УДК 004.056.5

Владислав ОСІДАК

Західноукраїнський національний університет

ПОВЕДІНКОВИЙ АНАЛІЗ У ЗАДАЧІ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОГРАМ

Вступ. Актуальність теми "Поведінковий аналіз у задачі виявлення шкідливих програм" обумовлена зростаючими загрозами кібербезпеки та складністю виявлення нових типів шкідливого ПЗ. Традиційні методи, які базуються на сигнатурах, часто не здатні ефективно розпізнати нові або модифіковані віруси.

Поведінковий аналіз дозволяє відстежувати дії програм у реальному часі, що дає змогу виявляти шкідливі програми навіть до їх поширення. Це робить методи поведінкового аналізу важливими для підвищення ефективності захисту від кіберзагроз.

Метою дослідження є аналіз ефективності застосування методів поведінкового аналізу для виявлення шкідливих програм. Дослідження також має на меті виявити основні переваги та обмеження цього підходу порівняно з традиційними методами. Окрім того, буде розглянуто можливості інтеграції поведінкового аналізу в сучасні системи кіберзахисту для покращення виявлення та нейтралізації нових загроз.

1. Підходи до аналізу шкідливого програмного забезпечення

Шкідливе програмне забезпечення вже довгий час є однією з основних загроз в галузі інформаційної безпеки. Підходи до аналізу та захисту від таких атак бувають різні. Загалом поділяють два підходи: статичний та динамічний аналіз.

Завдання статичного аналізу - пошук шаблонів шкідливого вмісту у файлі чи пам'яті процесу. Це можуть бути рядки, фрагменти закодованих або стислих даних, послідовності компільованого коду. Може здійснюватися пошук як окремих шаблонів, а й їх комбінацій з додатковими умовами (наприклад, з прив'язкою до місця знаходження сигнатури, перевіркою відносної відстані в розміщені один від одного).

Динамічний аналіз – це аналіз поведінки програми. Варто зазначити, що програма може бути запущена в так званому емульованому режимі. Передбачається безпечне інтерпретування дій без завдання пошкоджень операційній системі. Інший спосіб - запуск програми у віртуальному середовищі (пісочниці). У такому разі буде чесне виконання дій на системі з подальшою фіксацією дзвінків. Ступінь подробиці логування - це свого роду баланс між глибиною спостереження та продуктивністю аналізуючої системи. На виході виходить журнал дій програми операційній системі (траса поведінки), який піддається подальшому аналізу.

Динамічний чи поведінковий аналіз дає ключову перевагу - незалежно від спроб заплутування програмного коду та прагнень приховати наміри зловмисника від вірусного аналітика шкідливий вплив буде зафіксовано. Зведення завдання

виявлення ВПО до аналізу дій дозволяє висунути гіпотезу про стійкість просунутого алгоритму виявлення шкідливих даних. А відтворюваність поведінки, завдяки тому самому початковому стану середовища для аналізу (зліпка стану віртуального сервера), спрощує вирішення завдання класифікації легітимного і шкідливого поведінки.

Часто підходи у поведінковому аналізі ґрунтуються на наборах правил. Експертний аналіз переноситься в сигнатури, на основі яких інструмент детекту шкідливого ПЗ та файлів робить висновки. Однак у такому разі може виникнути проблема: можуть враховуватися лише ті атаки, які суворо відповідають написаним правилам, а атаки, які не виконують ці умови, але все ще шкідливі, можна пропустити. Та ж проблема виникає у разі змін одного й того ж шкідливого ПЗ. Вирішити це можна за допомогою більш м'яких критеріїв спрацьовування, тобто можна написати більш загальне правило, або за допомогою великої кількості правил під кожен шкідливість. У першому сценарії ми ризикуємо отримати багато помилкових спрацьовувань, а другий вимагає серйозних витрат за часом, що може призвести до запізнення необхідних оновлень.

