Лабораторна робота № 8(3.1.19)

Продвинуті пошуки

Хід роботи:

Частина 1: Google Advanced Searches (Dorking)

Таблиця операторів Google Advanced Search

ОПЕРАТОР	ОПИС
allintext:	Обмежує результати сторінками з усіма ключовими словами в тексті сторінки
filetype:	Обмежує результати сторінками вказаного типу файлу (.pdf, .ppt, .doc тощо)
intitle:	Обмежує результати сторінками з певним словом (або словами) в заголовку
inurl:	Обмежує результати сторінками з певним словом (або словами) в URL
site:	Обмежує результати сторінками з вказаного домену

Крок 1: Дослідження Google dorking

Базові команди для Google dorking:

Звичайний пошук (без операторів)

ethical hacker

Пошук на конкретному сайті

ethical hacker site:pearson.com

Пошук конкретного типу файлів на сайті

ethical hacker site:pearson.com filetype:pdf

Пошук за словом у заголовку

ethical hacker intitle:certification

Пошук за словом в URL

ethical hacker inurl:free

Пошук всіх слів у тексті сторінки

allintext:free ethical hacker practice test questions

Пошук точної фрази (в лапках)

"**ethical hacker certification**"

Питання: Що спільного у всіх результатах пошуку ethical hacker site:pearson.com?

Відповідь:

Всі результати мають спільне те, що вони:

Походять з домену pearson.com - всі сторінки знаходяться на сайті Pearson

Містять терміни "ethical" та "hacker" - обидва слова присутні на сторінках

Пов'язані з освітніми матеріалами - Pearson є видавцем освітньої літератури

Можуть включати книги, курси, або матеріали про етичний хакінг

Питання: Який тип файлу відкривається кожним з результатів ethical hacker site:pearson.com filetype:pdf?
Відповідь:
Всі результати відкривають **PDF файли** (Portable Document Format). Це можуть бути:
PDF-документи книг про етичний хакінг
Навчальні матеріали та посібники
Зразки розділів або оглядів книг
Презентації або технічна документація
Крок 2: Проведення пошуків через Google Advanced Search форму
Доступ до форми:
В Google пошуку ввести:
advanced search
Або прямий URL:
https://www.google.com/advanced_search
Еквіваленти між формою та операторами:

ФОРМА ADVANCED SEARCH	**ОПЕРАТОР**
all these words	(звичайний пошук)
this exact word or phrase	"точна фраза"
any of these words	OR
none of these words	- (мінус)
site or domain	site:
file type	filetype:
terms appearing	allintext:, intitle:, inurl:

Крок 3: Проведення пасивної розвідки з операторами advanced search
Команди для пасивної розвідки:
1. Пошук адміністративних сторінок
site:examplecompany.com inurl:admin
Варіації:
site:examplecompany.com inurl:administrator
site:examplecompany.com inurl:cpanel
site:examplecompany.com inurl:dashboard
site:examplecompany.com inurl:control
site:examplecompany.com inurl:manage
2. Пошук сторінок входу
site:examplecompany.com intitle:login
Варіації:
site:examplecompany.com intitle:"log in"
site:examplecompany.com intitle:signin
site:examplecompany.com intitle:"sign in"
site:examplecompany.com intitle:authentication
3. Пошук PDF файлів
site:examplecompany.com filetype:pdf
Варіації з різними типами файлів:

site:examplecompany.com filetype:doc
site:examplecompany.com filetype:docx
site:examplecompany.com filetype:xls
site:examplecompany.com filetype:xlsx
site:examplecompany.com filetype:ppt
site:examplecompany.com filetype:pptx
site:examplecompany.com filetype:txt

4. Пошук документів з конкретним текстом

site:examplecompany.com intext:employee filetype:pdf

Варіації:

site:examplecompany.com intext:"confidential" filetype:pdf
site:examplecompany.com intext:"internal use only" filetype:doc
site:examplecompany.com intext:password filetype:xls
site:examplecompany.com intext:"do not distribute" filetype:pdf
site:examplecompany.com intext:salary filetype:xlsx

