정보보안 부트캠프 - Red Team 입문 읽을거리

초심자용 해킹개론
OWASP Top 10 2017, DVWA, Kali Linux

비유로 이해하는 웹 보안과 레드팀 기초

공격 절차보다 개념 이해, 방어 관점, 안전한 실습을 우선합니다.

버전: Beginner Friendly Edition
작성일: 2026-06-23
대상 독자: 정보보안 부트캠프의 레드팀 입문자
사용 범위: 허가된 교육, 실습망, DVWA/WebGoat/CTF 등 안전한 환경

들어가기 전에: 이 문서의 약속

이 문서는 실전 공격 지시서가 아닙니다

목표는 초심자가 보안 개념을 이해하고, 허가된 실습 환경에서 안전하게 배울 수 있도록 돕는 것입니다. 실제 서비스, 학교/회사 네트워크, 공용 와이파이, 타인의 계정과 장비를 대상으로 한 실험은 허가 없이는 해서는 안 됩니다.

보안 공부는 처음에는 낯선 단어가 많다. Injection, XSS, Sniffing, MITM, LAND, DVWA, Kali 같은 단어가 한꺼번에 나오면 마치 외국어 수업처럼 느껴진다. 그래서 이 문서는 기술 용어를 바로 던지기보다, 건물, 택배, 주방, 출입증, 게시판 같은 일상 비유로 먼저 설명한다.

레드팀은 “나쁜 일을 잘하는 팀”이 아니다. 더 정확히는 허가된 범위 안에서 공격자의 사고방식을 빌려 조직의 약점을 찾아 주는 팀이다. 소방훈련에서 불을 내는 사람이 아니라, 불이 났을 때 대피로와 경보기가 제대로 작동하는지 확인하는 훈련 담당자에 가깝다.

원칙	초심자식 표현	실무 의미
허가	남의 집 문고리를 돌려보지 않는다	명확한 범위와 서면 승인 없이 테스트하지 않는다
최소 영향	문이 열리는지만 보고 물건은 만지지 않는다	취약점은 필요한 만큼만 입증하고 서비스 장애를 만들지 않는다
기록	무엇을 언제 왜 했는지 일지로 남긴다	재현 가능한 증적, 타임라인, 영향 범위, 권고안을 남긴다
보고	발견하면 떠벌리지 말고 책임자에게 알린다	취약점 공개와 공유는 정해진 절차를 따른다

검증 기준

이 문서는 OWASP Top 10 2017 공식 문서, OWASP Cheat Sheet Series, DVWA 공식 GitHub, Kali Linux 공식 문서, NIST/MITRE의 네트워크 스니핑 설명, Juniper의 LAND 공격 설명을 바탕으로 작성했다. OWASP Top 10은 2026년 현재 더 최신 판본도 있으나, 여기서는 요청 범위에 맞추어 2017 항목을 중심으로 다룬다. 최신 흐름과 비교할 때는 OWASP Top 10 Project 페이지를 함께 보는 것이 좋다. [1][2]

읽는 순서

- 1. 보안 세계를 보는 지도
- 2. 웹과 네트워크 기초
- 3. OWASP Top 10 2017 초심자 해설
- 4. 기본 해킹 기법을 순하게 이해하기

1. 보안 세계를 보는 지도

1.1 레드팀, 블루팀, 퍼플팀을 축구로 비유하기

보안 조직을 축구팀으로 비유하면 이해가 쉽다. 레드팀은 상대 공격수 역할을 맡아 수비의 빈틈을 찾는다. 블루팀은 골키퍼와 수비수처럼 실제 서비스를 지키고 탐지와 대응을 담당한다. 퍼플팀은 공격수와 수비수가 따로 노는 것이 아니라, 훈련 영상을 같이 보며 “이 공격은 어떻게 보였고, 다음에는 어떻게 막을까”를 함께 개선하는 방식이다.

구분	하는 일	초심자 비유	성과물
레드팀	허가된 범위에서 공격자 관점으로 약점을 검증	건물 안전 점검에서 일부러 비상구를 찾아보는 사람	취약점, 공격 경로, 영향도, 재현 증적
블루팀	탐지, 차단, 로그 분석, 사고 대응	CCTV 관제실과 경비팀	경보, 차단 정책, 사고 보고서, 개선 조치
퍼플팀	레드와 블루가 함께 학습하며 방어 체계를 개선	훈련 후 영상을 같이 보며 전술을 고치는 코치 회의	탐지 룰, 플레이북, 개선된 통제

1.2 보안은 벽 하나가 아니라 여러 겹의 문이다

초심자는 “해커가 뚫었다”라는 표현 때문에 보안을 단 하나의 벽으로 상상하기 쉽다. 실제 보안은 건물처럼 여러 겹이다. 외벽, 현관, 출입증, 엘리베이터, 각 방의 잠금장치, CCTV, 방문자 기록, 비상 연락망이 모두 필요하다. 웹 서비스도 마찬가지다. HTTPS, 로그인, 권한, 입력 검증, 서버 설정, 로그, 백업, 모니터링이 함께 움직인다.

중요한 사고방식

레드팀 초심자는 ‘어떻게 뚫지?’보다 ‘어떤 가정이 깨지면 위험해질까?’라고 질문해야 한다. 예를 들어 ‘로그인한 사용자는 항상 착하다’라는 가정은 깨지기 쉽다. ‘내부망은 안전하다’라는 가정도 깨지기 쉽다.

1.3 취약점, 위협, 위험의 차이

세 단어는 비슷해 보이지만 실무에서는 구분이 중요하다. 취약점은 약한 문고리다. 위협은 그 문고리를 흔들 수 있는 사람이나 사건이다. 위험은 그 문고리가 흔들렸을 때 실제로 벌어질 손해의 크기다.

용어	비유	예시
취약점 Vulnerability	문고리가 헐거움	세션 토큰이 예측 가능함
위협 Threat	문고리를 흔들 수 있는 사람	계정 탈취를 노리는 공격자
위험 Risk	문이 열렸을 때 생기는 피해	고객 개인정보 유출, 서비스 중단, 평판 손상
통제 Control	문고리를 보강하고 CCTV를 다는 조치	MFA, 권한 검사, 로그 경보, 패치

1.4 레드팀원이 처음 가져야 할 질문 7개

1. 이 시스템은 누구를 믿고 있는가?
2. 외부에서 들어오는 값은 어디에 저장되고 어디에서 다시 사용되는가?
3. 로그인한 사용자와 관리자 사이의 경계는 어디인가?
4. 실패했을 때 오류 메시지가 너무 많은 것을 알려주지 않는가?
5. 중요한 정보가 저장 중, 전송 중, 로그 안에서 보호되는가?
6. 공격을 당했을 때 누가, 얼마나 빨리, 무엇을 보고 알아차리는가?
7. 이 테스트가 허가 범위와 안전 기준을 벗어나지는 않는가?

> **초심자에게 가장 좋은 태도**
> 툴을 먼저 외우지 말고, 데이터가 어디서 와서 어디로 가는지 선으로 그려라. 그 선 위에 신뢰 경계가 있다.

2. 웹과 네트워크 기초

2.1 웹은 식당 주문 시스템과 비슷하다

브라우저는 손님, 웹 서버는 식당, 데이터베이스는 냉장고와 장부, 애플리케이션 코드는 주방 규칙이라고 생각하자. 손님은 메뉴판을 보고 주문한다. 서버는 주문을 받아 주방 규칙에 따라 장부를 확인하고 음식을 내준다.

웹 구성요소	초심자 비유	보안에서 보는 포인트
브라우저 Client	손님	쿠키, 로컬 저장소, 입력값, 인증서 경고
웹 서버	주문 접수대	HTTPS, 헤더, 라우팅, 오류 페이지
애플리케이션	주방 규칙	입력 검증, 권한 검사, 비즈니스 로직
데이터베이스	장부와 냉장고	질의 안전성, 권한, 백업, 암호화
API	직원끼리 쓰는 주문서 양식	인증, 권한, 요청 제한, 데이터 최소화

2.2 HTTP 요청과 응답

HTTP 요청은 “이 주소의 이 정보를 주세요”라고 쓰인 주문서다. 응답은 서버가 돌려주는 음식과 영수증이다. URL, 메서드, 헤더, 쿠키, 본문이 요청에 들어가고, 상태 코드, 헤더, 본문이 응답에 들어간다.

