해킹개론 읽을거리

OWASP Top 10 2017, 기본 공격 기법, DVWA, Kali Linux - 정보보안 부트캠프 레드팀 입문

작성: GPT-5.5 Pro

2026-06-23

목차

1 사용 범위와 읽는 법

이 문서는 정보보안 부트캠프의 레드팀 입문자가 읽는“해킹개론”자료를 목표로 한다. 주제는 네 가지다.

1. OWASP Top 10 2017 의 각 항목을 공격 관점과 방어 관점에서 정리한다.
2. LAND 공격, 스니핑, 스캐닝, 인증 공격, 웹 취약점 등 기본 해킹 기법을 개념 중심으로 설명한다.
3. DVWA 가 무엇인지, 어떻게 설치하고 안전하게 실습하는지 정리한다.
4. “Kali Red”라는 표현을 Kali Linux 와 Kali Purple 의 관계 속에서 정리하고, Kali Linux 사용법을 입문 수준에서 설명한다.

이 문서는 실전 공격 매뉴얼이 아니다. 모든 실습은 본인이 소유하거나 명시적으로 허가받은 시스템, 또는 DVWA 같은 취약점 실습용 랩에서만 수행해야 한다. 무단 스캐닝, 무단 패킷 캡처, 무단 계정 공격, 타인 네트워크에서의 중간자 공격, 서비스 거부 유발 행위는 법적 책임을 만들 수 있다.

OWASP Top 10 은 개발자와 웹 애플리케이션 보안을 위한 표준적인 인식 문서로 설명되며, 웹 애플리케이션의 중요한 보안 위험에 대한 폭넓은 합의를 제공한다 [S1]. 현재 OWASP Top 10 최신판은 별도로 존재하지만, 이 문서는 요청 범위에 맞춰 2017 판을 기준으로 정리한다. OWASP 2017 판은 A1 부터 A10 까지의 위험 항목과 각 위험의 개념, 취약 여부 판단 기준, 예방 방법을 제공한다 [S2].

1.1 이 문서의 원칙

- 공격 절차보다 원리, 징후, 방어, 보고를 우선한다.
- 도구 이름은 학습 맥락에서 소개하되, 인터넷 대상 무단 공격에 바로 쓰이는 절차는 제공하지 않는다.
- DVWA 설치처럼 명시적 실습 환경을 만드는 절차는 구체적으로 제공한다.
- 취약점은“찾는 것”보다“허가된 범위에서 증명하고 안전하게 보고하는 것”이 중요하다.
- 레드팀은 공격자 흉내만 내는 팀이 아니라, 조직의 방어 체계를 검증하고 개선점을 증거로 제시하는 팀이다.

1.2 빠른 학습 경로

처음 읽는 사람은 다음 순서를 권장한다.

1. 1 장과 2 장을 읽고 합법적 범위와 기본 용어를 잡는다.
2. 3 장의 OWASP Top 10 2017 전체 요약을 훑는다.
3. 4 장의 기본 해킹 기법을 읽으며 네트워크, 웹, 인증, 운영체제 계층의 차이를 이해한다.
4. 5 장에서 DVWA 를 로컬 랩으로 설치한다.
5. 6 장에서 Kali Linux 를 VM 으로 구성한다.
6. 7 장의 체크리스트를 이용해 실습 보고서 형식으로 정리한다.

2 레드팀 입문자가 먼저 알아야 할 큰 그림

2.1 레드팀, 펜테스트, 취약점 진단의 차이

세 용어는 비슷하게 쓰이지만 목적과 산출물이 다르다.

취약점 진단은 시스템, 애플리케이션, 네트워크의 알려진 약점과 설정 오류를 찾는 활동이다. 자동 스캐너, 설정 점검, 버전 점검, 코드 리뷰, 인증/권한 체크가 포함된다. 산출물은 보통 취약점 목록, 위험도, 조치 권고다.

침투 테스트는 발견한 취약점이 실제로 보안 목표를 우회할 수 있는지 제한된 범위에서 검증한다. NIST SP 800-115 는 기술적 보안 시험과 평가를 계획, 수행, 분석하고 완화 전략을 개발하기 위한 실무 지침을 제공한다 [S22]. 즉, 취약점이“있어 보인다”에서 멈추지 않고“이 취약점이 어떤 영향으로 이어지는가”를 확인하는 활동이다.

레드팀은 더 넓다. 현실적인 공격자 시나리오를 바탕으로 탐지, 대응, 로그, 보안 운영, 의사결정, 물리/인적 절차까지 검증한다. 레드팀의 성공은“뚫었다”가 아니라“조직이 몰랐던 방어 격차를 안전하게 증명했고, 개선 가능한 형태로 전달했다”에 있다.

2.2 기본 공격 흐름

전형적인 공격 흐름은 다음처럼 단순화할 수 있다. 실제 업무에서는 범위와 규칙에 따라 일부 단계가 빠지거나 순서가 달라진다.

1. **범위 확인**: 대상 IP, 도메인, 계정, 테스트 시간, 금지 행위, 연락망, 중단 조건을 확인한다.
2. **정찰**: 공개 정보, DNS, 인증서, 노출 서비스, 애플리케이션 구조를 파악한다.
3. **스캐닝/열거**: 살아 있는 호스트, 포트, 서비스, 버전, 기술 스택, 사용자/권한 경계를 찾는다.
4. **취약점 분석**: 입력 검증, 인증, 권한, 암호화, 설정, 컴포넌트, 로그 부재를 확인한다.
5. **제한된 검증**: 허가된 범위에서 취약점의 영향도를 최소한의 증거로 입증한다.
6. **권한/영향 분석**: 노출 데이터, 계정 권한, 내부 이동 가능성, 업무 영향도를 정리한다.
7. **보고와 개선**: 재현 조건, 영향, 원인, 위험도, 조치, 검증 방법을 제시한다.

MITRE ATT&CK 는 공격자가 능동적으로 대상 인프라를 탐색하는 활동을 Active Scanning 으로 분류하며, 이는 직접적인 네트워크 트래픽으로 정보를 수집하는 정찰 방식이다 [S25]. 이 관점은 스캐닝이 단순한 도구 사용이 아니라 탐지와 법적 범위가 얽힌 행위임을 보여준다.

2.3 허가 범위가 가장 중요한 이유

보안 교육에서 가장 먼저 배워야 하는 문장은“대상이 내 것이거나 허가받은 것인가”이다. 다음 기준을 충족하지 못하면 실습을 중단해야 한다.

- 대상이 본인 소유, 강의 랩, CTF, 버그바운티 범위, 계약서 범위 중 하나인가?
- 테스트 시간, 테스트 IP, 계정, 금지 기술, 서비스 거부 금지, 데이터 열람 금지 조건이 명확한가?
- 의도치 않은 장애가 발생했을 때 연락할 담당자와 중단 절차가 있는가?
- 실습 로그와 증거를 안전하게 보관하고 민감 정보를 마스킹할 준비가 되어 있는가?

레드팀의 기술 역량은“무엇을 할 수 있는가”가 아니라“무엇을 해도 되는지 정확히 알고, 필요한 만큼만 증명하는가”에서 평가된다.

2.4 보안 테스트의 산출물

초급 레드팀원이 처음부터 익혀야 할 산출물은 다음과 같다.

- **대상과 범위**: 무엇을 테스트했고 무엇은 테스트하지 않았는가.
- **취약점 요약**: 취약점명, 위치, 심각도, CWE/OWASP 매핑, 발견 방법.
- **영향 설명**: 공격자가 얻는 권한, 접근 가능한 데이터, 업무 영향.
- **증거**: 스크린샷, 로그, 요청/응답 일부, 설정값. 단, 비밀번호, 토큰, 주민번호 등은 마스킹한다.
- **재현 조건**: 담당자가 안전하게 재확인할 수 있는 최소 조건. 인터넷 전체에 적용 가능한 공격 레시피가 아니라 해당 시스템의 검증 정보여야 한다.
- **조치 방안**: 근본 원인과 코드/설정/운영 개선책.
- **재검증 기준**: 조치 후 어떤 상태면 해결로 볼 것인가.

3 OWASP Top 10 2017 전체 요약

3.1 OWASP Top 10 2017 의 위치

OWASP Top 10 은 취약점 백과사전이 아니라 위험 인식 문서다. 2017 판은 웹 애플리케이션에서 반복적으로 나타나는 중요한 위험을 10 개 범주로 묶었다. OWASP 는 각 항목에 대해 공격 가능성, 취약점의 보편성, 탐지 난이도, 기술적 영향, 비즈니스 영향을 함께 고려하도록 안내한다 [S3].

2017 판의 항목은 다음과 같다 [S2].

번호	항목	핵심 한 줄
A1	Injection	신뢰할 수 없는 데이터가 명령이나 쿼리로 해석된다.
A2	Broken Authentication	인증과 세션 관리 결함으로 계정 탈취가 가능해진다.
A3	Sensitive Data Exposure	민감 데이터가 저장, 전송, 처리 중 충분히 보호되지 않는다.
A4	XML External Entities (XXE)	취약한 XML 파서가 외부 엔터티를 처리해 파일 노출, SSRF, DoS 를 유발한다.
A5	Broken Access Control	사용자가 허용된 권한 밖의 기능이나 데이터에 접근한다.
A6	Security Misconfiguration	기본 설정, 과도한 오류 메시지, 불필요한 기능, 패치 누락 등이 위험을 만든다.
A7	Cross-Site Scripting (XSS)	브라우저가 공격자의 스크립트를 피해자 권한으로 실행한다.
A8	Insecure Deserialization	조작된 직렬화 객체를 역직렬화해 로직 우회나 원격 코드 실행으로 이어진다.
A9	Using Components with Known Vulnerabilities	취약한 라이브러리와 프레임워크가 애플리케이션 권한으로 실행된다.
A10	Insufficient Logging & Monitoring	공격이 기록, 탐지, 대응되지 않아 침해가 장기화된다.

3.2 위험 점수 요약

OWASP 2017 의 Risk Factor Summary 는 각 항목을 일반화된 위험 점수로 정리한다. 이 점수는 모든 조직에 그대로 적용되는 절대 순위가 아니다. OWASP 도 실제 위험은 조직의 위협 행위자와 비즈니스 영향을 함께 고려해야 한다고 설명한다 [S3].

번호	항목	OWASP 2017 점수
A1	Injection	8.0
A2	Broken Authentication	7.0
A3	Sensitive Data Exposure	7.0
A4	XML External Entities (XXE)	7.0
A5	Broken Access Control	6.0
A6	Security Misconfiguration	6.0
A7	Cross-Site Scripting (XSS)	6.0
A8	Insecure Deserialization	5.0
A9	Using Components with Known Vulnerabilities	4.7
A10	Insufficient Logging & Monitoring	4.0

3.3 A1:2017 - Injection

3.3.1 개념

Injection 은 사용자 입력, HTTP 헤더, 쿠키, JSON, XML, 파일, 내부 서비스 응답처럼 신뢰할 수 없는 데이터가 SQL, NoSQL, LDAP, OS 명령, XML 파서, ORM 질의 등의“명령 구조”안으로 섞일 때 발생한다. OWASP 는 SQL, LDAP, XPath, NoSQL 쿼리, OS 명령, XML 파서, SMTP 헤더, 표현식 언어, ORM 쿼리 등에서 Injection 이 발견된다고 설명한다 [S4].

핵심은“데이터와 명령의 분리 실패”다. 애플리케이션은 사용자 입력을 단순한 값으로 처리해야 하지만, 취약한 구현은 입력을 쿼리나 명령의 일부로 해석한다. 그 결과 공격자는 원래 의도되지 않은 조회, 변경, 삭제, 인증 우회, 파일 접근, 명령 실행을 유도할 수 있다.

3.3.2 흔한 원인

- 문자열 결합으로 SQL 이나 OS 명령을 만든다.
- 파라미터 바인딩 없이 동적 쿼리를 실행한다.
- 입력 검증을 클라이언트 JavaScript 에만 의존한다.
- 오류 메시지가 내부 쿼리 구조를 노출한다.
- ORM 을 쓰더라도 검색 조건에 사용자 입력을 구조적으로 삽입한다.
- XML, LDAP, 템플릿 엔진, 표현식 언어 같은 덜 눈에 띄는 인터프리터를 간과한다.

