**졸업 논문**

**주제: 웹 애플리케이션 및 보안 문제**

****

**지도교수님: 정인상**

**이름: 부이 반 안**

**학번: 1692177**

**전공: 컴퓨터공학부**

**2017년 05월 10일**

**논문 개요**

요즘 인터넷은 매우 친숙해 졌으며 국가가 이미지를 소개하거나 자신을 소개 할 사람의 개인 웹 사이트를 소개하는 유용한 도구가 되었습니다. 모두 웹 애플리케이션의 지속적인 개발로 이어졌습니다. 그리고 점차 웹 애플리케이션의 개념이 대중화 되었습니다. 인터넷에서 웹 응용 프로그램이 대중화되고 널리 적용되면서 웹 응용 프로그램 공격도 매우 복잡해졌습니다. 이를 통해 웹 응용 프로그램 및 사용자의 개인 정보에 대한 정보 보안을 보장하는 방법을 알아야합니다. 웹 응용 프로그램 전문 지식 및 웹 응용 프로그램 공격 개념도 산업 문서에서 점점 더 일반화되고 있습니다. 웹 개발자 및 웹 마스터 도구는 웹 응용 프로그램 취약점을 찾는 데 도움이 되는 것처럼 보이지만 웹 응용 프로그램의 빠른 개발을 따라 가지 못합니다. 물론 점점 더 다양한 공격이 웹 응용 프로그램 버그, 관리자 또는 웹 응용 프로그램 프로그래머를 완전히 악용하므로 웹 응용 프로그램 공격을 완전히 방지 할 수는 없습니다.

통계에 따르면 인터넷 공격의 75 %가 웹 응용 프로그램 공격이며 막대한 피해를 초래합니다. 따라서 웹 애플리케이션 공격에 대해 배우는 것이 필수적입니다. 관리자가 웹 응용 프로그램을 예방하고 보호하기위한 효과적인 조치를 취할 수 있습니다.

웹 응용 프로그램의 보안 허점을 이해하고 분석하여 교정 솔루션을 제안하기 위해 논문을 수행했습니다. 동시에 이 논문은 경험이 없는 웹 프로그래머가 응용 프로그램을 만드는 과정에서 오류를 피할 수 있도록 도와줍니다.

목 차

[**제 1 부 : 서론** 7](#_Toc40637378)

[**제 1 장: 웹 애플리케이션 개요** 8](#_Toc40637379)

[**1.1: 웹 응용 프로그램 개념(web application)** 8](#_Toc40637380)

[**1.2: 구조, 기능, 작동 원리** 9](#_Toc40637381)

[**1.2.1: 웹 애플리케이션의 기본 구조** 9](#_Toc40637382)

[**1.2.2: 웹 애플리케이션의 기본 기능** 11](#_Toc40637383)

[**1.2.3: 웹 애플리케이션의 기본 운영 원리** 11](#_Toc40637384)

[**1.2.4: 웹 애플리케이션 보안 문제** 13](#_Toc40637385)

[**제 2 장: 관련 용어 및 개념 소개** 15](#_Toc40637386)

[**2.1: 웹 애플리케이션의 일반적인 개념** 15](#_Toc40637387)

[**2.2: 해킹 및 웹 응용 프로그램 보안과 관련된 개념 및 용어** 16](#_Toc40637388)

[**2.2.1: 해커 개념** 16](#_Toc40637389)

[**2.2.2: HTTP 헤더** 16](#_Toc40637390)

[**2.2.3: SESSION** 17](#_Toc40637391)

[**2.2.4: 쿠키** 17](#_Toc40637392)

[**2.2.5: 프록시** 19](#_Toc40637393)

[**제 3 장: 기본 웹 애플리케이션 공격 기술** 19](#_Toc40637394)

[**3.1: 웹 액세스 제어(Web Access Control)** 19](#_Toc40637395)

[**3.1.1: 뒷문을 통해 시스템에 들어감(Back door)** 19](#_Toc40637396)

[**3.2:세션 점유(Session Mangement )** 20](#_Toc40637397)

[**3.2.1: 세션 수정(Session Fixation )** 20](#_Toc40637398)

[**3.2.2: 세션 도난 (Session Hijacking)** 20](#_Toc40637399)

[**3.3: 유효한 데이터 입력 확인시 오류 활용( Input Validation)** 20](#_Toc40637400)

[**3.3.1: 브라우저 측 언어로 데이터의 정확성 확인(Client-Side Validation)** 20](#_Toc40637401)

[**3.3.2: 버퍼 오버플로(Buffer overflow)** 20](#_Toc40637402)

[**3.3.3: URL 인코딩 ( URL Encoding)** 20](#_Toc40637403)

[**3.3.4: 메타 문자(Meta-characters)** 20](#_Toc40637404)

[**3.3.5: 링크를 통과(Path Traversal)** 20](#_Toc40637405)

[**3.3.6: 대상 브라우저에 실행 코드 삽입(Cross- Site Scripting)** 20](#_Toc40637406)

[**3.3.7: 시스템의명령어 추가( OS Command Injection)** 21](#_Toc40637407)

[**3.3.8: SQL 쿼리 삽입(SQL Injection)** 21](#_Toc40637408)

[**3.3.9: 서버 측 언어(Server Side Includes)** 21](#_Toc40637409)

[**3.3.10: 공문자(Null Characters)** 21](#_Toc40637410)

[**3.3.11: 파라미터 전송으로 작동 (Parameter Manipulation)** 21](#_Toc40637411)

[**3.4: 티나는 정보(Informational)** 21](#_Toc40637412)

[**3.5: 서비스 거부( Denial Of Service(DoS))** 22](#_Toc40637413)

[**제 2 부:** 23](#_Toc40637414)

[**웹 애플리케이션 공격 및 보안기술** 23](#_Toc40637415)

[**제 4 장: 파라미터 전송으로 조작** 23](#_Toc40637416)

[**4.1: URL 조작** 24](#_Toc40637417)

[**4.1.1:개념** 24](#_Toc40637418)

[**4.1.2: 고치는 방법** 24](#_Toc40637419)

[**4.2:숨겨진 폼(form)변수 조작** 24](#_Toc40637420)

[**4.2.1:개념** 24](#_Toc40637421)

[**4.2.2: 고치는 방법** 25](#_Toc40637422)

[**4.3: 쿠키(COOKIE) 조작** 26](#_Toc40637423)

[**4.3.1:개념** 26](#_Toc40637424)

[**4.3.2: 고치는 방법** 26](#_Toc40637425)

[**제 5 장: SQL 심문 삽입( SQL INJECTION)** 27](#_Toc40637426)

[**5.1. 개념** 28](#_Toc40637427)

[**5.2: 데이터베이스 모델 소개** 28](#_Toc40637428)

[**5.3: 공격 방법** 28](#_Toc40637429)

[**5.3.1: SQL Injection 공격 기술** 28](#_Toc40637430)

[**5.3.2: SELECT 문을 기반으로 공격** 30](#_Toc40637431)

[**5.3.3: HAVING 문을 기반으로 한 공격** 31](#_Toc40637432)

[**5.3.4: UNION 조합 문에 기반으로 한 공격** 31](#_Toc40637433)

[**5.3.5: INSERT 명령 기반으로 한 공격** 32](#_Toc40637434)

[**5.3.6: STORED PROCEDURE을 기반으로 공격** 33](#_Toc40637435)

[**5.4: 예방하는 방법** 33](#_Toc40637436)

[**제 6 장: 세션 점유(Possessing The Session)** 34](#_Toc40637437)

[**6.1: SESSIONID의 개요** 34](#_Toc40637438)

[**6.2: 세션 고정(Session Fixation)** 35](#_Toc40637439)

[**6.3: 세션 하이재킹(Session Hijacking)** 36](#_Toc40637440)

[**제 7 장: 버퍼 오버플로(Buffer Overflow)** 38](#_Toc40637441)

[**7.1:개념** 38](#_Toc40637442)

[**7.2: 일반적인 버퍼 오버플로 오류 유형** 38](#_Toc40637443)

[**7.3: 예방 조치** 39](#_Toc40637444)

[**제 8 장: 서비스 거부** 40](#_Toc40637445)

[**8.1:개념** 40](#_Toc40637446)

[**8.2: DOS의 공격 대상** 40](#_Toc40637447)

[**8.3: 공격 기술** 40](#_Toc40637448)

[**8.3.1: 타입 1 공격** 40](#_Toc40637449)

[**8.3.2: 타입 2 공격** 40](#_Toc40637450)

[**8.3.3: 시스템 리소스에 대한 공격 유형** 41](#_Toc40637451)

[**8.4: 예방 조치** 41](#_Toc40637452)

[**제 9 장: 공격 프로세스 요약 및이를 방지하는 방법** 42](#_Toc40637453)

[**9.1: 해커 공격 프로세스 요약** 42](#_Toc40637454)

[**9.1.1: 대상 인프라에서 정보를 수집** 42](#_Toc40637455)

[**9.1.2: 웹 애플리케이션 설문 조사** 43](#_Toc40637456)

[**9.1.3:공격** 44](#_Toc40637457)

[**9.2: 예방 조치 요약** 44](#_Toc40637458)

[**9.2.1: 네트워크 관리자와 함께** 44](#_Toc40637459)

[**9.2.2: 웹 애플리케이션 디자이너와 함께** 45](#_Toc40637460)

[**9.2.3: 웹 애플리케이션 사용자와 함께** 45](#_Toc40637461)

[**제 10 장:** 46](#_Toc40637462)

[**공격 및 정보 보안에 일반적인 소프트웨어 소개** 46](#_Toc40637463)

[**10.1: NMAP** 46](#_Toc40637464)

[**10.1.1: 개요** 46](#_Toc40637465)

[**10.1.2:가능** 46](#_Toc40637466)

[**10.1.3:활용법** 46](#_Toc40637467)

[**10.2: SQLMAP** 48](#_Toc40637468)

[**10.2.1: 개요** 49](#_Toc40637469)

[**10.2.2:가능** 49](#_Toc40637470)

[**10.2.3:활용법** 50](#_Toc40637471)

[**제 3 부: 결론** 52](#_Toc40637472)

[**I: 달성 된 문제** 53](#_Toc40637473)

[**II: 논문의 개발 방향** 53](#_Toc40637474)

[**참고 문헌** 54](#_Toc40637475)

그림 목차

[그림 1: 웹 응용 프로그램 9](#_Toc40636699)

[그림 2: MVC (Model-View-Controller) 모델 10](#_Toc40636700)

[그림 3: 웹 응용 프로그램 서비스의 모델 11](#_Toc40636701)

[그림 4: 웹 응용 프로그램의 작업 단계 13](#_Toc40636702)

[그림 5: 웹 응용 프로그램의 운영 모델 14](#_Toc40636703)

[그림 6: nmap의 인터페이스 45](#_Toc40636704)

[그림 7: nmap의 빠른 스캔 유형 설명 46](#_Toc40636705)

[그림 8: nmap 부분 사용시 결과 (nmap output) 46](#_Toc40636706)

[그림 9: 검사중인 호스트의 이미지 47](#_Toc40636707)

[그림 10: sqlmap 48](#_Toc40636708)

# **제 1 부 : 서론**

# **제 1 장: 웹 애플리케이션 개요**

## **1.1: 웹 응용 프로그램 개념(web application)**



그림 1: 웹 응용 프로그램

웹 응용 프로그램은 HTTP 프로토콜을 사용하여 다른 사용자 나 시스템과 상호 작용하는 클라이언트 / 클라이언트 응용 프로그램입니다.

