УДК 004.056

Г.Г. Киричек, канд. техн. наук, доц.,
В.Ю. Гаркуша, студент
Національний університет «Запорізька політехніка», м. Запоріжжя, Україна
kirichek@zntu.edu.ua

Процес виявлення зловживань і аномалій в мережі

В роботі запропоновано метод реалізації системи виявлення зловживань і аномалій з використанням тренування звичайних і атакуючих пакетів відповідно. Метод, який використовується при навчанні на застосування атаки є комбінацією непідконтрольної та контрольованої нейронної мережі для системи виявлення вторгнень. У непідконтрольній мережі атаки класифікуються за меншими категоріями з урахуванням їх особливостей та використанням самоорганізуючої карти Кохонена (SOM). Для керування кластерами застосовується нейронна мережа на основі Backpropagation.

Ключові слова: нейронна мережа, виявлення аномалії, вторгнення, навчання, SOM.

DOI: 10.31474/1996-1588-2019-1-28-42-46

Вступ

Ймовірність виникнення загроз в комп'ютерних мережах збільшується з кожним роком і є достатньо серйозним питанням при забезпеченні мережної безпеки. Сучасні фільтри мережевого трафіку, системи виявлення і заходи протидії стають дедалі менш ефективними при роботі з великими обсягами трафіку у високошвидкісних мережах, а також непридатними для розпізнавання нових типів і методів атак на комп'ютерні системи та мережі [1].

Система виявлення вторгнень (Intrusion Detection System (IDS)) контролює трафік мережі, яка досліджується, за підозрілою діяльністю, а також попереджає саму систему або системного адміністратора про можливі атаки. Основною метою IDS є захист доступності, конфіденційності та цілісності критичних мережних інформаційних систем. Зазвичай використовуються два основні підходи до системи виявлення вторгнень: виявлення зловживань та аномалій [2].

Виявлення невірного використання базується на описі відомих небезпечних дій [2]. Цей опис часто моделюється як набір правил, які іменуються підписами атаки. Ідентифікатор виявлення аномалій шукає загрози та застосовує правила або, визначені попередньо, поняття: нормальна та аномальна активність системи. В подальшому використовуємо це для виявлення відмінності загроз від звичайної поведінки системи, для проведення моніторингу звіту або для блокування загроз, коли вони виникають [3, 4].

Для виявлення аномалій є більш раціональною розробка інтерактивної IDS, ніж звичайні правила та програми, що працюють за звичайним принципом виявлення та протидії аномаліям в мережі. Тому є сенс в інтеграції класичних підходів IDS та підходів з аналізу даних використовуючи нейронні мережі. Це більш гнучким підходом при аналізі та класифікації даних [2].

Постановка задачі

Метою роботи є реалізація прототипу системи IDS на основі гібридної нейронної мережі для виявлення аномалій та загроз з боку мережі. Об'єктом дослідження є процес реалізації модулів виявлення загроз та аномалій. Предметом є моделі, методи та програмні засоби побудови прототипу системи.

Проведення досліджень дозволило визначити, що програмне забезпечення повинне виконувати аналіз та проводити розділення звичайних і небезпечних пакетів вхідних даних. Після виявлення небезпечних даних воно ще повинне виконувати і класифікацію типу загрози.

В якості основи для виконання поставлених завдань обрано підхід з використанням нейронних мереж (перцептронів), на основі самоорганізуючих мап Кохонена. Дані які використовувались в роботі для навчання перцетронів це набір даних (датасет) лабораторії Лінкольна Масачусетського технологічного університету. Даний набір створено для оцінювання систем виявлення вторгнень DARPA і вважається еталоном для проведення досліджень IDS [5, 6].

При проведенні експерименту використовуємо набір даних, який складається з пакетів п'яти класів: звичайні пакети, пакети для зондування та сканування інфраструктури, пакети, які викликали відмову в обслуговуванні обладнання, пакети які збільшили привілеї користувача до супер користувача та пакети зовнішніх загроз.

