СТЕГАНОГРАФІЯ

ЛІТНЯ ШКОЛЯ З КІБЕРБЕЗПЕКИ 2026

ПРАКТИЧНЕ ЗАННЯТЯ №1

Завдання 1. Загублена флешка.

Опис завдання: Після заняття в комп'ютерному класі знайдено загублену флешку з навчальними файлами. Частина файлів має неправильні розширення, можливо через ручне перейменування.

Ваше завдання: Провести первинний аналіз файлів, визначити реальний тип кожного файлу та підтвердити його за допомогою файлової сигнатури. Перевірити чи містять файли читабельні рядки. Для виконання завдання потрібно використати інструменти `file`, `strings`, `hexdump`. Вони є корисними як перші інструменти під час аналізу CTF-файлів. Не рекомендується відкривати файли подвійним кліком до завершення аналізу.

Інструмент	Опис	Приклад
`file`	Визначає реальний тип файлу	`file *` `file sea_secret.png`
`strings`	Шукає читабельні рядки	`strings bird_secret.jpg | tail`
`hexdump`	Показує байти файлу в hex-форматі	`hexdump -C bird_secret.jpg | head`

Файлова сигнатура – це перші байти файлу, за якими можна визначити його справжній формат.

Приклади сигнатур:

- .jpg / .jpeg – FF D8 FF
- .png – 89 50 4E 47 0D 0A 1A 0A
- .pdf – 25 50 44 46 / %PDF
- .zip – 50 4B 03 04 / PK
- .docx / .xlsx / .pptx – 50 4B 03 04 / PK
- .html – 3C 21 44 4F 43 54 59 50 45 / <!DOCTYPE

Питання для обговорення:

1. Чому під час аналізу файлів не можна довіряти лише розширенню або іконці файлу?

2. Що робити, якщо розширення файлу не збігається з його сигнатурою?

Завдання 2. Рекламна кампанія.

Опис завдання: Студенти готують рекламну кампанію для нового курсу з кібербезпеки. Дизайнер передав кілька зображень і PDF-флаєр. Перед публікацією потрібно перевірити, чи не залишилися у файлах службові коментарі, чернеткові назви або внутрішні підказки.

Ваше завдання: Проаналізувати метадані файлів, знайти приховані службові повідомлення та визначити, чи є серед них закодовані. Якщо повідомлення закодоване, декодуйте його за допомогою CyberChef або іншого інструмента. Для перевірки використайте команди `exiftool`, `strings`. Не змінюйте і не видаляйте метадані файлів під час аналізу.

Інструмент	Опис	Приклад
`exiftool`	Перегляд, редагування та аналіз метаданих	`exiftool forest_secret.jpg` `exiftool -Comment forest_secret.jpg`
`strings`	Може також прочитати метадані	`strings forest_secret.jpg | grep flag`
CyberChef	Універсальний інструмент для аналізу, перетворення та декодування даних	From Base64

Метадані – це службова інформація про файл, яка не завжди видима під час звичайного відкриття. Вони можуть містити автора, дату створення, назву програми, коментарі, ключові слова, опис або внутрішні нотатки.

`Exiftool` відображає дані структуровано. `Strings` показує, що знайдено, а `exiftool` допомагає зрозуміти, де саме це лежить.

Питання для обговорення:

1. Яку інформацію можуть містити метадані файлів?
2. Як зрозуміти, що знайдений рядок може бути закодований у Base64?

Завдання 3. Політика безпеки.

Опис завдання: Новим співробітникам кампанії розіслали PDF-документ із політикою інформаційної безпеки. Файл виглядає як звичайний документ: правила паролів, робота з поштою, використання зовнішніх носіїв. Однак один із аналітиків помітив, що файл має нетипово великий розмір для кількох сторінок тексту. Є підозра, що до PDF могли додати прихований архів.

Ваше завдання: Перевірити структуру PDF-файлу за допомогою `binwalk`, знайти можливий вкладений ZIP-архів, витягнути його (`unzip` або `binwalk -e`) та проаналізувати вміст. Якщо всередині є QR-код або повідомлення у Base64, потрібно прочитати (Parse QR Code) або декодувати його за допомогою CyberChef чи іншого інструмента.

