МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Одеська юридична академія»

ЄВРОПЕЙСЬКІ ОРІЄНТИРИ РОЗВИТКУ УКРАЇНИ: НАУКОВО-ПРАКТИЧНИЙ ВИМІР В УМОВАХ ВОЄННИХ ВИКЛИКІВ

МАТЕРІАЛИ
Міжнародної науково-практичної конференції

Одеса, 26 квітня 2024 року

Одеса
«Фенікс»
2024

УДК 005.332.2(4):316.42(477)“364”“20”(062.552)
Є 24

Рекомендовано до друку вченою радою
Національного університету «Одеська юридична академія»
(протокол № 8 від 15.05.2024 р.)

За загальною редакцією ***С. В. Ківалова***

Відповідальний за випуск ***М. Р. Аракелян***

Є 24 **Європейські орієнтири розвитку України: науково-практичний вимір в умовах воєнних викликів** : матеріали Міжнар.наук.-практ. конф. (Одеса, 26 квітня 2024 р.) / за заг. ред. С. В. Ківалова. – Одеса : Фенікс, 2024. – 1000 с.
ISBN 978-617-8430-05-4

У збірнику відображено наукові напрацювання вчених, практиків, військовослужбовців у теоретичній та емпіричній площині у сферах філософських основ, загальної теорії та історичних досліджень держави і права, актуальних проблем світових соціально-політичних процесів, соціології та психології в умовах повномасштабного військового вторгнення. Висвітлено питання загроз національній безпеці в їх конституційному вимірі у рамках міжнародного та європейського права, трудового права та права соціального забезпечення, земельного, аграрного та екологічного права, пов’язані з функціонуванням економіки та підприємництва в умовах європейського вибору України. Розглянуто проблеми інформатизації та цифровізації суспільства, захисту інформації та кібербезпеки в умовах військового вторгнення, питання методики викладання іноземних мов, теорії та практики перекладу, проблеми лінгвістики та журналістики. Відображено наукові напрацювання у сферах адміністративного права та процесу, фінансового, морського та митного права, організації та вдосконалення судоустрою, прокуратури, інших правоохоронних органів, адвокатури. Висвітлено питання кримінального права, кримінально-процесуальних аспектів кримінального провадження та кримінологічних особливостей протидії злочинності в умовах воєнного стану, криміналістики, судової експертизи, психології та медицини у забезпеченні судочинства, цивільного та сімейного права, інтелектуальної власності та патентної юстиції, цивільного судочинства, господарського права та процесу. Розглянуто актуальні питання діяльності сучасних бібліотек у закладах вищої освіти, фізичної підготовки здобувачів вищої освіти.

Збірник розраховано на наукових і науково-педагогічних працівників, здобувачів вищої освіти, практичних працівників у сферах юридичної, економічної, соціологічної, політологічної, психологічної, філологічної наук, журналістики та кібербезпеки тощо.

УДК 005.332.2(4):316.42(477)»364»»20»(062.552)

Матеріали видано в авторській редакції

ISBN 978-617-8430-05-4

СЕКЦІЯ 24. КІБЕРБЕЗПЕКА ОСОБИ, СУСПІЛЬСТВА ТА ДЕРЖАВИ ЯК ФУНДАМЕНТАЛЬНИЙ ЕЛЕМЕНТ ЗАБЕЗПЕЧЕННЯ ОБОРОНОЗДАТНОСТІ УКРАЇНИ

СЕКЦІЯ 25. ПИТАННЯ МЕТОДИКИ ВИКЛАДАННЯ ІНОЗЕМНИХ МОВ, ТЕОРІЇ ТА ПРАКТИКИ ПЕРЕКЛАДУ ЯК ЗАСОБУ НАЛАГОДЖЕННЯ КОМУНІКАЦІЇ

– дата та час створення первісного необробленого файлу;
– місце створення файлу (вимагає прив'язки до геолокації користувачів, їх координати, за певних умов конфіденційності персональних даних можна використовувати поштовий індекс міста, що дозволить ідентифікувати місце події).

