웹 해킹 및 정보보안 실습

Chapter 1. 실습환경 구축 & Chapter 2. OWASP Top 10

학습 핵심 키워드 (Keyword)

본 과정에서 주로 다루게 될 핵심 개념들을 미리 확인합니다.

주요 키워드 개요

#WebGoat

취약성이 의도적으로 포함된 애플리케이션으로, 실제 해킹 공격을 테스트할 수 있는 실습 대상입니다.

#Docker Container

PC 환경에 구애받지 않고 언제 어디서나 동일한 실습 환경을 실행할 수 있도록 도와주는 플랫폼입니다.

#OWASP Top 10

웹 애플리케이션 보안 취약점 중 가장 영향력이 큰 10가지를 선정한 글로벌 보안 지표입니다.

Chapter 1. 실습환경 구축

모의 해킹을 위한 웹 해킹의 실습 환경을 조성해 보고 구성 요소들을 상세히 알아봅니다.

Section 01. 실습 프로그램 설치 과정

- **학습 목표:** 웹 해킹의 안전한 실습 환경을 로컬 PC에 구축합니다.
- **핵심 요소 1:** 실습용 웹 애플리케이션인 WebGoat 코드를 준비합니다.
- **핵심 요소 2:** 애플리케이션을 가동하기 위한 Docker Desktop을 설치합니다.
- **기대 효과:** 복잡한 서버 구성 없이 컨테이너 기반으로 즉시 실습에 돌입할 수 있습니다.

01. WebGoat와 Docker Desktop

실습 환경 구성 요소에는 실습용 웹 애플리케이션인 **WebGoat**와 이를 실행할 **Docker**가 있습니다.

사용자마다 PC 환경이 달라 발생할 수 있는 예상치 못한 오류를 방지하기 위해, Docker를 이용해 격리된 실습용 Ubuntu 컨테이너를 생성하고 그 안에서 WebGoat를 동작시킵니다.

WebGoat의 주요 특징

- **취약성 내재:** 흔히 발견되는 취약성을 테스트하도록 의도적으로 불안정하게 개발되었습니다.
- **개발 기반:** Java Spring Boot 프레임워크를 바탕으로 만들어졌습니다.
- **편의성:** 직접 애플리케이션을 개발할 필요 없이 손쉽게 공격 기법을 수행해 볼 수 있습니다.
- **오픈소스:** 코드가 공개되어 있어 취약점을 직접 조치해 보는 방어 실습에도 용이합니다.

윈도우에서 Docker Desktop 설치 (1)

01

공식 접속

구글 크롬을 실행해 검색란에 'Docker'를 입력하고 검색합니다. 결과 화면에서 상단의 Docker 공식 홈페이지(www.docker.com)에 접속합니다.

홈 화면 상단 메뉴 바에서 **[Products]** → **[Docker Desktop]**을 차례로 클릭합니다.

윈도우에서 Docker Desktop 설치 (2)

02

파일 다운

Docker Desktop 화면의 다운로드 버튼에 마우스를 올리면 운영체제별 다운로드 항목이 나타납니다.

이 중에서 **[Download for Windows]** 버튼을 클릭하여 설치용 실행 파일을 다운로드하고 실행합니다.

윈도우에서 Docker Desktop 설치 (3)

설치 파일을 실행하면 'Installing Docker Desktop' 창과 함께 Configuration(구성) 설정이 나타납니다.

여기서 가장 중요한 점은 **[Use WSL 2 instead of Hyper-V (recommended)]** 항목의 선택을 반드시 **해제**한 후 [OK]를 클릭하는 것입니다.

[잠깐!] WSL2 옵션을 해제하는 이유

- **WSL이란?** 윈도우 운영체제의 가상화 기능을 활용해 Linux를 바로 사용하는 기술입니다.
- **오류 변수:** 윈도우 버전이나 업데이트 상태에 따라 WSL2 설치가 실패하거나 정상 동작하지 않는 경우가 종종 발생합니다.
- **안정성 최우선:** 예상치 못한 윈도우 의존적 오류로 실습이 중단되는 것을 막기 위해, 보다 안정적인 기존 가상화 방식인 **Hyper-V**를 사용하도록 설정합니다.

