УДК 004.056 (045)

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МОНІТОРИНГУ ТА БЛОКУВАННЯ НЕСАНКЦІОНОВАНИХ ПРОЦЕСІВ В АВТОМАТИЗОВАНИХ СИСТЕМАХ

А. Б. Петренко, канд. техн. наун, доц.; ***А. В. Цьопич***

Національний авіаційний університет

notindenmark2012@ukr.net

Визначено, що великий спектр погроз в автоматизованих системах реалізується шляхом виконання тих чи інших програм. Описано процес блокування роботи таких програм. Для вирішення проблеми запропоновано розробити дискретне розмежування доступу за моделлю ХРУ. Розмежування доступу полягає в розробленні білих та чорних списків — набору дозволених та заборонених програм для окремого користувача комп'ютера. Описано механізм моніторингу та блокування несанкціонованих процесів в автоматизованих системах згідно з розробленими списками. Наведено варіант програмної реалізації такого механізму.

Ключові слова: програмне забезпечення, моніторинг, безпека інформації, несанкціоновані процеси, комп'ютерна мережа, активні процеси, політика безпеки.

The broad spectrum of threat in automatic system are realized because of running a lot of programs. In this work described the necessary of blocking such programs. For decision this problem created discrete delimetation access model according the HRU model. The delimetation of access based on white and black lists. This white and black lists contain the kit of resolved and prohibited programs for some user. Also described the way of monitoring and blocking unauthorized processes in automatic system according created lists*. In the end was offered a sample code to make such monitoring and blocking.*

Keywords: software, monitoring, information security, unauthorized process, network, active process, security policy.

Вступ

У багатьох працях було досліджено вразливості інформаційних систем та погрози, а також розроблено довідники типових погроз, вразливостей та ризиків. Багато погроз для інформаційних систем реалізується під час розроблення спеціалізованого програмного забезпечення, яке спрамовано на виконання дій, які прямо чи опосередковано призводять до витоку інформації.

У найпростішому випадку це можуть бути віруси, які потрапили в комп'ютер під час заходження в мережу Internet чи використання зйомних носіїв інформації, які сприяють витоку інформації або ж призводять до інших негативних наслідків. Наприклад, займають багато оперативної пам'яті або процесорного ресурсу. Існують списки подібних відомих додатків.

Так, їх можна виявити, аналізуючи процеси комп'ютера. Реалізація контролю виконуваних процесів в інформаційних системах може попередити великий спектр погроз, якому сприяють несанкціоновано запущені процеси.

Новина теми, що розглядається, полягає в такому. Існує багато програм, які певним чином дають змогу працювати з активними процесами на комп'ютері.

Одними з таких додатків є Radmin, Actual Spy, ProcessExplorer. Для прикладу наведемо деякі з можливостей цих додатків.

ProcessExplorer дозволяє працювати з активними процесами. Використовується кольорове підкреслення процесів. Додаток дає можливість отримати багато інформації за обраним процесом. Для кожного запущеного процесу можна визначити розмір використовуваної пам'яті, застосування ресурсів процесора у відсотковому відношенні, розмір вхідних та вихідних файлів для додатка та ін.

Додаток Actual Spy сприяє відслідковуванню роботи інших користувачів за комп'ютером. Програма відслідковує всі натиснуті клавіші, виконує зйомку екрану та відсилає дану інформацію на e-mail по локальній мережі чи на FTP.

Radmin — програма віддаленого користування. На віддаленому комп'ютері потрібно запустити Radmin. Після цього з'являється можливість роботи на віддаленому комп'ютері, використовуючи засоби Radmin на власному комп'ютері. Спектр дій на віддаленому комп'ютері визначається з поставлених цілей.