클라이언트 클라이언트는 일반적으로 Internet Explorer 또는 Netscape Navigator와 같은 웹 브라우저입니다. 자동화 된 브라우저 역할을 하는 사용자 에이전트 프로그램 일 수도 있습니다. 사용자는 웹 페이지에 미치는 영향을 통해 서버와 정보를 주고받습니다. 프로그램은 교환, 포럼, 전자 우편 송수신 페이지 일 수 있습니다.

웹 애플리케이션 구축 기술의 개발 속도 또한 매우 빠르게 성장하고 있습니다. 과거에는 종종 CGI (Common Gateway Interface)를 사용하여 구축 된 웹 응용 프로그램이 웹 서버에서 실행되었으며 동일한 서버의 간단한 데이터베이스에 연결할 수 있었습니다. 오늘날 웹 응용 프로그램은 일반적으로 Java (또는 유사한 언어)로 작성되며 여러 데이터 소스에 연결하는 분산 서버에서 실행됩니다.

이 개념의 의미를 완전히 이해하기 위해 웹 응용 프로그램의 기능 및 작업 구조에 대해 알아 보겠습니다.

## **1.2: 구조, 기능, 작동 원리**

### **1.2.1: 웹 애플리케이션의 기본 구조**

간단한 웹 응용 프로그램의 모델은 MVC (Model-View-Controller) 모델입니다.

- 모델 : 모든 처리 방법, 데이터베이스 액세스, 클래스와 같은 데이터 설명 개체, 처리 기능 등을 포함하는 구성 요소입니다.

- 뷰 : 정보 표시 및 사용자와의 상호 작용을 담당하는 부분입니다. 또는 응용 프로그램의 인터페이스임을 대략 이해할 수 있습니다.

- 컨트롤러 : 컨트롤러는 데이터베이스를 처리 할 클라이언트의 작업을 처리하고 탐색하는 부분입니다 (있는 경우). 다시 말해, 컨트롤러는 뷰와 모델 사이의 브릿지 입니다.

작동을 더 잘 이해하려면 다음 이미지를 참조하십시오.

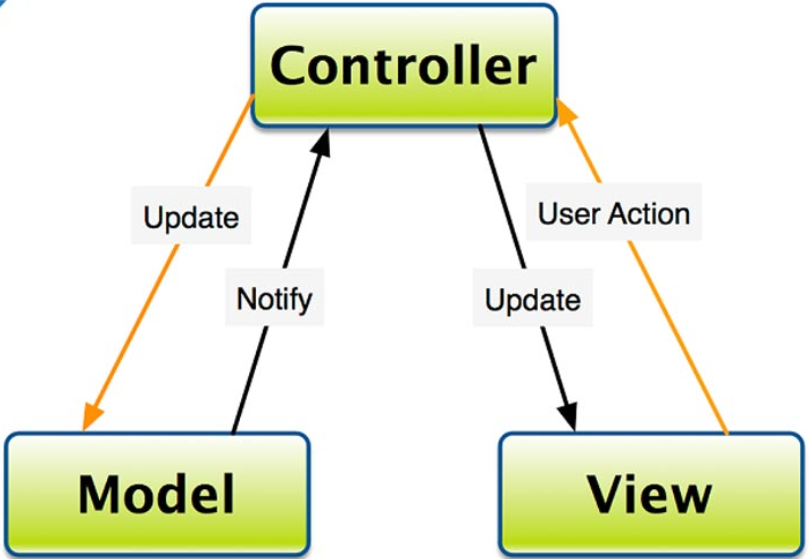


그림 2: MVC (Model-View-Controller) 모델

그리고 이것은 웹 응용 프로그램 서비스의 모델:

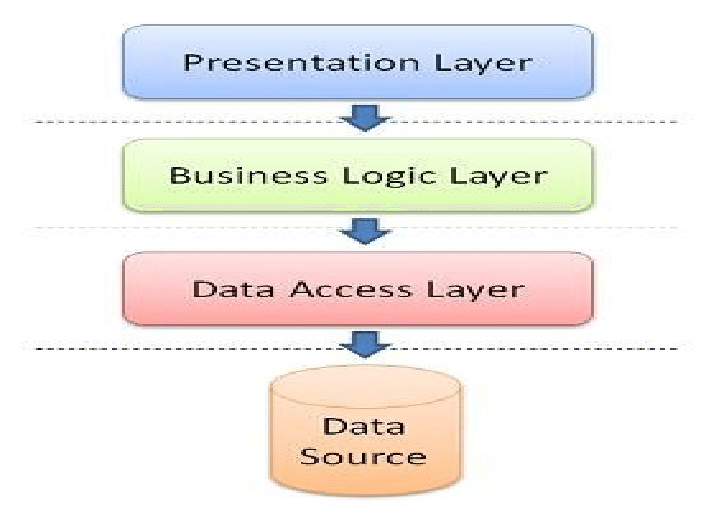


그림 3: 웹 응용 프로그램 서비스의 모델

- 프리젠테이션 계층 : 사용자와 통신하는 데 사용되며 주요 작업은 데이터를 표시하고 사용자로부터 데이터를 받는 것입니다.

- 비즈니스 로직 층 : 주요 임무는 소프트웨어의 기능을 제공하는 것입니다.

- 데이터 계층 : 데이터를 저장하고 Business Logic 계층이 데이터베이스를 검색, 추출, 업데이트 할 수 있습니다.

### **1.2.2: 웹 애플리케이션의 기본 기능**

프로그램의 어느 곳에서나 존재하는 인기있는 웹 응용 프로그램. 수천 대의 컴퓨터에 소프트웨어를 배포 및 설치하지 않고 웹 응용 프로그램을 업데이트하고 유지 관리 할 수 있는 기능이 인기의 주된 이유입니다. 웹 애플리케이션은 웹 메일, 온라인 판매, 온라인 경매, 위키, 토론 포럼, 웹 로그, MMORPG, 고객 관계 관리 시스템 및 기타 여러 기능을 구현하는 데 사용됩니다.

웹은 또한 크고 작은 수천 개의 조직 및 비즈니스를 위한 스마트 판매 채널입니다. 오늘날 10 억 명 이상의 인터넷 사용자 (소스 : Computer Industry Almanac 2006)가있는 미국 전자 상거래는 2006 년 거래에 약 102 억 달러를 사용합니다 (출처 : comScore Networks 2007).

이러한 모든 데이터는 즉시 또는 나중에 사용할 수 있는 방식으로 포장, 저장, 처리 및 전송되어야 합니다. 등록, 제출, 쿼리, 로그인, 판매 및 컨텐츠 관리 시스템 분야의 웹 응용 프로그램은 원하는 모든 작업을 수행 할 수 있는 위젯 웹 사이트입니다.

웹은 비즈니스가 온라인 세계에서 온라인 이미지를 향상시켜 잠재적 고객 및 기존 고객과의 장기적인 수익성 있는 관계를 만들고 유지하는 데 도움이 되는 기본 요소입니다.

웹 애플리케이션이 전세계 어디에서나 유비쿼터스 되었다는 것은 의심의 여지가 없습니다. 그러나 첨단 기술과 복잡한 자연 요소로 인해 일상 생활에서 심각하게 오해 조차도 정확하게 알려져 있지 않습니다.

오늘날 웹 사이트는 19 세기 이전의 그래픽 및 정적 텍스트 유형과는 다릅니다. 최신 웹 사이트에서는 사용자가 자신의 설정 및 환경 설정에 따라 개인화 된 동적 컨텐츠를 다운로드 할 수 있습니다. 또한, 클라이언트에서 스크립트를 실행할 수도 있고, 이메일, 대화 형 매핑 소프트웨어 (Yahoo Mail, Google Maps)와 같은 응용 프로그램의 인터페이스로 인터넷 브라우저를 "변경"할 수 있습니다.

가장 중요한 것은 민감한 고객 데이터 (예 : 개인 정보, 신용 카드 번호, 사회 보장 정보 등)를 패키징, 처리, 저장 및 전송할 수 있는 최신 웹 사이트를 사용할 수 있습니다. 지금 또는 이후에. 그리고 이것은 웹 응용 프로그램을 통해 수행됩니다. 웹 메일 구성 요소 (이메일), 로그인 페이지, 지원 프로그램 및 제품 요청 양식 또는 구매 및 판매, 컨텐츠 관리 시스템, 최신 웹 사이트 개발, 제공 기업은 잠재 고객과 기존 고객에게 연락해야 함을 의미합니다.

### **1.2.3: 웹 애플리케이션의 기본 운영 원리**

이전 형식의 클라이언트-서버 컴퓨팅에서 각 응용 프로그램에는 사용자 인터페이스 역할을 하는 자체 클라이언트 프로그램이 있으며 각 사용자의 개인용 컴퓨터에 별도로 설치해야 합니다. 응용 프로그램의 서버 부분을 업그레이드하려면 비용을 지원하고 생산성을 낮추는 것 외에도 각 사용자 워크 스테이션에 설치된 모든 클라이언트를 업그레이드 해야합니다.