윈도우에서 **Docker Desktop** 설치 완료

- 대화상자에 'Unpacking files...'가 보이면 정상적으로 설치가 진행 중인 것입니다.
- 잠시 후 **'Installation succeeded'** 메시지가 나타나면 설치가 무사히 완료된 것입니다.
- **[Close and restart]** 버튼을 클릭하여 윈도우를 재시작한 뒤, Docker Desktop을 실행할 준비를 마칩니다.

03. 맥 OS에서 Docker Desktop 설치방법

- Mac용 설치를 위해 다운받은 .dmg 파일을 더블클릭하여 마운트합니다.
- 나타나는 창에서 Docker 아이콘을 Applications(응용 프로그램) 디렉토리로 드래그 앤 드롭하여 복사합니다.
- SpotLight 또는 Launch pad를 실행한 후 'Docker'를 검색해 실행합니다.
- 인터넷에서 다운로드된 앱 경고창이 나타나면 **[열기]**를 클릭해 승인합니다.

04. Docker Desktop 초기 실행 및 설정

OK

약관 동의

처음 실행 시 나타나는 이용 약관(Subscription Service Agreement) 대화상자에서 **[Accept]** 버튼을 클릭합니다.

이어서 나타나는 사용자 정보 수집 설문(Tell us about the work you do)은 작성할 필요 없이 하단의 **[Skip]**을 눌러 넘깁니다.

05. 실습용 컨테이너 실행 준비

이제 GUI가 아닌 명령어 기반의 **CLI (Command Line Interface)** 환경을 이용해 실습용 컨테이너를 가동합니다.

윈도우 하단 검색창에서 **'powershell'**을 검색한 뒤, 반드시 **[관리자로 실행]**을 눌러 터미널을 오픈합니다. (Mac은 터미널 앱 실행)

컨테이너 가동 명령어 입력

Windows PowerShell 터미널이 열리면, 다음 명령어를 띄어쓰기에 주의하여 정확히 입력합니다.

```
PS C:\> docker run -it --name webgoat -p 8080:8080 -p 9090:9090 ubuntu
```

- 로컬에 이미지가 없다면 `Pulling from library/ubuntu` 메시지와 함께 다운로드를 시작합니다.
- 완료 후 프롬프트가 `root@컨테이너ID:/#` 로 변경되며 컨테이너 내부로 접속됩니다.

Docker 명령어 상세 분석

- **docker run:** 컨테이너를 새롭게 생성하고 즉시 실행합니다.
- **-it 옵션:** 컨테이너 입력 설정(-i)과 터미널 쉘 접근(-t)을 합쳐, 실시간으로 명령어를 칠 수 있게 합니다.
- **--name webgoat:** 관리하기 쉽도록 생성할 컨테이너의 이름을 'webgoat'로 명명합니다.
- **-p 8080:8080:** 호스트 PC의 특정 포트(앞)를 컨테이너 내부 포트(뒤)와 포트포워딩하여 연결합니다.
- **ubuntu:** 컨테이너의 바탕이 되는 기본 운영체제 이미지입니다.

[TIP] Docker 메모리 부족 오류 해결법

"Not enough memory to start Docker Desktop"

Docker 엔진이 요구하는 메모리에 비해 현재 PC의 가용 시스템 메모리(RAM)가 부족할 때 발생하는 에러입니다. Docker 설정에서 리소스 사용 상한선을 낮추면 해결됩니다.

메모리 제한 설정 변경 단계

- 윈도우 우측 하단 작업표시줄의 [숨겨진 아이콘 표시]를 클릭하여 Docker 아이콘을 찾습니다.
- 아이콘을 우클릭한 후 메뉴에서 **[Settings]**를 클릭합니다.
- Settings 화면 좌측에서 **[Resources]** 탭을 클릭합니다.
- **[Memory]** 영역 슬라이드 바를 본인 PC 가용 RAM에 맞춰 무리가 안 가는 선으로 조절한 뒤 **[Apply & Restart]**를 누릅니다.

[TIP] Hyper-V 관련 오류 해결법

"Hardware assisted virtualization and data execution protection must be enabled..." 오류 발생 시 관리자 권한 PowerShell에 아래 명령을 입력하여 설정을 auto로 바꿉니다.

```
PS C:\> bcdedit /set hypervisorlaunchtype auto
```

재부팅 후에도 실패한다면, 아래 명령어로 윈도우 기능에서 Hyper-V를 강제로 활성화합니다.

```
PS C:\> Enable-WindowsOptionalFeature -Online -FeatureName
Microsoft-Hyper-V-All
```

Section 02. 실습 프로그램 구축 과정

다운로드한 Docker 컨테이너 위에서 소스코드를 받아오고,
WebGoat를 직접 빌드하여 실행하는 방법을 다룹니다.