3.3.3 영향

Injection 은 단순 정보 노출에서 끝나지 않는다. 데이터베이스 전체 유출, 데이터 변조, 인증 우회, 로그 변조, 서비스 거부, 경우에 따라 호스트 장악으로 이어질 수 있다. OWASP 도 Injection 이 데이터 손실, 부패, 무단 공개, 책임성 상실, 접근 거부, 때로는 완전한 호스트 탈취로 이어질 수 있다고 설명한다 [S4].

3.3.4 방어 원칙

- 데이터와 명령을 분리한다. SQL 은 Prepared Statement 와 파라미터 바인딩을 기본으로 사용한다.
- 허용 목록 기반 입력 검증을 적용한다. 예를 들어 숫자 ID 는 숫자만 허용하고, 정렬 컬럼은 사전 정의된 컬럼명만 허용한다.
- DB 계정 권한을 최소화한다. 웹 애플리케이션 계정이 불필요하게 DDL, 파일 접근, 관리자 권한을 갖지 않게 한다.
- 오류 메시지를 사용자에게 상세히 노출하지 않는다.
- SAST, DAST, 코드 리뷰를 CI/CD 에 포함한다.
- WAF 는 보조 방어일 뿐이며, 안전한 코드가 우선이다.

3.3.5 레드팀 관찰 포인트

- 입력값이 서버에서 어떤 인터프리터로 전달되는지 추적한다.
- 오류 메시지, 응답 시간 차이, 결과 집합 변화, 상태 코드 변화를 본다.
- 검증은 최소 영향으로 수행한다. 데이터 삭제, 대량 조회, 시스템 명령 실행 같은 고위험 행위는 사전 허가 없이는 하지 않는다.
- 보고서에는 취약 입력 위치, 영향받는 데이터, 필요한 권한, 권장 수정 방식을 명확히 쓴다.

3.4 A2:2017 - Broken Authentication

3.4.1 개념

Broken Authentication 은 로그인, 세션, 비밀번호 재설정, 계정 복구, 토큰 관리, MFA, 로그아웃 같은 인증 기능의 설계나 구현이 잘못되어 계정 탈취가 가능해지는 상태를 말한다. OWASP 는 인증과 세션 관리가 잘못 구현되면 공격자가 비밀번호, 키, 세션 토큰을 침해하거나 다른 사용자의 신원을 가장할 수 있다고 설명한다 [S2].

3.4.2 흔한 원인

- 기본 계정이나 기본 비밀번호가 남아 있다.
- 약한 비밀번호를 허용한다.
- 로그인 실패 제한, 지연, 탐지, 알림이 없다.
- 세션 ID 가 URL 에 노출된다.
- 로그인 후 세션 ID 가 재발급되지 않는다.
- 로그아웃 또는 비활성 시간 후 세션이 무효화되지 않는다.
- 비밀번호 재설정 질문이 추측 가능하다.
- MFA 가 없거나 고위험 작업에만 부분적으로 적용된다.

OWASP 는 credential stuffing, brute force, 기본/약한 비밀번호 허용, 약한 비밀번호 복구, 평문 또는 약한 해시 비밀번호 저장, MFA 부재, 세션 ID URL 노출, 세션 미무효화 등을 Broken Authentication 의 취약 조건으로 제시한다 [S5].

3.4.3 영향

계정 하나가 탈취되어도 영향은 크다. 관리자 계정이면 시스템 설정, 사용자 데이터, 결제 정보, 운영 로그, 배포 파이프라인까지 영향을 줄 수 있다. 일반 사용자 계정도 개인정보, 사내 자료, 수평 권한 이동의 시작점이 될 수 있다.

3.4.4 방어 원칙

- 가능한 곳에 MFA 를 적용한다.
- 기본 계정과 기본 비밀번호를 배포하지 않는다.
- 비밀번호 정책은 길이와 유출 비밀번호 차단 중심으로 설계한다.
- 로그인 실패에 rate limiting, 점진적 지연, 탐지, 알림을 둔다.
- 세션 ID 는 서버측 안전한 세션 관리자로 생성하고, 로그인 후 재발급한다.
- 세션 ID 를 URL 에 넣지 않는다.
- 로그아웃, idle timeout, absolute timeout 을 명확히 적용한다.
- 계정 복구 절차는 계정 열거를 유발하지 않도록 동일한 메시지를 사용한다.

3.4.5 레드팀 관찰 포인트

- 로그인 실패 응답이 사용자 존재 여부를 노출하는지 본다.
- 세션 쿠키 속성 Secure, HttpOnly, SameSite 가 적절한지 확인한다.
- 로그인 전후 세션 식별자가 바뀌는지 확인한다.
- 비밀번호 재설정 링크의 수명, 일회성, 추측 가능성을 본다.
- brute force 나 credential stuffing 시도는 서비스에 부하를 주므로 강의 랩이 아닌 경우 반드시 별도 허가와 속도 제한 조건이 필요하다.

3.5 A3:2017 - Sensitive Data Exposure

3.5.1 개념

Sensitive Data Exposure 는 민감 데이터가 저장, 전송, 표시, 로그, 캐시, 백업, 오류 메시지에서 충분히 보호되지 않는 위험이다. OWASP 는 많은 웹 애플리케이션과 API 가 금융, 의료, 개인정보 같은 민감 데이터를 제대로 보호하지 못한다고 설명한다 [S2].

민감 데이터는 조직마다 다르다. 일반적으로 비밀번호, 인증 토큰, 세션 쿠키, 주민등록번호, 여권번호, 신용카드, 계좌, 의료정보, 위치 정보, 내부 API 키, 암호화 키, 영업 비밀, 고객 목록은 민감 데이터로 취급해야 한다.

3.5.2 흔한 원인

- HTTP, FTP, Telnet, 평문 SMTP 등으로 민감 정보를 전송한다.
- 비밀번호를 평문, 단순 해시, 빠른 해시로 저장한다.
- 오래된 TLS 버전, 약한 cipher suite, 잘못된 인증서 검증을 사용한다.
- 암호화 키가 코드 저장소, 이미지, 환경 파일에 노출된다.
- 로그에 세션 토큰, 비밀번호, 개인정보가 남는다.
- 응답 캐싱으로 민감 페이지가 브라우저나 프록시에 저장된다.
- 데이터 분류가 없어 무엇을 보호해야 하는지 모른다.

OWASP 는 데이터가 평문으로 전송되는지, 약한 알고리즘과 프로토콜이 사용되는지, 기본 키나 약한 키가 쓰이는지, 인증서 검증이 되는지 확인하라고 제시한다 [S6].

3.5.3 방어 원칙

- 데이터를 분류하고 필요한 데이터만 수집한다.
- 민감 데이터는 저장하지 않는 것이 최선이다. 저장이 필요하면 강한 암호화와 키 관리가 필요하다.
- 비밀번호는 비밀번호 저장용 KDF 를 사용한다. 일반 해시 함수만으로는 충분하지 않다.
- 전송 구간은 TLS 를 강제하고 HSTS 를 적용한다.
- 민감 응답의 캐싱을 제한한다.
- 로그와 분석 도구로 민감 정보가 흘러가지 않도록 마스킹한다.
- 키와 비밀값은 별도 비밀 관리 시스템에서 관리한다.

3.5.4 레드팀 관찰 포인트

- 로그인, 비밀번호 변경, 개인정보 조회, 파일 다운로드 흐름에서 평문 전송이 있는지 본다.
- 브라우저 개발자 도구, 프록시, 로그, 오류 응답에서 토큰과 개인정보가 노출되는지 확인한다.
- 취약점 보고 시 실제 민감 데이터를 과도하게 수집하지 않는다. 증거는 일부 마스킹으로 충분해야 한다.

3.6 A4:2017 - XML External Entities (XXE)

3.6.1 개념

XXE 는 XML 파서가 외부 엔터티를 처리할 때 발생한다. 오래되었거나 잘못 설정된 XML 파서는 XML 문서 안의 외부 엔터티 참조를 평가할 수 있다. OWASP 는 외부 엔터티가 내부 파일 공개, 내부 파일 공유 접근, 내부 포트 스캐닝, 원격 코드 실행, 서비스 거부로 이어질 수 있다고 설명한다 [S2].

3.6.2 왜 중요한가

요즘 애플리케이션은 JSON 을 많이 쓰지만 XML 은 아직 남아 있다. SOAP API, SAML, RSS, SVG, DOCX/XLSX 같은 압축된 XML 기반 파일, 레거시 연동, 대기업 내부 시스템에서 XML 처리가 흔하다. 초급자는“우리 서비스는 JSON 만 쓴다”고 단정하지 말고 업로드 파일, 외부 연동, 라이브러리 내부 처리를 확인해야 한다.

3.6.3 흔한 원인

- XML 외부 엔터티와 DTD 처리를 비활성화하지 않았다.
- XML 파서 기본값이 안전하지 않다.
- 업로드 파일 내부의 XML 처리를 검증하지 않는다.
- 라이브러리나 프레임워크가 내부적으로 XML 을 파싱한다는 사실을 모른다.
- 네트워크 접근이 가능한 서버에서 XML 파서가 외부 URL 을 조회할 수 있다.

3.6.4 방어 원칙

- 가능하면 XML 대신 단순한 데이터 형식을 사용한다.
- XML 파서에서 DTD, external entity, XInclude, 외부 schema loading 을 비활성화한다.
- XML 입력 크기와 중첩 깊이를 제한한다.
- 서버의 outbound network 접근을 제한한다.
- 파일 업로드 처리에서 확장자만 보지 말고 실제 콘텐츠와 파서 설정을 확인한다.
- XML 관련 라이브러리를 최신 상태로 유지한다.

3.6.5 레드팀 관찰 포인트

- XML 을 받는 엔드포인트, 파일 업로드, 문서 변환, SAML 처리, SOAP API 를 찾는다.
- 테스트는 반드시 허가된 랩에서 수행한다. 내부 파일 읽기나 내부망 포트 접근은 민감도가 높으므로 실제 업무에서는 별도 승인 없이는 시도하지 않는다.
- 보고서에는“외부 엔터티 처리 가능성”과“서버가 외부 리소스를 요청할 수 있는지”를 분리해 설명한다.

3.7 A5:2017 - Broken Access Control

3.7.1 개념

Broken Access Control 은 사용자가 자기 권한 밖의 기능이나 데이터에 접근하는 상태다. OWASP 는 접근 통제 실패가 무단 정보 공개, 데이터 수정/삭제, 허용 범위를 벗어난 업무 기능 수행으로 이어질 수 있다고 설명한다 [S8].

인증은“당신이 누구인가”를 확인한다. 인가는“당신이 이 행동을 해도 되는가”를 확인한다. 로그인에 성공했다고 모든 리소스를 볼 수 있는 것은 아니다.

3.7.2 흔한 유형

- URL 파라미터의 사용자 ID 만 바꿔 다른 사용자의 정보를 본다. 흔히 IDOR 라고 부른다.
- 관리자 페이지가 메뉴에는 숨겨져 있지만 직접 URL 접근은 막지 않는다.

- 클라이언트의 hidden field, JWT claim, 쿠키 값을 신뢰한다.
- API 의 GET 만 막고 POST, PUT, DELETE 는 허용한다.
- 파일 경로를 조작해 의도하지 않은 파일에 접근한다.
- CORS 설정이 과도하게 허용되어 외부 사이트가 API 를 호출한다.
- 객체 소유권 검사가 서비스 계층이 아니라 화면 계층에만 있다.

3.7.3 방어 원칙

- 기본은 deny by default 다.
- 접근 통제는 서버 측 신뢰 경계 안에서 수행한다.
- 객체 단위 소유권 검사를 한다. “로그인했는가”만으로 충분하지 않다.
- 관리자 기능, API, 파일 다운로드, 검색 결과, export 기능까지 일관되게 검사한다.
- 권한 실패를 로그로 남기고 반복 실패를 탐지한다.
- 접근 제어 테스트를 단위 테스트와 통합 테스트에 포함한다.