5. Пошук конфігураційних файлів

site:examplecompany.com filetype:xml inurl:config
site:examplecompany.com filetype:conf
site:examplecompany.com filetype:cnf
site:examplecompany.com filetype:ini
site:examplecompany.com filetype:env

6. Пошук backup файлів

site:examplecompany.com filetype:bak
site:examplecompany.com filetype:backup
site:examplecompany.com filetype:old
site:examplecompany.com ext:sql

7. Пошук інформації про співробітників

site:examplecompany.com intext:"email" filetype:pdf
site:examplecompany.com intext:"phone" filetype:doc
site:examplecompany.com intext:directory filetype:pdf

8. Пошук на LinkedIn

site:linkedin.com intitle:"example company"
site:linkedin.com "example company" "CEO"
site:linkedin.com "example company" "IT Manager"
site:linkedin.com "example company" "Security"

9. Пошук на соціальних мережах

site:facebook.com "example company"
site:twitter.com "example company"
site:instagram.com "example company"
site:github.com "example company"

10. Пошук технічної інформації

site:examplecompany.com intext:"powered by"
site:examplecompany.com intext:"running on"
site:examplecompany.com intext:"version"

Питання: Який тип інформації може отримати хакер з цього типу dork (LinkedIn search)?
Відповідь:
З пошуку LinkedIn хакер може отримати дуже цінну інформацію:
1. Організаційна структура:
- Імена керівників та менеджерів
- Структура відділів (IT, Security, HR, Finance)
- Кількість співробітників у кожному відділі
- Ієрархія організації
2. Інформація про співробітників:
- Повні імена співробітників
- Посади та ролі
- Email адреси (часто у форматі firstname.lastname@company.com)
- Номери телефонів (інколи)
- Дати працевлаштування (скільки років працюють)
3. Технологічний стек:
- Які технології використовує компанія (з навичок співробітників)
- Операційні системи (Windows, Linux, macOS)
- Мови програмування
- Frameworks та tools
- Cloud providers (AWS, Azure, GCP)
4. Проекти та ініціативи:
- Поточні та минулі проекти
- Технологічні трансформації
- Security initiatives
5. Контакти для соціальної інженерії:
- Targets для phishing атак
- Зв'язки між співробітниками
- Контактна інформація
6. Patterns для credential stuffing:
- Формат email адрес (firstname.lastname@company.com)
- Naming conventions для username
Приклад використання:
LinkedIn dorks:
site:linkedin.com "Acme Corporation" "Chief Information Security Officer"
site:linkedin.com "Acme Corporation" "IT" "Administrator"
site:linkedin.com "Acme Corporation" "Network" "Engineer"
site:linkedin.com "Acme Corporation" intitle:"CEO" OR intitle:"CTO" OR intitle:"CISO"
Збір email patterns:
site:linkedin.com "Acme Corporation" "@acme.com"
Пошук нових співробітників (легші цілі):
site:linkedin.com "Acme Corporation" "started new position"
site:linkedin.com "Acme Corporation" "joined" "month"

Частина 2: The Google Hacking Database (GHDB)
Крок 1: Дослідження GHDB головної сторінки
https://www.exploit-db.com/google-hacking-database
Категорії GHDB:
Footholds - точки входу в системи
Files Containing Usernames - файли з іменами користувачів
Sensitive Directories - чутливі директорії
Web Server Detection - виявлення веб-серверів
Vulnerable Files - вразливі файли
Vulnerable Servers - вразливі сервери
Error Messages - повідомлення про помилки
Files Containing Juicy Info - файли з цінною інформацією
Files Containing Passwords - файли з паролями
Sensitive Online Shopping Info - інформація про онлайн-покупки
Network or Vulnerability Data - мережеві дані або дані про вразливості
Pages Containing Login Portals - сторінки з порталами входу
Various Online Devices - різні онлайн-пристрої
Advisories and Vulnerabilities - рекомендації та вразливості
Крок 2: Використання Quick Search для пошуку конкретних dorks
Питання: Яка інформація надається про Dorks?
Відповідь:
Про кожен Dork в GHDB надається наступна інформація:
1. Dork query (пошуковий запит):
- Фактичний Google search query, який можна скопіювати