01. WebGoat 소스코드 준비

- 구글 크롬에서 'webgoat github'를 검색하여 공식 레포지토리(Repository)에 접속합니다.
- 좌측의 [Switch branches/tags] 메뉴를 열고, 기본 'main branch'를 **v2023.3** 태그로 변경합니다. (버전 업데이트에 따른 실습 오류 예방 목적)
- **[Code]** 버튼을 클릭하여 제공되는 HTTPS Clone 주소를 클립보드에 복사해 둡니다.

컨테이너 패키지 매니저 세팅

Ubuntu 쉘에서 패키지 목록을 최신화하기 위해 `apt update` 명령어를 먼저 실행합니다.

그다음 소스코드 복사용 Git과 파일 수정을 위한 Vim 에디터를 설치합니다.

```
root@...:/# apt install -y git vim
```

Java (OpenJDK 17) 설치

JDK

Version 17

WebGoat는 Java Spring Boot 환경(JDK 17)으로 작성되었습니다. 따라서 컨테이너 내부에 Java 개발 키트를 설치합니다.

```
root@...:/# apt install -y openjdk-17-jdk
```

설치 직후 `java -version` 명령어를 입력하여 17 버전이 맞게 설치되었는지 확인합니다.

소스코드 복사 (Git Clone)

루트(root) 계정의 홈 디렉터리로 이동한 후, 복사해 둔 주소와 특정 버전(-b v2023.3)을 지정해 소스를 다운받습니다.

```
root@...:/# cd ~

     root@...:~# git clone -b v2023.3
https://github.com/WebGoat/WebGoat.git
```

- 작업이 완료되면 `cd WebGoat` 명령어로 디렉터리에 진입합니다.
- `git describe`를 쳐서 **v2023.3**이 정확히 나오는지 버전을 확인합니다.

WebGoat 접근 허용 설정 변경

기본 설정은 컨테이너 내부(127.0.0.1)만 접근이 허용되어 호스트 PC에서 접속이 불가합니다. 속성 파일을 편집해 외부 개방을 해줍니다.

```
root@...:~/WebGoat# vi src/main/resources/application-
webgoat.properties
```

- 에디터 안에서 `webgoat.host=${WEBGOAT_HOST:127.0.0.1}` 줄을 찾아 앞에 #을 붙여 주석 처리합니다.
- 그 아래 줄에 `webgoat.host=0.0.0.0` 을 수동으로 추가하고 저장합니다.

WebWolf 접근 허용 설정 변경

보조 유틸리티인 WebWolf도 동일한 위치에 있는 설정 파일을 열어 접근 권한을 0.0.0.0으로 개방해주어야 합니다.

```
root@...:~/WebGoat# vi src/main/resources/application-
webwolf.properties
```

- webwolf.host=${WEBWOLF_HOST:127.0.0.1} 줄을 찾아 #으로 주석 처리합니다.
- 그 아래에 webwolf.host=0.0.0.0 을 명시한 뒤 :wq를 입력해 저장 및 종료합니다.

Maven Wrapper를 이용한 자동 빌드

로컬에 Maven 환경이 없더라도, 소스에 기본 포함된 **mvnw (Maven Wrapper)**를 실행하여 의존성을 설치하고 소스를 빌드할 수 있습니다.

```
./mvnw clean install
```

- 라이브러리 다운로드와 컴파일이 차례로 진행됩니다.
- 터미널 로그 마지막 줄에 **BUILD SUCCESS** 문구가 출력되면 준비가 완료된 것입니다.

WebGoat 서비스 실행

RUN

Spring Boot

성공적으로 빌드된 애플리케이션을 구동하기 위해 아래 명령어를 입력합니다.

```
./mvnw spring-boot:run
```

출력되는 긴 로그 중에 **"Started StartWebGoat in ..."**라는 메시지가 등장하면 웹 서버가 정상적으로 켜진 것입니다.