Існують розроблені додатки, які напрямлені на блокування активних процесів. Найпростішим прикладом є антивірусне програмне забезпечення. Антивіруси виявляють за різними «ознаками». Це можуть бути номери портів, що використовуються; надмірний та однотипний трафіки (за назвою програми). Назви процесів, які антивірус вважає за небажані, містяться в базах, які завантажуються з певних ресурсів, на

які користувач ніяким чином впливати не може. Отже, користувач не може формувати список процесів, які підлягають моніторингу. Функціонують зазначені додатки на локальному комп'ютері та інколи можуть виконувати моніторинг процесів в автоматизованих системах другого та третього класів (АС-2 та АС-3).

Постановка проблеми

Таким чином, постає необхідність у розробленні програмного забезпечення, яке виконує моніторинг процесів згідно з певним зазначеним адміністратором списку. Передбачається, що програмне забезпечення виконуватиме моніторинг в АС-2 та АС-3. Повинна бути передбачена можливість аналізу процесів, які запущені чи можуть бути запущені за так званим білим та чорним списками.

У чорному списку зазначається, які процеси повинні бути заблоковані, робота всіх інших дозволяється.

При використанні білого списку активними лишаються всі процеси, які занесені до нього, робота всіх інших процесів припиняється. Формування білого списку вимагає особливої уваги. До нього обов'язково потрібно додати всі системні процеси, без яких комп'ютер не буде адекватно працювати [2]. Також до білого списку включається все необхідне для роботи окремого користувача.

Програмне забезпечення повинно мати *серверну* та *клієнтську* частини.

Клієнтська частина виконує роль агента, який безпосередньо реалізує сканування активних процесів комп'ютера, на якому вони виконуються. При цьому використовуються білий чи чорний списки залежно від налаштувань агента.

Результати виконаної роботи періодично передаються до сервера. Звіт має містити назву закритого процесу, дату та час, коли це відбулося, тип списку, за яким виконувався моніторинг автоматизованої системи.

За допомогою сервера відбувається формування білих та чорних списків для кожного клієнта окремо.

Відправлення сформованих списків клієнту відбувається під час внесення змін до списку або за запитом клієнта. Також варто дозволити вносити зміни до білого та чорного списків самому корисутвачу, якщо у нього є такі права.

Для визначення прав користувача формується таблиця з іменем користувача та наданими йому правами. Наприклад, користувач під іменем адміністратора має право на внесення довільних змін до списків.

Визначення імені користувача можна реалізувати двома шляхами: ім'я, під яким відбувалося входження до системи або логін і пароль, введений при запуску самого клієнтського додатка.

Аналіз останніх досліджень і публікацій

Питанню погроз та вірусам в автоматизованих системах приділено достатньо багато уваги [3–5]. Тобто, наводяться описи типів вірусів та їх загальний механізм роботи. Однак рідко наводять способи боротьби з ними, окрім як використання антивірусного програмного забезпечення. Також розглядаються питання розмежування доступу та надання прав певним користувачам.

При вирішенні таких питань йдеться про необхідність визначення ПЗ, яке може чи не може використовувати певний користувач. Ряд подібних питань розглянуто в працях [1; 8]. Але комплексного дослідження, присвяченого даному питанню, в спеціалізованій науковій літературі [1–9] не виявлено. Також не описано, як контролювати роботу активних процесів в АС у цілому.

Таким чином, як видно з наведеного аналізу літератури [1–9], питанню блокування несанкціонованих процесів приділено достатньо багато уваги, однак ці питання розглядаються на рівні локального комп'ютера та оминають АС другого класу. Отже, вирішення зазначеної проблеми є важливим питанням, у рамках якого розроблення ПЗ моніторингу та фільтрації списку виконуваних процесів в АС-1 та АС-2 є актуальним.

Методика дослідження

Мета — організація моніториингу та фільтрації активних процесів на комп'ютері чи в локальній комп'ютерній мережі [5]. У контексті даної роботи під активними процесами розумітимемо запущені системні та прикладні програми.

Запущення програми характеризується створенням нового процесу [2]. Закриття такого процесу означає припинення роботи самої програми. В операційній системі (ОС) Windows список активних процесів можна переглянути, використовуючи спеціалізоване програмне забезпечення або ж у вікні диспетчера задач, який можна викликати натисненням клавіш «Ctrl-Shift-Esc» чи іншими способами.