HTML / XHTML과 같은 널리 사용되는 브라우저에서 지원하는 표준 형식으로 다양한 웹 문서를 생성하는 유연한 웹 응용 프로그램입니다. JavaScript와 같은 표준화 된 언어의 사용자측 스크립팅 언어는 종종 사용자 인터페이스에 더 역동적 인 요소를 갖도록 추가됩니다. 일반적으로 모든 단일 웹 페이지는 안정적인 문서로 사용자에게 전송되지만 페이지 순서는 사용자가 입력 한 내용이 템플릿 구성 요소를 통해 반환 될 때 시각적 의미를 제공 할 수 있습니다. 웹은 북마크에 포함됩니다. 트랜잭션 중에 웹 브라우저는 페이지를 해석하고 표시하며 모든 웹 응용 프로그램의 공유 사용자 역할을 합니다.

MVC 모델 (모델-보기-컨트롤러)을 따릅니다.

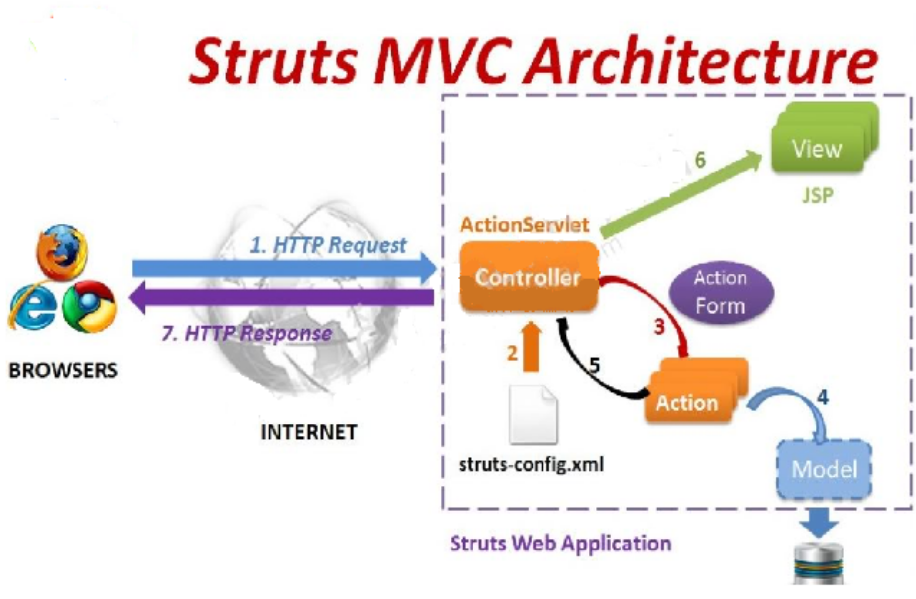


그림 4: 웹 응용 프로그램의 작업 단계

**간단한 웹 응용 프로그램의 작업 단계:**

* 1 단계 : 브라우저가 웹 애플리케이션으로 전송 된 HTTP 요청을 작성합니다.
* 2 단계 : Struts의 기본 컨트롤러는이 요청을 캡처하고 해당 URL을 분석 한 후 struts-config.xml 파일을 사용하여 해당 요청을 해당 Action 클래스로 보내는 ActionServlet 클래스입니다.
* 3 단계 : 액션 클래스는 Struts Framework의 클래스입니다. 우리의 응용 프로그램은 일반적 으로이 클래스에서 확장되고 해당 비즈니스를 처리하는 코드를 작성합니다. 예를 들어, 사용자 로그인 및 로그 아웃을 처리하기위한 LoginAction이 있습니다.
* 4 단계 : 필요한 경우 조치 클래스가 데이터베이스에 액세스하여 업데이트 할 수 있습니다.
* 5 단계 : Action 클래스가 비즈니스 처리를 완료하면 데이터 (있는 경우)와 함께 전달 / 리디렉션 요청을 컨트롤러에 보냅니다.
* 6 단계 : 컨트롤러가보기 계층의 해당 JSP 페이지로 제어를 전달합니다. 이 JSP 페이지가 데이터를 사용하는 경우 컨트롤러가이를 제공합니다 (이것은 조치가 컨트롤러에 대해 작성하고 경주 한 데이터입니다).
* 7 단계 : JSP 페이지가 준비되면 컨트롤러가 HTTP 응답을 작성하여 브라우저로 보내 브라우저가 화면에 표시되도록 합니다.

### **1.2.4: 웹 애플리케이션 보안 문제**

웹 응용 프로그램은 모든 측면에서 매우 빠르게 개발되므로 적용 범위가 넓고 웹 응용 프로그램의 보안 문제도 더욱 주의를 기울여야합니다. 현재 상당한 개선이 진행되고 있음을 부정하지는 않지만 웹 응용 프로그램의 보안 문제는 지속적으로 증가하고 있습니다. 원인은 부적절한 코드 때문일 수 있습니다. 많은 심각한 취약점이나 취약점으로 인해 해커는 민감한 데이터를 추출하는 데이터베이스에 직접 액세스하고 액세스 할 수 있습니다. 많은 데이터베이스에는 개인 정보, 재무 정보와 같은 중요한 정보가 포함되어있어 대부분의 해커에게 정기적 인 대상이 됩니다. 비즈니스 방해 행위는 여전히 정기적으로 일어나고 있지만. 그러나 해커들은 이제 데이터 판매로 인한 큰 이익 때문에 데이터베이스 서버에 있는 민감한 데이터에 액세스하는 능력을 향상시키는 것을 선호합니다.

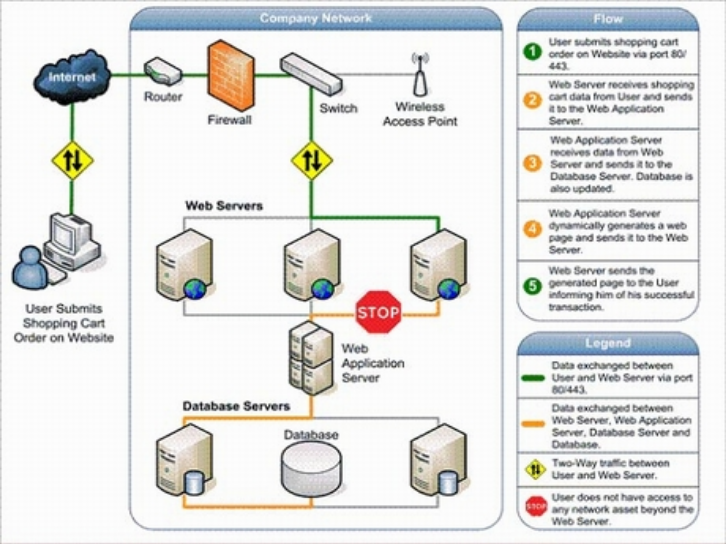


그림 5: 웹 응용 프로그램의 운영 모델

위에서 설명한 활동 상자에서 해커가 약간의 창의력으로 데이터베이스의 정보에 빠르게 액세스하는 것이 얼마나 쉬운 지 알 수 있습니다. 더 운이 좋으면 웹 응용 프로그램의 태만 또는 사용자 오류로 인해 취약점이 발생할 수 있습니다.

언급 된 바와 같이, 웹 사이트는 요청 된 정보를 사용자에게 전달하기 위해 데이터베이스에 의존한다. 웹 응용 프로그램이 안전하지 않은 경우 (예 : 취약성, 일종의 해킹 기술 발생) 민감한 정보가 포함 된 전체 데이터베이스는 심각한 위험에 노출됩니다.

일부 해커는 악의적 인 코드를 취약한 웹 응용 프로그램에 주입하여 사용자를 속여 피싱 웹 사이트로 유도 할 수 있습니다. 크로스 사이트 스크립팅이라고 하는이 기술은 웹 서버 자체와 데이터베이스 저장소에 취약점이 없는 경우에도 사용할 수 있습니다.

최근 연구에 따르면 사이버 공격의 75 %가 웹 응용 프로그램 수준에서 수행됩니다. 고객 요구 사항, 직원, 공급 업체 및 기타 여러 관련 담당자의 요청에 따라 서비스를 제공하려면 웹 사이트 및 관련 웹 응용 프로그램을 항상 24/7 사용할 수 있어야합니다.

ZF 방화벽, SSL은 웹 사이트에 대한 액세스가 공개되어 있어야 누구나 웹 사이트를 방문 할 수 있기 때문에 웹 응용 프로그램을 해킹으로부터 보호 할 수 없습니다. 모든 최신 데이터베이스 시스템 (예 : Microsoft SQL Server, Oracle, MySQL)은 특정 포트 (예 : 포트 80, 443)를 통해 액세스 할 수 있습니다. 원하는 경우 운영 체제의 보안 메커니즘을 무시할 때 누군가가 데이터베이스에 직접 연결할 수 있습니다. 이 포트는 합법적 인 네트워크 트래픽과 통신 할 수 있도록 개방 된 상태로 유지되므로 주요한 위험한 결점을 형성합니다.

 웹 응용 프로그램은 종종 고객 데이터베이스와 같은 최종 데이터에 액세스하고 중요한 데이터를 제어하므로 절대적으로 보안하기가 어렵습니다. 현재 데이터 액세스에는 일반적으로 패키징 및 데이터 전송을 허용하는 스크립트가 수반되지 않습니다. 해커가 스크립트의 약점을 인식하면 다른 영역으로의 트래픽을 쉽게 다시 열고 사용자의 개인 정보를 불법으로 분할 할 수 있지만 때로는 의도적으로 하지는 않습니다.

대부분의 웹 응용 프로그램은 자체 제작되므로 유사한 소프트웨어보다 인증 테스트가 적습니다. 따라서 사용자 지정 응용 프로그램이 더 취약한 경우가 많습니다.

웹 응용 프로그램은 데이터베이스, 특히 사용자 지정 응용 프로그램의 게이트웨이라고 할 수 있습니다. 정기적 인 보안 검사를 거칠 필요가 없으므로 최상의 보안으로 개발되지 않았습니다. 일반적으로 "웹 사이트의 어느 부분이 안전하지만 공격에 노출되어 있다고 생각합니까?"라는 질문에 대답해야 합니다. "우리가 하지 말아야 할 일 중 일부를 수행하도록 응용 프로그램에 어떤 데이터를 가져 오나요?".