Моделювання системи

Для зручного масштабування та використання програмного забезпечення з незалежними модулями, застосуємо два перцептрони, спростив

не тільки здатність до масштабування, а й зменшив концентрацію відповідальності на кожному з модулів системи. Тобто навчати та налагоджувати два менших перцептрони значно легше ніж один великий. При розробці аналізатору мережних загроз використовуємо підхід гібридної мережі на основі нейронної мережі без вчителя (перший модуль) та нейронної мережі з вчителем (другий модуль). В якості мережі без вчителя застосуємо самоорганізуючі мапи Кохонена (SOM) [7], які адаптовані із використанням навчання без вчителя (без кінцевого результату). Метод зворотного поширення є методом навчання, який контролюється тренінгом штучних нейронних мереж [8]. Метою зворотного розповсюдження є підготовка мережі для досягнення балансу між здатністю правильно реагувати на вхідні моделі, які використовуються для навчання та здатністю давати правильні відповіді на вхід, як при навчанні (рис.1).

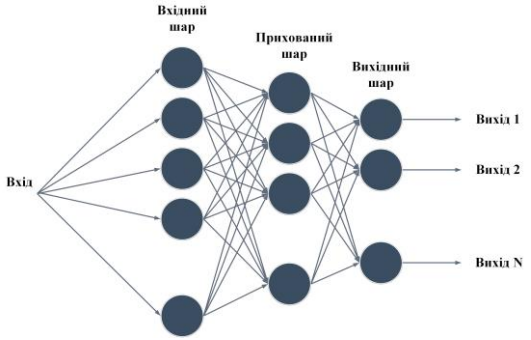

Рисунок 1 – Метод зворотного поширення

Процес навчання без вчителя в SOM включає три етапи: вагами підключення призначаються малі випадкові числа та здійснюється вибір параметру швидкості навчання; фіксується найкращий блок узгодження; ваги оновлюються, згідно з формулою 1.

$$\omega_{ij}(new) = \omega_{ij}(old) + \alpha\,[x_i - \omega_{ij}(old)], \qquad (1)$$

де: x_i – і-й вхідний параметр вектору, ω_{ij} – j-й стовпець вагової матриці, α – швидкість навчання (зменшується), навчання продовжується [9].

Оновлення ваги нейронів відбувається тільки для активних вихідних нейронів. Дозволяється навчати одиницю, ваговий вектор якої є найближчим до вхідного вектору [9]. Сам процес навчання продовжується до тих пір, поки не будуть оброблені всі вхідні вектори. Критерій конвергенції в нейронних мережах є епохою. Це одна ітерація в процесі навчання. Епоха визначає, скільки ітерацій усі вхідні вектори повинні подаватися в SOM для навчання.

Алгоритм зворотного поширення помилки, при навчанні, використовує два поширення мережею – пряме та зворотнє. На початку застосовано прямий прохід – генерується набір вихідних сигналів, який визначає реакцією мережі на вхідні дані. Під час прямого проходу усі синаптичні ваги мережі є фіксованими. Другим етапом є зворотній прохід – параметри (синаптичні ваги) налаштовуються відповідно за правилами корекції помилок. Від очікуваних вихідних значень віднімається отримане значення фактичного виходу і в результаті такої операції формується сигнал помилки. Оскільки ваги мережі ініціалізовані до встановлення випадкових значень, малоймовірно, що перед тренуванням небезпечні результати роботи потрапляють до мережі.

Система, яка моделюється є засобом ідентифікації аномальних і загальних пакетів в мережі. Весь процес розробки системи можна розділити на два етапи. Першим є етап підготовки, в якому нейронні мережі SOM та зворотного розповсюдження помилки пройшли навчання на протязі певної кількості часу (епох) (рис. 2).