Перелічені дані дозволять визначити реальну дату та місце зйомки, модель пристрою – визначити рівень та оснащеність тієї особи, яка вела запис, а це за непрямими ознаками дозволяє зрозуміти мету поширення фейків. Додатково можуть бути внесені відомості щодо авторства, поточних змін щодо обробки мультимедійного контенту (час та дата змін, якою програмою були внесені зміни), які також дозволили б скласти повне уявлення про сутність мультимедійного контенту.

Розробка програмної реалізації методів вбудовування метаданих в контент мультимедійних файлів має враховувати наступне:
– нечутливі мінімальні зміни в структурі даних мультимедійного файлу. Якщо це відео, зображення, то вбудовування інформації ніяким чином не має впливати на візуальну якість результату (заповненого контейнеру), в аудіо даних не має бути чутно перешкод;
– стійкість до атак стисненням, афінних перетворень, шуму, просторових перетворень, адже великий обсяг інформації, що зберігається і передається, передбачає стиснення для зменшення розміру файлів, а в процесі передачі можливі зайві сторонні впливи, шуми, тощо;
– можливість додавання нових метаданих щодо виконаних над мультимедійним контентом змін та модифікацій.

Така стеганографічна система дозволить вбудувати всі відомості щодо створення мультимедійних файлів та для звичайних користувачів надасть змогу перевірити, коли, де, ким були створені мультимедійні дані.

Список використаних джерел

1. Про судову практику у справах про захист гідності та честі фізичної особи, а також ділової репутації фізичної та юридичної особи : постанова Пленуму Верховного Суду України № 1 від 27.02.2009. URL: https://zakon.rada.gov.ua/laws/show/v_001700-09/conv#o47
2. Aspose. Total App Product Family. URL: https://products.aspose.app/
3. Сімейство додатків GroupDocs.Metadata App. URL: https://products.groupdocs.app/uk/metadata/family
4. ExifTool for photo and video. URL: https://play.google.com/store/apps/details?id=com.exiftool.free&hl=uk

Ключові слова: кібербезпека, стеганографія, мультимедійні дані, фейки, метадані
Keywords: cybersecurity, steganography, multimedia data, fakes, meta data

Бойко Віктор Дмитрович
Національний університет «Одеська юридична академія»,
доцент кафедри кібербезпеки, кандидат технічних наук, доцент

БІБЛІОТЕКА NUMPY – ОСНОВА ІНСТРУМЕНТАЛЬНОГО СТЕКУ НАУКОВИХ РОЗРАХУНКІВ SCIPY

У роботах [1; 2] розглядалось створення дослідних систем на базі інструментального стеку мови python. Збільшення обчислювальних потужностей, поряд з розширенням обся-

гу та характеру інформації, що збирається, зробило актуальними використання спеціалізованих методів роботи з великими даними, дата-майнінгу та дослідницького моделювання. Наші рекомендації здебільшого було орієнтовано на користувачів POSIX-сумісних систем, що мають у своєму складі розвинену командну оболонку сценаріїв (bash, zsh і т.д.), до яких можна віднести і системи під керуванням MacOS, як сумісні з стандартом POSIX. Як було показано в попередніх роботах, для організації стеку наукових та дослідницьких розрахунків потрібні зовнішні бібліотеки python. Зупинимося докладніше на роботі з бібліотекою NumPy.

Фундаментальною бібліотекою, на якій засновано весь науковий стек, є NumPy [3]. Ця бібліотека була створена для того, щоб подолати невисоку швидкість обчислень у базовому Python. Модулі бібліотеки були написані мовою C, а алгоритми широко використовують векторизацію (vectorization) та окремий тип даних (array). Завдяки цьому з одного боку досягається висока швидкість обробки даних (особливо це стосується різних матричних обчислень), з іншого боку, користувач використовує бібліотеку, як частину мови python, а не мови C.

Ця бібліотека часто використовується не тільки в наукових, а й у прикладних обчисленнях, зокрема в різних обробках великих обсягів інформації (наприклад, системних журналів, логів веб-серверів тощо) [4].

