

수업 내용 정리 보고서

작성일: 2026년 7월 2일

챕터 구성

- 웹 보안 표준(OWASP Top 10) 및 핵심 공격 기술의 이해
- WebGoat 실습 환경 구축 및 시스템 보안 설정 검증
- 웹 보안 방어 전략 및 실무 운영 점검 가이드

웹 보안 표준(OWASP Top 10) 및 핵심 공격 기술의 이해

웹 애플리케이션 보안을 학습하는 과정에서 가장 먼저 접하게 되는 것은 **OWASP(Open Web Application Security Project)**입니다. OWASP는 2001년 설립된 비영리 오픈소스 웹 보안 커뮤니티로, 전 세계 보안 전문가들이 참여하여 실제 발생 빈도가 높고 치명적인 웹 보안 위협을 정리하여 발표합니다. 이들이 발표하는 "**OWASP Top 10**"은 단순히 위협 목록을 넘어, 전 세계 기업의 보안 감사(Audit)와 개발자 보안 교육의 국제 표준(Industry Standard)으로 활용됩니다.

1. OWASP Top 10의 진화와 최신 트렌드

웹 보안 위협은 기술의 발전과 함께 계속해서 변화합니다. 따라서 OWASP Top 10은 약 3~4년 주기로 업데이트되며, 최신 공격 트렌드(예: 클라우드, AI, 공급망 공격)를 반영합니다. 2017년, 2021년, 그리고 최신 2025년판의 주요 변화를 비교하면 다음과 같습니다.

순번	2017년판	2021년판	2025년판 (최신)
1	Injection	Broken Access Control	Broken Access Control
2	Broken Authentication	Cryptographic Failures	Security Misconfiguration (↑3)
3	Sensitive Data Exposure	Injection	Software Supply Chain Failures (신규)
4	XML External Entities (XXE)	Insecure Design (신규)	Cryptographic Failures (↓2)
5	Broken Access Control	Security Misconfiguration	Injection
6	Security Misconfiguration	Vulnerable and Outdated Components	Insecure Design (↓2)
7	XSS	Auth Failures	Authentication Failures
8	Insecure Deserialization	Software and Data Integrity Failures (신규)	Software or Data Integrity Failures
9	Known Vulnerable Components	Logging & Monitoring Failures	Logging & Alerting Failures
10	Insufficient Logging & Monitoring	SSRF (신규)	Mishandling of Exceptional Conditions (신규)

위 표를 통해 알 수 있듯이, 최근 보안의 중심은 단순한 코드 버그(Injection)에서 **설계 결함(Insecure Design)**, **공급망 보안(Supply Chain)**, 그리고 **AI 서비스의 보안(Prompt Injection)**으로 확장되고 있습니다. 특히 2025년판에서는 클라우드 환경의 확산에 따라 설정 오류(Security Misconfiguration)가 2위로 급상승했으며, 예상치 못한 예외 상황을 이용한 공격(Mishandling of Exceptional Conditions)이 신규 항목으로 추가되었습니다.

2. 핵심 공격 기술의 심층 분석

OWASP Top 10 항목을 이해하기 위해서는 각 취약점이 실제로 어떻게 작동하는지 그 원리를 파악해야 합니다. 실무에서 가장 빈번하게 다뤄지는 핵심 공격 기술들을 정리합니다.

가. 데이터와 명령의 경계 붕괴: 인젝션(Injection)

인젝션은 사용자로부터 입력받은 '데이터'가 서버에서 실행되는 '명령어'로 변환될 때 발생합니다. 대표적으로 **SQL 인젝션(SQL Injection)**은 로그인 폼이나 검색 창에 ' OR '1'='1'과 같은 구문을 넣어 인증을 우회하거나 데이터베이스 전체를 탈취합니다. 이는 개발자가 입력값을 검증 없이 쿼리문에 직접 결합(Concatenation)할 때 발생하며, 이를 방지하기 위해서는 반드시 **Prepared Statement(파라미터화 쿼리)**를 사용하여 데이터와 명령을 엄격히 분리해야 합니다.