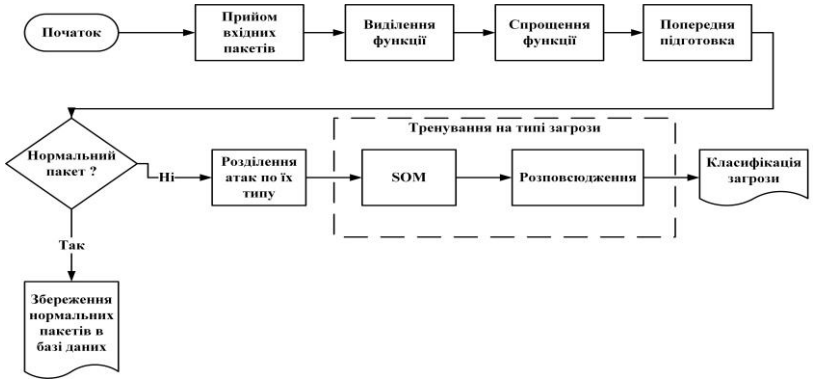

Рисунок 2 – Алгоритм фази навчання

Наступним етапом є етап виявлення або тестування. Цей метод наведено на рисунку 3. Оскільки операції звичайних пакетів вказані і вони демонструють очікувану поведінку, ми можемо на основі знань ініціювати визначення (неправильне використання), тоді як нетипова активність пакета (ймовірно, вторгнення продемонструє нетипову поведінку роботи пакету) постійно розробляється і не може розглядатися як визначена атака, тому виявлення аномалій IDS виконується над атаками.

Неконтрольована нейронна мережа на основі SOM поділяє класифікацію загрози на менші категорії з урахуванням їх схожих особливостей, а потім, на основі помилки зворотного розповсюдження нейронної мережі, виконується кластеризація загроз.

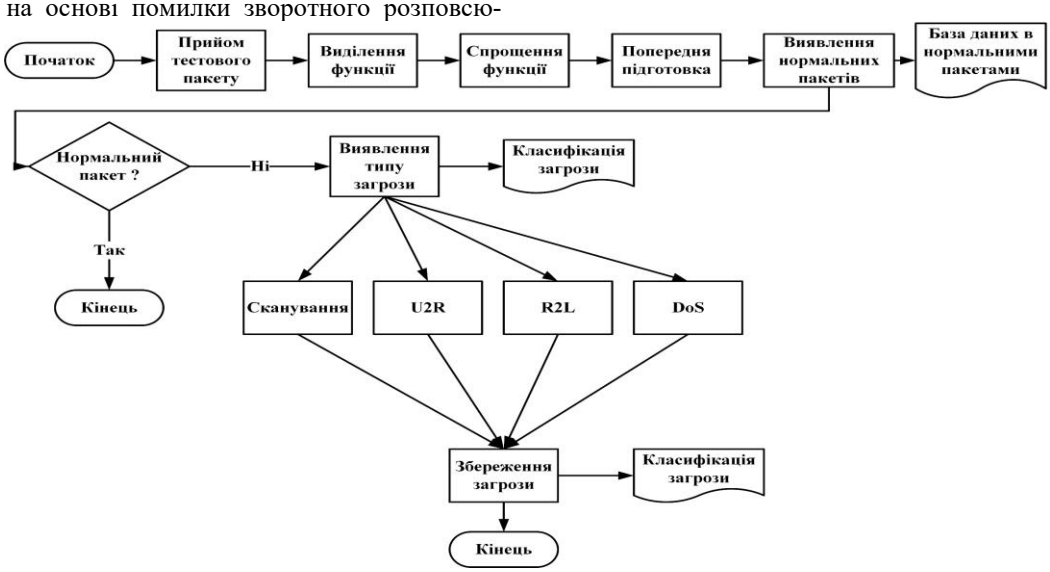

Рисунок 3 – Алгоритм фази виявлення

Програмна реалізація

В процесі проектування та програмної реалізації системи авторами реалізоване програмне забезпечення, що дозволяє виявити та класифікувати загрози у мережному трафіку (пакетах).

