якщо блок заповнений (тобто містить 64 адреси), то використовується Bitcoin витискання. Дане витискання дозволяє зловмисникові помістити свої адреси в таблицю перевірених адрес атакованого вузла, що, в свою чергу, дозволяє йому отримати контроль над усіма з'єднаннями атакованого вузла і здійснювати широкий спектр атак на протокол криптовалюти.

Поданий у роботі метод протидії атакам на таблиці маршрутизації містить у собі ряд дій, спрямованих на підвищення складності реалізації атаки для зловмисника. Цей метод базується на архітектурах

ботнетів. У разі сценарію, при якому таблиця перевірених адрес від початку не заповнена, в ході використання менше 5000 ботів зловмисник повністю заповнює таблицю. При сценарії Bitcoin витискання повне заповнення таблиці перевірених адрес досягається зловмисником в ході використання близько 7000 ботів. Показник успіху 50% завжди вимагає, щоб зловмисник заповнював таблиці перевірених адрес, що вимагає близько 3680 тимчасових вузлів у ботнет-атаці. Поєднуючи це з детермінованим випадковим витісненням, цифра підвищиться до 10194 ботів для ймовірності успіху 50%. Отримуємо підвищення складності більше, ніж на 40%. Метод, наведений у роботі, передбачає опціональне використання окремих дій. Наприклад, шоста дія змушує зловмисника для атаки інфраструктури подвоювати кількість груп для того, щоб очікувано заповнити ту ж частину таблиці перевірених адрес. Аналогічно і для атаки ботнету необхідно подвоїти кількість ботів.

Таким чином, під час розробки децентралізованих систем на базі децентралізованої платформи, яка є аналогічною платформі Bitcoin, слід звертати увагу на перший рівень архітектури – на однорангову мережу. Вразливості першого рівня архітектури збільшують кількість вразливостей і простоту їх реалізації на останньому рівні. Стосовно атак на таблиці маршрутизації слід зазначити, що основним вразливим місцем є механізм зберігання і управління мережевою інформацією.

Література:

[1] *Anceaume E., Busnel Y., Gambs S.* On the power of the adversary to solve the node sampling problem. In Transactions on Large Scale Data and Knowledge Centered Systems XI. Springer, 2013, pp. 102-126.

[2] *Bakker A., Van Steen M. Puppetcast*: A secure peer sampling protocol. In European Conference on Computer Network Defense (EC2ND) (2008), IEEE, pp. 3-10.

[3] *Bortnikov E., Gurevich M., Keidar I.* Brahms: Byzantine resilient random membership sampling. Computer Networks 53, 13 (2009), pp. 2340-2359.

[4] *Jesi G. P., Montresor A., Van Steen M.* Secure peer sampling. Computer Networks 54, 12 (2010), pp. 2086-2098.

[5] *Rossow C., Andriesse D., Werner T.* Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In IEEE Symposium on Security and Privacy (2013), IEEE, pp. 97-111.

[6] *Nakamoto S.*: Bitcoin: A peer-to-peer electronic cash system // *https://bitcoin.org/bitcoin.pdf*. – 2008. – 9 p. – 09.09.2016.

[7] Amazon web services elastic ip. http://aws.amazon.com/ec2/faqs/#elastic-ip. Accessed: 2016-06-18.

[8] Microsoft azure ip address pricing http://azure.microsoft.com/en-us/pricing/details/ip-addresses/. Accessed: 2016-11-18.

[9] Rackspace: Requesting additional ipv4 addresses for cloud servers. http://www.rackspace.com/knowledge_center /article/requesting-additional-ipv4-addresses-for-cloud-servers. Accessed: 2016-11-18.

[10] *Johnson B., Laszka, A., Grossklags, J.* Game-theoretic analysis of ddos attacks against Bitcoin mining pools. In Financial Cryptography and Data Security. Springer, 2014, pp. 72-86.

[11] *King L.* Bitcoin hit by 'massive' ddos attack as tensions rise. Forbes http://www.forbes.com/sites/leoking/2014/02/12/bitcoin-hit-by-massive-ddos-attack-as-tensions-rise/. (December 2 2015).

[12] *Vasek M., Thornton M., Moore T.* Empirical analysis of denial-of-service attacks in the Bitcoin ecosystem. In Financial Cryptography and Data Security. Springer, 2014, P. 57 – 71.