OWASP 는 접근 통제는 공격자가 수정할 수 없는 서버 측 코드에서 강제되어야 하며, 공개 리소스를 제외하고 기본 거부, 재사용 가능한 접근 통제 메커니즘, 객체 소유권 검사, 디렉터리 목록 비활성화, 접근 통제 실패 로그를 권장한다 [S8].

3.7.4 레드팀 관찰 포인트

- 같은 기능을 사용자 A 와 사용자 B 로 비교한다.
- URL, 경로, 객체 ID, 파일명, 조직 ID, 권한 값을 변경했을 때 서버가 거부하는지 본다.
- UI 에 버튼이 없어도 API 가 허용되는지 확인한다.
- 보고서에는“필요 권한”, “실제 접근 가능 권한”, “침해되는 객체”를 명확히 구분한다.

3.8 A6:2017 - Security Misconfiguration

3.8.1 개념

Security Misconfiguration 은 보안 설정의 누락, 기본값 유지, 불필요한 기능 노출, 패치 누락, 오류 메시지 과다 노출 같은 광범위한 설정 문제다. OWASP 는 보안 설정 오류가 네트워크 서비스, 플랫폼, 웹 서버, 애플리케이션 서버, 데이터베이스, 프레임워크, 커스텀 코드, 사전 설치된 VM/컨테이너/스토리지 등 애플리케이션 스택의 어느 수준에서도 발생할 수 있다고 설명한다 [S9].

3.8.2 흔한 유형

- 기본 관리자 계정과 비밀번호가 남아 있다.
- 디버그 모드가 운영 환경에서 켜져 있다.
- 디렉터리 리스팅이 가능하다.
- 예외 stack trace 와 내부 경로가 사용자에게 노출된다.
- 사용하지 않는 포트, 서비스, 기능, 샘플 페이지가 남아 있다.
- 보안 헤더가 누락되어 브라우저 방어 기능이 약하다.
- 클라우드 스토리지 버킷, 데이터베이스, 관리 콘솔이 공개되어 있다.
- 컨테이너 이미지에 비밀값, 빌드 도구, 불필요한 패키지가 포함되어 있다.

3.8.3 방어 원칙

- 개발, 테스트, 운영 환경의 설정을 분리하고 운영은 최소 기능으로 구성한다.
- 기본 계정, 샘플 파일, 설치 페이지를 제거한다.
- 모든 계층의 패치와 하드닝 기준을 관리한다.
- 인프라를 코드로 관리하고 변경 이력을 남긴다.
- 오류 메시지는 사용자 친화적으로, 상세 진단 정보는 서버 로그에만 남긴다.
- 정기적으로 구성 검사를 자동화한다.

3.8.4 레드팀 관찰 포인트

- HTTP 응답 헤더, 디렉터리 목록, 기본 페이지, 버전 노출, 에러 페이지를 본다.

- 관리자 콘솔과 개발용 포트가 외부에 열려 있는지 확인한다.
- 스캐너 결과를 맹신하지 말고 실제 노출 범위와 인증 요구 여부를 확인한다.
- 보고서에는“설정 위치”, “현재 값”, “권장 값”, “위험한 이유”를 함께 적는다.

3.9 A7:2017 - Cross-Site Scripting (XSS)

3.9.1 개념

XSS 는 애플리케이션이 신뢰할 수 없는 데이터를 웹 페이지에 삽입하면서 적절한 검증이나 인코딩을 하지 않아, 브라우저가 공격자의 스크립트를 실행하는 취약점이다. OWASP 는 XSS 가 피해자 브라우저에서 스크립트를 실행해 세션 탈취, 사이트 변조, 악성 사이트 리다이렉션 등을 유발할 수 있다고 설명한다 [S2].

3.9.2 주요 형태

- **Reflected XSS**: 요청에 포함된 입력이 즉시 응답에 반사되어 실행된다.
- **Stored XSS**: 게시글, 프로필, 댓글, 관리자 메모처럼 서버에 저장된 값이 여러 사용자에게 실행된다.
- **DOM XSS**: 서버 응답보다 브라우저의 JavaScript 가 URL fragment, localStorage, postMessage 같은 값을 위험하게 처리한다.

3.9.3 흔한 원인

- 출력 위치에 맞는 encoding 을 하지 않는다.
- HTML, 속성, JavaScript, CSS, URL 컨텍스트 차이를 무시한다.
- 프레임워크의 자동 escape 기능을 끈다.
- Markdown, WYSIWYG, HTML sanitizer 를 잘못 사용한다.
- 사용자 입력을 `innerHTML` 처럼 위험한 DOM API 에 넣는다.

3.9.4 방어 원칙

- 출력 컨텍스트에 맞는 encoding 을 적용한다.
- 안전한 템플릿 엔진과 프레임워크 기본 escape 기능을 유지한다.
- HTML 허용이 필요하면 검증된 sanitizer 를 사용하고 허용 태그와 속성을 최소화한다.
- CSP 는 보조 방어로 사용한다.
- 쿠키에는 HttpOnly, Secure, SameSite 를 적용해 피해를 줄인다.
- DOM 조작에서는 `textContent` 같은 안전한 API 를 우선한다.

3.9.5 레드팀 관찰 포인트

- 입력값이 화면 어디에 출력되는지, 어떤 컨텍스트인지 먼저 본다.
- stored XSS 는 다른 사용자와 관리자에게 영향을 줄 수 있으므로 실제 서비스에서는 검증 범위를 엄격히 제한한다.
- 증거는 alert 같은 단순 실행 확인보다“어떤 권한의 브라우저에서 어떤 데이터 접근이 가능한가”를 설명하는 쪽이 좋다. 단, 실제 쿠키 탈취나 토큰 전송은 허가 없이는 수행하지 않는다.

3.10 A8:2017 - Insecure Deserialization

3.10.1 개념

Serialization 은 객체나 데이터 구조를 저장/전송 가능한 형태로 바꾸는 과정이고, Deserialization 은 이를 다시 객체로 복원하는 과정이다. Insecure Deserialization 은 공격자가 조작한 직렬화 데이터를 애플리케이션이 신뢰하고 역직렬화할 때 발생한다. OWASP 는 공격자가 제공한 악의적이거나 변조된 객체를 역직렬화하면 애플리케이션 로직 변경이나 원격 코드 실행 같은 결과가 나올 수 있다고 설명한다 [S11].

3.10.2 왜 어려운가

초급자가 보기에는 JSON, 쿠키, 토큰, 캐시, 메시지 큐, 세션 저장소가 단순 데이터처럼 보인다. 하지만 Java, .NET, PHP, Python 등에서 객체 직렬화가 사용되면 복원 과정에서 클래스 로딩, magic method, gadget chain, 파일 접근, 네트워크 접근이 발생할 수 있다. 그래서 단순 입력 검증만으로는 충분하지 않은 경우가 많다.

3.10.3 흔한 원인

- 클라이언트가 보낸 직렬화 객체를 신뢰한다.
- 서명이나 무결성 검증 없이 세션 객체를 복원한다.
- 불필요한 클래스와 라이브러리가 애플리케이션 classpath 에 많다.
- 타입 제한, allowlist, sandbox 가 없다.
- 역직렬화 과정에서 실행되는 메서드의 부작용을 고려하지 않는다.

3.10.4 방어 원칙

- 가능하면 신뢰할 수 없는 데이터의 native object deserialization 을 피한다.
- JSON 처럼 단순 데이터 형식을 쓰더라도 schema validation 을 적용한다.
- 직렬화 데이터에는 전자서명이나 MAC 으로 무결성을 보장한다.
- 역직렬화 가능한 타입을 allowlist 로 제한한다.
- 애플리케이션 권한과 네트워크 접근을 최소화한다.
- 의존성 최소화와 패치 관리를 병행한다.

3.10.5 레드팀 관찰 포인트

- 쿠키, hidden field, API body, 메시지 큐, 캐시, 파일 업로드에서 직렬화 흔적을 찾는다.
- 검증은 서비스 영향이 크므로 실제 객체 실행보다 무결성 검증 부재, 타입 변조 가능성, 예외 메시지, 로직 변화 같은 낮은 위험 증거를 우선한다.
- 보고서에는 사용된 직렬화 포맷, 신뢰 경계, 무결성 검증 여부, 예상 영향, 안전한 대체 설계를 적는다.

3.11 A9:2017 - Using Components with Known Vulnerabilities

3.11.1 개념

현대 애플리케이션은 라이브러리, 프레임워크, 패키지, 컨테이너 이미지, 운영체제 패키지, 프론트엔드 컴포넌트, 플러그인 위에 만들어진다. A9 는 알려진 취약점이 있는 컴포넌트를 사용해 전체 애플리케이션이 위험해지는 문제다. OWASP 는 컴포넌트가 애플리케이션과 같은 권한으로 실행되며, 취약한 컴포넌트가 악용되면 데이터 손실이나 서버 장악으로 이어질 수 있다고 설명한다 [S2].

3.11.2 흔한 원인

- 사용 중인 컴포넌트 목록, 버전, 라이선스, 출처를 모른다.
- 패치 정책이 없다.
- 오래된 CMS 플러그인, 테마, JavaScript 라이브러리, 서버 프레임워크가 남아 있다.
- 취약한 컨테이너 base image 를 계속 사용한다.
- SCA 도구의 경고를 무시하거나 우선순위를 정하지 못한다.
- 직접 사용하지 않는 transitive dependency 취약점을 놓친다.

3.11.3 방어 원칙

- SBOM 또는 dependency inventory 를 유지한다.
- 공식 저장소와 신뢰할 수 있는 출처만 사용한다.
- 업데이트 정책과 긴급 패치 절차를 둔다.
- SCA, container scanning, image signing, lockfile 검사를 CI/CD 에 포함한다.
- 사용하지 않는 의존성을 제거한다.
- 취약점 심각도만 보지 말고 실제 노출 경로와 실행 권한을 함께 평가한다.

3.11.4 레드팀 관찰 포인트

- HTTP 헤더, 정적 파일 경로, JavaScript 파일명, 오류 메시지, `/vendor`, `/node_modules`, CMS 버전 노출을 확인한다.
- 취약점 데이터베이스의 PoC 가 존재하더라도 운영 대상에서 곧바로 재현하지 않는다. 무중단 검증이 가능한 버전 증거와 노출 조건부터 제시한다.

- 보고서에는“취약한 버전”, “해당 컴포넌트가 실제 요청 경로에서 사용되는지”, “권장 버전”, “패치 영향”을 적는다.

3.12 A10:2017 - Insufficient Logging & Monitoring

3.12.1 개념

A10 은 공격을 기록, 모니터링, 탐지, 대응하지 못하는 위험이다. OWASP 는 충분하지 않은 로깅과 모니터링이 거의 모든 주요 사고의 기반이며, 공격자는 탐지와 적시 대응의 부재를 이용해 목표를 달성한다고 설명한다 [S13].

3.12.2 흔한 원인

- 로그인, 실패 로그인, 권한 실패, 관리자 기능, 고가치 거래가 기록되지 않는다.
- 로그 메시지가 불명확하거나 사용자/세션/IP/요청 ID 가 없다.
- 로그가 서버 로컬에만 있고 중앙 수집이 없다.
- 경고 임계값과 대응 절차가 없다.
- DAST, 침투 테스트, 취약점 스캔이 탐지를 유발하지 않는다.
- 애플리케이션이 실시간 또는 근실시간 공격을 탐지하지 못한다.

OWASP 는 로그인, 접근 통제 실패, 서버 측 입력 검증 실패를 충분한 사용자 컨텍스트와 함께 로그로 남기고, 중앙 로그 관리 시스템에서 소비 가능한 포맷으로 생성하며, 고가치 거래의 audit trail 과 효과적 모니터링/알림을 구축하라고 권장한다 [S13].

3.12.3 방어 원칙

- 무엇을 기록할지 미리 정한다. 인증, 권한, 민감 데이터 접근, 관리자 행동, 설정 변경, 결제/이체/삭제, API rate limit 초과는 기본 후보이다.
- 로그는 구조화된 형식으로 남긴다. 시간, 사용자 ID, 세션 ID, 요청 ID, 소스 IP, user-agent, 결과, 실패 이유를 포함한다.
- 비밀번호, 토큰, 개인정보는 로그에 남기지 않는다.
- 로그 무결성과 보관 기간을 관리한다.
- SIEM 또는 중앙 로그 시스템에서 상관 분석과 알림을 설정한다.
- 침투 테스트 후 보안팀이 실제로 탐지했는지 복기한다.