> **비유**
>
> GET은 ‘메뉴판 보여 주세요’에 가깝고, POST는 ‘이 주문을 접수해 주세요’에 가깝다. 단, 실제 의미는 서버 구현에 달려 있으므로 상태를 바꾸는 기능을 GET으로 만들지 않는 것이 좋다.

2.3 쿠키와 세션

웹은 원래 매 요청을 따로 보는 경향이 있다. 그래서 서버는 “방금 로그인한 사람이 같은 사람인지” 기억할 방법이 필요하다. 쿠키는 브라우저가 들고 다니는 작은 표식이고, 세션은 서버가 그 표식을 보고 사용자를 기억하는 방식이다.

초심자 비유로는 놀이공원 팔찌가 좋다. 입장할 때 팔찌를 받으면 놀이기구마다 다시 신분증을 보이지 않아도 된다. 하지만 팔찌가 복제되거나 도난당하면 다른 사람이 나처럼 행동할 수 있다.

2.4 HTTPS와 TLS

HTTP가 엽서라면 HTTPS는 봉인된 편지에 가깝다. 엽서는 중간의 배달 경로에서 주소와 내용이 쉽게 보일 수 있지만, 봉인된 편지는 내용 확인이 어렵다. HTTPS는 통신 상대 확인과 암호화를 통해 스니핑과 중간자 공격 위험을 줄인다.

2.5 입력값은 모두 외부인이다

검색어, 로그인 ID, 파일명, JSON 본문, HTTP 헤더, 쿠키, URL 파라미터는 모두 외부에서 들어오는 값이다. 초심자는 “입력창에서 들어온 값만 위험하다”고 생각하기 쉽지만, 실제로는 서버가 받는 모든 값이 입력이다.

외우기

입력값은 손님이고, 서버는 직원이다. 친절한 직원도 손님에게 창고 열쇠를 넘기면 안 된다.

2.6 신뢰 경계 Trust Boundary

신뢰 경계는 “여기부터는 믿는 쪽, 저기부터는 확인해야 하는 쪽”을 나누는 선이다. 브라우저와 서버 사이, 웹 서버와 DB 사이, 내부망과 외부망 사이, 일반 사용자와 관리자 사이가 대표적인 신뢰 경계다.

레드팀 관점에서는 이 선을 찾는 일이 매우 중요하다. 많은 취약점은 “선 밖에서 온 값을 선 안의 것처럼 믿었기 때문에” 발생한다.

> **초심자 그림 연습**
> 종이에 브라우저, 서버, DB를 세 칸으로 그리고 화살표를 그어 보라. 화살표마다 ‘누가 보냈나?’, ‘무엇을 믿나?’, ‘실패하면 어떤 피해가 나나?’를 적으면 이미 훌륭한 보안 분석의 시작이다.

3. OWASP Top 10 2017 초심자 해설

3.1 OWASP Top 10 2017은 무엇인가

OWASP Top 10은 웹 애플리케이션에서 특히 중요하게 다뤄야 할 보안 위험을 정리한 인식 문서다. OWASP는 Top 10을 개발자와 웹 애플리케이션 보안을 위한 표준적인 인식 문서로 설명한다. 2017 버전은 아래 10개 항목을 중심으로 구성된다. [1][2]

번호	항목	짧은 뜻	비유
A1	Injection	입력값이 명령어처럼 해석되는 문제	손님 메모가 주방 레시피를 바꾸는 상황
A2	Broken Authentication	로그인과 세션 관리가 허술한 문제	경비원이 출입증을 대충 보는 상황
A3	Sensitive Data Exposure	민감정보가 암호화나 보호 없이 노출되는 문제	현금 봉투를 투명 비닐에 넣어 보내는 상황
A4	XXE	XML 해석기가 외부 자료를 마음대로 읽는 문제	도서관 직원이 쪽지대로 비밀 서가를 찾아주는 상황
A5	Broken Access Control	사용자가 해도 되는 일과 안 되는 일을 서버가 제대로 막지 못하는 문제	호텔 엘리베이터가 아무 층이나 열어주는 상황
A6	Security Misconfiguration	기본값, 오류 메시지, 권한, 패치 등 설정이 느슨한 문제	새 아파트 문 비밀번호가 아직 0000인 상황
A7	XSS	사용자 입력이 다른 사람의 브라우저에서 코드처럼 실행되는 문제	게시판에 원격 조종 쪽지를 붙이는 상황
A8	Insecure Deserialization	받은 데이터를 객체로 되살리는 과정에서 위험한 동작이 생기는 문제	모르는 사람이 보낸 조립식 로봇을 검수 없이 켜는 상황
A9	Known Vulnerable Components	이미 알려진 취약점이 있는 라이브러리와 프레임워크를 계속 쓰는 문제	리콜된 잠금장치를 계속 쓰는 상황
A10	Insufficient Logging & Monitoring	기록과 감시가 부족해 침해를 늦게 알아차리는 문제	CCTV는 달았지만 녹화 버튼을 안 누른 상황

읽는 법

각 항목을 '정의 - 비유 - 레드팀 관찰점 - 방어 방법 - 암기 문장' 순서로 읽으면 된다. 처음부터 모든 기술 세부사항을 외우려 하지 말고, 위험의 모양을 먼저 잡는 것이 좋다.

A1: Injection - 입력값과 명령어가 섞일 때

비유로 이해하기

식당에서 손님이 “김치찌개 하나”라고 주문해야 하는데, 주문서 칸에 “냉장고 문도 열고 재고표도 바꿔라” 같은 주방 지시문을 적었다고 생각하자. 좋은 식당 시스템은 손님 메모를 음식 이름으로만 다루지만, 나쁜 시스템은 그 메모를 주방 명령처럼 실행한다.

핵심 개념

Injection은 신뢰할 수 없는 입력값이 SQL, NoSQL, 운영체제 명령, LDAP 질의 같은 해석기(interpreter)에 “데이터”가 아니라 “명령 조각”처럼 전달될 때 생긴다. OWASP 2017은 Injection을 A1로 배치했고, 적절한 권한 없이 데이터에 접근하거나 의도하지 않은 명령을 실행하게 만들 수 있다고 설명한다. [1]

초심자식 이해

초심자는 “입력창은 모두 편지함”이라고 외우면 된다. 편지함에 들어온 내용은 그냥 편지여야 한다. 편지를 받은 직원이 그 안의 문장을 회사 규칙으로 실행하면 사고가 난다.

레드팀 관찰 포인트

레드팀 관점에서는 검색창, 로그인 폼, API 파라미터, 파일명, 헤더처럼 “밖에서 들어오는 값”을 먼저 본다. 단, 실제 서비스에는 절대 무단으로 값을 넣어 시험하지 않는다. 허가된 DVWA, WebGoat, CTF, 내부 실습망에서만 관찰한다.

방어 관점

방어는 “데이터와 명령을 분리하기”가 핵심이다. 매개변수화된 질의, ORM의 안전한 사용, 허용 목록 기반 입력 검증, DB 계정 최소권한, 자세한 오류 메시지 숨김, 입력값과 결과 로그를 통한 이상 징후 추적이 기본이다.

> **암기 문장**
> 암기 문장: 입력값은 손님 메모이지, 주방장 지시서가 아니다.

A2: Broken Authentication - 문지기가 사람을 잘못 확인할 때

비유로 이해하기

회사 로비 경비원이 사진이 흐릿한 출입증도 통과시키고, 방문증을 잃어버린 사람에게 새 방문증을 아무 확인 없이 내준다면 어떨까. 로그인은 문지기이고 세션은 방문증이다.