[13] *Plohmann D., Gerhards-Padilla E.* Case study of the miner botnet. In Cyber Conflict (CYCON), 2012 4th International Conference (2012), IEEE, pp. 1-16.

[14] *STOCK, B., GOBEL, J., ENGELBERTH, M., FREILING, F. C., AND HOLZ, T.* Walowdac: Analysis of a peer-to-peer botnet. In European Conference on Computer Network Defense (EC2ND) (2009), IEEE, P. 13 – 20.

[15] IANA. Iana ipv4 address space registry. http: //www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml, January 2015.

[16] *Dingledine R., Hopper N., Kadianakis G.* One fast guard for life (or 9 months). In 7th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2014) (2014).

[17] *Biryukov A., Pustogarov I.* Bitcoin over Tor isn't a good idea. arXiv preprint arXiv:1410.6079 (2014).

[18] *Miller A., Litton J., Pachulski A.* Discovering Bitcoin's network topology and influential nodes. Tech. rep., University of Maryland, 2015.

[19] *Dillon J.* Bitcoin-development mailinglist: Protecting bitcoin against network-wide dos attack. http://sourceforge.net/ bitcoin/mailman/message/31168096, 2013. Accessed: 2016-10-11.

Стеценко Павло Ігорович, аспірант кафедри БІТ ХНУРЕ. Область наукових інтересів: захист інформації у децентралізованих системах.

Халімов Геннадій Зайдулович, доктор технічних наук, професор кафедри БІТ ХНУРЕ. Область наукових інтересів: криптографія на групах.

УДК 004.056.5

Метод противодействия атакам на таблицы маршрутизации на основе архитектур ботнетов для одноранговой пиринговой сети Bitcoin / П.И. Стеценко, Г.З. Халимов // Прикладная радиоэлектроника науч.-техн. журнал. – 2016. – Том 15, №. 3. – С. 232 – 239.

Представлен метод противодействия атакам на таблицы маршрутизации для одноранговой пиринговой сети Bitcoin. Метод основан на архитектурах ботнетов. Класс атак на таблицы маршрутизации включает в себя атаки затмения, ботнет-атаки и атаки инфраструктуры. Метод включает в себя последовательность из десяти действий. Действия 1 – 4 рекомендуется применять в связке, действие 5 является самостоятельным, действия 6 – 10 являются опциональными. Рассмотрены уязвимости механизма адресации пиринговой сети Bitcoin. Рассмотрены 4 сценария занятия злоумышленником таблицы проверенных адресов узла, рассчитаны сравнительные оценки ожидаемого количества перезаписанных адресов. Метод позволяет значительно повысить сложность реализации атак на таблицы маршрутизации, отдельные действия повышают необходимое количество ресурсов для атаки вдвое.

Ключевые слова: криптовалюта Bitcoin, одноранговая пиринговая сеть, таблица маршрутизации, ботнет-атака, одноранговый узел, таблица проверенных адресов, таблица новых адресов.

Табл.: 01. Ил.: 01. Библиогр.: 19 наим.

UDC 004.056.5

Method of countering attacks on routing tables based on the botnet architectures for Bitcoin peer-to-peer network / P.I. Stetsenko, G.Z. Khalimov // Applied Radio Electronics: Sci. Journ. – 2016. – Vol. 15, №.3 – P. 232 – 239.

A method of countering attacks on routing tables for the Bitcoin peer-to-peer network is presented. The method is based on the botnet architectures. A class of attacks on the routing tables includes eclipse attacks, botnet-attacks, and infrastructure attacks. The method includes a sequence of ten operations. Operations 1-4 are recommended to be used in conjunction, operation 5 is independent, and operations 6-10 are optional. The vulnerability of Bitcoin peer-to-peer network addressing mechanism has been considered. Four various scenarios of the attacker's occupation of a table with checked addresses have been considered, the comparative assessment of the expected number of the rewritten addresses has been calculated. The method can significantly increase the complexity of the implementation of the attacks on the routing tables, certain operations raise the required number of resources to an attack by a factor of 2.

Keywords: Bitcoin cryptocurrency, peer-to-peer network, routing table, botnet-attack, peer, table with checked addresses, table with new addresses.

Tab.: 01. Fig.: 01. Ref.: 19 items.