Моніторинг активних процесів означає періодичне визначення списку активних процесів та зіставлення отриманого списку з чорним чи білим. Отримання списку активних процесів на комп'ютері не викликає ніяких труднощів, оскільки є масиви, елементи яких містять назви активних процесів та інформацію про кожен з них. Тип списку, за яким відбувається сканування, визначається в налаштуваннях додатка.

Фільтрація передбачає припинення роботи певних активних процесів, назв яких немає в чорному списку або які є в білому списку. Закриття процесу повинно супроводжуватися відповідним записом у файл-звіт.

Для вирішення поставленої проблеми потрібно розробити два незалежних додатки: серверний та клієнтський [3]. У процесі роботи ці додатки взаємодіють між собою. Запускається серверна частина програмного забезпечення (ПЗ). Після цього запускають клієнтські додатки, кожен з яких ініціює з'єднання з серверним додатком. Для кожного нового клієнта виділяється окремий канал, по якому буде виконуватися взаємодія клієнтського з серверним додатком. Зв'язок повинен бути дуплексний. Тобто, після встановлення з'єднання в довільний момент часу і клієнт, і сервер можуть відправити один одному певний запит або одержати відповідь. Вибір мови програмування, на якій буде розроблено таке ПЗ, залежить від уподобань розробника. Однак варто зазначити, що багато проблемних питань виникає в разі вибору мови програмування C#. Для цієї мови роз-роблені набори бібліотек, деякі з яких направлені на роботу саме з процесами коп'ютера. Доволі просто виконати описані дії, використовуючи сокети для встановлення з'єднання. Як параметри використовується IP-адреса та номер порту.

Під час написання ПЗ моніторингу та блокування несанкціонованих процесів ключовим моментом є саме функція моніторингу та блокування. Функцію моніторингу та блокування представимо у вигляді алгоритму дій (рис. 1), до якого наводиться відповідний опис.

Рис. 1. Алгоритм роботи функції моніторингу та фільтрації активних процесів

Схематично зображений алгоритм виконує разовий моніторинг та фільтрацію активних процесів. Отже, потрібно ініціювати його періодичне повторнення. Програмно це можна реалізувати використатнням системного таймеру [3]. Розроблений відповідно до алгоритму програмний код повинен бути розташований в окремій функції, яка й виконується щоразу при спрацюванні таймера. Опис призначення кожного з блоків схеми наведено нижче.

Блок «Початок» є умовним та позначає момент часу, коли починає виконуватися алгоритм. Щоразу виконуючи цей алгоритм, відбувається формування списку активних процесів у поточний момент. Далі під час виконання функції використовуватимуться як імена активних процесів елементи сформованого списку. Відбувається перехід до блоку 2, аналізується, за яким списоком відбувається сканування — чорним або білим.

Зчитуються всі записи обраного списку. Блок «3» передбачає розшифрування елементів чорно-білого списку. Блок 4 направлений на формування потоку для запису, в який будуть вноситися записи про виявлений невідповідний процес, момент його закриття та вид сканування, при якому його було виявлено. Блок 5 сигналізує про запуск циклу з числом ітерацій, що дорівнює кількості активних процесів. Тобто, кожен елемент списку активних процесів буде почергово перевірено на відповідність у циклі 6.

Блок 6 — цикл з числом ітерацій, що дорівнює кількості елементів чорно-білого списку. В даному циклі буде почергово використовуватися кожен елемент чорно-білого списку. На сьомому етапі порівнюється обраний процес з обраним елементом чорно-білого списку. Якщо відбувається сканування за білим списком, то активний процес буде заблоковано в разі збігу елементів обох списків; при скануванні за чорним списком активний процес блокується, якщо збігу виявлено не було.

Припинення роботи процесу та запис в журнал виконується в блоці 8. Отже, для кожного елемента списку активних процесів виконується кількість порівнянь з елементом чорно-білого списку, яке дорівнює кількості елементів чорно-білого списку.