나. 클라이언트 측 공격: XSS(Cross-Site Scripting)

XSS는 공격자가 악성 JavaScript 코드를 웹 페이지에 삽입하여, 해당 페이지를 열람하는 다른 사용자의 브라우저에서 코드가 실행되게 하는 공격입니다. 게시판에 악성 스크립트를 저장하는 **저장형(Stored) XSS**가 가장 위험하며, 이를 통해 사용자의 세션 쿠키를 탈취하여 계정을 도용할 수 있습니다. 방어 방법으로는 사용자 입력값의 특수문자를 HTML 엔티티로 변환하는 **인코딩(Encoding)**과 **CSP(Content Security Policy)** 설정이 필수적입니다.

다. 클라우드 및 내부망 위협: SSRF(Server-Side Request Forgery)

SSRF는 서버가 외부 URL을 요청하는 기능을 악용하여, 공격자가 직접 접근할 수 없는 서버 내부 리소스나 클라우드 메타데이터 서버(예: AWS의 169.254.169.254)에 대신 요청을 보내게 만드는 공격입니다. 이를 통해 방화벽을 우회하여 내부 시스템 정보를 탈취할 수 있습니다. 2025년판에서는 이 위협이 접근 제어(Broken Access Control) 항목에 통합되어 관리됩니다.

라. 현대적 고위험 위협: 공급망 공격 및 AI 공격

- 공급망 공격(Supply Chain Attack):** 기업이 신뢰하는 오픈소스 라이브러리나 업데이트 서버를 먼저 해킹하여, 정상적인 배포 경로를 통해 악성코드를 유포합니다. (예: SolarWinds 사례)
- 프롬프트 인젝션(Prompt Injection):** AI(LLM) 서비스에서 사용자가 악의적인 지시를 입력하여 AI가 원래의 시스템 지침을 무시하고 기밀 정보를 유출하게 만드는 신종 공격입니다.

3. [실습] WebGoat 환경 구축 및 취약점 점검

이론으로 배운 취약점들을 실제로 확인하기 위해, 의도적으로 보안 결함을 포함하도록 설계된 실습 환경인 **WebGoat**를 구축합니다. 실습 환경 구축 과정에서는 보안 설정이 완료되어야 하므로, 시스템의 방화벽 및 SELinux 상태를 먼저 점검해야 합니다.

Step 1: 시스템 보안 설정 완화 및 검증

실습을 위해 서비스 간 통신을 방해할 수 있는 방화벽(firewalld)과 강제 접근 제어 시스템인 SELinux를 비활성화합니다. 먼저 방화벽의 현재 상태를 확인합니다.

실습 화면에서 확인되는 명령 및 결과 목록:

- 명령어: `systemctl status firewalld`
- 상태값: `Loaded: disabled, Active: inactive (dead)`

방화벽이 비활성화된 것을 확인했다면, 다음으로 SELinux의 상태를 점검합니다. 보안 정책이 강하게 적용되어 있으면 실습 중 서비스 동작이 차단될 수 있습니다.

실습 화면에서 확인되는 설정 내용:

- 파일 경로: `/etc/selinux/config`
- 설정값: `SELINUX=disabled`

두 설정이 모두 완화되었다면, 이제 WebGoat 서비스를 구동할 준비가 되었습니다. 서비스가 어떤 방식으로 실행되는지 설정 파일을 통해 확인합니다.

실습 화면에서 확인되는 서비스 정의 내용:

- 서비스 명: `WebGoat and WebWolf Security Lab Service`
- 실행 명령어: `/usr/bin/java -jar webgoat-2023.8.jar --server.address=0.0.0.0`
- 로그 경로: `/var/log/webgoat/webgoat.log`

설정 파일의 `ExecStart` 항목을 보면, 자바(Java)를 사용하여 특정 JAR 파일을 실행하며 모든 네트워크 인터페이스(0.0.0.0)에서 접속을 허용하도록 설정되어 있음을 알 수 있습니다. 이제 서비스를 시작하고 정상 구동 여부를 확인합니다.