02. 실습환경 웹 접속 및 회원가입

- **웹사이트 접속:** 호스트 PC의 브라우저 주소창에 `http://localhost:8080/WebGoat`를 입력하여 이동합니다.
- **회원가입 진입:** 나타난 로그인 화면 아래의 **[or register yourself as a new user]** 링크를 클릭합니다.
- **계정 생성 완료:** 임의의 Username과 Password를 넣고, 약관 동의 체크 후 **[Sign up]**을 눌러 실습 계정을 생성합니다.

WebGoat 및 WebWolf 연동 확인

WebGoat 포털

생성한 계정으로 로그인 시 'What is WebGoat?' 문구와 함께 좌측에 해킹 실습 카테고리가 뜨면 정상입니다.

WebWolf 확인

새 브라우저 탭에 `http://localhost:9090`을 입력해 접속 시 WebWolf 홈 화면이 보인다면 완벽히 성공한 것입니다.

[TIP] 종료된 컨테이너 재실행 (CLI & GUI)

실습을 exit로 마친 후 나중에 컨테이너를 재가동하여 접속하는 방법입니다.

- **CLI 방법:** 터미널에서 `docker ps -a`로 목록을 확인하고 `docker start webgoat`로 서버를 구동합니다. 이어서 `docker attach webgoat`를 치면 쉘로 접속됩니다.
- **GUI 방법:** Docker Desktop의 'Containers' 메뉴에서 webgoat 우측 **[▶ Play]** 버튼을 누른 뒤, 터미널에서 `docker attach webgoat` 명령어로 쉘에 진입합니다.

Chapter 2. OWASP Top 10

본격적인 실습 전, 웹 해킹 취약점의 핵심 종류와 공격 트렌드를
10가지 순위를 통해 명확히 파악합니다.

Section 01. OWASP Top 10 이란?

- **주체:** Open Web Application Security Project의 약자로, 취약점을 연구하는 비영리 보안 재단입니다.
- **핵심 지표:** 3~4년 주기로 고위험 취약점의 공격 가능성과 파급력을 분석하여 상위 10개를 공식 발표하며 이는 글로벌 표준이 됩니다.
- **트렌드 반영:** WebGoat 실습 환경 역시 최신 표준인 **2021 OWASP Top 10**을 기준으로 코스웍이 구성되어 있습니다.

2017 vs 2021 순위 주요 차이점

2021년도 발표에서 가장 눈여겨볼 변화점 4가지입니다.

- 절대적 1위였던 **Injection**이 3위로 소폭 하락했습니다.
- 개별 항목이던 **XSS**가 Injection 내부로 흡수되었습니다.
- **XXE** 항목이 Security Misconfiguration으로 흡수되었습니다.
- **Insecure Deserialization**이 Software and Data Integrity Failures로 통합되었습니다.

OWASP Top 10 (1위)

Broken Access Control (부적절한 인가)

1st

2017년 5위에서 2021년 **가장 치명적인 1위**로 등극한 취약점입니다.

사용자의 인가 여부를 결정하는 접근제어 로직에 발생하는 결함입니다. 예측 불가한 난수 세션/토큰을 쓰지 않고, 조작이 쉬운 **HTTP 요청 헤더나 파라미터 값만으로 권한을 검증**하려다 주로 발생합니다.

프레임워크 발전으로 기술적 취약점은 줄었으나, 개발자의 구현 단계 실수로 인해 빈번히 발생합니다.

OWASP Top 10 (2위)

Cryptographic Failures (암호화 문제)

2nd

2017년 3위였던 '민감한 정보 노출' 항목이 결과 중심에서 원인 중심 명칭으로 변경되며 2위로 상승했습니다.

하드웨어 연산 속도의 폭발적 발전으로 과거의 **오래된 암호화 알고리즘이 쉽게 파훼**되고 있습니다.

하지만 여전히 수많은 서비스들이 취약한 암호화 알고리즘을 사용하며 개선할 의지가 없어 빈번히 문제가 발생하고 있습니다.

OWASP Top 10 (3위)

3rd

Injection (인젝션)

오랜 기간 부동의 1위였던 인젝션이 3위로 내려왔습니다.

가장 큰 변화는 단독 항목이던 **XSS (Cross-Site Scripting)가 Web Frontend code Injection으로 해석되어 이곳으로 편입**된 것입니다.

순위가 하락한 이유는 최근 개발 프레임워크의 고도화와 **ORM(Object Relational Mapping) 등의 보편화**로 인해 기본 보안 모듈의 자체 방어가 훌륭해졌기 때문입니다.