З'являється потреба у поширенні вже наявних знань інші схожі випадки. Тобто ті, які раніше ми не зустрічали і не обробляли правилами, але на основі схожості деяких ознак можемо зробити висновок, що активність може бути шкідливою. Тут і допомагають алгоритми машинного навчання.

ML-моделі під час коректного навчання мають узагальнюючу здатність. Це означає, що навчена модель не просто вивчила всі приклади, на яких навчалася, а здатна приймати рішення для нових прикладів на основі закономірностей із навчальної вибірки.

Однак для того, щоб узагальнююча здатність працювала, необхідно враховувати два основні фактори на етапі навчання:

Набір ознак повинен бути якомога повнішим (щоб модель могла бачити якнайбільше закономірностей, відповідно, краще поширювала свої знання на нові приклади), але не надлишковим (щоб не зберігати і не обробляти ознаки, які не несуть у собі корисну інформацію для моделі).

Набір даних має бути репрезентативним, збалансованим та регулярно оновлюваним.

2. Процес переносу експертного знання в моделі машинного навчання

У контексті аналізу шкідливого програмного забезпечення вихідні дані - це самі файли, а проміжні дані - це створені ними допоміжні процеси. Процеси, у свою чергу, здійснюють системні виклики. Послідовності таких викликів є дані, які нам необхідно перетворити на набір ознак.

Складання датасету розпочалося на експертній стороні. Було обрано ознаки, які, на думку експертів, мають бути значущими з погляду виявлення ШПЗ. Усі ознаки можна було звести до виду n-грам за системними викликами.

Далі за допомогою моделі проведена оцінка, тих ознак які роблять найбільший внесок у виявлення, відкинули зайве і отримали підсумкову версію датасета.

Вихідні дані:

{"count":1,"PID":"764","Method":"NtQuerySystemInformation","unixtime":"1639557419.628073","TID":"788","plugin":"syscall","PPID":"416","Others":"REST:
,Module=\"nt\",vCPU=1,CR3=0x174DB000,Syscall=51,NArgs=4,SystemInformationClass=0x53,SystemInformation=0x23BAD0,SystemInformationLength=0x10,ReturnLength=0x0","ProcessName":"windows\\system32\\svchost.exe"}

{"Key":"\\registry\\machine","GraphKey":"\\REGISTRY\\MACHINE","count":1,"plugin":"regmon","Method":"NtQueryKey","unixtime":"1639557419.752278","TID":"3420","ProcessName":"users\\john\\desktop\\e95b20e76110cb9e3ecf0410441e40fd.exe","PPID":"1324","PID":"616"}

{"count":1,"PID":"616","Method":"NtQueryKey","unixtime":"1639557419.752278","TID":"3420","plugin":"syscall","PPID":"1324","Others":"REST:
,Module=\"nt\",vCPU=0,CR3=0x4B7BF000,Syscall=19,NArgs=5,KeyHandle=0x1F8,KeyInformationClass=0x7,KeyInformation=0x20CD88,Length=0x4,ResultLength=0x20CD98","ProcessName":"users\\john\\desktop\\e95b20e76110cb9e3ecf0410441e40fd.exe"}

3. Покращення якості моделі з кожним оновленням

Вважаємо вибірку найбільш коректною, тому що приклади цієї вибірки перевіряються і розмічуються експертами вручну, і з кожним оновленням перевіряється в першу чергу те, що гарантується 100% точності на цій вибірці. Тестування in the wild підтверджує, що точність покращується.

Досягається це за рахунок очищення навчальної вибірки від еталонних даних, що суперечать. Під даними, що суперечать, ми розуміємо приклади, накопичені з потоку, які досить близькі по векторній відстані до трас з еталонної вибірки, але при цьому мають протилежну мітку.

Експерименти показали, що такі приклади є викидами навіть з погляду даних із потоку, оскільки після видалення їх із навчальної вибірки з метою підвищення точності на еталонній вибірці, зростала і точність на потоці.