현재 해커는 기본 기술에서 첨단 기술과 기술이 필요한 웹 응용 프로그램까지 웹 응용 프로그램을 공격 할 수 있는 여러 가지 방법이 있습니다. 도구 (도구) 지원은 점점 더 태어 났으며 네트워크 관리자를 크게 지원하고 적시에 패치를 제공 할 수 있는 구멍을 찾았지만 동시에 양날의 칼 이기도 합니다. 해커는 이러한 도구를 사용하여 웹 응용 프로그램의 취약성을 감지 할 수 있으며 이 취약성에 대한 해당 공격이 발생하여 많은 손실을 초래할 수 있습니다.

따라서 기본 공격 기술 및 도구에 대한 연구는 웹 응용 프로그램 보안에 대한 연구에서 반드시 필요한 것입니다.

# **제 2 장: 관련 용어 및 개념 소개**

## **2.1: 웹 애플리케이션의 일반적인 개념**

웹 브라우저는 사용자가 데이터를 쿼리하고 웹 사이트 내의 웹 페이지에 있는 콘텐츠와 상호 작용할 수 있는 소프트웨어 응용 프로그램입니다.

 정적 웹 페이지 : 사용자가 리소스에 대한 요청을 제출하면 서버는 해당 리소스를 반환합니다. 웹 페이지는 형식화되고 배포 된 텍스트 일뿐입니다. 브라우저의 경우 정적 웹 페이지는 어려운 문제가 아니며 웹 페이지는 원래 인터넷을 통한 이벤트, 주소 또는 일정에 대한 정보만을 위한 것입니다. 웹 페이지.

동적 웹 사이트 : 웹 기반 소프트웨어 또는 http 프로토콜로 실행되는 프로그램이 지원하는 웹 사이트를 지칭하는 용어입니다. 본질적으로 동적 웹 사이트는 웹 소프트웨어 (웹용 응용 프로그램 모듈)와 "연결된"정적 웹 사이트를 의미합니다. 이 소프트웨어 프로그램을 사용하면 웹 사이트 소유자는 전문 인력에 의존하지 않고도 웹 사이트를 운영하고 정보를 편집하고 업데이트 할 권리가 있습니다.

도메인 이름 (도메인 이름)은 인터넷에서 웹 사이트의 식별자입니다. 도메인 이름은 종종 회사의 회사 이름 및 브랜드 이름과 관련이 있습니다. 도메인 이름은 고유하며 처음 등록한 사람에게 할당됩니다.

두 가지 유형의 이름이 있습니다

- 국제 도메인 이름 형식 : www.tencongty.com (.net, .biz .org, .info ...)

- 국가 도메인 이름 형식 : www.hansung.ac.kr

- 웹 호스팅은 온라인 데이터 저장 및 공유 서비스로, 월드 와이드 웹 (www), 파일 전송 (FTP), 메일 등의 인터넷 서비스가 설치된 서버의 공간이며 내부를 포함 할 수 있습니다 해당 공간의 웹 페이지 또는 데이터를 사용하십시오.

## **2.2: 해킹 및 웹 응용 프로그램 보안과 관련된 개념 및 용어**

### **2.2.1: 해커 개념**

해커는 네트워크 시스템을 방해하는 사람들을 지칭하는 용어입니다. 해커는 일반적으로 컴퓨터 전문가입니다. 해커는 시스템에 허점을 만들지 않지만 운영 체제, 데이터 관리 시스템, 프로그래밍 언어에 익숙한 사람입니다. 이들은 지식을 사용하여 네트워크 취약점을 탐색하고 악용합니다.

해커의 개념은 지난 세기의 50 ~ 60 년대로 거슬러 올라가며, 지금까지 해커는 오랜 시간 동안 네 가지 유형으로 발전해 왔습니다.

- 화이트 햇 해커 : 안전한 시스템을 수정하고 구축하기 위해 소프트웨어 버그를 찾는 전문 프로그래머입니다.

- 블랙 햇 해커 : 파괴적이고 자기를 찾는 사람들 인 화이트 햇 해커의 반대.

- 그린 햇 해커 : 유능한 프로그래머로서 Microsoft와 같은 회사에서 소프트웨어 버그를 찾기 위해 노력하고 있습니다.

- 회색 모자 해커 또는 갈색 모자 : 때때로 흰색 모자 해커의 작업을 수행하지만 여전히 검은 모자 해커의 작업을 수행하는 사람들입니다.

### **2.2.2: HTTP 헤더**

HTTP 헤더는 클라이언트와 서버가 서로에게 보내는 정보의 헤더입니다. 서버로 전송 된 클라이언트 정보를 HTTP 요청 (요청)이라고 하고 서버가 클라이언트에게 HTTP 응답 (응답)을 보냅니다. 일반적으로 HTTP 헤더는 매개 변수 이름과 값을 각각 포함하는 여러 줄로 구성됩니다. 일부 매개 변수는 요청 및 응답 헤더 모두에 사용할 수 있는 반면, 다른 매개 변수는 각 유형에만 사용할 수 있습니다.

### **2.2.3: SESSION**

HTTP는 상태 비 저장 객체 지향 프로토콜이며 서버와 클라이언트간에 작동 상태를 저장하지 않습니다. 서버가 클라이언트의 이전 상태를 모르기 때문에 일부 웹 응용 프로그램을 관리하기가 어렵습니다. 이 문제를 해결하기 위해 사람들은 SESSION (세션) 개념을 HTTP 프로토콜에 도입했습니다.

SessionID는 세션을 인증하기위한 문자열입니다. 일부 서버는 서버에서 웹 페이지를 볼 때 사용자에게 세션을 할당합니다.

* sessionID 세션을 유지하려면 일반적으로 다음에 저장됩니다.

+ URL의 변수

+에서 숨겨

+ 쿠키

- 세션은 허용 된 시간 내에 만 존재하며,이 시간은 서버 또는 실행 가능한 응용 프로그램에 의해 지정되며 서버는 자동으로 세션을 해제하여 시스템 리소스를 복원합니다.

### **2.2.4: 쿠키**

서버와 사용자 브라우저간에 공유되는 작은 구조화 된 데이터 조각입니다.

쿠키는 사이트를 방문한 사용자와 사이트를 통해 방문한 영역을 식별하기 위해 추적을 저장하기 위해 응용 프로그램에서 생성 한 작은 텍스트 파일로 저장됩니다. 이 정보에는 하드 디스크에 저장하기 위해 사용자 브라우저가 허용하는 사용자 정보, 계정, 비밀번호 등 쿠키가 포함될 수 있습니다. 많은 브라우저는 쿠키를 자동으로 저장하지 않지만 쿠키를 수락할지 여부는 사용자에게 달려 있습니다.

다음에 사이트를 방문하면 응용 프로그램은 사용자가 로그인하거나 추가 정보를 제공하지 않아도 쿠키의 정보 (관련 계정 정보)를 다시 사용할 수 있습니다.

 해당 웹 사이트를 이후에 방문 할 때 응용 프로그램은 사용자가 게시 작업을 다시 수행 할 필요없이 쿠키의 정보 (예 : Yahoo Messenger 로그인과 관련된 정보 등)를 재사용 할 수 있습니다. 다른 정보를 입력하거나 제공하십시오.

쿠키는 두 가지 범주로 분류: secure/nonsecure 및 persistent/non-persistent그래서 쿠키에는 4 가지 유형이 있습니다.

- Persistent và Secure

- Persistent và Non-Secure

- Non-Persistent và Secure

- Non-Persistent và Non-Secure

* 영구 쿠키는 지정된 기간 동안 클라이언트에 .txt 파일로 저장됩니다 (예 : NetscapeNavigator 브라우저는 쿠키를 cookie.txt 파일로 저장하고 Internet Explorer는 각 파일이 쿠키 인 \* .txt 파일로 저장 됨).
* 비 영구 쿠키는 클라이언트의 RAM에 저장되며 웹 사이트가 닫히거나 사이트에서 수신되면 취소됩니다.
* 보안 쿠키는 HTTPS (SSL)를 통해서만 보낼 수 있습니다.
* 비보안 쿠키는 HTTPS 또는 HTTP 프로토콜을 사용하여 보낼 수 있습니다. 실제로, 보안 쿠키의 경우 서버는 보안 전송 모드를 제공합니다.

**쿠키의 구성 요소는 다음과 같습니다.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Domain** | **Flag** | **Path** | **Secure** | **Expiration** | **Name** | **Value** |
| www.redhat. | FALSE | / | FALSE | 1154029490 | Apache | 64.3.40.151.16 |
| com |  |  |  |  |  | 018996349247 |
|  |  |  |  |  |  | 480 |

* 도메인 : 쿠키를 만든 웹 사이트의 도메인 (위 예에서)
* 플래그 : TRUE / FALSE-도메인 이름이 동일한 다른 시스템이 쿠키에 액세스 할 수 있는지 여부를 결정합니다.
* 경로 : 액세스 할 수있는 주소 범위 (예 : 경로가 "/ tracuu"인 경우 / tracuu 디렉토리의 주소와 / tracuu / baomat와 같은 모든 하위 폴더에 액세스 할 수 있습니다. 이 쿠키에. 값이 "/"이면 쿠키를 만든 사이트의 모든 주소가 쿠키에 액세스합니다.
* Sercure : TRUE / FALSE 값-보안 쿠키인지 여부를 결정하면 연결에서 SSL을 사용하는지 여부를 의미합니다.
* 만료 : 1970 년 1 월 1 일 00:00:00 GMT 이후 쿠키 만료 시간 (초) 이 값을 설정하지 않으면 브라우저는 쿠키를 비 영구 쿠키로 해석하고 RAM 메모리에만 저장하며 브라우저를 닫을 때 쿠키를 삭제합니다.
* 이름 : 변수 이름 (이 경우 Apache)
* 값 : 위에서 만든 쿠키에서 Apache의 값은 64.3.40.151.16018996349247480이고 만료 날짜는 2006 년 7 월 27 일이며 도메인 이름은 http://www.redhat.com입니다.
* 쿠키와 Cookies.txt 파일에 배치 된 Netscape (NS) 쿠키, 링크와 함께 Cookies.txt 파일에 배치 된 Netscape (NS) 쿠키

C : \ Program Files \ Netscape \ Users \ UserName \ Cookies.tx

* IE 쿠키는 여러 파일로 저장되며 각 파일은 쿠키입니다.