На рис. 2 наведено алгоритм, який треба реалізувати на обраній мові програмування. Наведемо приклад реалізації запропонованого алгоритму, використовуючи мову програмуванням C#.

```
1  private void timer1_Tick(object sender, EventArgs e)
2  {
3      Process[] stop_proc = Process.GetProcesses();
4      count_of_current_Procceses = stop_proc.Length;
5      StreamReader read_list = new StreamReader(path_ToWhiteList);
6      int count = Convert.ToInt16(read_list.ReadLine());
7      string[] list_Black = new string[count];
0      for (int i = 0; i < count; i++)
9      {
10         list_Black[i] = DECRYPT_text_bef_wr(read_list.ReadLine().ToString(), count);
11         read_list.ReadLine();
12     }
13     read_list.Close();
14     StreamWriter report = new StreamWriter(path_ToReport, true);
15     string temp_procName;
16     for (int i = 0; i < count_of_current_Procceses; i++)
17         {
18             for (int j = 0; j < count; j++)
19             {
20                 temp_procName = list_Black[j];
21                 if (stop_proc[i].ProcessName == temp_procName)
22                 {
23                     stop_proc[i].Kill();
24                     report.WriteLine("WhiteList:: " + temp_procName + " was closed");
25                     count_of_current_Procceses = count_of_current_Procceses - 1;
26                 }
27             }
28         }
29         report.Close();
30         FillThe_Process();
31 }
```

Рис. 2. Програмний код функції моніторингу та фільтрації активних процесів на мові C#

Код розробленої функції моніторингу та блокування несанкціонованих процесів складається з 31 рядка коду. Наведена функція реалізовує моніторинг за білим списком. Під час роботи з чорним списком у коді змінилися б деякі непринципові моменти. Код доволі простий та невеликий. Це позитивно впливає на швидкість виконання запропонованої функції.

Опишемо призначення кожного рядка запропонованого коду, що дасть загальне розуміння роботи функції.

Рядок 1 являє собою оголошення функції. Дана функція є обробником події спливання часу на системному таймері. Тобто вона викликається щоразу, як встановлений інтервал часу системного таймеру спливає. Це виконується періодично, поки запущений таймер.

Починаючи виконання функції, передусім слід визначити список активних процесів комп'ютера. Він отримується шляхом використання методу GetProcesses() та записується до масиву stop_proc (рядок 3). Далі в змінну count_of_current_Processes записується кількість активних процесів.

На наступному кроці (рядок 5) створюється потік для читання, який як параметри обирає шлях до білого списку.

Код у рядках 6–7 є оголошенням масиву, який буде містити елементи білого списку. Код у рядках 8–13 забезпечує запис у вище створений масив усіх елементів білого списку в розшифрованому вигляді.

На наступному кроці створюється потік для запису (рядок 14). Цей потік буде виконувати запис до файлу-звіту назву заблокованого процесу, час блокування та тип списку, за яким відбувалося сканування.

Рядок 16 являє собою цикл з числом ітерацій, що дорівнює кількості процесів на момент початку виконання функції. В рядку 18 оголошується новий цикл з числом ітерацій, рівному кількості елементів у білому списку. Даний цикл виконується повністю при кожній новій ітерації циклу, оголошеного в рядку 16. Тобто, це є вкладений цикл.

Рядок 21 являє собою порівняння назви *i*-го активного процесу з *j*-им елементом білого списку. При використаній організації циклу та вкладеного циклу для кожного *i*-го активного процесу виконується порівняння з усіма елементами білого списку. Так, якщо при порівнянні назви активного процесу з елементом білого списку виявлено збіг, виконується код 22–26.

Наведена частина програми зумовлює припинення роботи *i*-го активного процесу. Рядок 23 являє собою безпосереднє припинення роботи *i*-го процесу. Після цього виконується відповідний запис у файл-звіт, який періодично відсилається на сервер (рядок 24). Після цього потрібно зменшити кількість активних процесів на один (рядок 25).