2. Category (категорія):
- До якої категорії належить dork

3. Date (дата):
- Коли dork був доданий до бази даних

4. Author (автор):
- Хто створив/додав цей dork

5. Description (опис):
- Що знаходить цей dork
- Яку інформацію розкриває
- Потенційні ризики безпеки

6. Google Search button:
- Пряма кнопка для запуску пошуку

7. Sometimes includes:
- Приклади результатів
- Screenshots
- Додаткові пояснення

Крок 3: Вибір категорій для пошуку цікавих Dorks
allinurl:tsweb/default.htm
Питання: Що повертає цей Dork?
Відповідь:

Dork allinurl:tsweb/default.htm повертає:
1. Terminal Services Web Access login pages:
- Сторінки входу для Microsoft Terminal Services
- Remote Desktop Web Access portals
- Windows Server Remote Desktop Services
2. Інформація, яку можна зібрати:
- Версії операційної системи - часто Windows 2000, 2003, 2008 Server
- Версії IIS - Internet Information Services
- Domain names - корпоративні домени
- Server names - імена серверів
- IP addresses - публічні IP адреси
3. Потенційні вразливості:
- Старі версії Windows (Windows 2000 - EOL з 2010)
- Відомі вразливості Terminal Services
- Слабкі конфігурації
- Exposed RDP services
4. Що це дозволяє хакеру:
- Brute force атаки на login портал
- Exploitation відомих вразливостей старих версій
- Enumeration користувачів
- Man-in-the-middle атаки
Приклад використання:
Результат може показати:
- URL: https://company.com/tsweb/default.htm
- Title: "Remote Desktop Web Connection"
- Server info: "Microsoft-IIS/5.0" (indicates Windows 2000)
- Login form з Domain/Username/Password fields
Атакуючий тепер знає:
- Server runs Windows 2000 (vulnerable, EOL)
- RDP is exposed to internet
- Domain name for the company
- Potential target for exploitation
Крок 4: Комбінування фільтрів категорій з пошуковими термінами
Приклади пошуків:
1. Files Containing Passwords + db_pass
Category: Files Containing Passwords
Search: db_pass
Корисні dorks з цієї категорії:
filetype:env "DB_PASSWORD"
filetype:yml "password:"
filetype:config "dbpassword"
filetype:php "mysql_connect"
2. Error Messages
Category: Error Messages

intext:"sql syntax near" | intext:"syntax error has occurred"
intext:"Warning: mysql_connect()"
intext:"Fatal error: Call to undefined function"
3. Login Portals
Category: Pages Containing Login Portals
intitle:"login" inurl:admin
intitle:"index of" "password.txt"
inurl:admin intitle:login
4. Vulnerable Files
Category: Vulnerable Files
filetype:sql intext:password
filetype:log username password email
allintext:"Index of /" +.htaccess
5. Network or Vulnerability Data
Category: Network or Vulnerability Data
filetype:log inurl:"password.log"
inurl:"/phpinfo.php"
ext:php intext:"$_GET" intext:"$_POST"
6. Sensitive Directories
Category: Sensitive Directories
intitle:"index of" "backup"
intitle:"index of" "/.git"
intitle:"Index of" ._history
Детальні приклади корисних dorks:
Пошук exposed backup файлів:
site:example.com ext:sql | ext:backup | ext:bak
site:example.com intitle:"index of" "backup"
filetype:bak inurl:"password"
Пошук конфігураційних файлів:
site:example.com filetype:env
site:example.com ext:conf inurl:"/etc/"
filetype:ini intext:env.production
Пошук credentials:
filetype:xls inurl:"password"
filetype:xlsx username password email
site:example.com intext:"connectionString" filetype:config
Пошук exposed directories:
intitle:"Index of /" +.htpasswd
intitle:"Index of" .ssh
intitle:"index of" "parent directory"
Пошук камер та IoT пристроїв:
inurl:"/view.shtml"
intitle:"webcamXP 5"
inurl:"ViewerFrame?Mode="