핵심 개념

Broken Authentication은 로그인, 비밀번호, 세션 토큰, 계정 복구, 자동화된 로그인 시도 방어가 제대로 구현되지 않을 때 생긴다. OWASP는 인증과 세션 관리의 구현 오류가 공격자에게 다른 사용자의 신분을 임시 또는 영구적으로 넘겨줄 수 있다고 설명한다. [1]

초심자식 이해

로그인은 “너 누구야?”를 묻는 절차이고, 세션은 “방금 확인한 사람이 계속 같은 사람 맞지?”를 기억하는 표식이다. 비밀번호만 강해도 세션이 약하면 소용이 없다.

레드팀 관찰 포인트

레드팀은 비밀번호 정책, 계정 잠금, MFA, 세션 만료, 비밀번호 재설정, 로그아웃 후 토큰 무효화 같은 “문지기의 습관”을 확인한다. 자동화 공격을 실제 계정에 시도하는 행위는 범위 밖이면 금지다.

방어 관점

강한 비밀번호 정책, 다중 인증, 로그인 속도 제한, 흔한/유출 비밀번호 차단, 안전한 비밀번호 재설정, TLS 사용, 예측 불가능한 세션 ID, 민감 기능 재인증이 기본이다. OWASP Authentication Cheat Sheet도 인증은 사용자가 주장한 신원이 맞는지 검증하는 과정이라고 정의하고 세션 관리의 중요성을 설명한다. [3]

> **암기 문장**
> 암기 문장: 비밀번호는 열쇠이고, 세션은 방문증이다. 둘 다 관리해야 한다.

A3: Sensitive Data Exposure - 귀중품을 투명 봉투에 넣을 때

비유로 이해하기

은행 카드, 주민번호, 건강기록을 택배로 보낸다고 하자. 봉투가 투명하고 잠금장치도 없다면 택배기사가 훔치지 않아도 지나가는 사람이 볼 수 있다.

핵심 개념

Sensitive Data Exposure는 금융정보, 의료정보, 개인정보 같은 민감 데이터가 저장 중 또는 전송 중 충분히 보호되지 않을 때 발생한다. OWASP는 민감 데이터가 암호화 같은 추가 보호 없이 노출될 수 있고 브라우저와 교환할 때도 특별한 주의가 필요하다고 설명한다. [1]

초심자식 이해

“비밀번호를 암호화한다”는 표현은 흔하지만, 실제로 비밀번호는 보통 복호화 가능한 암호화가 아니라 해시와 솔트를 이용해 보관한다. 즉, 원본을 다시 꺼내 쓰는 금고가 아니라 원본을 맞혀보기 어렵게 만든 지문 보관함에 가깝다.

레드팀 관찰 포인트

레드팀은 HTTPS 미사용, 민감정보가 URL에 들어가는지, 로그에 카드번호나 토큰이 남는지, 백업 파일이 열려 있는지, 브라우저 캐시에 중요한 값이 남는지 확인한다.

방어 관점

데이터 분류, 최소 보관, TLS, 안전한 키 관리, 저장 데이터 암호화, 비밀번호 해시, 로그 마스킹, 캐시 제어, 백업 보호, 권한 분리가 기본이다. 가장 안전한 데이터는 애초에 모으지 않는 데이터다.

> **암기 문장**
> 암기 문장: 귀중품은 투명 봉투가 아니라 잠긴 금고와 추적 가능한 배송으로 보낸다.

A4: XXE - 문서 안의 심부름 쪽지를 믿을 때

비유로 이해하기

도서관 직원이 반납 쪽지에 “비밀 서가 3층의 금고 문서를 가져와라”라고 적혀 있으면 그대로 가져온다고 생각하자. 원래 쪽지는 책 제목만 알려줘야 하는데, 심부름 명령까지 들어가면 위험해진다.

핵심 개념

XXE(XML External Entity)는 오래되었거나 잘못 설정된 XML 처리기가 XML 문서 안의 외부 엔터티 참조를 평가하면서 내부 파일 노출, 내부 포트 탐색, 원격 코드 실행, 서비스 거부 같은 문제가 생길 수 있는 취약점이다. OWASP 2017은 이를 A4로 분류했다. [1]

초심자식 이해

XML은 데이터를 담는 서류 양식이다. 문제는 서류 안에 “외부 자료를 찾아오라”는 지시가 있을 때, 처리기가 그 지시를 너무 착하게 따르는 경우다.

레드팀 관찰 포인트

레드팀은 XML 업로드, SOAP API, 문서 변환 기능, SAML 처리처럼 XML을 받는 지점을 찾는다. 실습은 취약 앱에서만 하고, 실제 서버의 내부 파일이나 내부망을 겨냥하는 테스트는 명시적 허가 없이는 금지다.

방어 관점

외부 엔터티와 DTD 처리 비활성화, 안전한 XML 파서 설정, 최신 라이브러리 사용, 입력 크기 제한, 네트워크 접근 제한, 오류 메시지 축소가 기본이다.

> **암기 문장**
> 암기 문장: 문서가 심부름을 시키게 두지 마라.

A5: Broken Access Control - 로그인은 했지만 아무 방이나 들어갈 때

비유로 이해하기

호텔 투숙객은 자기 객실 층만 갈 수 있어야 한다. 그런데 엘리베이터가 객실 카드만 보면 모든 층을 열어 준다면, 투숙객 인증은 되었지만 권한 통제는 깨진 것이다.

핵심 개념

Broken Access Control은 인증된 사용자가 할 수 있는 행동 제한이 서버에서 제대로 강제되지 않을 때 생긴다. OWASP는 이런 결함이 다른 사용자의 계정, 민감 파일, 데이터 수정, 권한 변경으로 이어질 수 있다고 설명한다. [1]

초심자식 이해

인증(Authentication)은 “누구인지 확인”이고, 인가/권한부여(Authorization)는 “무엇을 해도 되는지 확인”이다. 로그인했다고 모든 버튼을 눌러도 되는 것은 아니다.

레드팀 관찰 포인트

레드팀은 사용자 역할, 관리자 기능, 주문/게시글/파일 ID, API 권한을 본다. 화면에서 버튼을 숨기는 것만으로는 충분하지 않다. 서버가 매 요청마다 권한을 확인해야 한다.

방어 관점

서버 측 권한 검사, 기본 거부(deny by default), 객체 소유권 확인, 역할 기반/속성 기반 접근제어, 권한 변경 로그, 관리자 기능 분리, 테스트 자동화가 기본이다. OWASP Authorization Cheat Sheet는 비즈니스 문맥에 맞는 강건한 권한 로직을 강조한다. [3]

> **암기 문장**
> 암기 문장: 로그인은 건물 출입이고, 권한은 각 방의 열쇠다.

A6: Security Misconfiguration - 좋은 자물쇠를 달고도 문을 열어 둘 때

비유로 이해하기

새 아파트에 튼튼한 도어락이 있어도 초기 비밀번호가 0000이고 관리실 마스터키가 책상 위에 놓여 있다면 보안은 실패다. 설정은 제품보다 중요할 때가 많다.

핵심 개념

Security Misconfiguration은 안전하지 않은 기본 설정, 불완전한 임시 설정, 열린 클라우드 저장소, 잘못된 HTTP 헤더, 민감한 정보가 담긴 자세한 오류 메시지, 미흡한 패치에서 자주 발생한다. OWASP 2017은 이를 가장 흔히 보이는 문제라고 설명한다. [1]

초심자식 이해

초심자가 놓치기 쉬운 점은 “설치 완료”가 “보안 완료”가 아니라는 것이다. 프로그램은 설치 후 청소, 잠금, 불필요 기능 끄기, 업데이트가 필요하다.

레드팀 관찰 포인트

레드팀은 기본 계정, 디렉터리 목록, 디버그 페이지, 과도한 오류 메시지, 관리 콘솔 노출, 불필요한 서비스, 오래된 서버 배너를 확인한다.