3.12.4 레드팀 관찰 포인트

- 테스트 후 블루팀/운영팀이 무엇을 탐지했는지 확인한다.
- 보고서에는“취약점 자체”와“탐지 실패”를 분리해 쓴다.
- 반복 실패, 권한 우회 시도, 비정상 user-agent, 짧은 시간의 많은 요청이 로그에 남는지 확인한다.
- A10 은 기술 취약점보다 운영 취약점에 가깝다. 좋은 보고서는 로그 샘플, 누락 필드, 알림 부재, 대응 지연을 구체적으로 지적한다.

4 기본적으로 알아야 할 해킹 기법

이 장은 공격 기술의 작동 원리, 방어 관점, 실습 시 주의점을 설명한다. 실제 공격 절차를 외우는 것보다“어떤 신뢰 경계가 깨지는가”를 이해하는 것이 중요하다.

4.1 수동 정찰과 능동 정찰

수동 정찰은 대상 시스템에 직접 트래픽을 보내지 않고 공개 정보를 수집하는 방식이다. 예를 들어 회사 홈페이지, 채용 공고, 공개 저장소, DNS 기록, 인증서 투명성 로그, 기술 블로그, 검색 엔진 캐시를 본다. 수동 정찰은 상대적으로 탐지가 어렵지만, 공개 정보만으로도 도메인, 기술 스택, 이메일 패턴, 외부 협력사, 클라우드 사용 흔적을 얻을 수 있다.

능동 정찰은 대상에게 직접 요청을 보내는 방식이다. 포트 스캔, 웹 크롤링, 디렉터리 검색, 버전 확인, 취약점 스캐닝이 여기에 속한다. MITRE ATT&CK 의 Active Scanning 은 대상 인프라에 네트워크 트래픽을 보내 정보를 수집하는 정찰 기술로 설명된다 [S25]. 능동 정찰은 법적 범위와 탐지 가능성이 높으므로 반드시 허가가 필요하다.

4.1.1 레드팀 체크포인트

- 정찰에서 얻은 정보의 출처를 기록한다.
- 수동 정보와 능동 확인 정보를 구분한다.
- 스캔 속도, 시간대, 대상 범위, 제외 대상, User-Agent, 연락망을 사전에 합의한다.
- 결과는“취약점”이 아니라“가설”로 취급하고 추가 검증한다.

4.2 포트 스캐닝과 서비스 열거

포트 스캐닝은 대상 호스트에서 어떤 네트워크 서비스가 열려 있는지 찾는 활동이다. 서비스 열거는 열린 포트의 프로토콜, 배너, 버전, 인증 방식, TLS 설정, 기본 페이지, 노출 파일을 확인하는 활동이다. NIST 의 target identification and analysis 기법은 시스템, 포트, 서비스, 잠재 취약점을 식별하기 위한 네트워크 발견, 포트/서비스 식별, 취약점 스캐닝 등을 포함한다 [S22].

포트 스캐닝 자체가 취약점은 아니다. 하지만 불필요하게 열린 포트, 오래된 서비스, 인증 없는 관리 인터페이스, 디버그 서비스는 공격 표면을 넓힌다.

4.2.1 방어 관점

- 외부 노출 포트는 비즈니스 필요성으로 설명 가능해야 한다.
- 관리 포트는 VPN, bastion, IP allowlist, MFA 뒤에 둔다.
- 배너와 오류 메시지에서 상세 버전을 숨기되, 숨기는 것만으로 패치를 대체하지 않는다.
- 네트워크 IDS/방화벽/클라우드 보안 그룹에서 비정상 스캔을 탐지한다.

4.3 취약점 스캐닝

취약점 스캐닝은 서비스 버전, 설정, 응답, 알려진 CVE, 정책 위반을 자동으로 점검한다. 스캐너는 빠르고 넓게 볼 수 있지만 오탐과 미탐이 있다. 스캐너 결과는“증거”가 아니라“검토할 후보”다.

4.3.1 스캐너 결과를 읽는 법

- CVSS 점수만 보지 말고 노출 위치와 인증 필요 여부를 본다.
- 취약 버전이 실제로 실행 중인지, 단순 파일 잔재인지 구분한다.
- PoC 존재 여부보다 조직의 자산 가치와 접근 경로를 본다.
- 패치가 불가능한 경우 임시 완화책을 제시한다.
- 운영 서비스에 부하를 줄 수 있는 intrusive check 는 사전 허가 없이 실행하지 않는다.

4.4 스니핑과 패킷 분석

스니핑은 네트워크 통신을 모니터링하고 패킷의 헤더와 페이로드를 분석하는 기술이다. NIST 는 Network Sniffing 을 네트워크 통신을 모니터링하고 프로토콜을 디코딩하며 관심 정보를 찾기 위해 헤더와 페이로드를 검사하는 수동 기법으로 정의한

다 [S23]. MITRE ATT&CK 도 Network Sniffing 을 유선 또는 무선 연결을 통해 전송되는 정보를 모니터링하거나 캡처하는 기술로 설명하며, 인증 정보나 환경 정보를 얻는 데 사용될 수 있다고 설명한다 [S24].

4.4.1 스니핑이 가능한 상황

- 같은 허브나 미러링 포트에 연결되어 있다.
- 스위치 SPAN/TAP 이 설정되어 있다.
- 무선 네트워크에서 관리자가 허가한 모니터링을 수행한다.
- 시스템 자체에서 로컬 트래픽을 캡처한다.
- 클라우드 환경에서 트래픽 미러링이 설정되어 있다.

4.4.2 볼 수 있는 것과 볼 수 없는 것

암호화되지 않은 HTTP, FTP, Telnet, 일부 메일 프로토콜은 계정 정보와 콘텐츠가 평문으로 보일 수 있다. 반면 TLS 가 올바르게 적용된 HTTPS 는 IP, 포트, SNI, 인증서, 패킷 크기, 타이밍 같은 메타데이터는 보일 수 있어도 본문은 보이지 않는다. 따라서 현대 환경에서 스니핑의 가치는“비밀번호 훔치기”보다 네트워크 구조, 프로토콜, 비정상 통신, 암호화 누락을 확인하는 데 있다.

4.4.3 방어 원칙

- 민감 서비스는 TLS 를 강제한다.
- 내부망이라고 평문 프로토콜을 허용하지 않는다.
- 무선망은 강한 암호화와 분리된 게스트 네트워크를 사용한다.
- 스위치 포트 보안, DHCP snooping, Dynamic ARP Inspection 같은 보호 기능을 검토한다.
- 중요 서버 간 통신은 네트워크 분리와 인증서 기반 상호 인증을 고려한다.
- 패킷 캡처 파일은 민감 정보가 포함될 수 있으므로 암호화 저장하고 공유 범위를 제한한다.

4.4.4 실습 주의

패킷 캡처는 타인의 통신을 볼 수 있는 행위다. 강의실에서도 반드시 랩 네트워크와 실습 계정만 대상으로 해야 한다. 실습용으로는 로컬 DVWA 에 접속할 때 브라우저와 서버 사이의 요청/응답 구조를 관찰하거나, TLS 적용 전후의 차이를 비교하는 수준이 적절하다.

4.5 중간자 공격 개념

중간자 공격은 공격자가 두 통신 주체 사이에 끼어 트래픽을 관찰하거나 변조하는 공격군이다. 흔한 예시는 ARP spoofing, DNS poisoning, rogue access point, 악성 프록시, TLS downgrade 시도, 인증서 검증 우회다.

4.5.1 핵심 원리

- 피해자가 공격자를 정상 게이트웨이, DNS 서버, AP, 프록시, 서버로 믿게 만든다.
- 암호화가 없거나 인증서 검증이 약하면 페이로드까지 볼 수 있다.
- 암호화가 있어도 DNS, SNI, 메타데이터, 트래픽 패턴은 일부 보일 수 있다.
- 공격자는 단순 관찰뿐 아니라 요청/응답 변조, 세션 탈취, 피싱 페이지 삽입을 노릴 수 있다.

4.5.2 방어 원칙

- HTTPS 와 HSTS 를 강제한다.
- 인증서 오류를 무시하는 앱/스크립트를 금지한다.
- 내부 DNS 와 DHCP 를 보호한다.
- 무선 AP 는 WPA2/WPA3 Enterprise, 인증서 검증, 게스트 분리를 적용한다.
- 중요 관리 접속은 VPN, SSH host key 검증, bastion 을 사용한다.
- 이상한 ARP 변경, 게이트웨이 MAC 변경, DNS 응답 변조를 모니터링한다.

4.6 LAND 공격

LAND 공격은 서비스 거부 공격의 고전적인 예다. 공격자는 source IP 와 destination IP 를 피해자 주소로 동일하게 만들고, source port 와 destination port 도 동일하게 만든 TCP SYN 패킷을 보낸다. CERT/CC 는 이 유형을 source/destination IP 와 source/destination port 가 같고 SYN flag 가 설정된 spoofed packet 을 이용하는 LAND 공격으로 설명한다 [S26]. Juniper 문서도 LAND 공격을 피해자의 IP 주소를 출발지와 목적지 주소 양쪽에 넣은 spoofed SYN packet 으로 설명한다 [S27].

4.6.1 왜 문제가 되었나

취약한 TCP/IP 스택은 자기 자신에게 연결을 시도하는 이상한 상태를 제대로 처리하지 못해 일시 정지, CPU 소모, 시스템 불안정, 서비스 거부가 발생할 수 있었다. 현대 운영체제와 네트워크 장비는 대부분 이런 비정상 패킷을 처리하거나 차단하지만, LAND 공격은“프로토콜 경계 조건”과“spoofing 차단”의 중요성을 보여주는 교육용 사례로 가치가 있다.

4.6.2 방어 원칙

- OS 와 네트워크 장비를 최신 상태로 유지한다.
- 방화벽과 라우터에서 source 와 destination 이 같은 비정상 패킷을 차단한다.
- ingress/egress filtering 으로 IP spoofing 을 줄인다.
- IDS/IPS 에서 LAND 패턴을 탐지한다.
- 외부에서 내부 주소를 출발지로 사용하는 패킷, 내부에서 외부 주소를 도착지로 위조한 패킷을 차단한다.

4.6.3 실습 주의

LAND 공격은 서비스 거부를 유발할 수 있으므로 실제 네트워크에서 테스트하면 안 된다. 강의에서는 패킷 캡처 예시나 시뮬레이터, 또는 장비 문서의 탐지 조건을 보는 방식으로 충분하다.

4.7 SYN Flood 와 서비스 거부

SYN Flood 는 TCP 연결 수립 과정의 상태 보존 특성을 악용한다. RFC 4987 은 SYN flooding 을 TCP 서버가 LISTEN 상태 포트에 SYN segment 를 받은 뒤 half-connection 상태를 일정 시간 보존하는 특성을 이용해, 정상 연결에 필요한 자원을 소진시키는 DoS 방식으로 설명한다 [S28].

4.7.1 DoS/DDoS 의 범주

- **프로토콜 자원 고갈**: SYN flood, UDP flood, ICMP flood.
- **애플리케이션 자원 고갈**: 비싼 검색, 대량 로그인 시도, 이미지 변환, 보고서 생성 API 남용.
- **대역폭 고갈**: 대량 트래픽으로 회선을 포화시킨다.
- **반사/증폭**: DNS, NTP, memcached 같은 서비스를 악용해 피해자에게 대량 응답을 보낸다.

4.7.2 방어 원칙

- SYN cookies, connection backlog 조정, rate limiting 을 적용한다.
- CDN, DDoS 보호 서비스, Anycast, WAF 를 활용한다.
- 애플리케이션 API 에는 인증, rate limit, quota, 캐싱, 비싼 작업 큐잉을 적용한다.
- 모니터링은 RPS, error rate, latency, CPU, memory, connection state 를 함께 본다.
- 레드팀 테스트에서는 서비스 거부를 유발하는 행위를 기본 금지로 두고, 별도 내구성 시험은 사전 승인된 환경에서만 한다.