[C :] \ 문서 및 설정 \ [사용자 이름] \ 쿠키 (Win2000)에 배치

win9x에서 쿠키 폴더는 [C :] \ Windows \ cookies 폴더에 있습니다.

- 쿠키의 최대 크기는 4kb입니다. 도메인의 최대 쿠키 수는 20 개의 쿠키입니다. 브라우저 가까이에서 파괴 된 쿠키를 "세션 쿠키"라고합니다.

### **2.2.5: 프록시**

프록시는 사용자에게 dual\_homed 호스트 또는 basion 호스트의 특정 프로토콜 또는 일련의 실행 프로토콜에 대한 인터넷 액세스를 제공합니다. 프록시 서버가 사용자의 클라이언트 프로그램을 중재하여 사용자가 통신해야하는 실제 서버를 대체합니다.

프록시 서버는 클라이언트의 요청을 결정하고 응답 여부를 결정합니다. 요청이 충족되면 프록시 서버는 클라이언트 대신 실제 서버에 연결하여 클라이언트에서 서버로 요청을 계속 전달합니다. 클라이언트에 대한 서버의 응답뿐만 아니라 따라서 프록시 서버는 서버와 클라이언트 사이의 매개체와 같습니다.

# **제 3 장: 기본 웹 애플리케이션 공격 기술**

## **3.1: 웹 액세스 제어(Web Access Control)**

### **3.1.1: 뒷문을 통해 시스템에 들어감(Back door)**

응용 프로그램을 설계하는 동안 응용 프로그램 개발자는 "후문"(back door)을 설치하여 나중에 시스템에 쉽게 침투 할 수 있습니다.

## **3.2:세션 점유(Session Mangement )**

### **3.2.1: 세션 수정(Session Fixation )**

해커가 유효한 세션 ID를 사용자에게 보내서 유효한 사용자를 가장 할 수 있는 공격 기술입니다. 사용자가 시스템에 성공적으로 로그인하면 해커는 해당 세션 ID를 다시 사용하고 유효한 사용자가 됩니다.

### **3.2.2: 세션 도난 (Session Hijacking)**

해커가 피해자가 시스템에 로그인 한 후 쿠키 또는 URL 매개 변수 (폼의 숨겨진 변수)에 저장된 세션 ID를 해독하여 시스템에 로그인 한 후 해커가 유효한 사용자를 가장하도록하는 공격 기술입니다.

## **3.3: 유효한 데이터 입력 확인 시 오류 활용( Input Validation)**

### **3.3.1: 브라우저 측 언어로 데이터의 정확성 확인(Client-Side Validation)**

브라우저 측 언어 (JavaScript, VBScript ..)가 브라우저에서 실행되므로 해커는 소스 코드를 수정하여 테스트를 비활성화 할 수 있습니다.

### **3.3.2: 버퍼 오버플로(Buffer overflow)**

버퍼 오버플로(buffer overflow) 또는 버퍼 오버런(buffer overrun)은 메모리를 다루는 데에 오류가 발생하여 잘못된 동작을 하는 프로그램 취약점이다. 컴퓨터 보안과 프로그래밍에서는 프로세스가 데이터를 버퍼에 저장할 때 프로그래머가 지정한 곳 바깥에 저장하는 것을 의미한다. 벗어난 데이터는 인접 메모리를 덮어 쓰게 되며 이때 다른 데이터가 포함되어 있을 수도 있는데, 손상을 받을 수 있는 데이터는 프로그램 변수와 프로그램 흐름 제어 데이터도 포함된다. 이로 인해 잘못된 프로그램 거동이 나타날 수 있으며, 메모리 접근 오류, 잘못된 결과, 프로그램 종료, 또는 시스템 보안 누설이 발생할 수 있다.

### **3.3.3: URL 인코딩 ( URL Encoding)**

해커는 URL에서 특수 문자의 표준 인코딩을 이용하여 이 제어를 우회하기 위해 스크립트 문자로 체크인 된 문자를 불법 문자로 자동 인코딩합니다.

### **3.3.4: 메타 문자(Meta-characters)**

특수 문자를 사용하면 해커가 XSS 기술의 <script>와 같은 명령 문자열에서 문자를 전송하여 명령을 실행할 데이터를 삽입 할 수 있습니다.

### **3.3.5: 링크를 통과(Path Traversal)**

URL의 파일에 액세스하여 경로를 활용하여 해커가 시스템의 파일 내용을 얻을 수 있는 브라우저로 결과를 반환하는 방법입니다.

### **3.3.6: 대상 브라우저에 실행 코드 삽입(Cross- Site Scripting)**

이것은 주로 서버 시스템이 아닌 사용자 컴퓨터의 정보를 대상으로 하는 공격 기술입니다. 해커는 임의 코드를 추가하여 (일반적으로 JavaScript, VBScript 등의 스크립팅 언어로 프로그래밍 됨) 쿠키와 같은 중요한 정보를 도용하여 응용 프로그램의 유효한 사용자가 될 수 있습니다. 이 도난 된 정보에 근거하여 ...를 사용하십시오. 크로스 사이트 스크립팅은 일종의 "session hijacking"공격입니다.

### **3.3.7: 시스템의명령어 추가( OS Command Injection)**

양식의 매개 변수, 쿠키, HTTP 헤더 요청 및 시스템의 위험한 데이터와 같은 엄격한 확인없이 매개 변수에 추가 된 시스템 명령 또는 코드 스 니펫을 실행할 수있는 기능 파일이 서버에 업로드됩니다.

이 기술의 성공으로 해커는 동일한 서버 권한으로 시스템 명령을 실행할 수 있었습니다.

### **3.3.8: SQL 쿼리 삽입(SQL Injection)**

데이터베이스를 사용한 프로그래밍에서 프로그래머는 입력 값을 확인하는 데 실수를했습니다. 해커는 유효하지 않은 쿼리 나 값을 추가하여 시스템에 쉽게 로그인 할 수 있습니다.

### **3.3.9: 서버 측 언어(Server Side Includes)**

임베디드 파일 (include file), 데이터베이스 액세스 (jdbc)와 같은 시스템 명령을 추가 할 수 있습니다. 해커는 일반적으로 웹 사이트에서 볼 수 없는 파일 및 데이터베이스에 액세스 할 수 있습니다.

### **3.3.10: 공문자(Null Characters)**

C ++와 같은 번역 프로그램을 사용하는 응용 프로그램에서는 C ++가 \0이 문자열의 끝이라고 생각하기 때문에 해커가 종종 응용 프로그램을 속이기 위해 추가하는 \0으로 끝나는 문자열을 활용합니다. 예를 들면 다음과 같습니다.

해커는 다음 문자열을 추가합니다.

**입력 필드** : 첫 번째 주제 \0 <script> alert(document.cookie) </ script>

응용 프로그램이 C ++ 프로그램을 사용하여 문자열의 정확성을 검사하는 경우 C ++는 "\0"이 문자열의 끝을 알기 때문에 위의 유효한 문자열이므로 검사하지 않습니다.

### **3.3.11: 파라미터 전송으로 작동 (Parameter Manipulation)**

서버와 브라우저간에 교환되는 정보는 URL 변수, 양식 변수 및 쿠키와 같은 변수에 저장됩니다. 변수 제어에 충분한주의를 기울이지 않았기 때문에 해커는 변수 값을 수정하여 사용자 세션을 훔치거나 항목 값을 변경할 수 있습니다.

## **3.4: 티나는 정보(Informational)**

웹 페이지의 소스 코드와 같은 중요한 정보가 포함 된 시스템의 파일 및 응용 프로그램 또는 시스템의 사용자 암호가 포함 된 파일은 항상 해커의 대상입니다. 또한 소스 코드의 주석은 해커에게 유용한 정보 소스이기도 합니다.

해커는 시스템의 HTTP 응답을 사용하여 파일 또는 응용 프로그램이 있는지 확인합니다.

예제 3.4-1:

• HTTP 200 : 파일이 존재합니다

• HTTP 404 : 파일이 없습니다.

## **3.5: 서비스 거부( Denial Of Service(DoS))**

일정 기간 동안 많은 수의 요청이 애플리케이션에 전송되어 시스템이 요청에 응답하지 않아 시스템 장애가 발생합니다.

# **제 2 부:**

# **웹 애플리케이션 공격 및 보안기술**

# **제 4 장: 파라미터 전송으로 조작**

매개 변수 매개 변수 조작은 쿠키, URL 또는 form의 숨겨진 변수에 대한 중요한 정보를 변경하는 기술입니다. Cross-Site Scripting, SessionID, SQL Injection, BufferOverflow 기술은 이러한 매개 변수를 사용하여 해커 단계를 완료해야 합니다. 전송 매개 변수는 애플리케이션 공격 중 모든 해커 활동의 초점이라고 할 수 있습니다. 이것이 이 섹션의 첫 번째 장이며, 다음 섹션의 프레젠테이션을보다 잘 지원하기위한 것입니다.

## **4.1: URL 조작**

### **4.1.1:개념**

HTML 양식을 입력하면 GET 또는 POST의 두 가지 방법으로 결과가 전송됩니다. GET을 사용하면 모든 변수 이름과 해당 값이 URL 문자열에 나타납니다.

예제 5.1.1 : 회원의 비밀번호를 변경할 수 있는 응용 프로그램 웹 사이트가 있습니다.

<http://www.nganhang.com/example?user=thang&newpass=123>

+ username은 비밀번호를 변경해야하는 사람의 이름입니다.

+ newpass는 사용자 이름의 새로운 비밀번호입니다

그러나 다음과 같이 매개 변수를 변경:

<http://www.nganhang.com/example?user=admin&newpass=111111>

해커가 새 비밀번호 (예제 : '1111111')로 관리자 비밀번호를 변경할 수 있었습니다.