방어 관점

하드닝 기준선, 자동화된 설정 점검, 패치 관리, 불필요 기능 제거, 환경별 설정 분리, 비밀값의 안전한 보관, 오류 메시지 정리, 클라우드 권한 점검이 기본이다.

> **암기 문장**
> 암기 문장: 보안은 설치 버튼이 아니라 마감 청소까지 포함한다.

A7: XSS - 게시판 글이 다른 사람 브라우저에서 행동할 때

비유로 이해하기

카페 게시판에 손님이 메모를 붙였다. 그런데 그 메모가 그냥 글이 아니라, 읽는 사람의 휴대폰을 조종하는 리모컨이라면 어떨까. XSS는 웹페이지에 붙은 글이 다른 사용자의 브라우저에서 코드처럼 실행되는 문제다.

핵심 개념

XSS는 애플리케이션이 신뢰할 수 없는 데이터를 적절히 검증하거나 이스케이프하지 않고 새 웹페이지에 넣거나, 브라우저 API를 통해 HTML/JavaScript를 만들 때 발생한다. OWASP는 XSS가 세션 탈취, 사이트 변조, 악성 사이트 리다이렉트로 이어질 수 있다고 설명한다. [1]

초심자식 이해

브라우저는 HTML과 JavaScript를 보면 “그려라, 실행하라”고 이해한다. 사용자가 쓴 글은 “표시할 내용”이어야지 “실행할 코드”가 되면 안 된다.

레드팀 관찰 포인트

레드팀은 검색 결과, 댓글, 프로필, 관리자 페이지 미리보기, 오류 메시지, URL 파라미터 반영 위치를 본다. 학습 환경에서는 입력이 어디에 다시 표시되는지 관찰하는 습관이 중요하다.

방어 관점

출력 위치에 맞는 인코딩, 안전한 템플릿 엔진, HTML 살균(sanitization), 위험한 DOM API 회피, Content Security Policy, 쿠키의 HttpOnly/SameSite/Secure 속성, 보안 리뷰가 기본이다.

> **암기 문장**
> 암기 문장: 남이 쓴 글은 화면에 보여도 되지만, 실행되면 안 된다.

A8: Insecure Deserialization - 택배 상자를 검수 없이 조립할 때

비유로 이해하기

택배 상자에 부품이 들어 있고 설명서에 따라 조립하면 로봇이 된다. 그런데 보낸 사람이 악의적이면, 로봇이 켜지는 순간 창문을 열거나 물건을 옮기도록 설계했을 수 있다.

핵심 개념

직렬화는 객체를 저장하거나 전송하기 쉽게 납작한 데이터로 바꾸는 것이고, 역직렬화는 그것을 다시 객체로 되살리는 것이다. OWASP는 안전하지 않은 역직렬화가 원격 코드 실행으로 이어질 수 있고, 재전송, 인젝션, 권한 상승에도 사용될 수 있다고 설명한다. [1]

초심자식 이해

초심자는 “저장된 게임 파일을 다시 불러오기”로 이해하면 된다. 저장 파일이 조작되어 있다면, 게임이 이상한 상태로 시작될 수 있다.

레드팀 관찰 포인트

레드팀은 쿠키, 토큰, 숨은 폼 값, 메시지 큐, 캐시, 세션 파일처럼 객체 상태가 저장되는 지점을 본다. 단, 실제 시스템에서 임의 객체를 조작하는 실험은 위험도가 높으므로 허가된 실습 앱에서만 한다.

방어 관점

신뢰할 수 없는 데이터 역직렬화 금지, 안전한 데이터 형식 사용, 무결성 서명, 허용된 타입만 처리, 라이브러리 업데이트, 최소권한 실행, 모니터링이 기본이다.

> **암기 문장**
> 암기 문장: 모르는 사람이 보낸 조립 로봇은 바로 켜지 않는다.

A9: Using Components with Known Vulnerabilities - 리콜된 부품을 계속 쓸 때

비유로 이해하기

건물 현관에 유명한 자물쇠를 달았는데, 뉴스에 “이 모델은 머리핀으로 열립니다”라고 공개되었다고 생각하자. 자물쇠 자체가 내 것이 아니어도, 그 자물쇠를 쓰는 건물은 내 책임이다.

핵심 개념

라이브러리, 프레임워크, 소프트웨어 모듈은 애플리케이션과 같은 권한으로 실행되는 경우가 많다. OWASP는 취약한 컴포넌트가 악용되면 심각한 데이터 손실이나 서버 장악으로 이어질 수 있다고 설명한다. [1]

초심자식 이해

현대 소프트웨어는 혼자 만든 집이 아니라 조립식 건물에 가깝다. 내가 직접 만든 벽이 아니어도, 부품 목록과 리콜 정보를 관리해야 한다.

레드팀 관찰 포인트

레드팀은 서버 헤더, 정적 파일, package-lock, pom, requirements, container image, CMS 플러그인처럼 버전 단서가 나오는 곳을 본다. 공개 취약점이 있다고 해서 실제 공격을 시도하는 것이 곧바로 허용되는 것은 아니다.

방어 관점

자산 목록, SBOM, 의존성 스캔, 패치 정책, 사용하지 않는 라이브러리 제거, 버전 고정과 검토, 컨테이너 이미지 업데이트, 보안 공지 구독이 기본이다.

> **암기 문장**
> 암기 문장: 남이 만든 부품도 내 서비스 안에 들어오면 내 책임이다.

A10: Insufficient Logging & Monitoring - 사고가 났는데 녹화가 없을 때

비유로 이해하기

가게에 도둑이 들었는데 CCTV는 꺼져 있고 출입 기록은 종이 메모로만 남아 있다면, 무엇이 사라졌는지 언제 알 수 있을까. 로그와 모니터링은 사고 후에야 빛나는 보험이 아니라, 사고 중에도 울려야 하는 화재 경보기다.

핵심 개념

OWASP는 부족한 로깅과 모니터링, 그리고 사고 대응 연계 부재가 공격자의 추가 공격, 지속성 확보, 측면 이동, 데이터 변조/탈취/파괴로 이어질 수 있다고 설명한다. 또한 침해 탐지에 200일 이상 걸리는 사례가 많다는 점을 언급한다. [1]

초심자식 이해

로그는 일기장이고, 모니터링은 초인종이다. 일기만 쓰고 아무도 읽지 않으면 늦고, 초인종만 있고 녹화가 없으면 나중에 설명할 수 없다.

레드팀 관찰 포인트

레드팀은 테스트 중 남긴 흔적이 탐지되는지, 경보가 누구에게 가는지, 대응 절차가 작동하는지 확인한다. 좋은 레드팀은 “들켰는지”도 결과로 보고한다.

방어 관점

로그 표준화, 인증/권한/관리자 행위 기록, 실패 이벤트 기록, 중앙 수집, 변조 방지, 경보 기준, 보존 기간, 개인정보 마스킹, 사고 대응 플레이북, 모의훈련이 기본이다.

> **암기 문장**
> 암기 문장: 보안은 막는 것뿐 아니라 빨리 알아차리는 것이다.

3.12 OWASP 항목을 한 번에 묶어 보는 법

10개 항목은 따로 외우는 것보다 “어디에서 신뢰가 깨졌는가”로 묶어 보면 쉽다.

묶음	해당 항목	질문
입력과 출력	Injection, XSS, XXE, Deserialization	외부에서 온 값이 명령이나 코드처럼 다뤄지는가?
신원과 권한	Broken Authentication, Broken Access Control	누구인지와 무엇을 해도 되는지가 구분되는가?

묶음	해당 항목	질문
설정과 부품	Security Misconfiguration, Known Vulnerable Components	기본값, 패치, 라이브러리가 안전하게 관리되는가?
데이터와 탐지	Sensitive Data Exposure, Logging & Monitoring	중요 데이터가 보호되고, 사고를 알아차릴 수 있는가?