Як програмний метод обрано Python. Він є інтерпретованою, об'єктно-орієнтованою мовою програмування та підтримує пакети модулів та декілька парадигм програмування: об'єктно-орієнтовану, процедурну, функціональну та аспектно -орієнтовану [10]. На вибір Python вплинули: підтримка Python об'єктно-орієнтованого підходу, простота синтаксису та наявність вбудованих функцій і структур даних.

В якості основного фреймфорку для проектування, розробки та навчання даних персептронів використовуємо PyBrain. Даний фреймфорк має достатню кількість рішень та алгоритмів для навчання, проектування і тестування нейронних мереж різних типів. Архітектура програмного забезпечення представлена з використанням процедурно-об'єктного підходу.

Через відсутність потреби у збереженні проміжного результату роботи системи (після навчання персептрон зберігається в файл), весь прогрес навчання зберігаємо у звичайні файли на жорсткому диску. Як основне програмне забезпечення, для спрощення побудови основної структури системи використовуємо setuptools. Програма складається з основної функції, яка викликається при запуску програми і після ініціалізації класів.

```
from __future__ import __all__
from scipy import random
from scipy.ndimage import minimum_position
from scipy import mgrid, zeros, tile, array,
floor, sum

……..
        Kohonen_SOM_Map.outputFullMap =
output_full_map
        Kohonen_SOM_Map.neurons =
random.random((neurons_num, neurons_num,
dim))
        Kohonen_SOM_Map.winner = zeros(2)
        Kohonen_SOM_Map.diff =
zeros(self.neurn.shape)
        Kohonen_SOM_Map.inputs_num = dim
        Kohonen_SOM_Map.neurons_num =
neurons_num
        Kohonen_SOM_Map.neighbours =
neurons_num

……..

    def _forward_err_implement(self, inbuf,
outbuf):
        Kohonen_SOM_Map.diff =
Kohonen_SOM_Map.neurons - tile(inbuf,
(Kohonen_SOM_Map.neurons_num,
Kohonen_SOM_Map.neurons_num, 1))
        error = sum(Kohonen_SOM_Map.diff **
2, 2)
        Kohonen_SOM_Map.winner =
array(minimum_position(error))
        if not Kohonen_SOM_Map.outputFullMap:
            outbuf[:] =
Kohonen_SOM_Map.winner

…..
    def _backward_err_implement(cls):
        n = floor(cls.neighbours)
        cls.neighbours *= cls.neighbourdecay
        tl = (cls.winner - n)
        br = (cls.winner + n + 1)
        tl[tl < 0] = 0
        br[br > cls.neurons_num + 1] =
cls.neurons_num + 1
```

```
…..
    def __dist_matrix_create(cls):
        if not cls.neurons_num:
            print ("Kohonen_map: not setted
neural layers")
        distx, disty =
mgrid[0:cls.neurons_num, 0:cls.neurons_num]
        cls.dist_matrix =
zeros((cls.neurons_num, cls.neurons_num, 2))
        cls.dist_matrix[:, :, 0] = distx
        cls.dist_matrix[:, :, 1] = disty
```

Метод _forward_err_implement призначає один з нейронів вхідними даними в буфері вводу і фіксує координати нейронів в буфері виводу, а також виконує обчислення нейрона з найбільшою вагою, вираховуючи дані з найменшою похибкою за допомогою використання квадрату різниці. Метод _backward_err_implement тренує мапу Кохонена в безконтрольному режимі, переміщуючи найближчий нейрон та суміжні з ним нейрони ближче до вхідного шаблону. Функція main виконує ініціалізацію екземпляру класу - створення об'єкту KohonenMap та присвоєння даного об'єкту змінній. Після чого, в буфер вводу, задаються дані для навчання та проведення тренування даного об'єкту. Після навчання, на протязі декількох епох, результат зберігається на жорсткий диск.

Всі реалізовані модулі використовуються на дистрибутивах Unix подібних систем. Рішення засновано на тому, що велика кількість мережного обладнання, що використовує операційні системи, застосовує саме Unix подібні системи, зокрема сімейство Linux дистрибутивів OpenWrt. Головною платформою при виявленні пакетів є набір утиліт Iptables. Iptables є утилітами командного рядку та стандартним інтерфейсом керування роботою міжмережного екрану.