Інструмент	Опис	Приклад
`binwalk`	Шукає вкладені файли всередині іншого файлу	`binwalk data_secret.pdf` `binwalk -e data_secret.pdf`
`unzip`	Переглядає або розпаковує ZIP-архів	`unzip data_secret.pdf -d extracted_zip`

Питання для обговорення:

1. Які ознаки можуть вказувати на те, що у файлі є прихований контейнер або архів?

2. Які рядки у виводі `binwalk` підтверджують наявність ZIP-архіву?

Завдання 4. Автор CTF-завдання.

Опис завдання: Ви – автор навчального CTF-завдання та досліджуєте утиліту `steghide` для приховування даних у JPG-зображеннях. На цьому етапі потрібно самостійно створити простий стегофайл: обрати зображення, підготувати коротке текстове повідомлення, приховати його в картинці та перевірити, що повідомлення можна успішно вилучити.

Ваше завдання: Обрати одне із запропонованих JPG-зображень або використати власну JPG-картинку. Створити короткий TXT-файл, приховати його в зображенні за допомогою `steghide` та отримати стегофайл. Після цього потрібно вилучити зі стегофайлу прихований текстовий файл і перевірити, що його вміст збігається з оригінальним файлом за допомогою `diff` або `sha256sum`. Під час приховування задайте пароль і запам’ятайте його, адже він потрібен для вилучення повідомлення.

Інструмент	Опис	Приклад

`steghide`	Приховує або вилучає файли у контейнерах JPG, BMP, WAV, AU	`steghide embed -cf cat.jpg -ef secret.txt -sf cat_secret.jpg`
		
		`steghide info cat_secret.jpg`
		
		`steghide extract -sf cat_secret.jpg`

Інструмент	Опис	Приклад
`diff`	Порівнює оригінальний і вилучений текст	`diff secret.txt extracted.txt`
`sha256sum`	Порівнює хеші файлів	`sha256sum secret.txt extracted.txt`

Рекомендований порядок виконання:

1. Перевірити, що обране зображення є JPG:

```
file image.jpg
```

2. Створити текстовий файл:

```
nano message.txt
```

3. Написати своє секретне повідомлення, наприклад:

```
flag{my_first_steghide_task}
```

Зберегти: `Ctrl + O → Enter → Ctrl + X`

4. Приховати TXT-файл у JPG-зображенні (запросить пароль):

```
steghide embed -cf image.jpg -ef message.txt -sf stego_image.jpg
```

5. Перевірити створений файл:

```
file stego_image.jpg
steghide info stego_image.jpg
```

6. Вилучити прихований файл:

```
steghide extract -sf stego_image.jpg -xf extracted_message.txt
```

7. Порівняти оригінал і вилучений файли:

```
diff message.txt extracted_message.txt
sha256sum message.txt extracted_message.txt
```

Питання для обговорення:

1. Чому після приховування повідомлення стегофайл усе ще відкривається як звичайне зображення?

2. Чому повідомлення, приховане через `steghide`, зазвичай не видно напряму через `strings`?

Завдання 5. LSB-листівка.

Опис завдання: Ви – фахівець зі стеганографії, який працює в парі з колегою та досліджує LSB-стеганографію в PNG-зображеннях. Це один із найпоширеніших видів стеганографії у CTF-завданнях. У методі LSB (Least Significant Bit) молодші біти каналів кольору пікселів замінюються на біти секретного повідомлення.

Ваше завдання: Обрати одне із запропонованих PNG-зображень або використати власну PNG-картинку. За допомогою StegOnline приховати в ній коротке повідомлення методом LSB, зберегти отримане стегозображення та обмінятися ним із напарником. Після цього потрібно проаналізувати отриманий PNG-файл за допомогою `zsteg` і вилучити приховане повідомлення.