Розглянемо різницю у роботі з традиційними засобами python та бібліотекою Numpy у прикладі для традиційного REPL-середовища оболонки python3 та для ipython.

Підключення бібліотеки може виконуватись різними способами.

Найбільш очевидна наступна схема:

>>> import numpy

Однак, при частому використанні префікс numpy засмічуватиме систему. Тому в деяких посібниках прийнято систему безпрефіксного імпорту.

>>> from numpy import *

Це зменшує засміченість namespace, проте збільшує ризик перекриття одних функцій іншими (numpy.sum/sum) і відповідно збільшує кількість помилок.

На наш погляд оптимальна схема, що склалася:

>>> import numpy as np

При ній ми працюємо з мінімальним префіксом (np.sum), при цьому не виникає ризику перекриття.

Розглянемо можливості numpy в порівнянні зі звичайною конвенційною роботою в python.

Для цього знадобиться бібліотека timeit, за допомогою якої можна вимірювати час виконання сніпетів коду.

Наприклад:

>>> import timeit

>>> timeit.timeit('[x for x in range(1000)]')

28.241173309999795

У середовищі ipython виклик даної бібліотеки буде більш інформативним:

In [1]: %timeit [x for x in range(1000)]

29 μs ± 245 ns per loop (mean ± std. dev. of 7 runs, 10,000 loops each)

Тут зрозуміло, що виклик запропонованого сніпету займає в середньому 29 мікросекунд плюс-мінус 245 наносекунд і що завмер йшов по 7 спроб по 10000 циклів кожна.

Час, який вимірює %timeit, буде трохи вищим, ніж той, який повідомляє модуль timeit.py. Це пов'язано з тим, що %timeit виконує оператор у просторі імен оболонки, порівняно з timeit.py, який використовує єдиний оператор налаштування для імпорту функції або створення змінних. Як правило, це не має значення, якщо результати з timeit.py не змішуються з результатами з %timeit.

Розглянемо, як обидві бібліотеки справляються із обробкою послідовних масивів. Для цього обчислимо послідовну суму квадратів від 0 до 99. Якщо ми використовуватимемо засоби конвенційного python, ми згенеруємо наступний список:

```
>>> a1 = [x for x in range(100)]
>>> a1
[0, 1, 2, 3, 4, 5, … , 97, 98, 99]
>>> [x ** 2 for x in a1]
[0, 1, 4, 9, 16, 25, … , 9409, 9604, 9801]
```

І загальна сума вийде рівною:

```
>>> sum([x ** 2 for x in b1])
328350
```

З використанням бібліотеки numpy ми отримаємо таку послідовність дій:

```
>>> a2 = np.arange(100)
>>> print(a2)
[ 0 1 2 3 4 5
…
96 97 98 99]
>>> a2 * a2
array([ 0, 1, 4, 9, 16, 25, …
…
7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 9216, 9409, 9604,
9801])
```

Далі ми маємо варіанти підсумовування – або конвенційним оператором python:

```
>>> sum(a2 * a2)
328350
```

Або засобами numpy:

```
>>> np.sum(a2 * a2)
328350
```

Виміряємо час виконання кожного з процесів.

```
>>> timeit.timeit('sum([x**2 for x in a1])', 'from __main__ import a1')
33.769974015000116
>>> timeit.timeit('sum(a2 * a2)', 'from __main__ import a2')
8.474411449999934
>>> timeit.timeit('np.sum(a2 * a2)', 'from __main__ import a2; from __main__ import np')
3.4669552790001035
```

Проведемо аналогічні вимірювання в середовищі ipython:

```
In [1]: import numpy as np
In [2]: a1 = [x for x in range(100)]
In [3]: a2 = np.arange(100)
In [4]: %timeit sum([x**2 for x in a1])
21.8 µs ± 360 ns per loop (mean ± std. dev. of 7 runs, 10,000 loops each)
In [5]: %timeit sum(a2 * a2)
8.48 µs ± 41.8 ns per loop (mean ± std. dev. of 7 runs, 100,000 loops each)
In [6]: %timeit np.sum(a2 * a2)
3.45 µs ± 51.2 ns per loop (mean ± std. dev. of 7 runs, 100,000 loops each)
```