Пошук vulnerable веб-додатків:
inurl:"/phpinfo.php" | inurl:"/info.php"
inurl:wp-content/uploads filetype:sql
"Index of" inurl:wp-content/backups
Пошук exposed API keys:
filetype:env "API_KEY"
"api_key" filetype:json
site:github.com "AWS_ACCESS_KEY_ID"
Частина 3: The Wayback Machine
Крок 1: Дослідження бази даних Wayback Machine
Доступ:
URL: https://web.archive.org
або
URL: https://archive.org/web/
Основні функції:
1. Пошук архівованих сторінок
2. Перегляд snapshots з різних часових періодів
3. Порівняння змін між версіями
4. Пошук конкретних файлів та URL
Крок 2: *Дослідження вкладки Calendar*
Функції Calendar tab:
1. Timeline graph - показує частоту crawling
2. Calendar view - конкретні дати snapshots
3. Circles/dots на датах:
   - Синій = є архів
   - Більший = більше snapshots того дня
4. Hover over date - показує кількість captures
Корисні команди для Wayback Machine:
CDX Server API для пошуку:
http://web.archive.org/cdx/search/cdx?url=example.com&output=json
Пошук конкретного URL:
http://web.archive.org/cdx/search/cdx?url=example.com/admin/*&output=json
Фільтр за типом файлу:
http://web.archive.org/cdx/search/cdx?url=example.com&matchType=domain&filter=mimetype:application/pdf
Пошук за датами:
http://web.archive.org/cdx/search/cdx?url=example.com&from=2015&to=2020
Питання: Як може бути вигідним для хакера збір інформації з архівованого сайту?
Відповідь:
Збір інформації з архівованих сайтів може бути надзвичайно вигідним для хакера з наступних причин:
1. Доступ до видаленої або змінної інформації:
Старі контактні дані співробітників (email, телефони)

Технічна документація, яка більше не публікується
API endpoints, які були видалені але можуть залишатися активними
Vulnerable pages, які були виправлені на live сайті
Subdomain names, які більше не видимі
2. Виявлення структури сайту:
Стара архітектура сайту та інфраструктури
Hidden directories та paths
Old admin panels та login pages
Forgotten endpoints (/test/, /dev/, /staging/)
Removed features, які можуть залишатися функціональними
3. Credentials та sensitive data:
Hardcoded passwords в JavaScript або HTML коментарях
API keys в old source code
Database connection strings
Old config files, які не були приховані належним чином
Email addresses співробітників для phishing
4. Технологічні деталі:
Old software versions з відомими вразливостями
Frameworks and libraries з CVE
Server information (Apache/IIS versions, PHP versions)
Third-party integrations та services
5. Соціальна інженерія:
Company history та milestones
Personnel information (імена, посади, проекти)
Press releases з чутливою інформацією
Old job postings, які розкривають технології
6. Business intelligence:
Product roadmaps та development plans
Financial information з old reports
Partnership information
Client lists та testimonials
Практичні приклади:
Приклад 1: Знаходження старих admin panels
На archive.org знайдено:
https://web.archive.org/web/20150301000000/example.com/admin/
Перевірити чи цей endpoint ще існує:
curl https://example.com/admin/
Якщо так - potential entry point!
Приклад 2: Витік API keys в old JavaScript
В архіві з 2018 знайдено:

```
<script>
 var apiKey = "sk_live_abc123xyz789";
 var endpoint = "https://api.example.com/v1/";
</script>
```

Перевірити чи key ще валідний:
curl -H "Authorization: Bearer sk_live_abc123xyz789"
https://api.example.com/v1/users
Приклад 3: Old employee emails для phishing
В архіві з 2016 знайдено:
Contact: John Smith - j.smith@company.com
Contact: Sarah Johnson - sarah.j@company.com
Використати для targeted phishing або credential stuffing
Приклад 4: Vulnerable software version
В старому robots.txt або comments:
WordPress 4.7.0 - vulnerable to REST API vulnerability
CMS: Joomla 3.4.5 - multiple known CVEs
***Крок 3:** Дослідження вкладки Collection*
Функції Collections tab:
1. Shows different crawl sources:
   - Internet Archive Bot
   - Alexa Crawls
   - Common Crawl
   - Специфічні колекції
2. Timeline по місяцях:
   - Jan-Dec показують активність
   - Кількість captures
3. Information about crawlers:
   - Who runs them
   - Crawl frequency
   - Coverage