실습 화면에서 확인되는 서비스 구동 상태:

- 명령어: `systemctl status webgoat`
- 상태값: `Active: active (running), Main PID: 8707 (java)`

서비스가 정상적으로 실행 중임을 확인했다면, 웹 브라우저를 통해 실제 서비스 페이지에 접속합니다. 이때 보안 연결(HTTPS)이 적용되지 않은 환경임을 주의 깊게 살펴야 합니다.

실습 화면에서 확인되는 브라우저 접속 상태:

- 접속 주소: `172.16.16.97:8080/WebGoat/login`
- 보안 상태: `Not secure (HTTP 사용)`

브라우저의 "**Not secure**" 표시는 데이터가 암호화되지 않은 평문(HTTP)으로 전송되고 있음을 의미합니다. 이는 앞서 배운 **암호화 실패(Cryptographic Failures)** 취약점에 해당하며, 공격자가 네트워크 중간에서 데이터를 가로채면 로그인 정보가 그대로 노출될 수 있습니다. WebWolf 서비스 역시 동일한 보안 결함을 가지고 있습니다.

이러한 보안 경고를 확인했다면, 이제 WebGoat의 각 모듈을 통해 실제 인젝션, XSS 등의 공격 시나리오를 수행하며 취약점의 작동 원리를 학습할 수 있습니다.

4. 학습 정리 및 점검 포인트

오늘 학습한 내용을 바탕으로, 실제 보안 업무나 시험에서 놓치지 말아야 할 핵심 포인트를 정리합니다.

실무/시험 관점의 체크리스트

- **인증 vs 권한 부여**: 인증(Authentication)은 '누구인지'를 확인하는 과정이고, 권한 부여(Authorization/Access Control)는 '무엇을 할 수 있는지'를 결정하는 과정입니다. 이 둘을 혼동하여 설계하면 접근 제어 취약점이 발생합니다.
- **데이터와 명령의 분리**: SQL 인젝션 방어를 위해 왜 `String` 결합이 아닌 `Prepared Statement`를 써야 하는지 그 이유를 설명할 수 있어야 합니다.
- **암호화의 범위**: 암호화 실패는 단순히 알고리즘의 문제를 넘어, **HTTPS 미사용, 하드코딩된 키, 취약한 난수 생성** 등을 모두 포함하는 넓은 개념입니다.
- **공급망 보안의 중요성**: 우리 회사의 코드가 안전하더라도, 우리가 사용하는 오픈소스 라이브러리가 오염되어 있다면 시스템 전체가 위협해질 수 있음을 인지해야 합니다.

⚠ 흔히 하는 실수 (Common Mistakes)

- **클라이언트 측 검증에만 의존**: JavaScript를 이용한 입력값 검증은 공격자가 쉽게 우회할 수 있습니다. 모든 보안 검증은 반드시 서버 측(Server-side)에서 수행되어야 합니다.
- **Blacklist 방식의 방어**: "특정 단어(예: <script>)만 막으면 되겠다"라는 생각은 위험합니다. 공격자는 다양한 인코딩 기법으로 이를 우회합니다. 허용된 값만 통과시키는 **Whitelist 방식**이 훨씬 안전합니다.

WebGoat 실습 환경 구축 및 시스템 보안 설정 검증

웹 보안 학습을 위해 의도적으로 취약한 환경을 구축하는 과정은 보안의 원리를 이해하는 데 매우 중요하다. 이번 실습에서는 웹 취약점 학습용 애플리케이션인 WebGoat와 이를 보조하는 도구인 WebWolf를 설치하고, 원활한 실습을 위해 시스템의 보안 설정(방화벽, SELinux 등)을 조정하는 과정을 진행했다.

1. 실습 환경 준비 및 기본 소프트웨어 설치

WebGoat는 다양한 웹 취약점 문제를 퀴즈 형태로 제공하는 사이트이며, WebWolf는 이러한 퀴즈를 풀기 위한 도구 역할을 한다. 실습을 시작하기 위해서는 다음과 같은 선행 조건이 필요하다.