4.8 SQL Injection

SQL Injection 은 Injection 의 대표 사례다. 사용자 입력이 SQL 쿼리 구조에 섞여 데이터베이스가 공격자의 의도대로 조회, 수정, 삭제, 인증 우회를 수행한다.

4.8.1 방어 관점에서 기억할 것

- Prepared Statement 와 파라미터 바인딩이 기본이다.
- ORM 을 사용해도 raw query 와 동적 조건 조합은 위험할 수 있다.
- DB 계정 권한을 최소화한다. 조회 전용 기능에는 쓰기 권한이 필요 없다.
- DB 오류 메시지를 사용자에게 그대로 보여주지 않는다.
- 입력값이 숫자, 날짜, enum, 정렬 컬럼이면 형식과 허용 목록을 검증한다.

4.8.2 레드팀 관찰 포인트

- 검색, 로그인, 필터, 정렬, 다운로드, 관리자 페이지, API query parameter 를 본다.
- 응답 오류, 데이터 수 변화, 응답 시간 변화, 권한 경계 변화를 확인한다.
- 대량 데이터 추출, 삭제, DB 함수 실행은 허가 범위를 벗어나기 쉽다. 증명은 최소화한다.

4.9 Command Injection

Command Injection 은 애플리케이션이 사용자 입력을 OS 명령에 섞어 실행할 때 발생한다. 네트워크 진단, 파일 변환, 이미지 처리, 백업, 압축, ping/traceroute 기능에서 발견될 수 있다.

4.9.1 방어 원칙

- OS shell 호출을 피하고 안전한 라이브러리 API 를 사용한다.
- 명령 실행이 필요하면 인자 배열 방식으로 전달하고 shell interpretation 을 피한다.
- 입력은 허용 목록 기반으로 제한한다.
- 애플리케이션 실행 계정 권한을 최소화한다.
- 컨테이너, seccomp, AppArmor, SELinux, chroot 등으로 피해 범위를 줄인다.
- 실행 가능한 명령과 경로를 고정한다.

4.9.2 레드팀 관찰 포인트

- 결과가 화면에 바로 보이지 않아도 시간 지연, 로그, 파일 생성, DNS callback 같은 side effect 가 있을 수 있다. 단, 외부 callback 이나 파일 생성은 승인된 랩에서만 사용한다.
- 보고서에는“사용자 입력이 OS 명령으로 전달되는 경로”와“shell 사용 여부”를 적는다.

4.10 파일 업로드 취약점

파일 업로드는 사용자 제공 파일이 서버에 저장되고, 파싱되고, 다른 사용자에게 제공되는 기능이다. 취약하면 악성 스크립트 업로드, 경로 조작, 파일 덮어쓰기, 악성 문서 파싱, 저장소 비용 폭증, XSS, 악성 콘텐츠 배포로 이어진다.

4.10.1 방어 원칙

- 확장자만 믿지 말고 MIME, magic bytes, 파일 구조를 확인한다.
- 업로드 파일은 웹 루트 밖에 저장하고 직접 실행되지 않게 한다.
- 파일명은 서버가 재생성하고 경로 구분자를 제거한다.
- 크기, 개수, 해상도, 압축 해제 후 크기 제한을 둔다.
- 이미지와 문서 파서는 최신 상태로 유지한다.
- 다운로드 시 Content-Type 과 Content-Disposition 을 명확히 설정한다.
- 악성 콘텐츠 스캔과 격리 저장을 고려한다.

4.10.2 레드팀 관찰 포인트

- 업로드 후 파일이 어디에 저장되고 어떻게 접근되는지 본다.
- 이미지 변환, 썸네일 생성, 문서 미리보기처럼 서버가 파일을 파싱하는 부분을 본다.
- 웹쉘 업로드 같은 고위험 검증은 강의 랩 외에서는 금지하거나 별도 승인이 필요하다.

4.11 경로 조작과 파일 포함

Path Traversal 은 `../` 같은 경로 조작으로 의도한 디렉터리 밖의 파일을 읽거나 쓰는 취약점이다. Local File Inclusion 과 Remote File Inclusion 은 애플리케이션이 파일 경로를 신뢰해 서버 파일이나 원격 파일을 포함하는 문제다.

4.11.1 방어 원칙

- 사용자 입력을 파일 시스템 경로로 직접 사용하지 않는다.
- 파일 ID 와 실제 경로를 서버 측 매핑 테이블로 관리한다.
- canonical path 를 계산한 뒤 허용 디렉터리 내부인지 확인한다.
- 다운로드 권한은 파일 소유권과 함께 확인한다.
- 템플릿, 로그, 업로드 파일이 실행 경로에 포함되지 않게 한다.

4.12 인증 공격: brute force, password spraying, credential stuffing

- **Brute force**: 한 계정에 많은 비밀번호를 시도한다.
- **Password spraying**: 많은 계정에 소수의 흔한 비밀번호를 천천히 시도한다.
- **Credential stuffing**: 다른 침해에서 유출된 ID/비밀번호 조합을 재사용해 로그인한다.

OWASP A2 는 credential stuffing, default administrative account lists, automated brute force, dictionary attack tools 가 공격자에게 널리 사용된다고 설명한다 [S5].

4.12.1 방어 원칙

- MFA 를 적용한다.
- 로그인 실패 제한과 지연을 적용한다.
- 비밀번호 재사용과 유출 비밀번호를 차단한다.
- 계정 열거를 막기 위해 응답 메시지를 통일한다.
- 비정상 로그인 위치, 장치, 속도, 실패 패턴을 탐지한다.
- 보안팀이 탐지하고 대응할 수 있도록 로그와 알림을 구성한다.

4.12.2 실습 주의

계정 공격은 실제 서비스에서 계정 잠금, 장애, 법적 문제를 만들 수 있다. 부트캠프에서는 DVWA 나 전용 인증 랩에서만 제한된 속도로 실습하고, 실제 조직 대상 테스트에서는 명시적 승인이 있어야 한다.

4.13 비밀번호 해싱과 크래킹 개념

비밀번호 크래킹은 탈취한 해시에서 원래 비밀번호를 추정하는 과정이다. 레드팀 입문자는 도구 사용보다 방어 원리를 먼저 알아야 한다.

4.13.1 중요한 개념

- **Hash**: 같은 입력은 같은 출력이지만 역산이 어렵도록 설계된 함수다.
- **Salt**: 같은 비밀번호가 같은 해시로 보이지 않도록 사용자별 무작위 값을 섞는다.
- **KDF**: 비밀번호 저장을 위해 계산 비용을 높인 함수다. 일반 해시보다 느리게 만들어 대량 추측을 어렵게 한다.
- **Pepper**: 애플리케이션 외부 비밀값을 추가로 섞어 DB 만 유출되어도 공격을 어렵게 한다.

4.13.2 방어 원칙

- 평문 비밀번호 저장은 절대 금지다.
- 빠른 일반 해시만으로 저장하지 않는다.
- 사용자별 salt 와 비밀번호 저장용 KDF 를 사용한다.
- MFA 와 유출 비밀번호 차단을 병행한다.
- 해시 유출 사고를 가정하고 비밀번호 재설정 절차를 준비한다.

4.14 웹 세션과 쿠키 공격

웹 세션은 사용자가 로그인 상태를 유지하게 해주는 핵심 메커니즘이다. 세션 관리가 약하면 세션 고정, 세션 탈취, CSRF, 쿠키 변조, 토큰 재사용이 가능해진다.

4.14.1 방어 원칙

- 로그인 후 세션 ID 를 재발급한다.
- 쿠키에 Secure, HttpOnly, SameSite 를 설정한다.
- 세션 만료와 로그아웃 무효화를 서버 측에서 처리한다.
- CSRF 토큰은 세션과 연동하고 재사용 위험을 줄인다.
- JWT 를 사용할 때는 서명 알고리즘, 만료, audience, issuer, key rotation 을 검증한다.
- 민감 작업에는 재인증이나 step-up authentication 을 적용한다.

4.15 CSRF

CSRF 는 사용자가 로그인한 상태라는 점을 이용해, 공격자 사이트가 피해자 브라우저로 정상 사이트에 상태 변경 요청을 보내게 하는 공격이다. OWASP 2017 Top 10 의 본 항목에는 없지만, OWASP Risk Factor Summary 는 추가로 고려할 위험 중 하나로 CSRF 를 언급한다 [S3].

4.15.1 방어 원칙

- 상태 변경 요청에는 예측 불가능한 CSRF 토큰을 사용한다.
- SameSite 쿠키 속성을 적용한다.
- GET 요청으로 상태 변경을 하지 않는다.
- Origin/Referer 검증을 보조 방어로 사용한다.
- 민감 작업에는 재인증이나 확인 절차를 둔다.

4.16 SSRF

SSRF 는 서버가 공격자가 지정한 URL 이나 내부 주소로 요청을 보내게 하는 취약점이다. 내부 메타데이터 서비스, 관리자 포트, 클라우드 credential, 내부 API 접근으로 이어질 수 있다. OWASP 2017 Risk Factor Summary 도 SSRF 를 추가로 고려할 위험 중 하나로 언급한다 [S3].

4.16.1 흔한 위치

- URL 미리보기
- 이미지 가져오기
- 웹훅 등록
- PDF/스크린샷 생성
- 원격 파일 import
- 서버 측 프록시

4.16.2 방어 원칙

- URL 허용 목록을 사용한다.
- private IP, loopback, link-local, metadata 주소 접근을 차단한다.
- DNS rebinding 을 고려해 요청 직전과 연결 후 IP 를 검증한다.
- 서버의 outbound network policy 를 최소화한다.
- 리다이렉트 허용 여부를 엄격히 관리한다.

4.17 피싱과 사회공학

피싱은 기술 취약점보다 사람과 절차를 노리는 공격이다. 레드팀에서 피싱 시뮬레이션은 교육 효과가 있지만 개인정보, 심리적 피해, 업무 신뢰 훼손 문제가 있으므로 별도 승인, 명확한 범위, 사후 교육, 민감 데이터 수집 금지가 필요하다.

4.17.1 방어 원칙

- MFA 를 적용한다.
- 메일 보안: SPF, DKIM, DMARC 를 운영한다.
- URL rewriting 과 첨부 파일 샌드박스를 적용한다.
- 보고 버튼과 피드백 체계를 만든다.
- 교육은 처벌이 아니라 학습 중심이어야 한다.
- 권한 상승 요청, 결제 변경 요청, 계정 복구 요청은 out-of-band 확인 절차를 둔다.

4.18 권한 상승

권한 상승은 이미 얻은 낮은 권한을 더 높은 권한으로 바꾸는 과정이다. 서버에서는 커널 취약점, SUID/권한 설정 오류, sudo 정책 오류, 서비스 계정 비밀값, cron, PATH 조작, 잘못된 파일 권한이 원인이 될 수 있다. 웹에서는 일반 사용자에서 관리자 권한으로 넘어가는 접근 통제 실패가 대표적이다.

4.18.1 방어 원칙

- 최소 권한과 역할 분리를 적용한다.
- 관리자 명령은 감사 로그를 남긴다.
- 비밀값을 파일, 환경 변수, 이미지에 방치하지 않는다.
- OS 와 패키지를 패치한다.
- 파일 권한, sudoers, 서비스 계정 권한을 정기 점검한다.
- 컨테이너는 rootless, read-only filesystem, capability 제한을 적용한다.

4.19 내부 이동과 피벗

내부 이동은 침해된 시스템을 발판으로 다른 시스템에 접근하는 단계다. 피벗은 네트워크 경로를 우회하거나 내부 자원 접근을 중계하는 행위다. 실제 레드팀에서 내부 이동은 큰 위험을 만들 수 있으므로 범위와 승인 없이는 수행하면 안 된다.

4.19.1 방어 원칙

- 네트워크 세그먼테이션을 적용한다.
- 서비스 계정과 관리자 계정의 재사용을 줄인다.
- 중요 시스템 접근에는 MFA 와 bastion 을 적용한다.
- 내부 동서 트래픽을 모니터링한다.
- EDR, Windows Event Log, Linux audit log, DNS log, proxy log 를 중앙 수집한다.