### **4.1.2: 고치는 방법**

URL 문자열의 내용이 변경되는 것을 방지하기 위해 응용 프로그램은 다음과 같은 조치를 취할 수 있습니다.

* 응용 프로그램은 해시 테이블 메커니즘을 사용합니다. 사용자가 사용자 이름으로 성공적으로 인증되면 응용 프로그램은 해당 키를 생성합니다. 이 키는 해시 객체의 username 변수와 함께 서버에 저장됩니다. 사용자가 응용 프로그램에 연결할 때마다이 키와 사용자 이름이 전송되어 해시 테이블의 키 및 사용자 이름과 비교됩니다. 데이터의 레코드에 해당하는 경우 유효합니다. 그렇지 않은 경우 서버는 사용자가 URL을 변경했음을 알고 있습니다.
* 또한 유용한 정보를 얻으려면 해커가 정보를 자유롭게 수정하지 못하도록 브라우저에 표시하기 전에이 정보를 인코딩해야합니다.

## **4.2:숨겨진 폼(form)변수 조작**

### **4.2.1:개념**

Hidden Form Field라는 form의 숨겨진 변수를 통해 정보를 변환 할 수 있습니다. 숨겨진 form 변수는 브라우저 화면에 표시되지 않지만 사용자는 "view source"에서 해당 컨텐츠를 찾을 수 있으므로 웹 페이지 컨텐츠를 브라우저에 저장하여 해커가 이용할 수있는 약점입니다. 페이지 내용을 변경하고 서버에 제출하십시오.

예제 4.2.1 : 원본 양식의 내용은 다음과 같습니다.

<form [action="http://www.tancong.com/cuahang.pl"](http://www.tancong.com/cuahang.pl) method="POST">

...

<input type="hidden" name="price" value="**99.99**">

...

</form>

변경 사항이 없는 경우 서버에 대한 요청에는 다음 컨텐츠가 있습니다.

POST /cuahang.pl HTTP/1.0

...

**price=99.99**

그러나 해커가 "price"필드에 다른 값을 할당하는 경우 :

<form [action="http://www.tancong.com/cuahang.pl"](http://www.tancong.com/cuahang.pl) method="POST">

...

<input type="hidden" name="price" value="**0.99**">

...

</form>

요청이 변경:

POST /cuahang.pl HTTP/1.0

...

**giaca=0.99**

### **4.2.2: 고치는 방법**

* 브라우저에 데이터를 표시 할 때는 폼의 숨겨진 변수 만 사용하고 응용 프로그램 처리에서 변수 값을 사용하여 조작하지 마십시오.
* HTTP\_REFERER 변수를 사용하여 수신 요청의 출처를 확인하지만 해커는 프록시를 사용하여 실제 출처를 숨길 수 있으므로 HTTP\_REFERER 변수에 너무 의존하여 확인하지 마십시오.
* 숨겨진 변수의 이름과 값을 단일 문자열로 결합합니다. MD5 암호화 알고리즘 또는 다른 단방향 해시 유형을 사용하여 해당 문자열을 합성하고 "Pattern String"이라는 숨겨진 필드에 저장하십시오.

폼의 값이 전송되면 이전에 정의한 것과 동일한 키를 사용하여 위의 작업이 다시 수행됩니다. 그런 다음 "모델 문자열"과 비교하십시오. 일치하지 않으면 양식의 값이 변경되었습니다.

* sessionID를 사용하여 데이터베이스에 저장된 정보를 참조하십시오.

## **4.3: 쿠키(COOKIE) 조작**

### **4.3.1:개념**

쿠키는 가장 안전한 정보 저장 구성 요소이므로 쿠키는 종종 양식 변수 및 URL 변수가 아닌 HTTP 상태를 유지하는 데 사용됩니다. 응용 프로그램 및 기타 세션 데이터를 사용할 때 사용자 정보를 저장하는 데에도 사용됩니다. 영구적 또는 비 영구적, 안전 또는 안전하지 않은 쿠키와 같은 모든 유형의 쿠키는 사용자가 수정하여 서버로 보낼 수 있습니다. 따라서 해커는 쿠키 콘텐츠를 변경하여 응용 프로그램을 우회 할 수 있습니다.

Winhex와 같은 무료 도구를 사용하면 비 영구 쿠키를 수정할 수 있습니다. SSL은 전송 중 쿠키 만 보호 할 수 있습니다.

예제 4.3.1 : 여행 정보 웹 앱에 대한 정보를 저장하는 데 사용되는 쿠키에 대한 정보 :

Cookie: lang=en-us; ADMIN=no; y=1 ; time=10:30GMT ;

이 사용자를 식별하는 쿠키는 관리자가 아니지만 (ADMIN = no) 해커가 이 필드를 변경하면 어떻게 됩니까? 해커는 다음으로 다시 변경할 수 있습니다.

Cookie: lang=en-us; ADMIN=yes; y=1 ; time=12:30GMT ;

해커는 이제 응용 프로그램의 관리자 역할을 합니다.

### **4.3.2: 고치는 방법**

* 세션 객체를 사용하여 서버에 중요한 정보를 저장하십시오. 애플리케이션이 사용자 정보를 확인해야하는 경우 애플리케이션은 사용자의 sessionID를 사용하여 캐시 또는 데이터베이스에서 해당 사용자의 정보를 참조합니다.
* 쿠키가 위조 된 것으로 알려진 잘못된 값을 찾기 위해 쿠키의 내용을 확인하는 메커니즘을 개발하십시오. 예를 들어, 쿠키에서 "admin"플래그 변수가 true로 설정되어 있지만 쿠키의 사용자 일련 번호 값이 "administrator"서수 값과 동일하지 않은 경우 서버.
* 마지막 방법은 쿠키를 암호화하는 것입니다. 대칭 (암호화 및 암호 해독에 고유 키 사용) 또는 비대칭 (암호화에는 별도의 키 2 개, 암호화를 위한 공유 키 및 디코딩을 위한 개인 키를 사용)과 같은 몇 가지 암호화 방법이 있습니다.

# **제 5 장: SQL 심문 삽입( SQL INJECTION)**

## **5.1. 개념**

SQL 주입은 데이터베이스 액세스에서 웹 프로그래밍 허점을 활용하는 방법입니다. 이는 SQL Server의 단일 결함 일뿐만 아니라 Oracle, MS Access 또는 IBM DB2와 같은 다른 모든 데이터베이스에 대한 일반적인 문제이기도 합니다.

해커가 폼을 통해 데이터를 보내면 웹 응용 프로그램은 데이터베이스와 관련된 쿼리 또는 오류 메시지의 결과를 실행하고 브라우저로 반환합니다. 또한이 정보 덕분에 해커는 데이터베이스의 내용을 알고 있으므로 전체 응용 프로그램 시스템을 제어 할 수 있습니다.

## **5.2: 데이터베이스 모델 소개**

프리젠테이션 기술의 내용을 보다 잘 표현하기 위해 이 논문은 User 테이블을 사용하여 공격 기술을 보여줍니다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | Variable name |  | datatypes | size | explain |
| 1 | Username | PRIMARY KEY | Text | 50 | Each user has 1 account to login. |
| 2 | Password |  | Text | 50 | Password to login |

**컨벤션 :**

**이 장에서 사용 된 프로그래밍 언어는 SQL Server 데이터베이스가 있는 ASP입니다.**

## **5.3: 공격 방법**

### **5.3.1: SQL Injection 공격 기술**

다음은 로그인 form을 무시하는 데 사용되는 가장 간단한 SQL Injection기술입니다.

예제 5.3.1-1 : 웹 응용 프로그램에 다음 코드가 있다고 가정합니다.

SQLQuery= “SELECT Username FROM User WHERE Username= ‘” & strUsername & “’ AND Password= ‘” & Password & “’”

flag= GetQueryResult (SQLQuery) if flag = “” then

check=FALSE else

check=TRUE

end if

위의 코드는 사용자 이름 및 비밀번호 입력 문자열을 확인합니다. User 테이블에 존재하면 check = true이고, 그렇지 않으면 check = false입니다.

한 값을 입력:

**Username:** ’ OR ‘’=’

**Password:** ’ OR ‘’=’

SQL 문은 다음과 같습니다.

SELECT Username FROM User WHERE Username= ‘’ OR ‘’=’‘ AND Password= ‘’ OR ‘’=’’

위의 진술은 항상 그렇습니다 ( '항상'과 같기 때문에 '). 따라서 WHERE 절의 조건문은 항상 참입니다. 표에서 첫 번째 행의 사용자 이름 값이 선택됩니다.

**SQL 특수 문자와 결합 :**

• 문자“; ": 쿼리의 끝을 표시

• 문자 "--": 같은 줄에서 문자열을 숨 깁니다.

예제 5.3.1-2 :

**Username**: ’; drop table User--

**Password**:

SQL 문은 다음과 같습니다.

SELECT Username FROM User WHERE Username= ‘’;drop table User-- AND Password= ‘” & Password & “’”

위의 문장으로 User 테이블이 완전히 삭제됩니다.

예제 5.3.1-3 :

**Username**: admin’--

**Password**:

SQL 문은 다음과 같습니다

SELECT Username FROM User WHERE Username= ‘admin’-- AND Password= ‘” & Password & “’”

위의 명령을 사용하면 암호없이 관리자 권한으로 시스템에 로그인 할 수 있습니다.

### **5.3.2: SELECT 문을 기반으로 공격**

**Direct Injection 개념:** 작은 따옴표 나 따옴표로 묶이지 않은 명령문에 추가 된 인수는 직접 삽입(direct injection)입니다.

StrSQL=“SELECT Username FROM User WHERE Username=”& tName

**Quote Injection 개념:** 따옴표 또는 따옴표 사이의 공백에 인수가 적용되는 경우는 따옴표 삽입(quote injection)의 경우입니다.

StrSQL=“SELECT Username FROM User WHERE Username=’”& tName & “’”

올바른 구문을 유지하면서 따옴표를 비활성화하고 명령문을 변경하려면 삽입 된 코드 문자열에 삽입 된 문자열 앞에 작은 따옴표가 있어야하고 명령문의 끝에 작은 따옴표가 와야 합니다.