- **Java SE 설치:** WebGoat는 Java 기반 애플리케이션이므로 적절한 버전의 Java Runtime Environment(JRE) 또는 Java Development Kit(JDK)가 설치되어 있어야 한다.
- **패키지 다운로드:** wget 명령어를 사용하여 WebGoat 실행 파일(.jar)을 서버로 다운로드한다. 만약 다운로드가 원활하지 않을 경우, 네트워크 차단을 방지하기 위해 시스템 방화벽 상태를 먼저 점검해야 한다.

2. 시스템 보안 설정 완화 (실습 환경 최적화)

실제 운영 환경에서는 시스템을 보호하기 위해 방화벽과 SELinux가 강력하게 작동하지만, 실습 중에는 이러한 보안 기능이 공격 시나리오를 차단하거나 의도적인 동작을 방해할 수 있다. 따라서 실습 목적에 따라 다음과 같이 보안 설정을 일시적으로 완화한다.

2.1 방화벽(Firewall) 설정 점검

WebGoat가 사용하는 포트(예: 8080, 9090)로의 통신이 차단되는 것을 방지하기 위해 firewalld 데몬의 상태를 확인한다. 실습 환경에서는 통신 원활화를 위해 방화벽을 비활성화한다.

systemctl status firewalld 명령어를 통해 현재 방화벽의 활성화 여부를 확인한 결과, 서비스가 disabled(비활성화) 및 inactive (dead)(중지) 상태임을 확인하였다.

방화벽이 꺼져 있어야 외부 혹은 로컬 네트워크에서의 요청이 차단 없이 서비스로 전달될 수 있다. 만약 서비스 접속이 안 된다면 이 단계에서 방화벽이 다시 활성화되어 있는지 확인해야 한다.

2.2 SELinux(Security-Enhanced Linux) 설정 조정

SELinux는 커널 수준에서 보안 정책을 강제하여 비정상적인 프로세스 접근을 차단하는 보안 메커니즘이다. 그러나 WebGoat 실습 중 발생하는 특정 파일 접근이나 프로세스 간 통신이 SELinux 정책에 의해 차단될 수 있으므로, 설정을 disabled로 변경한다.

/etc/selinux/config 파일을 확인하여 SELINUX=disabled로 설정되어 있는지 검증한다.

SELinux를 비활성화하면 커널 수준의 강제 접근 제어가 해제되어, 실습에서 의도하는 취약점 공격(예: 파일 읽기, 권한 상승 등)이 정책에 막히지 않고 정상적으로 수행될 수 있다.

3. WebGoat 서비스 등록 및 구동 확인

설치된 WebGoat를 시스템 서비스로 등록하여 관리 효율성을 높이고, 백그라운드에서 안정적으로 구동되도록 설정한다.

3.1 Systemd 서비스 유닛 설정

WebGoat를 시스템 서비스로 관리하기 위해 systemd 유닛 파일을 작성한다. 서비스 파일 내에는 Java 실행 경로, 실행할 JAR 파일명, 그리고 네트워크 인터페이스 설정이 포함된다.

작성된 설정 파일의 핵심 내용은 다음과 같다.

```
ExecStart=/usr/bin/java -jar webgoat-2023.8.jar --server.address=0.0.0.0
```

여기서 --server.address=0.0.0.0 옵션은 특정 IP가 아닌 모든 네트워크 인터페이스로부터의 접속을 허용하도록 설정하는 매우 중요한 값이다.

3.2 서비스 활성화 및 실행 상태 검증

서비스 등록 후 systemctl start webgoat 명령으로 서비스를 시작하고, systemctl status webgoat 명령을 통해 실제 프로세스가 메모리에 적재되어 동작 중인지 확인한다.

상태 확인 결과, 서비스가 active (running) 상태이며, 특정 PID(Process ID)를 할당받아 정상적으로 구동 중임을 확인하였다.