4.20 지속성

Persistence 는 시스템 재부팅, 비밀번호 변경, 세션 종료 후에도 접근을 유지하려는 기법이다. 악성 계정 생성, SSH key 추가, scheduled task, service 등록, 웹쉘, 브라우저 토큰 탈취 등이 포함될 수 있다.

4.20.1 방어 원칙

- 계정 생성과 권한 변경을 알림으로 연결한다.
- 서비스 등록, cron, scheduled task, startup folder 변경을 모니터링한다.
- 파일 무결성 모니터링을 적용한다.
- 관리자 접근은 bastion 과 감사 로그로 통제한다.
- 레드팀 실습에서는 지속성 설치를 기본 금지하고, 필요 시 별도 승인과 제거 절차를 문서화한다.

4.21 데이터 유출과 증거 관리

Exfiltration 은 공격자가 데이터를 외부로 빼내는 단계다. 레드팀에서는 실제 민감 데이터 유출을 재현하지 않아도 영향도를 입증할 수 있어야 한다.

4.21.1 안전한 증거 원칙

- 민감 데이터는 일부 샘플과 마스킹으로 충분하다.
- 대량 다운로드를 하지 않는다.
- 토큰, 세션, 비밀번호, 개인정보는 보고서에 원문으로 넣지 않는다.
- 증거 파일은 암호화하고 공유 범위를 제한한다.
- 테스트 종료 후 랩 계정, 임시 파일, 토큰, 접근 권한을 정리한다.

5 DVWA 란?

5.1 정의와 목적

DVWA 는 Damn Vulnerable Web Application 의 약자다. 공식 GitHub 저장소는 DVWA 를 PHP/MariaDB 기반의 의도적으로 취약한 웹 애플리케이션으로 설명하며, 보안 전문가가 합법적인 환경에서 기술과 도구를 테스트하고, 웹 개발자가 웹 애플리케이션 보안 절차를 이해하며, 학생과 교사가 통제된 교실 환경에서 웹 보안을 배우도록 돕는 것이 목적이라고 설명한다 [S14].

DVWA 의 장점은 단순하다.

- 설치가 비교적 쉽다.
- SQL Injection, XSS, CSRF, Command Injection, File Upload, Brute Force 등 기본 웹 취약점을 반복 실습할 수 있다.
- 난이도 수준을 낮음, 중간, 높음, 불가능에 가깝게 바꿔 방어 코드 차이를 비교할 수 있다.
- 강의자가 같은 화면과 같은 취약점을 기준으로 설명할 수 있다.

5.2 가장 중요한 경고

DVWA 는 의도적으로 취약하다. 공식 README 도 DVWA 를 public html 폴더나 인터넷에 노출된 서버에 올리지 말라고 경고하며, NAT 네트워크 모드의 VM 같은 격리 환경 사용을 권장한다 [S14]. 실습자는 다음 원칙을 지켜야 한다.

- 인터넷 공개 서버에 설치하지 않는다.
- 기본 포트가 외부에 바인딩되지 않게 한다.
- 가능하면 `127.0.0.1` 또는 host-only 네트워크에서만 접근한다.
- 실습용 계정과 실습용 데이터만 사용한다.
- 실습 후 컨테이너와 볼륨을 정리하거나 스냅샷을 되돌린다.

5.3 권장 랩 구조

초급자에게 가장 안전한 구조는 다음 중 하나다.

구조 A: 호스트 PC 에 DVWA Docker, Kali 는 VM

```
[Host OS]
  - Docker Desktop / Docker Engine
  - DVWA: http://127.0.0.1:4280

[Kali VM]
  - NAT 또는 Host-only network
  - 브라우저/프록시/패킷 분석 도구로 DVWA 접근
```

구조 B: 별도 Ubuntu/Debian VM 에 DVWA Docker, Kali VM 과 Host-only network

```
[Host OS]
  +-- [Kali VM] ---------------+
  |                            |  Host-only Lab Network
  +-- [DVWA Ubuntu VM] -------+
```

구조 B 는 네트워크 실습을 이해하기 좋지만 설정이 더 복잡하다. 초급자는 구조 A 로 시작해도 충분하다.

5.4 Docker 기반 설치법

DVWA 공식 README 는 Docker 와 Docker Compose 를 전제 조건으로 제시하고, 저장소를 클론하거나 다운로드한 뒤 DVWA 폴더에서 `docker compose up -d` 를 실행하면 `http://localhost:4280` 에서 접근할 수 있다고 안내한다 [S14]. 아래 절차는 그 흐름을 실습용으로 정리한 것이다.

5.4.1 1 단계: Docker 와 Compose 확인

Docker Desktop 을 쓰면 Docker 와 Docker Compose 가 함께 설치되는 경우가 많다. Linux 에서 Docker Engine 을 쓰는 경우에는 공식 Docker 설치 문서를 따라 설치한다. 터미널에서 다음 명령으로 설치 여부를 확인한다.

```
docker version
docker compose version
```

버전이 출력되면 다음 단계로 진행한다. 명령을 찾을 수 없으면 Docker 설치가 먼저 필요하다.

5.4.2 2 단계: DVWA 공식 저장소 받기

공식 저장소를 사용한다.

```
git clone https://github.com/digininja/DVWA.git
cd DVWA
```

Git 이 없다면 GitHub 에서 ZIP 을 다운로드해 압축을 풀어도 된다. 공식 README 는 여러 버전이 돌아다니지만 지원되는 버전은 공식 GitHub 저장소의 최신 소스라고 설명한다 [S14].

5.4.3 3 단계: 컨테이너 실행

```
docker compose up -d
```

실행 후 브라우저에서 다음 주소를 연다.

```
http://localhost:4280
```

DVWA README 는 Docker 컨테이너 실행 시 웹 서버가 일반적인 80 번 포트가 아니라 4280 포트에서 동작한다고 안내한다 [S14].

5.4.4 4 단계: 로그인과 DB 초기화

공식 README 의 기본 로그인 정보는 다음과 같다 [S14].

```
Username: admin
Password: password
```

로그인 후 `Setup DVWA` 메뉴에서 `Create / Reset Database` 버튼을 눌러 데이터베이스를 생성하거나 초기화한다. 공식 README 도 이 버튼을 통해 데이터베이스를 생성/초기화한다고 설명한다 [S14].

5.4.5 5 단계: 보안 레벨 설정

DVWA 에는 보안 레벨이 있다. 일반적으로 다음처럼 이해하면 된다.

- **Low**: 취약점을 이해하기 쉬운 가장 취약한 구현.
- **Medium**: 일부 방어가 있지만 우회 가능한 구현.
- **High**: 더 강한 방어가 있지만 완전하지 않을 수 있는 구현.
- **Impossible**: 안전한 구현 예시를 보여주는 수준.

처음에는 Low 에서 취약점의 원리를 이해하고, Medium/High/Impossible 로 올라가며 어떤 방어가 추가되는지 비교한다. 단, “Low 에서 잘 된다”가 실무 능력을 의미하지 않는다. 실무 능력은 취약한 코드와 안전한 코드의 차이를 설명하고 수정 방향을 제시하는 데 있다.

5.4.6 6 단계: 로그 확인과 종료

공식 README 는 Docker 환경에서 로그를 터미널로 확인할 수 있다고 안내한다 [S14].

```
docker compose logs
```

실습을 멈추려면 다음 명령을 사용한다.

```
docker compose stop
```

컨테이너를 제거하려면 다음을 사용한다.

```
docker compose down
```

강의 실습 후에는 스냅샷을 되돌리거나 컨테이너를 정리하는 습관을 들인다.

5.5 수동 설치 개요

Docker 가 어려운 환경에서는 LAMP/LEMP 에 수동 설치할 수 있다. DVWA README 는 Debian 계열 Linux 에서 apache2, MariaDB, PHP, php-mysqli, php-gd, libapache2-mod-php 같은 패키지를 필요로 한다고 안내한다 [S14]. 다만 초급자에게는 Docker 가 더 안전하고 재현성이 좋다.

수동 설치는 다음 위험이 있다.

- 웹 루트에 잘못 배치해 외부 노출될 수 있다.
- DB 계정과 권한 설정 실수가 생길 수 있다.
- PHP 버전과 모듈 문제로 실습 환경이 달라질 수 있다.
- 실습 후 제거가 Docker 보다 번거롭다.

5.6 DVWA 로 무엇을 배워야 하는가

DVWA 의 목적은 도구 버튼을 눌러“공격 성공”을 보는 것이 아니다. 다음 질문에 답할 수 있어야 한다.

- 취약점은 어떤 입력에서 발생했는가?
- 서버는 그 입력을 어떤 신뢰 경계 안으로 가져갔는가?
- 취약한 구현과 안전한 구현의 코드 차이는 무엇인가?
- 공격자가 얻는 권한과 데이터는 무엇인가?
- 로그에는 어떤 흔적이 남는가?
- 방어 코드는 어디에 있어야 하는가? 클라이언트, 서버, DB, 프록시, WAF 중 무엇이 핵심인가?

5.7 DVWA 실습 주제와 OWASP 매핑

DVWA 주제	관련 OWASP 2017	배울 점
Brute Force	A2	로그인 실패 제한, 계정 열거, 세션 관리
Command Injection	A1	OS 명령과 사용자 입력 분리
CSRF	추가 위험	상태 변경 요청과 CSRF 토큰
File Inclusion	A5/A6	경로 검증, 서버 파일 접근 제한
File Upload	A6 및 추가 위험	업로드 파일 검증, 웹 루트 분리
SQL Injection	A1	파라미터 바인딩, DB 권한 최소화
XSS Reflected/Stored/DOM	A7	출력 인코딩, 브라우저 신뢰 경계
Weak Session IDs	A2	세션 예측 가능성, 토큰 난수성
CSP Bypass	A7 보조 방어	CSP 의 가치와 한계

5.8 실습 보고서 예시 구조

DVWA 실습도 실무 보고서처럼 쓰는 습관을 들이면 좋다.

```
제목: DVWA SQL Injection 취약 구현 분석
환경: DVWA Docker, Security Level Low, Localhost only
취약 위치: /vulnerabilities/sqli/
원인: 사용자 입력이 파라미터 바인딩 없이 SQL 문에 결합됨
영향: 조건 조작을 통한 비인가 데이터 조회 가능
증거: 요청/응답 캡처 일부, 화면 캡처, 코드 비교
```

```
방어: Prepared Statement, 입력 검증, DB 최소 권한, 오류 메시지 제한
재검증: 동일 입력에서 SQL 구조가 변하지 않고 안전한 오류/정상 응답을 반환해야 함
```

6 Kali Red 란? Kali Linux 설명

6.1 공식 명칭 관점

“Kali Red”라는 표현은 부트캠프나 커뮤니티에서“레드팀용 Kali”또는“공격 보안 중심의 기존 Kali”를 부르는 비공식 표현으로 쓰일 수 있다. 그러나 공식 문서에서 일반적으로 쓰는 명칭은 **Kali Linux** 다. Kali Linux 공식 문서는 Kali Linux 를 오픈소스, Debian 기반 Linux 배포판으로 설명하며, 고급 침투 테스트와 보안 감사를 수행할 수 있게 한다고 소개한다 [S15].

반면 **Kali Purple** 은 Kali 가 방어 보안 영역으로 확장하기 위해 소개한 프로젝트다. Kali 2023.1 릴리스 글은 Kali Purple 을 defensive security 로 확장하는 기술 프리뷰로 설명하며, SOC 분석, threat hunting, security control testing, blue/red/purple teaming exercises 를 위한 방향을 제시한다 [S21]. 따라서 부트캠프에서“Kali Red”라고 부른다면 다음처럼 이해하면 된다.

- **Kali Linux**: 공식 배포판. 침투 테스트, 보안 연구, 포렌식, 리버스 엔지니어링 등에 쓰인다.
- **Kali Purple**: 방어/블루팀/퍼플팀을 겨냥한 Kali 계열 프로젝트.
- **Kali Red**: 공식 제품명이라기보다 기존 Kali 의 offensive security/레드팀 워크스테이션 용도를 가리키는 비공식 표현으로 보는 것이 안전하다.