Як бачимо, результати загалом збігаються. Навіть на таких простих прикладах даних (одномірний масив), numpy за рахунок векторизації дозволяє отримувати на порядок більшу продуктивність. Цікаво, що різниця між результатами sum та np.sum обумовлена різницею в алгоритмах – оператор sum викликає "магічний" дандер __add__, а np.sum використовує більш складний підхід, пов'язаний з урахуванням точності підсумовування float типів.

Таким чином, бібліотека numpy дозволяє ефективно здійснювати розрахунки з великими масивами чисел, при цьому зберігаючи зручність використання python мови, що дозволяє швидко і ефективно виконувати аналіз великих обсягів даних.

Список використаних джерел

1. Бойко В. Д. Інструменти моделювання та аналізу даних у навчальному та дослідному процесі. *Європейські орієнтири розвитку україни в умовах війни та глобальних викликів XXI століття. Синергія наукових, освітніх та технологічних рішень :* матеріали Міжнар. наук.-практ. конф. 19 трав. 2023 р. / ред. С. В. Ківалов. Одеса, 2023. Р. 618–621.

2. Бойко В. Д. Використання interactive python для моделювання, аналізу та обробки даних у навчальному та дослідницькому процесах. *Інформаційне суспільство: Проблеми та перспективи* : матеріали VIII всеукр. наук.-практ. конф. (м. Одеса, 2 черв. 2023 р.). Одеса, 2023. Р. 51–56.

3. Harris C. R. et al. Array programming with NumPy. *Nature. Springer Science; Business Media LLC*. 2020. Vol. 585, N 7825. P. 357–362.

4. Beazley D. M. Python distilled. Pearson, 2021. 352 p.

Ключові слова: аналіз даних, відкрите програмне забезпечення, відкриті формати даних, відкриті протоколи зв'язку, аналіз логів, аналіз логів веб-серверів.

Keywords: python, numpy,ipython, bigdata, data explore, data mining, data analysis, open software, open data formats, open communication protocols, log analysis, web server logs, analysis of web server logs, threats and risks.

Кухаренко Сергій Вікторович
Національний університет «Одеська юридична академія»,
доцент кафедри кібербезпеки, кандидат технічних наук, доцент

ПОЛІТИКИ ТА ПРОЦЕДУРИ РЕАГУВАННЯ НА ПОРУШЕННЯ БЕЗПЕКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Політика безпеки інформаційної системи визначає сукупність правил, вимог і керівних принципів в області інформаційної безпеки, якими керується організація в своїй діяльності та безпосередньо встановлює основні положення і завдання щодо системи управління інформаційною безпекою. Політика безпеки є нормативною основою для захисту інформаційних активів організації з метою забезпечення їх цілісності, конфіденційності та доступності [1].

Подія щодо кібербезпеки це – зміна стану кібербезпеки, яка може вплинути на діяльність організації.

Інцидент у кібербезпеки це – окрема або серія небажаних або неочікуваних подій з кібербезпекою або таких, що є можливими ознаками кібератаки які можуть поставити під загрозу роботу організації.

Існує багато типів інцидентів інформаційної безпеки, які можна класифікувати як інциденти кібербезпеки, починаючи від серйозних атак кібербезпеки на критично важливу національну інфраструктуру та великих організованих кіберзлочинів, через хакерство та базові атаки зловмисного програмного забезпечення (шкідливий програмний код), до внутрішнього неправомірного використання систем і збою програмного забезпечення.

Відповідний перелік типів кіберінцидентів розроблений з використанням та відповідає рекомендації Європейської агенції з кібербезпеки (ENISA Reference Incident Classification Taxonomy), а також спільному документу ENISA та Європейського центру боротьби з кіберзлочинністю Європолу (Common Taxonomy for Law Enforcement and The National Network