Крок 4: Дослідження вкладки Changes
Функції Changes tab:
1. Візуалізація змін між captures:
   - Grey = мало змін
   - Blue = значні зміни
2. Comparison tool:
   - Вибрати 2 captures
   - Натиснути "Compare"
   - Highlights що змінилося
3. Useful для:
   - Tracking content changes
   - Finding removed information
   - Detecting security patches

Крок 5: Дослідження вкладки Summary
Функції Summary tab:
1. MIME type breakdown:
   - text/html
   - image/jpeg, image/png

- application/pdf
- application/javascript
- video/mp4
- audio/mpeg

2. Date range selector:
   - Year Start
   - Year End
3. Data type buttons:
   - All
   - text
   - application
   - image
   - message
   - audio
   - video

Крок 6: Дослідження вкладки Site Map

Функції Site Map tab:

1. Visual representation:
   - Center circle = root
   - Rings = page depth
   - Further from center = more complex
2. Complexity over time:
   - Click through years
   - See site growth
3. Interactive:
   - Click rings/cells
   - Open archived pages

Крок 7: Дослідження вкладки URLs

Команди для пошуку через URLs tab:

Фільтри для пошуку цікавих файлів:

1. Backup files

*.bak
*.backup
*.old
*.tmp

2. Archive files

*.zip
*.rar
*.tar
*.gz
*.7z

3. Configuration files

*.config
*.conf

*.cfg
*.ini
*.env
4. Database files
*.sql
*.db
*.sqlite
*.mdb
5. Spreadsheets and documents
*.csv
*.xls
*.xlsx
*.doc
*.docx
*.pdf
6. Code files
*.php~
*.inc
*.asp
*.aspx
7. Admin paths
/admin/
/administrator/
/cpanel/
/manage/
8. API paths
/api/
/v1/
/v2/
/rest/
9. Development paths
/dev/
/test/
/staging/
/beta/
10. Hidden files
.htaccess
.htpasswd
.git
.svn

Додаткові техніки з Wayback Machine:

Wayback Machine CDX API для автоматизації:

Отримати всі URLs для домену:

curl "http://web.archive.org/cdx/search/cdx?url=example.com/*&output=json" > urls.json

Фільтрувати по типу файлу:

curl "http://web.archive.org/cdx/search/cdx?url=example.com/*&filter=mimetype:application/pdf&output=json"

Пошук за ключовими словами в URL:

curl "http://web.archive.org/cdx/search/cdx?url=example.com/*&filter=original:.*admin.*&output=json"

Пошук за датами:

curl "http://web.archive.org/cdx/search/cdx?url=example.com/*&from=20150101&to=20151231&output=json"

Обмежити кількість результатів:

curl "http://web.archive.org/cdx/search/cdx?url=example.com/*&limit=100&output=json"

Скрипт для автоматизованого аналізу:

!/bin/

wayback_recon.sh

DOMAIN=$1

OUTPUT_DIR="wayback_${DOMAIN}"

mkdir -p $OUTPUT_DIR

echo "[+] Fetching all URLs from Wayback Machine..."

curl -s "http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&output=json" > ${OUTPUT_DIR}/all_urls.json

echo "[+] Filtering interesting files..."