실무 감각

레드팀 보고서에서 '취약점 이름'만 적으면 초심자 보고서다. 좋은 보고서는 '왜 위험한지, 누가 피해자인지, 어떤 통제가 없었는지, 어떻게 고칠지'를 함께 적는다.

4. 기본 해킹 기법을 순하게 이해하기

4.1 먼저 기억할 안전선

기법을 아는 것과 실행하는 것은 다르다. 이 장은 개념, 비유, 방어 관점을 익히기 위한 장이다. 실제 패킷 캡처, 스캔, 부하 테스트, 인증 우회 실험은 허가된 랩에서만 해야 한다.

초심자 안전 규칙

- 내 장비, 내 계정, 내 실습망, 명시적으로 허가받은 대상만 다룬다.
- 툴을 실행하기 전에 로그와 영향 범위를 생각한다.
- 성공보다 피해 방지가 우선이다.
- 결과는 스크린샷보다 원인, 영향, 재현 조건, 조치 방안으로 설명한다.

4.2 기본 기법 사전

정찰 Reconnaissance

비유 건물에 들어가기 전 공개된 안내판, 회사 홈페이지, 채용공고, 기술 블로그를 읽는 단계다.

개념 공개 정보 수집은 합법적 범위에서도 가능하지만, 수집 목적과 보관 방식은 윤리와 계약 범위를 따라야 한다.

방어 방어는 불필요한 정보 노출 줄이기, 공개 저장소 비밀값 점검, 직원 교육, 외부 노출 자산 목록 관리다.

스캐닝과 열거 Scanning & Enumeration

비유 복도에서 어느 문이 열리는지, 문패에 어떤 부서명이 적혀 있는지 확인하는 단계에 비유할 수 있다.

개념 스캔은 시스템에 부하를 줄 수 있으므로 허가된 범위, 시간, 속도 제한이 필요하다. 실습망 밖에서는 금지다.

방어 방어는 외부 노출 최소화, 방화벽, 서비스 배너 축소, 취약점 관리, 스캔 탐지 로그다.

스니핑 Sniffing

비유 엽서를 배달하는 길목에 서서 주소와 내용을 보는 것과 비슷하다. 암호화되지 않은 통신은 엽서에 가깝고, TLS로 보호된 통신은 봉인된 편지에 가깝다.

개념 NIST는 네트워크 스니핑을 네트워크 통신을 모니터링하고 프로토콜과 헤더/페이로드를 해석하는 수동 기법으로 정의한다. MITRE ATT&CK도 인증 자료나 환경 정보를 얻기 위해 네트워크 트래픽을 수동으로 캡처할 수 있다고 설명한다. [8][9]

방어 방어는 TLS, 안전한 Wi-Fi, 스위치/세그먼트 구성, 인증서 경고 무시 금지, 민감 프로토콜 제거, 트래픽 모니터링이다.

중간자 공격 MITM/AiTM

비유 두 사람이 대화하는 사이에 가짜 통역사가 앉아 말을 살짝 바꾸거나 몰래 듣는 상황이다.

개념 개념적으로는 통신 당사자 사이에 공격자가 끼어드는 것이다. 실제 네트워크에서 시도하면 심각한 불법 행위가 될 수 있으므로 실습 환경에서만 이해해야 한다.

방어 방어는 HTTPS, HSTS, 인증서 검증, MFA, 피싱 저항형 인증, 네트워크 신뢰 경계 축소, 사용자 교육이다.

스푸핑 Spoofing

비유 전화번호 표시를 조작해 은행이나 회사 대표번호처럼 보이게 하는 것과 비슷하다.

개념 IP, 이메일, DNS, 웹사이트, 발신자 이름 등 여러 층에서 신분이 위장될 수 있다.

방어 방어는 SPF/DKIM/DMARC, 인증서 검증, DNS 보안, 네트워크 anti-spoofing, 로그 상관분석이다.

비밀번호 공격

비유 열쇠 꾸러미를 들고 문에 하나씩 맞춰보거나, 이미 유출된 열쇠를 다른 건물에도 꽂아보는 상황이다.

개념 무차별 대입, 사전 대입, 크리덴셜 스터핑, 패스워드 스프레이는 모두 인증 체계를 시험하거나 악용하는 방식이다. 실제 계정에 무단 시도하면 안 된다.

방어 방어는 MFA, 속도 제한, 계정 잠금 정책, 유출 비밀번호 차단, 로그인 알림, 비밀번호 관리자, 긴 패스프레이즈다.

피싱과 사회공학

비유 가짜 택배 문자로 사람을 속여 문을 열게 하는 방식이다. 시스템의 버그보다 사람의 신뢰와 바쁨을 노린다.

개념 레드팀 훈련에서는 사전 승인, 대상 범위, 금지 문구, 심리적 부담 완화, 사후 교육이 중요하다.

방어 방어는 보고 버튼, 확인 절차, 콜백 문화, 보안 교육, 도메인 보호, 메일 인증, 임원 사칭 대응 훈련이다.

DoS/DDoS

비유 가게 문 앞을 인파로 막아 정상 손님이 들어오지 못하게 하는 상황이다.

개념 서비스 거부는 시스템 자원, 네트워크 대역폭, 애플리케이션 처리량을 소모시킨다. 실습도 부하를 만들 수 있어 격리된 랩이 아니면 위험하다.

방어 방어는 용량 계획, 캐싱, 속도 제한, WAF/CDN, 큐잉, 백프레셔, DDoS 대응 업체, 장애 훈련이다.

LAND 공격

비유 장난전화가 자기 번호로 자기에게 걸려 오도록 만들어 전화기가 스스로 응답하느라 바빠지는 모습에 비유할 수 있다.

개념 LAND는 스푸핑된 SYN 패킷에서 출발지와 목적지 IP가 피해자 주소로 같아지는 형태의 DoS 공격이다. Juniper 설명은 수신 시스템이 자신에게 SYN-ACK를 보내 빈 연결이 생기고, 이런 연결이 많아지면 시스템을 압도할 수 있다고 설명한다. [10]

방어 방어는 최신 TCP/IP 스택, 방화벽/IDS의 LAND 필터, 불가능한 출발지 주소 차단, 네트워크 장비 패치, anti-spoofing이다.

권한 상승 Privilege Escalation

비유 일반 사원 출입증으로 들어온 사람이 우연히 마스터키를 찾아 임원실까지 들어가는 상황이다.

개념 취약한 설정, 약한 파일 권한, 취약한 서비스, 잘못된 sudo/관리자 권한이 원인이 될 수 있다.

방어 방어는 최소권한, 패치, 비밀값 관리, 권한 변경 감사, EDR/로그, 관리자 계정 분리다.

측면 이동 Lateral Movement

비유 한 방에 들어간 뒤 복도와 내부 문을 따라 다른 방으로 이동하는 상황이다.

개념 처음 진입한 시스템 하나가 전체 내부망으로 이어지지 않게 만드는 것이 핵심이다.

방어 방어는 네트워크 분리, 내부 인증 강화, 관리자 계정 보호, 원격 접속 제한, 동서 트래픽 모니터링이다.

파일 업로드 취약점

비유 물품보관소에 맡긴 가방을 직원이 열어 보지도 않고 주방 한가운데 놓는 상황이다.

개념 업로드 파일은 이름, 확장자, 실제 내용, 저장 위치, 실행 가능성, 접근 권한을 모두 따져야 한다.

방어 방어는 파일 형식 검증, 실행 권한 제거, 웹루트 밖 저장, 무작위 파일명, 악성코드 검사, 크기 제한, 다운로드 시 안전 헤더다.

경로 탐색 Path Traversal

비유 손님이 '3번 사물함'만 열 수 있어야 하는데, 쪽지에 '위층 사무실 캐비닛'이라고 적어도 직원이 찾아주는 상황이다.

개념 파일명을 입력받아 서버 파일을 읽는 기능에서 자주 생긴다. 실제 경로를 추측해 접근하는 행위는 무단이면 불법이다.