서비스가 active 상태임에도 웹 접속이 불가능하다면, java 실행 경로가 올바른지, 혹은 지정된 포트가 이미 다른 프로세스에 의해 사용 중인지 점검해야 한다.

4. 취약점 관점에서의 결과 분석: 암호화 실패(Cryptographic Failures)

서비스가 정상적으로 구동되면 웹 브라우저를 통해 WebGoat(:8080) 및 WebWolf(:9090) 로그인 페이지에 접속한다. 이 과정에서 보안 관점의 중요한 결함을 발견할 수 있다.

브라우저 주소창을 확인하면, https가 아닌 http 프로토콜을 사용하고 있으며, 브라우저는 이를 "Not secure(안전하지 않음)"로 표시한다.

이 현상은 OWASP Top 10 - A04: Cryptographic Failures(암호화 실패) 항목과 직결된다. 통신 구간에서 TLS/SSL 암호화가 적용되지 않았기 때문에, 사용자가 입력하는 아이디와 비밀번호가 평문(Plaintext)으로 네트워크를 통해 전송된다. 공격자가 중간에서 패킷을 도청(Sniffing)할 경우, 사용자의 자격 증명을 그대로 탈

취할 수 있는 매우 위험한 상태이다.

5. 실습 요약 및 점검 체크리스트

성공적인 실습 환경 구축을 위해 다음 사항을 최종 점검한다.

- **환경 점검:** Java 환경이 구성되었으며, wget으로 필요한 파일을 정상적으로 확보했는가?
- **보안 완화 확인:** firewallld와 SELinux가 실습 목적에 맞게 비활성화되었는가?
- **서비스 상태:** systemctl status 명령 시 서비스가 active (running)으로 표시되는가?
- **접근성 확인:** 브라우저에서 http://[IP]:8080 및 http://[IP]:9090으로 접속이 가능한가?
- **보안 인식:** 브라우저의 "Not secure" 경고를 통해 암호화 미적용(A04)의 위험성을 인지하였는가?

웹 보안 방어 전략 및 실무 운영 점검 가이드

현대 웹 애플리케이션 보안은 단일 기술의 도입만으로는 완성될 수 없습니다. 보안은 설계(Design), 구현(Implementation), 운영(Operation)의 전 과정이 유기적으로 결합되어야 하는 체계적인 활동입니다. 본 가이드는 글로벌 보안 표준인 OWASP(Open Web Application Security Project)의 위협 모델을 바탕으로, 실제 실습 환경 구축부터 취약점별 방어 전략 및 실무 운영 시의 점검 포인트를 다룹니다.

1. 보안 실습 환경 구축 및 시스템 보안 설정 점검

웹 취약점을 안전하게 학습하기 위해서는 의도적으로 보안이 약화된 환경(WebGoat 등)을 구축해야 합니다. 이 과정에서 시스템 서비스의 상태를 관리하고, 학습 도구가 네트워크를 통해 원활히 통신할 수 있도록 방화벽 및 보안 모듈을 조정하는 절차가 선행됩니다.

1.1 서비스 유닛(Service Unit) 구성

학습 도구인 WebGoat를 백그라운드에서 안정적으로 구동하기 위해 systemd 서비스를 등록합니다. 이때 서비스 설정 파일 내의 실행 경로와 네트워크 바인딩 주소를 정확히 확인해야 합니다.

```
[Service]
Type=simple
User=wang
WorkingDirectory=/usr/local
ExecStart=/usr/bin/java -jar webgoat-2023.8.jar --server.address=0.0.0.0
Restart=on-failure
```

위 설정에서 --server.address=0.0.0.0 옵션은 특정 IP가 아닌 모든 네트워크 인터페이스로부터의 접속을 허용함을 의미합니다. 이는 외부 클라이언트가 실습 서버에 접근할 수 있도록 하기 위한 설정입니다.