6.2 Kali Linux 의 특징

Kali Linux 는 단순히“해킹 도구 모음”이 아니다. 공식 문서는 Kali 가 여러 플랫폼에서 실행되고, 정보보안 전문가와 취미 사용자 모두에게 무료로 접근 가능하며, 컴퓨터 포렌식, 리버스 엔지니어링, 취약점 탐지 같은 작업에 집중할 수 있도록 수백 개의 도구, 설정, 스크립트를 포함한다고 설명한다 [S15].

입문자가 이해할 특징은 다음과 같다.

- Debian 기반이다.
- penetration testing 과 security auditing 에 최적화되어 있다.
- 도구가 많지만, 도구의 의미를 모르면 위험한 버튼 모음일 뿐이다.
- 기본 사용자는 과거처럼 root 고정이 아니다. Kali 는 2020.1 이후 기본 non-root 정책으로 바뀌었다 [S19].
- 공식 이미지와 체크섬 검증이 중요하다. Kali 공식 문서는 공식 소스 외에서 이미지를 다운로드하지 말고 SHA256 체크섬을 확인하라고 강조한다 [S17].

6.3 어떤 이미지로 설치할까?

Kali 공식 문서는 x86-64 용으로 Installer, NetInstaller, Live, Everything 같은 이미지 유형을 설명하며, 의심되면 Installer 이미지를 사용하라고 안내한다 [S16]. 초급 부트캠프에서는 다음을 권장한다.

6.3.1 권장 1: Pre-built VM

처음에는 VMware 또는 VirtualBox 용 pre-built VM 이 가장 편하다. 공식 Kali 다운로드 페이지도 VMware 와 VirtualBox 이미지가 VM 설치를 선호하거나 필요한 사용자에게 제공된다고 설명한다 [S17]. VM 은 스냅샷, 격리, 복구가 쉽다.

6.3.2 권장 2: Installer ISO

리눅스 설치와 파티션, 부트로더, 네트워크를 배우고 싶다면 Installer ISO 를 사용한다. 단, 호스트 PC 에 직접 설치하는 bare metal 은 초급자에게 권장하지 않는다. 실수로 개인 파일을 지우거나 네트워크 설정을 망칠 수 있다.

6.3.3 권장하지 않는 시작 방식

- 업무용 노트북의 주 OS 로 Kali 를 쓰는 것.
- 검증되지 않은 블로그나 파일 공유 사이트에서 Kali 이미지를 받는 것.
- 인터넷에 노출된 클라우드 Kali 인스턴스를 기본 비밀번호로 운영하는 것.

6.4 안전한 다운로드와 검증

Kali 공식 문서는 다운로드한 이미지가 진짜 Kali 인지 확인하기 위해 SHA256SUMS 와 SHA256SUMS.gpg 파일을 함께 받고, Kali 공식 키로 서명을 검증하는 절차를 설명한다 [S18]. 초급자는 최소한 다음 습관을 가져야 한다.

- Kali 이미지는 공식 다운로드 페이지에서 받는다.
- 파일 해시를 확인한다.
- 가능하면 GPG 서명 검증까지 수행한다.
- 다운로드 후 보관 파일명과 해시를 실습 기록에 남긴다.

6.5 Kali 기본 계정

Kali 공식 문서는 Live Boot 또는 pre-created image, 즉 VM/ARM 같은 미리 만들어진 이미지의 기본 자격증명을 `kali/kali` 로 안내한다 [S19]. 다만 설치형 이미지는 설치 과정에서 표준 사용자 계정을 만들도록 요구한다. 실습 시작 후에는 다음을 수행한다.

- 기본 비밀번호를 바꾼다.
- VM 스냅샷을 만든다.
- 업데이트를 수행한다.
- 클립보드 공유, 드래그 앤 드롭, 공유 폴더는 필요한 경우에만 켠다.

6.6 업데이트와 메타패키지

Kali 공식 문서는 메타패키지를 여러 패키지를 한 번에 설치하기 위한 의존성 목록으로 설명한다. Kali 는 사용자가 얼마나 많은 도구를 설치할지 선택할 수 있게 메타패키지를 제공한다 [S20]. 문서 예시는 시스템 업데이트 후 `kali-linux-default` 같은 메타패키지를 설치하는 흐름을 보여준다 [S20].

기본 업데이트 명령은 다음과 같다.

```
sudo apt update
sudo apt full-upgrade -y
```

예시 메타패키지 설치는 다음처럼 할 수 있다.

```
sudo apt install -y kali-linux-default
```

입문자는 모든 도구를 설치하려고 하기보다 기본 도구를 이해하는 것이 먼저다. `kali-linux-everything` 은 용량과 관리 부담이 크므로 강의 목적이 아니라면 신중히 선택한다.

6.7 Kali 도구 분류

Kali 도구는 기능별로 이해하는 것이 좋다.

분류	대표 예시	학습 목표
정보 수집	Nmap, Amass, theHarvester	대상 범위와 노출면 이해
웹 분석	Burp Suite, OWASP ZAP	HTTP 요청/응답, 세션, 입력 검증 이해
패킷 분석	Wireshark, tcpdump	프로토콜, 평문/암호화 차이 이해
취약점 분석	Nuclei, OpenVAS/GVM	스캐너 결과 검증과 오탐 판단
비밀번호 감사	John the Ripper, Hashcat	해시와 비밀번호 정책 이해
무선 보안	Aircrack-ng 계열	RF 실습 환경과 암호화 이해
익스플로잇 프레임워크	Metasploit	취약점 영향 검증과 안전한 범위 관리
리버스/포렌식	Ghidra, binwalk, Volatility 계열	파일, 메모리, 바이너리 분석 기초
보고	CherryTree, Markdown, 스크린샷 도구	증거 정리와 재현 가능한 보고

도구 이름을 안다고 레드팀원이 되는 것은 아니다. 각 도구가 보내는 네트워크 트래픽, 생성하는 로그, 실패 조건, 법적 위험, 시스템 부하를 이해해야 한다.

6.8 Kali 운영 위생

- VM 스냅샷을 자주 찍는다.
- 실습별 폴더를 나누고 증거를 정리한다.

- 개인 계정과 실습 계정을 섞지 않는다.
- 브라우저에 개인 로그인 세션을 저장하지 않는다.
- 클라우드 API 키나 회사 VPN 설정을 Kali 실습 VM 에 넣지 않는다.
- 도구 실행 전 대상 범위를 다시 확인한다.
- 실습 후 토큰, 로그, 다운로드 파일을 정리한다.

6.9 Kali 를 배우는 올바른 순서

1. Linux 기본 명령, 파일 권한, 프로세스, 네트워크 설정을 익힌다.
2. HTTP, DNS, TCP/IP, TLS, 쿠키, 세션을 이해한다.
3. Burp Suite 나 ZAP 으로 요청/응답을 읽는다.
4. DVWA 에서 웹 취약점의 원리와 방어 코드를 비교한다.
5. Nmap 결과를 읽고 서비스별 위험을 설명한다.
6. 스캐너 결과를 검증하고 보고서로 정리한다.
7. 탐지 로그를 확인하며 블루팀 관점까지 연결한다.

7 부트캠프 실습 운영 가이드

7.1 기본 랩 체크리스트

실습 전 다음을 확인한다.

- □ 대상은 DVWA, CTF, 강의 VM, 본인 소유 시스템 중 하나다.
- □ 외부 인터넷에 노출되어 있지 않다.
- □ VM 스냅샷이 있다.
- □ 실습 계정과 비밀번호는 개인 계정과 다르다.
- □ 패킷 캡처는 랩 인터페이스에서만 수행한다.
- □ 실습 후 컨테이너와 임시 파일을 정리한다.
- □ 보고서에 민감 정보 원문을 남기지 않는다.

7.2 HTTP 요청/응답 읽기

웹 해킹 입문에서 가장 중요한 기술은 도구가 아니라 HTTP 를 읽는 능력이다.

- Method: GET, POST, PUT, DELETE 가 어떤 의미인지 본다.
- Path 와 query string: 어떤 리소스와 파라미터가 전달되는지 본다.
- Headers: Host, Cookie, Authorization, Content-Type, User-Agent, Origin, Referer 를 본다.
- Body: form-urlencoded, JSON, multipart/form-data 를 구분한다.
- Response code: 200, 302, 400, 401, 403, 404, 500 을 구분한다.
- Set-Cookie: Secure, HttpOnly, SameSite, Path, Domain, Expires 를 본다.
- Cache: Cache-Control, Pragma, Expires 를 본다.

7.3 취약점 하나를 분석하는 프레임

어떤 취약점이든 다음 질문으로 정리한다.

1. **입력**: 공격자가 통제할 수 있는 값은 무엇인가?
2. **신뢰 경계**: 그 값이 서버, DB, OS, 브라우저, 내부망 중 어디로 넘어가는가?
3. **검증 실패**: 어떤 검증, 인코딩, 권한 확인, 무결성 검증이 빠졌는가?
4. **영향**: 공격자가 읽기, 쓰기, 실행, 가장, 서비스 거부 중 무엇을 할 수 있는가?
5. **권한**: 익명, 일반 사용자, 관리자, 내부망 중 어느 권한이 필요한가?
6. **증거**: 최소한의 증거로 영향이 입증되는가?
7. **방어**: 코드를 고쳐야 하는가, 설정을 바꿔야 하는가, 운영 탐지를 강화해야 하는가?
8. **재검증**: 어떤 결과면 수정 완료로 볼 수 있는가?

7.4 실습별 권장 기록

실습 기록은 나중에 보고서가 된다.

```
날짜/시간:
환경:
대상 URL 또는 호스트:
테스트 계정:
보안 레벨:
관찰한 요청:
관찰한 응답:
취약 원인:
영향:
방어 코드 또는 설정:
남은 질문:
```

7.5 블루팀과 함께 복기하기

레드팀 실습이 진짜 가치가 있으려면 탐지와 대응으로 이어져야 한다.

- 공격성 요청이 웹 로그에 남았는가?
- 로그인 실패가 인증 로그에 남았는가?
- 관리자 기능 접근 실패가 alert 로 이어졌는가?
- WAF 나 IDS 가 무엇을 탐지했는가?
- SIEM 에 사용자, IP, 요청 ID 가 연결되었는가?
- 보고서의 시간대와 로그의 시간대가 일치하는가?

A10 이 중요한 이유는 여기에 있다. 취약점을 고치는 것만큼 공격 흔적을 볼 수 있는 능력이 중요하다.

8 핵심 용어 사전

8.1 공격 표면

공격자가 상호작용할 수 있는 모든 지점이다. 포트, API, 로그인, 파일 업로드, 관리자 콘솔, 메일, DNS, 클라우드 저장소, 직원 계정, 협력사 연동까지 포함한다.

8.2 신뢰 경계

데이터나 권한의 신뢰 수준이 바뀌는 지점이다. 브라우저에서 서버로 넘어갈 때, 웹 서버에서 DB 로 넘어갈 때, 외부 URL 을 서버가 호출할 때, 일반 사용자 요청이 관리자 기능으로 들어갈 때 신뢰 경계가 생긴다.

8.3 취약점과 익스플로잇

취약점은 약점이다. 익스플로잇은 그 약점을 이용하는 구체적 방법이다. 레드팀 보고서는 익스플로잇 자체보다 약점의 원인과 영향, 조치 방안을 우선해야 한다.

8.4 PoC

Proof of Concept 의 약자다. 취약점이 실제로 영향을 줄 수 있음을 보여주는 최소 증거다. 좋은 PoC 는 짧고 안전하며, 원인을 설명할 수 있어야 한다.

8.5 CVE 와 CWE

CVE 는 공개적으로 알려진 특정 취약점 식별자다. CWE 는 취약점 유형 분류다. 예를 들어 어떤 제품의 특정 버전 원격 코드 실행은 CVE 가 될 수 있고, 그 원인이 입력 검증 부재라면 관련 CWE 에 매핑될 수 있다.

8.6 CVSS

취약점 심각도 점수 체계다. 점수는 참고자료일 뿐이다. 실제 위험은 노출 위치, 인증 필요 여부, 데이터 민감도, 보상 통제, 공격자 능력, 업무 영향과 함께 봐야 한다.