방어 방어는 허용된 디렉터리 고정, 경로 정규화, 파일 ID 매핑, 권한 분리, 민감 파일 접근 차단이다.

CSRF

비유 식당 손님이 주문하지 않았는데 옆 사람이 웨이터에게 '저 사람 카드로 결제해 주세요'라고 말하는 상황이다.

개념 브라우저가 로그인 쿠키를 자동으로 붙여 보내는 특성을 악용할 수 있다.

방어 방어는 CSRF 토큰, SameSite 쿠키, 중요한 요청의 재확인, GET 요청으로 상태 변경 금지다.

SSRF

비유 외부 손님이 사내 배달 직원에게 '내부 금고 옆 방에서 서류를 가져와 달라'고 부탁하는 상황이다.

개념 서버가 대신 URL을 가져오는 기능에서 내부망이나 메타데이터 서비스에 접근할 위험이 생긴다.

방어 방어는 목적지 허용 목록, 내부 IP 차단, 리다이렉트 제한, egress 제어, 클라우드 메타데이터 보호다.

악성코드, C2, 유출

비유 집 안에 몰래 들어온 작은 로봇이 주인의 지시가 아니라 외부 리모컨 신호를 듣고 물건을 밖으로 옮기는 상황이다.

개념 이 문서에서는 제작이나 운용 방법을 다루지 않는다. 초심자는 감염 경로, 명령 통신, 권한, 탐지 신호를 개념적으로 이해하면 충분하다.

방어 방어는 EDR, 애플리케이션 제어, 이메일/웹 필터링, 백업, 네트워크 egress 감시, 최소권한, 사고 대응 훈련이다.

4.3 공격 흐름을 영화처럼 보지 말고 업무 흐름으로 보기

영화에서는 공격이 한 번의 멋진 명령으로 끝나는 것처럼 보이지만, 실제 보안 평가는 훨씬 더 지루하고 문서적이다. 범위 확인, 자산 파악, 안전한 테스트 설계, 취약점 검증, 영향도 산정, 보고, 재검증이 반복된다.

단계	공격자 관점 질문	방어자 관점 질문
정찰	무엇이 밖에 보이는가?	불필요하게 공개된 정보가 있는가?
초기 접근	어떤 문이 약한가?	어떤 인증과 입력 검증이 필요한가?
권한 확대	더 큰 권한을 얻을 수 있는가?	최소권한이 지켜지는가?
이동	다른 시스템으로 이어지는가?	망 분리와 내부 탐지가 있는가?
목표 달성	데이터 접근이나 서비스 영향이 가능한가?	중요 자산이 보호되는가?
탐지/대응	나의 행동이 보이는가?	경보와 플레이북이 작동하는가?

초심자에게 권하는 실습 방향

공격 명령을 많이 외우기보다, DVWA 같은 취약 앱에서 '요청이 어떻게 바뀌면 서버가 어떻게 반응하는지'를 관찰하라. 그리고 같은 취약점을 고친 코드나 설정을 비교하라. 이것이 실무 이해에 훨씬 오래 남는다.

5. DVWA란 무엇이고 어떻게 안전하게 설치하는가

5.1 DVWA란 무엇인가

DVWA(Damn Vulnerable Web Application)는 일부러 취약하게 만든 웹 애플리케이션이다. 공식 GitHub는 DVWA의 목적을 간단한 인터페이스와 여러 난이도로 흔한 웹 취약점을 연습하는 것이라고 설명한다. 동시에 “매우 취약하므로 인터넷에 공개된 서버나 호스팅 공간에 올리지 말라”고 경고한다. [4]

> **비유**
>
> DVWA는 실제 도로가 아니라 운전 연습장이다. 일부러 표지판도 틀리고 장애물도 놓여 있다. 그래서 초보 운전자가 사고 없이 위험 상황을 배울 수 있다.

5.2 DVWA를 쓰는 이유

- 실제 서비스를 건드리지 않고 취약점의 모양을 배울 수 있다.
- 난이도를 낮음에서 높음으로 바꾸며 같은 기능이 어떻게 달라지는지 볼 수 있다.
- 입력값, 요청, 응답, 서버 코드, 방어 코드를 함께 비교하기 좋다.
- 보고서 작성 연습에 적합하다. “취약점 발견 - 영향 - 원인 - 조치” 흐름을 반복할 수 있다.

5.3 설치 전 안전 설계

DVWA는 이름 그대로 취약하다. 따라서 설치보다 격리가 먼저다. 초심자는 아래 구조를 권장한다.

선택	권장 여부	이유
내 노트북의 Docker Desktop에서 localhost로만 실행	권장	설정이 간단하고 외부 노출을 줄일 수 있다
VirtualBox/VMware의 Kali VM 안에서 실행	권장	실습 환경을 스냅샷으로 되돌리기 쉽다
클라우드 서버나 공인 IP에 설치	비권장	취약 앱이 인터넷에 노출되어 실제 침해로 이어질 수 있다
회사/학교 네트워크에 연결된 서버에 설치	비권장	내부망 위험과 정책 위반 가능성이 있다

5.4 Docker 기반 설치 절차

공식 DVWA GitHub는 Docker와 Docker Compose를 전제 조건으로 안내하고, 저장소를 받은 뒤 해당 폴더에서 Docker Compose로 실행하면 기본적으로 `http://localhost:4280`에서 접근된다고 설명한다. [4]

```
# 1) Docker와 Compose가 보이는지 확인
# 출력에 버전 정보가 보이면 준비 완료입니다.
docker version
docker compose version

# 2) 공식 저장소 받기
git clone https://github.com/digininja/DVWA.git
cd DVWA

# 3) DVWA 실행
docker compose up -d

# 4) 브라우저에서 접속
# http://localhost:4280
```

로그인과 초기화

공식 문서의 기본 계정은 username = admin, password = password이다. 로그인 뒤 메뉴에서 Setup DVWA를 열고 Create / Reset Database를 눌러 실습 데이터를 만든다. 기본 계정은 실습용이므로 공개 환경에서는 절대 사용하면 안 된다. [4]

5.5 멈추는 방법과 정리 방법

```
# 잠시 멈춤
docker compose stop

# 컨테이너 제거. 실습 데이터도 정리될 수 있으니 필요한 기록은 먼저 남기세요.
docker compose down

# 로그 확인. 오류가 날 때 원인을 보는 용도입니다.
docker compose logs
```

5.6 DVWA 학습 순서

1. **로그인과 화면 구조 익히기:** 메뉴, 보안 레벨, Setup DVWA 위치를 확인한다.
2. **HTTP 요청 관찰:** 브라우저 개발자 도구나 프록시를 이용해 요청과 응답이 어떻게 생겼는지 본다. 이때 외부 사이트가 아니라 DVWA만 대상으로 한다.
3. **취약점 설명 읽기:** 먼저 설명을 읽고, “무엇을 믿었기 때문에 위험해졌는가”를 적는다.

4. **낮은 난이도에서 개념 확인:** 공격 성공보다 입력과 응답의 관계를 이해하는 데 집중한다.
5. **높은 난이도와 비교:** 방어 코드나 설정이 어떻게 바뀌었는지 비교한다.
6. **보고서 작성:** 취약점 이름, 영향, 재현 조건, 원인, 조치, 재검증 결과를 기록한다.

5.7 자주 막히는 지점

증상	확인할 것	비유
localhost:4280이 열리지 않음	컨테이너가 실행 중인지, 포트가 충돌하지 않는지 확인	가게는 열었는데 간판 주소를 잘못 본 상황
로그인이 안 됨	기본 계정 admin/password인지 확인, DB 초기화 여부 확인	출입증은 있는데 방문자 명단이 아직 비어 있는 상황
DB 오류	Setup DVWA에서 Create / Reset Database, 로그 확인	주방은 열었지만 장부가 아직 없는 상황
너무 쉽게 풀림	보안 레벨을 Medium/High/Impossible로 변경	연습장 장애물 난이도를 올리는 상황

절대 하지 말 것

- DVWA를 공인 IP, 포트포워딩, 회사/학교 서버에 공개하지 않는다.
- 기본 계정을 공개망에 노출하지 않는다.
- DVWA에서 배운 요청을 실제 서비스에 복사해 시험하지 않는다.
- 실습 스크린샷에 개인 토큰이나 실제 계정 정보를 포함하지 않는다.