ExecStart 항목의 java 실행 명령어 및 서버 주소 설정값

```
[Unit]
Description=WebGoat and WebWolf Security Lab Service
After=network.target

[Service]
Type=simple
User=wang
WorkingDirectory=/usr/local
ExecStart=/usr/bin/java -jar webgoat-2023.8.jar --server.address=0.0.0.0
Restart=on-failure
StandardOutput=append:/var/log/webgoat/webgoat.log
StandardError=append:/var/log/webgoat/webgoat.log

[Install]
WantedBy=multi-user.target
```

이 절의 실습 결과를 확인하는 화면

설정된 유닛 파일이 정상적으로 로드되었는지 확인하는 과정은 서비스 가용성을 보장하는 첫 번째 단계입니다.

1.2 시스템 보안 통제 해제 및 검증

실습 도중 발생하는 네트워크 차단이나 접근 거부를 방지하기 위해, 테스트 환경에서는 다음과 같이 시스템 보안 기능을 일시적으로 조정합니다. ※ 주의: 실제 운영 환경에서는 절대로 권장되지 않는 방식입니다.

- **방화벽(Firewall) 관리:** firewallld 데몬이 활성화되어 있으면 실습용 포트 접근이 차단될 수 있으므로 상태를 확인하고 비활성화합니다.
- **SELinux(Security-Enhanced Linux) 설정:** 커널 수준의 강제 접근 제어(MAC) 정책이 작동하면 서비스 실행에 제약이 생길 수 있으므로, 설정 파일에서 정책을 disabled로 변경합니다.

먼저 systemctl 명령어를 통해 방화벽의 현재 상태를 점검합니다.

Active: inactive (dead) 및 disabled 상태 표시

```
[root@localhost local]# systemctl status firewalld
○ firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; preset: en>
   Active: inactive (dead)
     Docs: man:firewalld(1)
   ESC0C
 firewall daemon
em/firewalld.service; disabled; preset: enabled)
```

systemctl 명령으로 확인된 firewalld 데몬의 비활성화 및 중지 상태

방화벽이 inactive (dead) 상태인 것을 확인했다면, 다음으로 SELinux 설정 파일을 확인하여 시스템 보안 정책이 적용되지 않도록 구성합니다.

SELINUX=disabled 텍스트 설정값

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
# See also:
# https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/using_selinux/changing-selinux-states-and-modes_using-selinux#changing-selinux-modes-at-bo
ot-time_changing-selinux-states-and-modes
#
# NOTE: Up to RHEL 8 release included, SELINUX=disabled would also
# fully disable SELinux during boot. If you need a system with SELinux
# fully disabled instead of SELinux running with no policy loaded, you
# need to pass selinux=0 to the kernel command line. You can use grubby
# to persistently set the bootloader to boot with selinux=0:
#
#   grubby --update-kernel ALL --args selinux=0
#
# To revert back to SELinux enabled:
#
#   grubby --update-kernel ALL --remove-args selinux
#
SELINUX=disabled
# SELINUXTYPE= can take one of these three values:
#   targeted - Targeted processes are protected,
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

이 절의 실습 결과를 확인하는 화면

모든 환경 설정이 완료된 후에는 WebGoat 서비스가 정상적으로 실행 중인지 최종적으로 확인합니다.

Active: active (running) 및 PID 정보

```
[root@localhost local]# systemctl status webgoat
● webgoat.service - WebGoat and WebWolf Security Lab Service
   Loaded: loaded (/etc/systemd/system/webgoat.service; enabled; preset: disabled)
   Active: active (running) since Thu 2026-07-02 10:36:48 KST; 3min 26s ago
 Main PID: 8707 (java)
    Tasks: 27 (limit: 48662)
  Memory: 517.7M (peak: 522.7M)
     CPU: 32.692s
   CGroup: /system.slice/webgoat.service
           └─8707 /usr/bin/java -jar webgoat-2023.8.jar --server.address=0.0.0.0

7월 02 10:36:48 localhost.localdomain systemd[1]: Started WebGoat and WebWolf Secur>
lines 1-11/11 (END)
```

systemctl 명령어로 확인한 WebGoat 서비스의 활성화 및 실행 정보

서비스가 active (running) 상태라면, 이제 브라우저를 통해 실제 웹 인터페이스에 접근하여 구동 여부를 판단합니다.