Backup files

curl -s "http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&filter=original:.*\\.bak&output=json" > ${OUTPUT_DIR}/backup_files.json

Config files

curl -s "http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&filter=original:.*\\.config&output=json" > ${OUTPUT_DIR}/config_files.json

SQL files

curl -s "http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&filter=original:.*\\.sql&output=json" > ${OUTPUT_DIR}/sql_files.json

Admin pages

curl -s "http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&filter=original:.*admin.*&output=json" > ${OUTPUT_DIR}/admin_pages.json

API endpoints

```
curl -s
"http://web.archive.org/cdx/search/cdx?url=${DOMAIN}/*&filter=original:.*/api/.*&o
utput=json" > ${OUTPUT_DIR}/api_endpoints.json
echo "[+] Done! Results in $OUTPUT_DIR/"
```

Корисні інструменти та ресурси

Інструменти для автоматизації Google Dorking:

1. Pagodo (Passive Google Dork)

```
git clone https://github.com/opsdisk/pagodo.git
cd pagodo
python3 pagodo.py -d example.com -g dorks.txt
```

2. GooDork

```
go get github.com/dwisiswant0/goodork
goodork -q "site:example.com" -p 5
```

3. Dork-cli

```
npm install -g dork-cli
dork -s "site:example.com filetype:pdf"
```

4. GooFuzz (Google Dorking + Fuzzing)

```
git clone https://github.com/m3n0sd0n4ld/GooFuzz
python3 GooFuzz.py -t example.com
```

5. DorkScout

```
git clone https://github.com/R4yGM/dorkscout
python dorkscout.py -d example.com
```

Інтеграція з іншими інструментами:

Комбінування з Subfinder для субдоменів:

```
subfinder -d example.com -silent | while read sub; do
    echo "Dorking $sub..."
    googler --json -n 20 "site:$sub filetype:pdf" >> results.json
done
```

Комбінування з Wayback Machine:

```
waybackurls example.com | grep -E "\\.pdf$|\\.doc$|\\.xls$" > interesting_files.txt
```

Експорт результатів у таблицю:

```
cat results.json | jq -r '.[] | [.url, .title, .abstract] | @csv' > results.csv
```

Reflection Question (Питання для рефлексії)
Питання: Чому пасивна розвідка настільки важлива для ефективного хакінгу та тестування на проникнення?
Відповідь:
Пасивна розвідка є критично важливою для ефективного хакінгу та пентестингу з наступних причин:
1. Stealth та Незручність виявлення:
- Не залишає слідів: Пасивна розвідка не взаємодіє безпосередньо з цільовою системою, тому не залишає логів, alertів або інших слідів у системах моніторингу
- Легальність: Використання публічної інформації є повністю легальним
- Безпека тестера: Мінімізує ризик правових наслідків або виявлення
- Не тригерить IDS/IPS: Системи виявлення вторгнень не можуть детектувати пасивну розвідку
2. Максимізація інформації за мінімальних зусиль:
- 80/20 principle: До 80% інформації, необхідної для успішної атаки, можна зібрати пасивно
- Ефективність часу: Автоматизовані інструменти можуть зібрати величезні обсяги даних за годину
- Low-hanging fruit: Виявлення очевидних вразливостей без active scanning
- Cumulative effect: Невеликі фрагменти інформації складаються в повну картину
3. Направляє Active Reconnaissance:
- Targeted approach: Пасивна розвідка визначає, що саме сканувати активно
- Зменшення noise: Менше непотрібних сканувань = менше шансів бути виявленим
- Optimized resources: Фокус на найбільш promising targets
- Attack surface mapping: Чітке розуміння всієї поверхні атаки
4. Виявлення слабких місць організації:
- Forgotten assets: Старі сервери, субдомени, які більше не підтримуються
- Shadow IT: Системи, про які IT відділ може не знати
- Third-party risks: Витоки через партнерів, contractors
- Human factor: Email адреси для phishing, інформація про співробітників
5. Social Engineering Foundation:
- Personnel information: Імена, посади, контакти співробітників
- Organizational structure: Хто кому підзвітний, key decision makers
- Company culture: З LinkedIn, соціальних мереж
- Current events: Злиття, поглинання, реорганізації
- Technical stack: Які технології використовуються (для pretexting)
6. Compliance та Legal Protection:
- Scope definition: Розуміння меж тестування
- Client validation: Перевірка, що клієнт дійсно володіє assets
- Documentation: Збір baseline information для звітів
- Risk assessment: Оцінка potential impact перед active testing
7. Виявлення критичних вразливостей:
- Data breaches: Compromised credentials через HIBP