OWASP Top 10 (4위)

Insecure Design (안전하지 않은 설계)

4th

2021년 신규 추가된 항목. **설계 초기 단계부터 보안을 신경 쓰지 않아** 발생하는 구조적 결함을 뜻합니다.

Shift Left (원점 회귀) 보안 철학: 개발의 초기 단계가 모든 취약점의 시작점입니다. 초기 단계를 간과하면 추후 배포 단계에서 걷잡을 수 없게 됩니다.

운영 중인 서비스의 취약점을 뒤늦게 조치하고 재배포할 경우, 서비스 가용성이나 실시간성에 매우 치명적인 장애를 유발할 위험이 있습니다.

OWASP Top 10 (5위)

Security Misconfiguration (보안 설정 미흡)

5th

2017년 6위에서 5위로 상승했습니다. 이름 그대로 보안과 관련한 설정이 미흡하여 발생하는 취약점입니다.

개발자가 편의를 위해 사용하던 **테스트 용도의 느슨한 설정을 운영용 서비스에 그대로 반영**하는 실수가 실무에서 매우 흔하게 발생합니다.

과거 단독 항목이었던 **XXE (XML External Entities)** 역시 외부 Entity 참조 설정을 잘못하여 발생하므로 이 항목으로 병합되었습니다.

OWASP Top 10 (6위)

Vulnerable and Outdated Components

6th

이미 취약점이 공개되었거나 공식 지원이 만료된 (EOL/EOS) 구형 컴포넌트를 사용할 때 발생합니다.

해커는 서비스에서 취약점이 노출된 라이브러리나 컴포넌트를 발견하게 되면 **조건 없이 즉각적인 공격을 수행**합니다.

예방을 위해 사용 중인 컴포넌트의 취약점 여부를 지속적으로 확인하고 패치 작업 및 기한 관리를 수행해야 합니다.

OWASP Top 10 (7위)

Identification and Authentication Failures

7th

과거 2위였던 '불충분한 인증' 항목으로, 로그인, 2차 인증, 게시물 비밀번호 인증 등의 로직이 우회되어 발생하는 취약점입니다.

여전히 필수로 지켜져야 할 항목이지만, **표준화된 프레임워크들의 강력한 기본 방어 지원** 덕분에 자체 발생 빈도가 줄어 7위로 대폭 하락했습니다.

OWASP Top 10 (8위)

Software and Data Integrity Failures

8th

2021년 신규 진입. 소프트웨어나 데이터를 **무결성 검증 과정 없이 신뢰**하여 발생합니다.

네트워크에서 공격당한 변조된 라이브러리를 무결성 체크 없이 설치하는 것이 대표적입니다.

과거 단독 항목이던 **Insecure Deserialization (불안전한 데이터 역직렬화)**가 여기에 흡수되었습니다. 조작된 직렬화 데이터를 그대로 역직렬화할 경우 원격 코드가 실행되는 등 치명적인 피해를 입습니다.

OWASP Top 10 (9위)

9th

Security Logging and Monitoring Failures

2017년 10위에서 9위로 상승했습니다. 시스템 보안 로깅과 모니터링 체계가 없거나 부족할 때 발생하는 위험입니다.

침해 사고가 발생했을 때, 로깅 정보가 없으면 해커의 경로 분석 및 피해 파악이 불가능해져 **사실상 사후 대응이 완전히 마비**되는 심각한 상황을 초래합니다.

OWASP Top 10 (10위)

Server Side Request Forgery (SSRF)

10th

2021년에 신규 추가됨. 클라이언트가 아닌 타겟 서버가 직접 내부의 다른 서버로 변조된 요청을 보내게 만드는 고도화된 기법입니다.

과거엔 내부 정보가 부족해 까다로웠으나, **온프레미스에서 클라우드 환경(AWS 등)으로 전환**이 보편화되며 공격 난이도가 낮아졌습니다.

특히 공개된 내부 메타데이터 서비스(IMDS) URI 등을 노려 서버 메타데이터를 탈취하는 사고가 빈번합니다.

수고하셨습니다.

OWASP Top 10의 생소한 보안 용어가 많더라도
실습을 통해 차근차근 구조를 파악해나가면 됩니다.

본격적인 해킹 방어 실습으로 넘어가 봅시다!