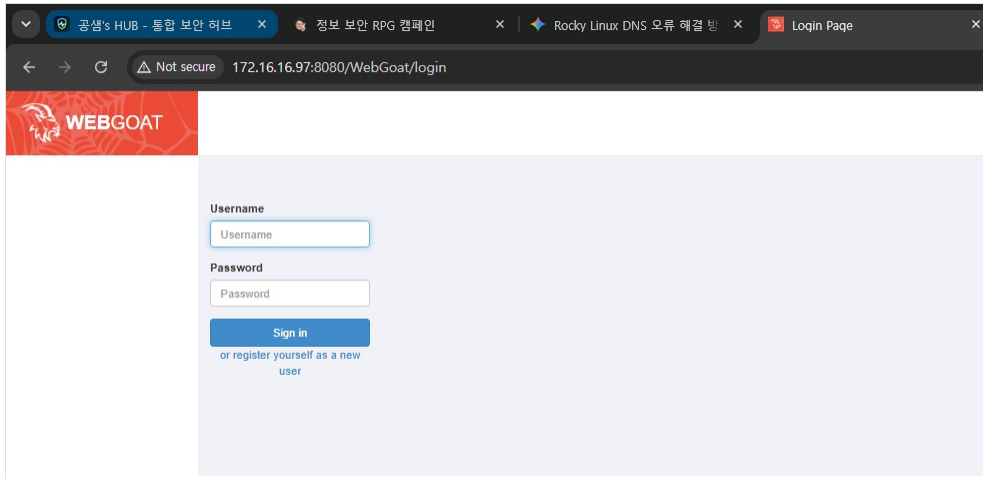
2. 주요 취약점별 방어 전략 (OWASP Top 10 기반)

보안 실무자는 취약점이 발생하는 원인을 이해하고, 이를 원천적으로 차단할 수 있는 방어 기법을 숙지해야 합니다.

2.1 암호화 실패 (Cryptographic Failures, A04)

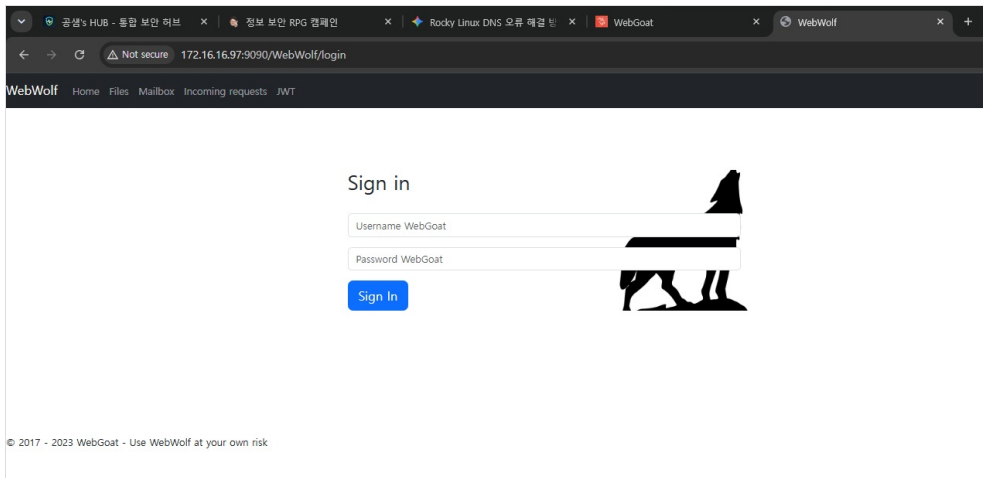
데이터 전송 시 암호화되지 않은 채로 정보를 주고받거나, 취약한 암호화 알고리즘을 사용하는 경우입니다. 실습 중 브라우저에서 "Not secure" 경고가 나타난다면 이는 전형적인 암호화 실패 사례입니다.