Отже, цикл, оголошений у рядку 16, виконуватиметься на одну ітерацію менше. Дії 18–27 виконуються *i*j* разів у максимальному випадку та забезпечують зіставлення кожному активному процесу кожного елемента білого списку. Рядок 29 направлений на закриття потоку для запису. Рядок 30 — оновлення списку активних процесів у вікні програми.

Основні результати

Розроблене ПЗ має високу швидкодію, яка визначається інтервалом спрацювання системного таймера. Інтервал можна задавати в налаштуваннях клієнтського програмного забезпечення. Під час кожного виклику таймера (спилювання зазначеного періоду часу) відбувається формування списку активних процесів та зіставлення кожного процесу білому чи чорному списку. У результаті виявлення відповідностей процес блокується та робиться відповідний запис у журнал звітності.

Після встановлення з'єднання з сервером клієнт у довільний момент часу може отримати новий білий чи чорний список, який заміняє існуючий.

Використання

Розроблене ПЗ може використовуватися в АС-1 та АС-2. На комп'ютері повинна бути ОС Windows з набором бібліотек не нижче NET.Framework 3.5. Саме ПЗ являє собою два окремих додатки — клієнтський та серверний.

Клієнтський запускається на кожному комп'ютері комп'ютерної мережі, де планується проводити моніторинг та блокування несанкціонованих процесів. На одному з комп'ютерів запускається серверний додаток. Доступ до цього комп'ютера має бути обмеженим, щоб обмежити коло осіб, які можуть працювати з серверним додатком розробленого ПЗ.

Клієнтські додатки можуть працювати автономно без підключення до сервера. У такому випадку у них мають бути попередньо сформовані чорний та білий списки.

У довільний момент часу можна виконати підключення до серверного додатку. Тоді буде можлим отримання нових білих та чорних списків, відправлення на сервер журналів звітності виконаної роботи. Також клієнт у довільний момент може відключитися від серверного додатка та продовжувати роботу автономно, якщо у нього є відповідні права.

Висновок

Розроблене ПЗ виконує моніторинг та фільтрацію списку виконуваних процесів в АС-1 та АС-2. ПЗ може функціонувати як на рівні окремого комп'ютера, так і в локальній комп'ютерній мережі.

Клієнтський додаток може функціонувати і автономно без підкдючення до сервера.

У такому випадку повинен бути попередньо отриманий від сервера чорний чи білий список або створений власноруч засобами того ж клієнтського додатка.

Це є зручним, оскільки в разі неможливості встановлення зв'язку з сервером, функція моніторингу та фільтрації лишається доступною.

Інтервал виявлення нових додатків зазначаєть при налаштуванні та може становити навіть 1 мс. Малоймовірно, що за такий час можуть бути виконані деякі небезпечні дії для комп'ютера, оскільки сам час запуску додатка є набагато більшим.

ЛІТЕРАТУРА

1. *Цирлов В. Л.* Основы информационной безопасности автоматизированных систем / В. Л. Цирлов // Издание «Феникс». — 2008. — С. 10–173.

2. *Шаханова М. В*. Апаратно-програмные средства и методы защиты инфорации: учеб. пособ./ М. В. Шаханова, С. К. Варлатая // — 2007. — С. 195–318

3. *Духан Е. И*. Применение программно-аппаратных средств защиты компьютерной информации / Е. И. Духан, Н. И. Синадский, Д. А. Хорьков // Федеральное агентство по образованию. — 2008. — С. 45–183

4. *Щеглов А. Ю*. Защита компьютерной информации от несанкционированного доступа / А. Ю. Щеглов // Наука и Техника. — 2004. — С. 208–384

5. *Цьопич А. В*. Моніторинг та формування списку виконуваних процесів в локальній комп'ютерній мережі / А. В. Цьопич, А. Б. Петренко // Мат. наук.-техн. конф. студентів та мол. учен. «Наукоємні технології». — 14–18 листопада 2011 р. — 2011. — С. 45.

Стаття надійшла до редакції 06.02.2012.