Головним принципом інтегрування утиліти аналізу даних та iptables є проведення аналізу логів утиліти iptables. При цьому попередньо змінюємо параметри логування iptables у файлі /var/log/kern.log, щоб відокремити запис логів для всієї операційної системи. При надходженні пакета у систему міжмережного екрану пакет починає проходити через систему із так званих чейнів таблиць фільтрації, які відноситься до даного походження пакета (nat, mangle, filther, security, raw, rawspot). Після проходження пакету через фільтрацію iptables, логи фіксуються та записуються у файл. Далі також налаштовується аналіз файлу логів, що фіксує роботу операційної системи у реальному часі та виконується обробка нових логів персептроном з проведенням аналізу є пакет безпечним чи небезпечним.

Похідні функції спрощуються для кожного з мережевих пакетів. Деякі з них є надлишковими, оскільки вони тісно пов'язані між собою, що зменшує не тільки швидкість, але й точність визначення загроз в комп'ютерній мережі [9].

Спрощення KDD-функцій, для кожного типу пакетів, виконано шляхом їх перетворення з текстової або символьної форми у числову. У цьому перетворенні для кожного символу призначається цілочисельний код. Наприклад, у випадку функції типу протоколу: нуль присвоюється протоколу tcp, одиниця – udp та двійка – icmp. Імена атак відповідають одному з п'яти класів: нуль для Normal, одиниця для Probe, двійка для DoS, трійка для U2R та четвірка для R2L.

Висновки

В процесі реалізації системи, заснованій на ідеї гібридизації нейронних мереж, вирішені завдання з забезпечення захисту внутрішньої мережі від зовнішніх загроз за допомогою фільтрації пакетів для таких загроз, як відмова в обслуговуванні та несанкціоноване збільшення привілеїв користувача. Використовуючи попередній аналіз пакетів на його небезпечність, збільшена ефективність методів захисту комп'ютерних мереж від шкідливого трафіку. Крім того, збільшена ефективність нейронної мережі за рахунок доставки на входи вже відфільтрованої інформації.

У подальшому розглядається можливість використання цього програмного забезпечення на операційній системі таких маршрутизаторів, як OpenWrt. Ця інтеграція не тільки підвищить ефективність системи при захисті мережі, але й підвищить точність персептрону за рахунок прийняття великої кількості мережного трафіку з самостійним вивченням нейронних мереж.

Список літератури

1. Garcia-Teodoro P., Diaz-Verdejo J., Maciá-Fernández G., Vázquez E. Anomaly based network intrusion detection: Techniques, systems and challenges. Computers & security, Vol. 28, 2009. pp.18-28.

2. Zhang Z. H. E. N. G., Manikopoulos C. Neural networks in statistical anomaly intrusion detection. Neural network world, Vol. 3, 2001, pp.305-316.

3. Kirichek G., Tymoshenko V., Rudkovskyi O., Hrushko S. Decentralized System for Run Services. //CMIS, 2019, pp.860-872.

4. DARPA Intrusion Detection Evaluation KDD dataset, URL: http://kdd.ics.uci.edu//databases/kddcup98/kddcup98.html.

5. Tavallaee M., Bagheri E., Lu W., Ghorbani A. A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp.1-6.

6. Akbar S., Rao K.N., Chandulal J.A. Intrusion detection system methodologies based on data analysis. International Journal of Computer Applications, 2010, 5(2), pp.10-20.
7. Kohonen T. The self-organizing map. Proceedings of the IEEE, 1990, 78(9), pp.1464-1480.
8. Ritter H., Martinetz T., Schulten K., Barsky D., Tesch M., Kates R. Neural computation and self-organizing maps: an introduction. Reading, MA: Addison-Wesley, 1992, pp. 141-161.
9. Bahrololum M., Salahi E., Khaleghi M. An improved intrusion detection technique based on two strategies using decision tree and neural network. JCIT, 2009, 4(4), pp. 96-101.
10. Kirichek, G., Kurai, V.: Implementation quadtree method for comparison of images. //TCSET 2018, Slavske, 2018, pp.129-132 doi: 10.1109/TCSET.2018.8336171.