- Exposed credentials: Паролі в GitHub, Pastebin
- Sensitive documents: Confidential info в PDF, DOC файлах
- Configuration leaks: API keys, database credentials
- Old vulnerabilities: Unpatched systems виявлені через version disclosure

8. Cost-Effectiveness:
- Free resources: Більшість пасивних інструментів безкоштовні
- Minimal infrastructure: Не потрібні потужні сервери для сканування
- Scalability: Можна досліджувати множину targets одночасно
- Repeatability: Можна повторювати без ризику detection

Практичний робочий процес:

Етап 1: PASSIVE RECON (1-3 дні)
```
├── Google Dorking
├── OSINT (LinkedIn, соціальні мережі)
├── DNS enumeration (пасивне)
├── Certificate transparency logs
├── Wayback Machine
├── Data breach checks
└── Metadata analysis
```
Аналіз зібраної інформації

Етап 2: SEMI-PASSIVE RECON (1-2 дні)
```
├── DNS queries
├── WHOIS lookups
├── Shodan/Censys searches
└── Public vulnerability databases
```
Виявлення цінних цілей

Етап 3: ACTIVE RECON (targeted)
```
├── Port scanning (тільки identified targets)
├── Service enumeration
├── Vulnerability scanning
└── Application testing
```
Експлойти виявлено

Статистика ефективності:

Згідно з досліджень та практикою:

70-80% успішних атак починаються з пасивної розвідки

60% вразливостей можуть бути виявлені без active scanning

90% phishing campaigns використовують інформацію з OSINT

Час: Пасивна розвідка = 20% часу, але надає 80% корисної інформації

Приклади успішних атак через пасивну розвідку:

Приклад 1: Exposed credentials

Google Dork виявив:

site:company.com filetype:env "DB_PASSWORD"

- Знайдено .env файл з database credentials
- Direct database access без сканування

Приклад 2: Forgotten subdomain
Certificate Transparency log показав:
dev.company.com (issued 2018, expired 2019)

- Subdomain ще активний але не оновлюється
- Old WordPress з known vulnerabilities

Приклад 3: Employee targeting
LinkedIn OSINT виявив:
John Smith - IT Administrator (працює 6 місяців)

- New employee, potentially less security aware
- Targeted phishing email
- Credential compromise

Приклад 4: API keys in GitHub
GitHub search знайшов:
site:github.com "company.com" "api_key"

- Old repository з hardcoded API key
- Key ще валідний
- Unauthorized API access

Пасивна розвідка є **foundation** будь-якої успішної атаки або пентесту. Вона:

- Мінімізує ризики виявлення
- Максимізує інформаційний output
- Направляє всі подальші етапи
- Виявляє low-hanging fruit
- Забезпечує context для соціальної інженерії
- Є cost-effective та scalable

Правило пентестингу: "Never rush into active reconnaissance without exhaustive passive reconnaissance first."

Висновок

У ході виконання лабораторної роботи було детально досліджено методи пасивної розвідки через Google Advanced Search (Google Dorking), Google Hacking Database (GHDB) та Wayback Machine. Google Dorking продемонстрував потужні можливості пошуку конфіденційної інформації, яка ненавмисно стала публічною, через використання спеціалізованих операторів (site:, filetype:, intitle:, inurl:, allintext:). GHDB надав структуризовану базу готових dorks для виявлення вразливостей, exposed credentials, admin panels та sensitive documents. Wayback Machine показав цінність історичних архівів сайтів для знаходження видаленої інформації, старих endpoints, forgotten subdomains та технічних деталей. Практична робота підкреслила критичну важливість пасивної розвідки як фундаменту для будь-якого пентесту - вона дозволяє зібрати до 80% необхідної інформації без прямої взаємодії з цільовою системою, залишаючись невидимою для систем моніторингу та захисту. Організації повинні регулярно проводити self-dorking та перевіряти свої архівні дані, щоб виявити та усунути потенційні витоки інформації до того, як їх знайдуть зловмисники.