5.8 DVWA 실습 노트 양식

항목	작성 예시
실습명	DVWA - SQL Injection 개념 이해
목표	입력값이 질의에 섞일 때 왜 위험한지 설명할 수 있다
환경	로컬 Docker, http://localhost:4280, 보안 레벨 Low
관찰	입력값과 응답이 연결되는 지점, 오류 메시지, 결과 변화
원인	데이터와 명령의 분리 부족
영향	권한 없는 데이터 조회 가능성
권고	매개변수화된 질의, 최소권한, 오류 메시지 축소
재검증	보안 레벨 또는 수정 코드에서 동일 문제가 줄어드는지 확인

6. Kali Linux와 “Kali Red” 표현 정리

6.1 Kali Linux란 무엇인가

Kali Linux는 정보보안 업무를 위해 만들어진 오픈소스 Debian 기반 Linux 배포판이다. Kali 공식 사이트는 침투 테스트, 보안 연구, 컴퓨터 포렌식, 리버스 엔지니어링 등 다양한 정보보안 작업에 맞춰져 있다고 설명한다. [5]

> **비유**
>
> Kali는 공구상자다. 망치, 드라이버, 전기 테스터가 들어 있지만, 공구상자를 들었다고 곧바로 기술자가 되는 것은 아니다. 공구보다 작업 허가, 안전수칙, 측정 방법, 보고서가 먼저다.

6.2 왜 초심자에게 VM을 권하는가

초심자는 Kali를 노트북에 바로 설치하기보다 VirtualBox, VMware, QEMU 같은 가상머신에서 시작하는 것이 안전하다. Kali 문서는 VirtualBox 안에서 Kali VM을 쓰면 호스트와 분리되고, 다른 VM과 상호작용할 수 있으며, 스냅샷으로 되돌리기 쉽다고 설명한다. [6]

방식	초심자 추천도	장점	주의점
Pre-built VM	높음	빠르게 시작, 스냅샷 편리	공식 출처와 체크섬 확인
Installer ISO	중간	설치 과정을 배움	파티션/부트 실수 주의
Live USB	중간	호스트 OS 변경 적음	저장 방식과 네트워크 권한 이해 필요
Bare metal	낮음	하드웨어 직접 접근	초심자에게 복구 부담이 큼
Docker	상황별	빠르고 가벼움	GUI/커널/네트워크 실습에는 한계

6.3 공식 VM의 기본 계정

Kali의 사전 구축 VM 이미지는 기본 자격 증명으로 `kali/kali`를 안내한다. 실습용 기본값이므로, 장기 사용하거나 네트워크에 연결한다면 비밀번호 변경과 업데이트가 필요하다. [7]

6.4 “Kali Red”는 공식 배포판 이름인가

보통 “Kali Red”라는 표현은 공식 배포판 이름이라기보다, Kali Linux를 레드팀/공격 보안 학습 용도로 쓰는 맥락에서 비공식적으로 부르는 말에 가깝다. 공식적으로 널리 쓰이는 이름은 Kali Linux이며, 방어 보안과 SOC

실습 맥락에서는 Kali Purple 프로젝트가 별도로 언급된다. Kali Purple 공식 GitLab 위키는 이를 “SOC-in-a-box” 커뮤니티 프로젝트로 소개한다. [12]

표현	정리	초심자에게 필요한 이해
Kali Linux	공식 배포판	정보보안 작업용 Linux 공구상자
Kali Red	일반적으로 비공식 표현	Kali를 레드팀/공격 보안 관점으로 쓴다는 의미로 이해
Kali Purple	방어/SOC/퍼플팀 학습 맥락의 프로젝트	탐지, 로그, 방어 실습을 함께 보는 방향

6.5 초심자용 Kali 실습망 설계

가장 안전한 구조는 “공격자 역할 Kali VM”과 “취약한 연습 대상 DVWA”를 같은 로컬 실습망 안에 두고, 취약한 대상이 인터넷에서 보이지 않게 하는 것이다.

구성요소	역할	권장 설정
Host OS	내 실제 노트북	중요 자료 백업, 관리자 권한 남용 금지
Kali VM	학습용 분석/테스트 도구 상자	NAT 또는 Host-only, 스냅샷 사용
DVWA	취약한 연습 대상	localhost 또는 실습망 내부에서만 접근
메모 공간	보고서와 증적 정리	날짜, 환경, 변경사항, 배운 점 기록

6.6 Kali 도구를 카테고리로만 먼저 이해하기

초심자는 개별 도구 명령어보다 도구의 역할 분류를 먼저 이해해야 한다.

도구 범주	역할	비유
정보 수집	대상 구조와 공개 정보를 파악	건물 안내도 보기
웹 프록시	브라우저와 서버 사이 요청/응답 관찰	주문서가 오가는 창구에서 복사본 보기
패킷 분석	네트워크 흐름 관찰	우편물의 봉투와 흐름 보기
취약점 스캐너	알려진 문제 후보를 자동 점검	건물 점검표로 빠르게 훑기
익스플로잇 프레임워크	취약점 검증 자동화	위험한 전동 공구. 허가된 랩에서만
포렌식/리버싱	파일, 메모리, 프로그램 동작 분석	사고 현장 감식
보고/문서화	증적과 개선안을 정리	점검 보고서 작성

6.7 초심자 Linux 생존 명령

다음은 공격 명령이 아니라 Linux 환경을 다루기 위한 기본 명령이다.

```
pwd         # 현재 위치 보기
ls          # 파일 목록 보기
cd 폴더명   # 폴더 이동
cat 파일명  # 짧은 파일 내용 보기
less 파일명 # 긴 파일을 천천히 보기
mkdir notes # 폴더 만들기
cp a b      # 파일 복사
mv a b      # 파일 이동 또는 이름 변경
rm -i 파일명 # 삭제 전 확인하며 삭제
ip addr     # 내 네트워크 주소 확인
history     # 이전에 입력한 명령 확인
```

Kali 초심자 금지 습관

- 인터넷 글에서 본 명령을 뜻도 모르고 복사해 붙여넣지 않는다.
- 관리자 권한 sudo를 습관처럼 붙이지 않는다.
- 대상 주소가 내 실습망인지 확인하지 않고 도구를 실행하지 않는다.
- 결과가 나왔다고 곧바로 성공이라고 생각하지 않는다. 오탐과 환경 차이를 확인한다.

7. 부트캠프식 학습 로드맵

7.1 4주 부트캠프 로드맵

아래 로드맵은 “툴 사용량”이 아니라 “이해한 개념을 말과 그림으로 설명할 수 있는가”를 기준으로 설계했다.

주차	목표	읽을 내용	실습	산출물
0주차	윤리, 법적 범위, Linux와 웹 기초	이 문서 1-2장	Kali VM 또는 Docker 준비, DVWA 설치	실습 환경 캡처, 안전 체크리스트
1주차	OWASP A1-A4 이해	Injection, Auth, Data Exposure, XXE	DVWA에서 요청/응답 관찰	각 취약점 한 페이지 요약
2주차	OWASP A5-A10 이해	Access Control, Misconfig, XSS, Deserialization, Components, Logging	난이도별 차이 비교	방어 체크리스트
3주차	네트워크와 기본 기법 이해	스니핑, MITM, DoS, LAND, 피싱 개념	내 실습망에서만 트래픽 흐름 관찰	위험-통제 매핑표
4주차	보고와 퍼플팀 관점	탐지, 로그, 재검증	가상의 취약점 보고서 작성	최종 보고서와 발표자료

7.2 하루 학습 루틴

1. **10분:** 오늘 다룰 용어를 비유로 설명한다.
2. **20분:** 공식 문서나 치트시트를 읽는다.
3. **30분:** DVWA 같은 실습망에서 요청/응답을 관찰한다.
4. **20분:** 원인, 영향, 방어를 노트로 정리한다.
5. **10분:** “오늘 내가 실제 서비스에서 하면 안 되는 행동”을 적는다.