References

1. Garcia-Teodoro P., Diaz-Verdejo J., Maciá-Fernández G., Vázquez E. Anomaly based network intrusion detection: Techniques, systems and challenges. Computers & security, Vol. 28, 2009. pp.18-28.
2. Zhang Z. H. E. N. G., Manikopoulos C. Neural networks in statistical anomaly intrusion detection. Neural network world, Vol. 3, 2001, pp.305-316.
3. Kirichek G., Tymoshenko V., Rudkovskyi O., Hrushko S. Decentralized System for Run Services. //CMIS, 2019, pp.860-872.
4. Akbar S., Rao K.N., Chandulal J.A. Intrusion detection system methodologies based on data analysis. International Journal of Computer Applications, 2010, 5(2), pp.10-20.
5. DARPA Intrusion Detection Evaluation KDD dataset, URL: http://kdd.ics.uci.edu//databases/kddcup98/kddcup98.html.
6. Tavallaee M., Bagheri E., Lu W., Ghorbani A. A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp.1-6.
7. Kohonen T. The self-organizing map. Proceedings of the IEEE, 1990, 78(9), pp.1464-1480.
8. Ritter H., Martinetz T., Schulten K., Barsky D., Tesch M., Kates R. Neural computation and self-organizing maps: an introduction. Reading, MA: Addison-Wesley, 1992, pp. 141-161.
9. Bahrololum M., Salahi E., Khaleghi M. An improved intrusion detection technique based on two strategies using decision tree and neural network. JCIT, 2009, 4(4), pp. 96-101.
10. Kirichek, G., Kurai, V.: Implementation quadtree method for comparison of images. //TCSET 2018, Slavske, 2018, pp.129-132 doi: 10.1109/TCSET.2018.8336171.

Надійшла до редакції 18.07.2019

Г.Г. КИРИЧЕК, В.Ю. ГАРКУША
Национальный университет "Запорожская политехника" (Украина)

ПРОЦЕСС ВЫЯВЛЕНИЯ ЗЛОУПОТРЕБЛЕНИЙ И АНОМАЛИЙ В СЕТИ

В работе предложен метод реализации системы обнаружения злоупотреблений и аномалий с использованием обучения обычных и атакующих пакетов, соответственно. Метод представляет собой комбинацию неподконтрольной и контролируемой нейронной сети для системы обнаружения вторжений. Для управления кластерами применяется нейронная сеть на основе Backpropagation.

Ключевые слова: нейронная сеть, выявление аномалий, вторжения, обучение, SOM.

G.G. KIRICHEK, V.Y. HARKUSHA
National University "Zaporizhia Politechnic" (Ukraine)

THE PROCESS OF DETECTING ABUSES AND ANOMALIES IN THE NETWORK

The realization method of the detection system of abuses and anomalies with introduction of training ordinary and attacking packages offers in the work. Method, used for teach an attack, is a combination of an uncontrollable and controlled neural network for an intrusion detection system. Neural network based on Backpropagation uses to manage clusters. The purpose of the work is to implement a prototype of the detecting abuse system on basis of a hybrid neural network. The research object is the implementation process of the detecting threats and anomalies modules. The subject is models, methods and software tools for constructing a system prototype. In the design process and program implementation of system, the authors developed of software that detects and classifies threats in network traffic (packets). The Python language selected as a software method. We use of PyBrain as the main framework for designing, developing and learning perceptron data. All implemented modules are using on distributions of Unix-like systems. The main platform for identifying packages is the Iptables utility suite. In the process of system implementation, based on the idea of neural networks hybridization, internal network secured from external threats by filtering packages for threats such as denial of service and unauthorized increase of user rights.

Keywords: neural network, detection of anomalies, invasion, training, SOM.