8.7 SAST, DAST, SCA

- **SAST**: 소스 코드나 바이너리를 실행하지 않고 분석한다.
- **DAST**: 실행 중인 애플리케이션에 요청을 보내 동작을 분석한다.
- **SCA**: 오픈소스/서드파티 의존성과 알려진 취약점을 분석한다.

8.8 WAF

Web Application Firewall 이다. 공격 패턴을 차단하거나 완화하지만 애플리케이션의 근본 취약점을 대체하지 않는다. WAF 는 보조 방어와 가시성 제공에 가깝다.

8.9 IDS/IPS

IDS 는 침입 탐지 시스템, IPS 는 침입 방지 시스템이다. 패킷, 흐름, 로그, 시그니처, 이상 행위를 기반으로 공격을 탐지하거나 차단한다.

8.10 SIEM

Security Information and Event Management 다. 여러 시스템의 보안 로그를 중앙에서 수집, 상관 분석, 알림, 조사할 수 있게 한다.

8.11 IOC 와 TTP

IOC 는 침해 지표다. IP, 도메인, 해시, 파일명 같은 관찰 가능한 흔적이다. TTP 는 전술, 기술, 절차다. 장기적으로는 IOC 보다 TTP 이해가 더 중요하다.

9 부트캠프 과제와 토론 질문

9.1 과제 1: OWASP Top 10 2017 한 페이지 요약

각자 A1 부터 A10 까지 다음 형식으로 10 줄 요약을 만든다.

```
A1 Injection: 데이터와 명령의 분리 실패. 대표 영향은 데이터 유출/변조/명령 실행. 핵심 방어는 파라미터 바인
↪ 딩과 최소 권한.
```

목표는 암기가 아니라“한 문장으로 설명하는 능력”이다.

9.2 과제 2: DVWA 취약 코드와 방어 코드 비교

DVWA 에서 같은 기능을 Low 와 Impossible 수준으로 비교한다.

- 어떤 입력이 취약한가?
- Low 에서는 어떤 검증이 빠졌는가?
- Impossible 에서는 어떤 방어가 추가되었는가?
- 그 방어가 실무에서도 충분한가?

9.3 과제 3: 패킷 캡처로 HTTP 이해하기

로컬 DVWA 접속을 대상으로만 패킷 또는 프록시 로그를 관찰한다.

- 로그인 요청의 Method 와 Body 구조를 설명한다.
- 세션 쿠키가 언제 발급되는지 본다.
- HTTP 와 HTTPS 의 관찰 가능한 정보 차이를 설명한다.
- 캡처 파일에 민감 정보가 포함될 수 있는 이유를 설명한다.

9.4 과제 4: 취약점 보고서 작성

DVWA 취약점 하나를 선택해 다음 구조로 보고서를 작성한다.

1. 제목
2. 요약
3. 영향
4. 재현 환경
5. 관찰 증거
6. 원인
7. 조치 방안
8. 재검증 기준
9. 참고 자료

9.5 토론 질문

- OWASP Top 10 의 점수가 낮은 A10 이 실제 사고에서 왜 중요할까?
- 취약점 스캐너 결과를 그대로 보고서에 넣으면 왜 위험할까?
- XSS 에서 서버 측 방어와 브라우저 측 방어의 역할은 어떻게 다른가?
- 비밀번호 정책에서 복잡도 강제와 유출 비밀번호 차단 중 무엇이 더 실용적인가?
- Kali Linux 를 일상 OS 로 쓰지 않는 것이 좋은 이유는 무엇인가?
- 레드팀이 성공했는데 블루팀이 아무것도 탐지하지 못했다면 산출물은 어떻게 달라져야 하는가?

10 1 페이지 요약본

10.1 OWASP 2017 핵심

- A1 Injection: 입력이 명령/쿼리가 되는 문제. 파라미터 바인딩과 안전한 API 가 핵심.
- A2 Broken Authentication: 인증/세션 관리 실패. MFA, 세션 재발급, 실패 제한, 기본 계정 제거.
- A3 Sensitive Data Exposure: 민감 데이터 보호 실패. 데이터 분류, 최소 수집, TLS, 암호화, 로그 마스킹.
- A4 XXE: XML 파서 외부 엔터티 처리. DTD/external entity 비활성화, outbound 제한.
- A5 Broken Access Control: 권한 밖 기능/데이터 접근. 서버 측 deny-by-default 와 객체 소유권 검사.
- A6 Security Misconfiguration: 기본값, 디버그, 노출 포트, 오류 메시지, 패치 누락. 하드닝과 자동 점검.
- A7 XSS: 브라우저에서 공격자 스크립트 실행. 컨텍스트별 출력 인코딩과 안전한 DOM API.
- A8 Insecure Deserialization: 조작 객체 역직렬화. 신뢰할 수 없는 객체 역직렬화 금지, 무결성 검증.
- A9 Known Vulnerable Components: 취약 컴포넌트 사용. 의존성 목록, SCA, 패치 정책.
- A10 Insufficient Logging & Monitoring: 공격 탐지/대응 실패. 구조화 로그, 중앙 수집, 알림, IR 계획.

10.2 기본 해킹 기법 핵심

- 정찰은 수동과 능동으로 나뉘며, 능동 정찰은 반드시 허가가 필요하다.
- 스니핑은 네트워크 통신을 관찰하는 기술이며, 평문 프로토콜과 메타데이터가 주요 분석 대상이다.
- LAND 공격은 source/destination IP 와 port 가 같은 spoofed SYN packet 을 이용하는 고전적 DoS 사례다.
- 인증 공격은 계정 잠금과 서비스 장애를 만들 수 있으므로 랩 외에서는 별도 승인이 필요하다.
- 파일 업로드, 경로 조작, SSRF, CSRF 는 OWASP 2017 범주와 함께 별도로 익혀야 한다.
- 좋은 레드팀 실습은 공격 성공보다 원인, 영향, 증거, 조치, 탐지 여부를 설명한다.

10.3 DVWA 핵심

- DVWA 는 PHP/MariaDB 기반의 의도적으로 취약한 웹 애플리케이션이다.
- 공식 GitHub 저장소의 최신 소스를 사용한다.
- Docker 설치 시 `docker compose up -d` 후 `http://localhost:4280` 에서 접근한다.
- 기본 계정은 `admin/password` 다.
- 인터넷에 노출하면 안 되며, 로컬 또는 격리 랩에서만 사용한다.

10.4 Kali Linux 핵심

- Kali Linux 는 Debian 기반의 침투 테스트/보안 감사용 Linux 배포판이다.
- “Kali Red”는 보통 기존 Kali 의 공격 보안 용도를 가리키는 비공식 표현으로 이해한다.
- Kali Purple 은 방어 보안/블루팀/퍼플팀 방향의 공식 프로젝트다.
- 초급자는 pre-built VM 과 스냅샷을 권장한다.
- 공식 이미지와 SHA256/GPG 검증을 습관화한다.
- 도구 사용보다 Linux, TCP/IP, HTTP, 인증, 로그 이해가 먼저다.

11 참고 자료

아래 자료를 기준으로 내용을 검증했다. 일부 설명은 교육 목적에 맞게 요약, 재구성했다.

- [S1] OWASP Top Ten Web Application Security Risks: https://owasp.org/www-project-top-ten/
- [S2] OWASP Top 10 2017 - Top 10 overview: https://owasp.org/www-project-top-ten/2017/Top_10
- [S3] OWASP Top 10 2017 - Risk factor summary: https://owasp.org/www-project-top-ten/2017/Details_About_Risk_Factors
- [S4] OWASP A1:2017 Injection: https://owasp.org/www-project-top-ten/2017/A1_2017-Injection
- [S5] OWASP A2:2017 Broken Authentication: https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication
- [S6] OWASP A3:2017 Sensitive Data Exposure: https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure
- [S7] OWASP A4:2017 XML External Entities (XXE): https://owasp.org/www-project-top-ten/2017/A4_2017-XML_External_Entities_%28XXE%29
- [S8] OWASP A5:2017 Broken Access Control: https://owasp.org/www-project-top-ten/2017/A5_2017-Broken_Access_Control
- [S9] OWASP A6:2017 Security Misconfiguration: https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration
- [S10] OWASP A7:2017 Cross-Site Scripting (XSS): https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_%28XSS%29
- [S11] OWASP A8:2017 Insecure Deserialization: https://owasp.org/www-project-top-ten/2017/A8_2017-Insecure_Deserialization
- [S12] OWASP A9:2017 Using Components with Known Vulnerabilities: https://owasp.org/www-project-top-ten/2017/A9_2017-Using_Components_with_Known_Vulnerabilities
- [S13] OWASP A10:2017 Insufficient Logging & Monitoring: https://owasp.org/www-project-top-ten/2017/A10_2017-Insufficient_Logging%2526Monitoring
- [S14] DVWA official GitHub repository: https://github.com/digininja/DVWA
- [S15] Kali Linux - What is Kali Linux?: https://www.kali.org/docs/introduction/what-is-kali-linux/
- [S16] Kali Linux - Which image should I download?: https://www.kali.org/docs/introduction/what-image-to-download/
- [S17] Kali Linux - Downloading Kali Linux and verifying images: https://www.kali.org/docs/introduction/download-official-kali-linux-images/
- [S18] Kali Linux - Download images securely: https://www.kali.org/docs/introduction/download-images-securely/
- [S19] Kali Linux - Default credentials: https://www.kali.org/docs/introduction/default-credentials/
- [S20] Kali Linux - Metapackages: https://www.kali.org/docs/general-use/metapackages/
- [S21] Kali Linux 2023.1 Release - Kali Purple: https://www.kali.org/blog/kali-linux-2023-1-release/
- [S22] NIST SP 800-115 - Technical Guide to Information Security Testing and Assessment: https://csrc.nist.gov/pubs/sp/800/115/final
- [S23] NIST CSRC glossary - Network Sniffing: https://csrc.nist.gov/glossary/term/network_sniffing
- [S24] MITRE ATT&CK T1040 - Network Sniffing: https://attack.mitre.org/techniques/T1040/
- [S25] MITRE ATT&CK T1595 - Active Scanning: https://attack.mitre.org/techniques/T1595/
- [S26] CERT/CC VU#396645 - Microsoft Windows vulnerable to DoS via LAND attack: https://www.kb.cert.org/vuls/id/396645
- [S27] Juniper documentation - Configuring TCP Land Attack Screen: https://www.juniper.net/documentation/us/en/software/ccfips22.2/cc-security-vSRX3.0/cc-security/topics/task/configuring-tcp-land-attack.html
- [S28] RFC 4987 - TCP SYN Flooding Attacks and Common Mitigations: https://datatracker.ietf.org/doc/rfc4987/

12 부록: 보고서 템플릿

```
[취약점 제목]

1. 개요
- 영향받는 시스템:
- 취약점 분류:
- OWASP/CWE 매핑:
- 심각도:

2. 영향
- 공격자가 가능한 행동:
- 필요한 권한:
- 노출될 수 있는 데이터:
- 업무 영향:

3. 재현 환경
- 테스트 일시:
- 테스트 계정:
- 대상 URL/IP:
- 브라우저/도구:
- 제한 사항:

4. 관찰 증거
- 요청 요약:
- 응답 요약:
- 로그/스크린샷:
- 민감 정보 마스킹 여부:

5. 원인 분석
- 누락된 검증:
- 신뢰 경계:
- 취약 코드/설정 위치:

6. 조치 방안
- 단기 완화:
- 장기 수정:
- 운영 탐지:

7. 재검증 기준
- 수정 후 기대 동작:
- 재테스트 방법:

8. 참고 자료
- 내부 티켓:
- 외부 레퍼런스:
```

13 부록: 실습 윤리 서약 예시

```
나는 본 교육에서 제공된 랩, CTF, 명시적으로 허가된 시스템 외에는 스캐닝, 공격, 패킷 캡처, 계정 시도, 취약점
↪ 검증을 수행하지 않는다.
나는 실습 중 발견한 민감 정보를 저장, 공유, 재사용하지 않는다.
나는 장애 유발 가능성이 있는 테스트를 사전 승인 없이 수행하지 않는다.
나는 실습 결과를 원인, 영향, 조치 방안 중심으로 보고한다.
```