주소창의 Not secure 경고 및 HTTP 프로토콜



HTTPS 미사용으로 인해 브라우저에서 보안 경고가 표시된 WebGoat 로그인 페이지

주소창의 Not secure 경고 및 HTTP 프로토콜



보안 연결이 적용되지 않아 브라우저 주의 문구가 나타난 WebWolf 로그인 화면

위 화면처럼 브라우저 상단에 보안 경고가 표시되는 것은 데이터가 평문(Cleartext)으로 전송되고 있음을 의미하며, 공격자가 중간자 공격(MITM)을 통해 민감 정보를 탈취할 수 있는 상태입니다.

☒ 방어 전략:

- **TLS/HTTPS 전면 적용:** 모든 통신에 강력한 암호화 프로토콜을 적용합니다.
- **강력한 알고리즘 사용:** MD5, SHA1과 같은 취약한 해시 함수 대신 SHA-256 이상의 알고리즘을 사용합니다.
- **비밀번호 솔팅(Salting):** 해시값 생성 시 랜덤한 값(Salt)을 추가하여 레인보우 테이블 공격을 방어합니다.

2.2 인젝션 (Injection, A05)

사용자의 입력값이 데이터가 아닌 시스템 명령어로 해석되어 실행되는 공격입니다. 대표적으로 SQL 인젝션이 있으며, 데이터베이스의 권한을 탈취하거나 데이터를 임의로 조작할 수 있습니다.

☒ 방어 전략:

- **파라미터화 쿼리(Prepared Statement) 사용:** 사용자 입력값이 쿼리의 구조를 변경하지 못하도록 데이터로만 처리되게 합니다.
- **입력값 검증:** 화이트리스트(Whitelist) 방식을 사용하여 허용된 형식의 데이터만 입력되도록 제한합니다.

2.3 접근 제어 취약점 (Broken Access Control, A01)

인증된 사용자가 자신의 권한을 넘어서는 데이터나 기능에 접근할 수 있는 결함입니다. 예를 들어, URL의 ID 값을 변경하여 타인의 개인정보를 조회하는 IDOR(Insecure Direct Object Reference) 공격이 이에 해당합니다.

☒ 방어 전략:

- **서버 측 권한 검증:** 클라이언트의 요청을 신뢰하지 말고, 매 요청마다 서버에서 해당 사용자의 권한을 재검증해야 합니다.
- **최소 권한 원칙(Principle of Least Privilege):** 사용자 및 프로세스에게 업무 수행에 필요한 최소한의 권한만 부여합니다.

3. 실무 운영 점검 및 대응 체크리스트

보안은 구축보다 유지가 중요합니다. 운영 환경에서는 정기적인 점검을 통해 보안 상태를 지속적으로 모니터링해야 합니다.

3.1 운영 점검 항목

점검 영역	주요 점검 포인트	판단 기준
시스템 설정	불필요한 서비스 및 포트 활성화 여부	사용하지 않는 포트는 모두 닫혀 있어야 함
인증 및 세션	MFA(다중 인증) 적용 및 세션 만료 정책	로그인 시 2차 인증이 필수적이어야 함
로그 및 모니터링	보안 이벤트(실패한 로그인 등) 기록 여부	이상 징후 발생 시 실시간 알림이 가능해야 함
소프트웨어 관리	사용 중인 라이브러리의 CVE 취약점 여부	최신 보안 패치가 적용된 버전 사용

3.2 실무자를 위한 핵심 요약

- **설계 단계:** 위협 모델링(Threat Modeling)을 통해 설계 단계부터 보안 요구사항을 반영하십시오.
- **개발 단계:** 입력값 검증과 파라미터화 쿼리를 습관화하여 인젝션 공격을 차단하십시오.
- **운영 단계:** 심층 방어(Defense in Depth) 원칙에 따라 다중 보안 계층을 구축하고 로그를 상시 모니터링하십시오.