4. Взаємодоповнення ML-підходу та поведінкових детектів у вигляді кореляцій

ML-модель дуже добре проявила себе у поєднанні з поведінковими детектами у вигляді кореляцій. Важливо зауважити, що саме в поєднанні, так як узагальнююча здатність моделі хороша у випадках, коли необхідно розширити рішення виявленням схожих та близьких інцидентів, але не у випадках, коли потрібен детект у рамках чіткого розуміння правил та критеріїв того, що є шкідливим ПЗ.

Прикладами, де ML-підхід зміг дійсно розширити рішення, стали:

– Аномальні ланцюжки підпроцесів. Саме собою велика кількість гіллястих ланцюжків - явище легітимне. Але аномальність у кількості вузлів, ступеня вкладеності, повторюваності чи повторюваності якихось конкретних імен процесів модель зауважує, а людина заздалегідь таке не нафантазує знайти шкідливим.

– Нестандартні параметри дзвінків за промовчанням. Найчастіше

аналітика цікавлять значні параметри функцій, у яких шукають ШПЗ. Інші параметри, грубо кажучи, значення за замовчуванням, вони не особливо цікавлять. Але в якийсь момент так виходить, що замість припустимо п'яти значень за умовчанням зустрічається шосте. Аналітик міг припустити, що таке можливо, а модель помітила.

– Нетипові послідовності викликів функцій. Той випадок, коли кожна функція окремо робить нічого шкідливого. Та й разом - теж. Але так сталося, що їхня послідовність не зустрічається в легітимному ПЗ. Аналітику буде потрібний гігантський досвід, щоб самостійно помітити таку закономірність. А модель помічає (і не одну), вирішуючи нестандартно завдання класифікації за ознакою, яка взагалі не закладалася як показник шкідливості.

Використання конкретного компонента одним викликом для шкідливої дії. Система використовує сотні об'єктів у різній варіативності, різною мірою. Вловити використання одного на тлі мільйона інших навряд чи вдасться - гранулярність аномалії все ж таки занизька. Проактивний детект за моделлю загроз. Вирішили, що певний вплив на певний об'єкт у системі хоча б один раз, неприпустимо. Модель може з першого разу не зрозуміти, що це значуще явище і буде шанс помилки чи невпевненого рішення на етапі класифікації чогось схожого. Обфускація послідовності процесів. Наприклад, може бути відомо, що потрібно зробити 3-4 дії у визначеному порядку. Не має значення, що буде між ними. Якщо накидати випадкові дії між 3-4 ключовими - це модель, рішення буде прийнято неправильно. При цьому розмірність числа ознак не дозволяє враховувати такі заплутування зберігання всіх комбінацій послідовностей викликів, а не тільки загальної кількості.

Висновок. Поведінковий аналіз є ефективним інструментом для виявлення шкідливих програм, оскільки дозволяє виявляти нові та модифіковані загрози, яких не можна розпізнати за допомогою традиційних методів. Він забезпечує гнучкість та адаптивність у боротьбі з кіберзагрозами, враховуючи динамічний характер сучасних вірусів. Однак для досягнення максимальної ефективності необхідно інтегрувати поведінковий аналіз із іншими методами захисту. У результаті, використання цього підходу може значно підвищити рівень безпеки в цифрових середовищах.

Перелік використаних джерел.

1. A. A. Selçuk, F. Orhan and B. Batur, "Undecidable problems in malware analysis," 2017 12th International Conference for Internet Technology and Secured Transactions (ICITST), Cambridge, UK, 2017, pp. 494-497, doi: 10.23919/ICITST.2017.8356458.

2. A. Afreen, M. Aslam and S. Ahmed, "Analysis of Fileless Malware and its Evasive Behavior," 2020 International Conference on Cyber Warfare and Security (ICCWS), Islamabad, Pakistan, 2020, pp. 1-8, doi: 10.1109/ICCWS48432.2020.9292376.

3. O. Or-Meir, A. Cohen, Y. Elovici, L. Rokach and N. Nissim, "Pay Attention: Improving Classification of PE Malware Using Attention Mechanisms Based on System Call Analysis," 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 2021, pp. 1-8, doi: 10.1109/IJCNN52387.2021.9533481.