다음과 같은 :

StrSQL=“SELECT Username FROM User WHERE Username=’’ and ‘’=’’”

위의 작업을 수행하면 오류 메시지는 "("와 관련이 있으며 삽입 된 문자열에는 ")"이 있어야합니다.

예제 5.3.2-1:

StrSQL=“SELECT Username FROM User WHERE (Username=’”& tName & “’”)

유효한 구문은 다음과 같습니다.

StrSQL=“SELECT Username FROM User WHERE (Username=’’) or ‘’=’’”

또한 % 문자는 정보 검색 사례에 자주 사용됩니다.

예제 5.3.2-2:

StrSQL=“SELECT Username FROM User WHERE Username like ‘% “ & tName & “’”

### **5.3.3: HAVING 문을 기반으로 한 공격**

GROUP BY 절과 함께 사용되는 HAVING은 테이블 및 필드 정보를 얻는 효과적인 방법입니다.

### **5.3.4: UNION 조합 문에 기반으로 한 공격**

SELECT 문은 데이터베이스에서 정보를 검색하는 데 사용됩니다. 일반적으로 SELECT 절에 삽입 할 수있는 위치는 WHERE 뒤에 있습니다. 테이블에서 여러 행의 정보를 리턴하려면 UNION SELECT를 삽입하여 WHERE 절의 조건을 변경하십시오.

예제 5.3.4-1:

StrSQL=“SELECT Username FROM User WHERE Username like ‘% “ & tName & “’UNION SELECT Password from User”

위의 명령은 User 테이블의 Username과 Password를 결합한 결과 집합을 반환합니다.

UNION 문을 삽입 한 후 리턴되는 구문 오류로 인해 각 필드의 유형을 알 수 있습니다.

다음은 HAVING, GROUP BY, UNION을 기반으로 데이터베이스 내용을 알 수없는 경우의 예입니다.

예제 5.3.4-2: 로그인에 필요한 쿼리를 반복:

SQLQuery= “SELECT Username,Password FROM User WHERE Username= ‘” & strUsername & “’ AND Password= ‘” & Password & “’”

먼저, 쿼리가 사용하는 테이블 이름과 필드 이름을 알려면 "HAVING"조건문을 사용하십시오.

예제 5.3.4-3: 입력 된 값 :

**Username:** ’having 1=1--

생성 된 오류 :

**[**Microsoft**]**[ODBC SQL Server Driver][SQL Server] Column’**User.Username**’is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.

이 오류로 인해 쿼리에 사용 된 테이블이 User이고 테이블에 Username이라는 필드가 있음을 알고 있습니다.

그런 다음 GROUP BY를 사용:

예제 5.3.4-3:

**Username**: ‘group by User.Username having 1=1--

생성 된 오류 :

**[**Microsoft**]**[ODBC SQL Server Driver][SQL Server] Column’**User.Password**’isinvalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.

따라서 Password는 User 테이블의 필드이며 쿼리에 사용됩니다.

쿼리에 참여하는 User 테이블의 모든 필드를 알 때까지 GROUP BY를 계속 사용하십시오.

GROUP BY 구문 오류가 더 이상 존재하지 않으면 테이블에있는 각 필드의 유형 검사 단계를 수행하십시오. 이제 UNION이 사용됩니다.

예제 2.3.4-4:

**Username:**’union select sum(Username) from User

sum 명령은 괄호 안의 인수에 대한 sum 명령입니다. 인수는 숫자 유형이어야합니다. 인수가 숫자 유형이 아닌 경우 다음 오류가 발생합니다.

따라서 위의 오류 메시지와 함께 Username은 "varchar"유형이어야합니다.

[Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation **cannot take a varchar data type** as an argument.

위의 방법을 사용하면 테이블의 각 필드 유형을 쉽게 결정할 수 있습니다. 위 정보를 수신 한 후 해커는 사용자 User에 정보를 쉽게 추가 할 수 있습니다.

예제 2.3.4-5:

**Username:’**; insert into User(Username,Password) values (‘admin’, ‘’)--

해커는 예제 2.3.2와 같은 컨텐츠를 인증 할 비밀번호가없는 네트워크 관리자가된다.

### **5.3.5: INSERT 명령 기반으로 한 공격**

INSERT 키워드는 데이터베이스에 정보를 넣는 데 사용됩니다. 일반적으로 INSERT 문은 사용자 등록 정보, 방명록(guestbook)과 같은 경우에 사용됩니다.

SELECT 문과 함께 사용되는 ";", "-"기술은 구문 오류를 방지하기 위해 올바른 수와 유형의 값을 입력해야합니다.

예제 5.3.5-1:

SQLString= “INSERT INTO User VALUES (‘” & strUsername & “’, ‘” & strName& “’, ‘” & strPassWord & “’,’”& strLimitSize & “’)”

### **5.3.6: STORED PROCEDURE을 기반으로 공격**

저장 프로시저(Stored Procedure)는 응용 프로그램의 복잡성을 줄이고 SQL Injection의 공격을 피하기 위해 웹 프로그래밍에 사용됩니다. 그러나 해커는 여전히 저장 프로시저를 활용하여 시스템을 공격 할 수 있습니다.

예제 5.3.6-1: 저장 프로 시저 sp\_login에는 username과 password라는 두 개의 매개 변수가 있습니다. 입력 한 경우 :

**Username:** nhimmap

**Password:** ‘;shutdown--

저장 프로 시저에 대한 호출은 다음과 같습니다.

exec sp\_login ‘nhimmap’,‘’;shutdown--’

shutdown 명령은 SQL Server를 즉시 중지합니다.

## **5.4: 예방하는 방법**

* 대부분의 브라우저에서 문자를 사용하기 전에 URL로 인코딩해야 합니다.
* SQL Injection 공격은 오류 메시지를 사용하므로 사용자에게 오류 메시지를 표시하지 않는 것이 가장 좋습니다. 응용 프로그램에서 오류가 발생할 때마다 알림 오류를 개발자가 디자인 한 페이지로 바꿉니다.
* 사용자 입력 값을 주의 깊게 확인하고‘; v..v ..
* " ',", /, \,;와 같은 메타 문자를 제거하고 다음에서 수신 한 문자열에서 NULL, CR, LF 등의 문자를 확장하십시오.
  + 사용자가 데이터를 제출했습니다.
  + URL의 매개 변수.
  + 쿠키 값.
* 숫자 값의 경우 SQL 쿼리를 실행하기 전에 이 값을 정수로 변환하거나 ISNUMERIC을 사용하여 정수인지 확인하십시오.
* 알고리즘을 사용하여 데이터를 암호화합니다.

**의견 :**

* 이 장을 통해 처리하기 전에 데이터를 확인해야합니다.
* 데이터의 정확성을 검사하는 것 외에도 데이터베이스 내부의 데이터를 인코딩해야 합니다. 오류가 웹 페이지를 내보내는 것을 방지하고 해커가 데이터베이스 정보를 수집 할 수 없도록 SQL 구문 오류의 내용을 보고하십시오.
* 네트워크 관리자의 직무 인 병렬

# **제 6 장: 세션 점유(Possessing The Session)**

## **6.1: SESSIONID의 개요**

세션은 브라우저와 서버 간의 작업 상태를 저장하는 데 사용됩니다. 세션 ID는 쿠키에 저장하거나 URL 또는 양식의 숨겨진 변수에 포함 할 수 있습니다.

각 유형의 호스팅에는 장단점이 있지만 쿠키를 사용하는 것이 여전히 가장 좋은 방법이며 가장 안전한 방법입니다.

일반적으로 사용자가 이름 / 암호와 같은 개인 정보를 기반으로 인증 된 후 세션 ID는 후속 요청에 대한 임시 정적 암호로 처리됩니다. 이것은 세션 ID를 해커에게 훌륭한 대상으로 만듭니다. 대부분의 경우 해커는 세션에 침입 할 수 있는 유효한 사용자 세션 ID를 얻습니다.

XSS는 또한 쿠키에 저장된 세션 ID를 캡처 할 수 있는 공격입니다. 이 공격을 "session hijacking"라고 합니다.

**한 세션에 대한 공격은 일반적으로 두 가지 주요 유형으로 수행됨:.**

* **세션 고정(Session Fixation)**
* **세션 하이재킹(Session Hijacking)**

## **6.2: 세션 고정(Session Fixation)**

세션 별 공격에서 해커는 시스템에 로그온하기 전에 피해자에게 세션 ID를 할당합니다. 그런 다음 해커는 이 세션 ID를 사용하여 피해자의 세션에 들어갑니다.

**공격 과정 요약 :**

• **1 단계 : 세션 ID를 설정하십시오**.

두 방향의 세션 관리 시스템 :

* 자유 방향 : 세션 ID를 수락합니다. 세션이 없으면 새 세션 ID를 만듭니다.
* 제한된 방향 : 이전에 등록 된 세션 ID 만 허용합니다.

프리 드라이브 시스템 해커는 세션 ID 만 설정하고 이 세션 ID를 기억 한 다음 재사용하면 됩니다. 제한된 방향으로 해커는 응용 프로그램에 세션 ID를 등록해야 합니다.

따라서 1 단계는 공격자가 서버에 요청을 보내 세션을 유지 관리하는 것입니다.

**• 2 단계 :이 ID를 대상 브라우저로 보냅니다.**

해커는 새로 만든 세션 ID를 사용자에게 보내고 세션 ID 교환은 응용 프로그램에 따라 달라지며 URL, 숨겨진 양식 변수 또는 쿠키를 통해 이루어질 수 있습니다.

* URL 매개 변수의 공격 세션 ID.
* 숨겨진 폼 변수로 세션 ID를 공격하십시오.
* 쿠키의 공격 세션 ID.

**• 3 단계 : 피해자의 세션에 침입합니다.**

피해자가 지정된 세션 ID를 통해 시스템에 로그인 한 후 응용 프로그램을 종료하지 않은 경우 해커는 이제 해당 세션 ID를 사용하여 피해자의 세션에 들어갑니다.