좋은 질문 예시

- 이 취약점은 어떤 신뢰가 깨진 것인가?
- 피해자는 사용자, 관리자, 서비스 운영자 중 누구인가?
- 공격자가 얻는 것은 데이터, 권한, 지속성, 서비스 중단 중 무엇인가?
- 방어자는 어떤 로그로 알아차릴 수 있는가?
- 개발자는 어느 줄의 코드 또는 어느 설정을 바꾸어야 하는가?

7.3 초심자에게 권하는 읽을거리

- OWASP Top 10 2017 공식 페이지: 항목별 위험의 큰 그림을 잡는다. [1]
- OWASP Cheat Sheet Series: 방어 관점의 실무 지침을 찾아본다. [3]
- DVWA GitHub README: 취약 앱을 안전하게 설치하고 경고사항을 확인한다. [4]
- Kali 공식 문서: 설치와 가상화, Docker, 기본 정책을 공식 출처에서 확인한다. [5][6][7]
- MITRE ATT&CK: 공격 기법을 탐지와 방어 언어로 번역하는 연습을 한다. [9]

7.4 실습 난이도 조절법

너무 어려우면 공격을 성공시키려 하지 말고 관찰 범위를 줄인다. 예를 들어 XSS가 어렵다면 “입력한 글자가 응답 HTML의 어디에 나타나는가”만 본다. Injection이 어렵다면 “사용자가 입력한 값이 DB 질의에 들어간다는 사실”만 이해해도 첫 단계는 성공이다.

너무 쉬우면 같은 취약점을 방어 관점으로 바꾸어 본다. “왜 Low에서는 위험했고, High 또는 Impossible에서는 왜 줄었는가?”를 설명하면 단순 풀이보다 훨씬 깊어진다.

8. 체크리스트, 보고서 템플릿, 용어집

8.1 실습 전 허가 체크리스트

체크	질문
범위	대상 IP, 도메인, 계정, 시간대가 명확한가?
허가	문서나 티켓으로 승인 근거가 남아 있는가?
금지행위	부하 테스트, 피싱, 계정 잠금 유발, 데이터 삭제 등 금지 항목이 정해졌는가?
연락망	장애가 나면 누구에게 즉시 연락하는가?
증적	민감정보를 최소 수집하고 안전하게 보관하는가?
복구	실습 환경을 중지하거나 되돌릴 방법이 있는가?

8.2 랩 격리 체크리스트

- DVWA, Metasploitable, WebGoat 같은 취약 대상은 인터넷에 노출하지 않는다.
- VM 스냅샷을 만든 뒤 실습한다.
- 실습 계정과 실제 개인/업무 계정을 분리한다.
- 실습 브라우저 프로필을 따로 만든다.
- 토큰, 쿠키, 비밀번호가 캡처 화면에 나오면 마스킹한다.
- 실습이 끝나면 컨테이너나 VM을 멈춘다.

8.3 취약점 보고서 템플릿

항목	작성 가이드
제목	무엇이 어디에서 잘못되었는지 한 줄로 쓴다
요약	비기술 담당자도 이해할 수 있게 피해 가능성을 설명한다
영향도	데이터 노출, 권한 상승, 서비스 중단, 계정 탈취 등으로 분류한다
대상/범위	어떤 URL, 기능, 계정, 환경에서 확인했는지 적는다
재현 조건	허가된 환경에서 필요한 최소 단계만 적는다
증적	스크린샷, 요청/응답 일부, 로그. 민감정보는 마스킹한다
원인	입력 검증 부재, 권한 검사 누락, 설정 오류 등 근본 원인을 적는다

항목	작성 가이드
권고	개발자가 실행할 수 있는 조치로 쓴다
재검증	수정 후 어떤 기준으로 해결을 확인할지 적는다

8.4 위험도 산정의 쉬운 방법

초심자는 CVSS 같은 정교한 점수를 바로 쓰기보다 아래 질문으로 먼저 감을 잡으면 좋다.

질문	낮음	중간	높음
공격 난이도	특수 조건이 많음	일부 조건 필요	쉽고 반복 가능
필요 권한	관리자 필요	일반 로그인 필요	비로그인 가능
피해 범위	일부 화면	일부 사용자/데이터	다수 사용자/핵심 데이터/서버
탐지 가능성	즉시 경보	로그 확인 필요	거의 흔적 없음
복구 난이도	설정 수정	코드/데이터 정리	침해 조사와 대규모 복구

8.5 초심자 용어집

용어	쉬운 뜻
취약점	시스템의 약한 부분
익스플로잇	약한 부분을 실제로 이용하는 방법 또는 코드
페이로드	취약점 확인 과정에서 전달되는 실제 내용. 이 문서에서는 구체적 공격 페이로드를 다루지 않는다
인증	너 누구야?를 확인
인가/권한	너 이거 해도 돼?를 확인
세션	로그인 상태를 기억하는 장치
토큰	권한이나 신원을 나타내는 표식
해시	원본을 되돌리기 어렵게 만든 지문 같은 값
암호화	키가 있어야 읽을 수 있게 만드는 것
로그	시스템의 일기장
탐지	이상한 일이 일어났음을 알아차리는 것
오탐	문제가 아닌데 문제라고 보는 것
미탐	문제인데 놓치는 것
스코프	테스트해도 되는 범위

용어	쉬운 뜻
하드닝	기본 설정을 안전하게 조이는 작업

8.6 마지막으로 남길 문장

> **레드팀 초심자의 기준**
> 좋은 레드팀원은 '뚫었다'고 자랑하는 사람이 아니라, 조직이 더 안전해지도록 원인과 개선책을 정확히 설명하는 사람이다.

참고문헌

아래 출처를 중심으로 사실관계를 확인했다. URL은 PDF에서 복사해 브라우저에 붙여 넣을 수 있도록 원문 주소를 그대로 표기했다.

1. [1] OWASP Top 10 2017.
   https://owasp.org/www-project-top-ten/2017/Top_10
2. [2] OWASP Top 10 Project.
   https://owasp.org/www-project-top-ten/
3. [3] OWASP Cheat Sheet Series.
   https://cheatsheetseries.owasp.org/
4. [4] DVWA 공식 GitHub.
   https://github.com/digininja/DVWA
5. [5] Kali Linux 공식 사이트.
   https://www.kali.org/
6. [6] Kali Linux VirtualBox 문서.
   https://www.kali.org/docs/virtualization/install-virtualbox-guest-vm/
7. [7] Get Kali - Pre-built Virtual Machines.
   https://www.kali.org/get-kali/
8. [8] NIST CSRC - Network Sniffing.
   https://csrc.nist.gov/glossary/term/network_sniffing
9. [9] MITRE ATT&CK T1040 - Network Sniffing.
   https://attack.mitre.org/techniques/T1040/
10. [10] Juniper LAND attack 설명.
    https://circitor.fr/Mibs/Html/JUNIPER-JS-SCREENING-MIB.php
11. [11] OWASP WebGoat.
    https://owasp.org/www-project-webgoat/
12. [12] Kali Purple 공식 GitLab 위키.
    https://gitlab.com/kalilinux/kali-purple/documentation/-/wikis/home

출처 사용 방식

OWASP 항목의 정의와 위험 분류는 OWASP Top 10 2017을 기준으로 삼았다. 방어 체크리스트는 OWASP Cheat Sheet Series의 방향성과 일반 보안 모범 사례를 초심자용 언어로 재구성했다. DVWA와 Kali 설치 관련 세부사항은 공식 GitHub와 공식 Kali 문서를 기준으로 했다.