Інструмент	Опис	Приклад
StegOnline	Онлайн-інструмент для роботи з LSB-стеганографією у зображеннях	Embed Files/Data, Extract Files/Data
`zsteg`	Пошук LSB-повідомлень у PNG/BMP-файлах	`zsteg sea_secret.png`

Рекомендований порядок виконання (приховування):

1. Відкрити StegOnline і перетягнути PNG-зображення у робочу область.
2. Обрати опцію Embed Files/Data.
3. Для приховування повідомлення в StegOnline використати нульовий біт каналів R, G, B. Налаштування: *Pixel Order – Row, Bit Order – MSB, Bit Plane Order – R G B, Pad Remaining Bits – No*.
4. У поле для повідомлення ввести короткий текст, наприклад:

```
flag{lsb_postcard_2026}
```

5. Зберегти зображення як нову картинку та передати його напарнику.

Рекомендований порядок виконання (вилучення):

1. Встановити `zsteg`*:*

```
sudo apt update
sudo apt install ruby ruby-dev
sudo gem install zsteg
zsteg --version
```

2. Спочатку перевірити, що файл залишився PNG:

```
file postcard_stego.png
```

3. Потім запустити базовий аналіз:

```
zsteg postcard_stego.png
```

4. У виводі потрібно шукати прихований текст, наприклад:

```
b1,rgb,msb,xy .. text: "flag{lsb_postcard_2026}"
```

Питання для обговорення:

1. Чому після LSB-приховування зображення візуально майже не змінюється?

2. Чому для цього завдання краще використовувати PNG, а не JPG?

Завдання 6. Шум, який треба побачити.

Опис завдання: Ви працюєте аналітиком у навчальній лабораторії з кібербезпеки та тестуєте різні приховані канали передавання інформації. Вам передано WAV-файл, який на слух звучить як звичайний фоновий шум. Однак є підозра, що повідомлення приховане не в текстових рядках і не в метаданих, а у візуальному представленні аудіосигналу.

Ваше завдання: Відкрити WAV-файл у Sonic Visualiser, побудувати спектрограму, знайти приховане повідомлення та пояснити, чому звичайний аналіз байтів або прослуховування аудіо не дає очевидного результату. Додатково перевірте файл через `file` та `strings`, щоб порівняти звичайний файловий аналіз із візуальним аналізом спектрограми.

Інструмент	Опис	Приклад
Sonic Visualiser	Програма для аналізу аудіофайлів і перегляду спектрограм	Layer → Add Spectrogram

Під час аналізу зверніть увагу на ділянки спектрограми, де можуть з’являтися літери, символи або короткі слова. Якщо повідомлення не видно одразу, спробуйте змінити колірну схему, контраст або масштаб спектрограми.

Питання для обговорення:

1. Чому повідомлення можна побачити на спектрограмі, але майже неможливо зрозуміти під час звичайного прослуховування?

2. Чому приховане повідомлення видно на спектрограмі, але не видно через `strings`?

Завдання 7. Невидимий пароль.

Опис завдання: Спеціальний агент мав передати пароль, але не міг написати його відкрито. Замість цього він надіслав звичайний TXT-файл із коротким текстом.

У видимому тексті немає жодного пароля, коду чи підказки. Єдина зачіпка – файл містить невидимі Unicode-символи.

Ваше завдання: Проаналізувати TXT-файл, знайти невидимі символи ZWSP та ZWNJ, відновити з них закодовану бінарну послідовність і декодувати приховане повідомлення за допомогою CyberChef.

Рекомендований порядок виконання:

1. Відкрити файл як звичайний текст і переконатися, що видимий текст виглядає безпечно:

```
cat stegotext.txt
```

2. Перевірити байти файлу:

```
hexdump -C stegotext.txt
```

3. Звернути увагу на повторювані послідовності байтів, що є невидимими Unicode-символами:

```
E2 80 8B – ZWSP
E2 80 8C – ZWNJ
```

4. Відкрити файл у CyberChef та додати такі операції:

To Hex

Find / Replace №1 – e2 80 8b → 1

Find / Replace №2– e2 80 8c → 0

Find / Replace №3 – \b[0-9a-fA-F]{2}\b → пусто (видалити всі інші символи)

From Binary

5. Запустити проєкт (Bake) та отримати приховане повідомлення.

Питання для обговорення:

1. Чому візуально TXT-файл може виглядати порожнім або звичайним, але містити приховані дані?

2. Чим приховування через невидимі Unicode-символи відрізняється від Base64 або LSB-стеганографії?