## **6.3: 세션 하이재킹(Session Hijacking)**

해커는 세션 별 공격과 달리 세션 중에 사용자의 세션 ID를 훔칩니다. 해커는 사용자의 세션 ID를 도용하기 위해 다음 사용할 수 있는방법:

* 세션을 예측
* 작업 세션 소진.
* 세션 스틸 코드 사용

**세션 하이재킹과 세션 고정의 차이점**

|  |  |  |
| --- | --- | --- |
|  | **Session hijacking** | **Session fixation** |
| 시간 | -피해자가 시스템에 로그인 한 후 피해자의 브라우저에서 공격했습니다. | 피해자가 시스템에 로그인하기 전에 피해자의 브라우저에서 공격했습니다. |
| 영향 | -한 번 액세스하십시오. | -해커는 피해자 세션의 각 공격에 대해 일회성, 임시 또는 장기 액세스 권한을 얻습니다. |
| 세션을 유지 | -세션 유지 관리가 필요하지 않습니다. | -피해자가 로그인 할 때까지 세션을 계속하도록 요청할 수 있습니다. |
| 공격 방향 | 1. 대상 컴퓨터에서 XSS 취약성을 악용합니다.  2. HTTP 헤더 참조에서 세션 ID를 캡처하여 다른 웹 서버로 전송하십시오.  3. 마이닝 네트워크 트래픽. | 1. 사용자에게 링크 또는 수정 된 양식을 통해 시스템에 로그인하도록 요청하십시오.  2. 피해자 도메인의 모든 서버에서 XSS 취약점을 악용합니다.  3. 대상 도메인의 모든 서버에서 <META> 태그의 취약점을 악용하십시오.  4. 대상 컴퓨터와 함께 도메인의 세션 ID를 만들 수있는 서버를 피해자의 DNS 서버에 추가하십시오.  5. 네트워크 트래픽 변경. |
| 목표 | -제출  -통신 링크 | -대상 도메인의 모든 서버  -DNS 서버  -제출  -통신 링크 |

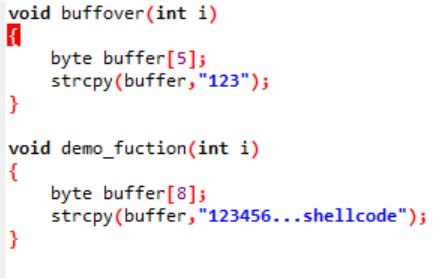
**의견 :**

이 공격 기법은 응용 프로그램 세션 세션을 관리 할 때 허점을 활용하며 웹 응용 프로그램에 액세스하는 부주의 한 사용자를 대상으로 합니다. 언급 된 장에서는 XSS 및 세션 관리 만이 사용자의 부주의를 이용합니다.

# **제 7 장: 버퍼 오버플로(Buffer Overflow)**

## **7.1:개념**

버퍼 오버 플로우는 버퍼 경계를 넘어서 데이터를 저장하는 프로세스의 길이가 고정 된 경우 비정상적인 상태입니다. 결과적으로 데이터가 인접한 메모리와 겹칠 수 있습니다. 덮어 쓴 데이터에는 다른 캐시, 변수 및 프로그램 흐름 제어 데이터가 포함될 수 있습니다.



**프로그램 및 응용 프로그램의 Buffer Overflow 오류 원인 :**

* 경계 검사 방법 (경계)이 완전히 구현되지 않았거나 생략되었습니다.
* C 언어와 같은 프로그래밍 언어 자체는 해커가 악용 할 수 있는 버그를 숨 깁니다.
* C 언어의 strcat (), strcpy (), sprintf (), bcopy (), gets (), canf () 메소드를 함수에 활용할 수 있습니다.
* 스택에 할당 된 버퍼가 버퍼에 복사 된 데이터보다 큰지 확인하지 않습니다.

## **7.2: 일반적인 버퍼 오버플로 오류 유형**

**스택에서 버퍼 오버플로 오류 악용 :**

* 스택의 버퍼 근처에 있는 로컬 변수를 재정 의하여 공격자가 쉽게 프로그램의 동작을 변경하십시오.
* 스택 프레임에서 리턴 주소를 대체하십시오. 함수가 반환되면 공격자가 지정한 주소, 일반적으로 사용자가 입력 한 데이터가 포함 된 버퍼에서 실행이 계속됩니다.

**힙에서 버퍼 오버 플로우 오류 악용 :**

* 힙 데이터 영역에서 발생하는 버퍼 오버 플로우를 힙 오버 플로우라고 하며 베어 스택 오류가 있는 다른 기술로 악용 될 수 있습니다.
* 힙 메모리는 런타임 응용 프로그램에 의해 할당되며 일반적으로 프로그램 데이터를 포함합니다.
* 마이닝은 응용 프로그램이 연결된 목록의 포인터와 같은 내부 데이터 구조를 덮어 쓰도록 특수한 방식으로 이 데이터를 분리하여 수행됩니다.

다른 악용 방법 :

* 프로그래밍 언어 (상업용 소프트웨어는 C 언어로 작성 됨)를 통해 소프트웨어 취약점을 악용합니다.
* 사용자 상호 작용은 있지만 사용자 이름, 비밀번호와 같은 입력 데이터를 바인딩하지 않는 웹 사이트를 악용하십시오.

## **7.3: 예방 조치**

버퍼 오버 플로우 오류를 악용 할 위험을 피하려면 다음 사용해야하는 예방 조치:

* 프로그래밍 언어 선택 : 프로그래밍 언어는 버퍼 오버플로 발생에 큰 영향을 미칩니다.
* 보안 메모리 관리 : 자동화 된 메모리 관리를 수행하는 추상 데이터 유형에 대해 잘 테스트되고 잘 작성된 라이브러리를 사용하십시오.
* 버퍼 오버 플로우를 방지하려면 입력 유효성 검사가 필수적입니다. 모든 필드, 등록 양식 또는 기타 입력 방법에는 잠재적 인 버퍼 오버 플로우가 있을 수 있습니다. 여기에는 양식 필드, HTTP 헤더, XML 인터페이스 또는 기타 입력 값이 포함되지 않습니다. 소프트웨어 응용 프로그램의 모든 영역에서 모든 입력 값을 생각하십시오.

# **제 8 장: 서비스 거부**

## **8.1:개념**

DoS 스타일 공격은 네트워크 서비스가 마비되어 요청에 응답 할 수 없는 공격입니다. 이러한 유형의 공격은 많은 시스템에 영향을 미치며 실행이 쉽고 DoS 공격으로부터 시스템을 보호하기가 매우 어렵습니다.

일반적으로 DoS 공격은 프로토콜을 기반으로 합니다. 예를 들어, ICMP 프로토콜을 사용하는 해커는 폭탄 전자 메일을 사용하여 메일 서버에서 시스템 리소스를 소모하기 위해 대역폭을 소비 할 목적으로 수천 개의 전자 메일 메시지를 보낼 수 있습니다. 또는 소프트웨어를 사용하여 서버에 일련의 요청을 보내 서버가 다른 합법적 인 요청에 응답하지 못하게 할 수 있습니다.

## **8.2: DOS의 공격 대상**

* SWAP SPACE 공격 : 대부분의 시스템에는 클라이언트의 요청을 처리하기 위해 수백 MB의 스왑 공간이 있습니다. 스왑 공간은 종종 단시간 하위 프로세스에 사용되므로 DoS는 스왑 공간을 채우는 방법을 기반으로 할 수 있습니다.
* BANDWIDTH 공격 : 각 시스템에서 사용할 수 있는 대역폭은 제한되어 있으므로 해커가 동시에 많은 요청을 시스템에 보내면 이러한 대량의 데이터에 대해 대역폭이 충분하지 않아 시스템이 손상 될 수 있습니다.
* RAM공격 : RAM의 많은 부분을 차지하는 Dos 공격은 시스템 파괴 문제를 일으킬 수 있습니다. BufferOverflow 공격이 이러한 파괴의 예입니다.
* DISKS 공격 : 일반적인 공격은 하드 디스크를 채우는 것입니다. 하드 디스크가 넘쳐 더 이상 사용할 수 없습니다.

## **8.3: 공격 기술**

### **8.3.1: 타입 1 공격**

해커 대역폭이 대상 시스템의 대역폭보다 크기 때문에 해커는 시스템을 완전히 넘칠 수 있습니다. 이러한 유형의 공격은 네트워크 속도에 의해 제한되지 않습니다.

예제8.3.1: 고속 T1 회선 (1,544-Mbps) 이상을 가진 해커는 56Kbps 회선으로 시스템을 쉽게 중단 할 수 있습니다.

### **8.3.2: 타입 2 공격**

이 유형의 공격은 해커의 네트워크 연결이 대상 시스템에 비해 너무 낮은 경우에 사용됩니다.

기존 DoS 공격과 달리 더 큰 대역폭 공격은 다른 시스템의 패킷을 활용하여 대상 시스템에 동시에 도달하여 대상 시스템의 전송을 더 이상 사용할 수 없게 합니다. 이에 대한 응답으로 서버는 더 이상 패킷을 수신 할 수 없습니다. 이 유형의 공격은 인터넷에서 대상 시스템을 제거합니다.

이는 서비스 거부 공격이지만 DDoS (분산 서비스 거부)라는 DoS는 아닙니다. 즉, 많은 컴퓨터가 동시에 시작되어 대상 컴퓨터로 패킷을 전송합니다. 대상 컴퓨터는 더 이상 패킷을 받을 수 없으며 인터넷에서 제거됩니다.

해커는 대상 컴퓨터의 IP 주소를 스푸핑하여 여러 패킷을 강력한 컴퓨터 시스템으로 온라인으로 보냅니다. 이러한 시스템은 이 가짜 SYN 패킷을 수신하고 연결을 수락하며 알림을 위해 SYN / ACK 패킷을 반환합니다. SYN 패킷의 IP 주소가 해커에 의해 대상 컴퓨터의 IP 주소로 수정되었으므로 SYN / ACK 패킷이 대상 컴퓨터로 전송됩니다. 동시에 여러 패킷을 수신하면 대상 시스템의 전송 회선이 응답 할 수 없었으며 대상 시스템이 패킷 수신을 거부했으며 이번에는 대상 컴퓨터 시스템이 다운되었습니다.

### **8.3.3: 시스템 리소스에 대한 공격 유형**

이러한 유형의 공격은 CPU, 메모리, 파일 시스템 및 프로세스와 같은 네트워크 리소스가 아닌 시스템 리소스를 대상으로 합니다. 해커는 시스템의 유효한 사용자이며 시스템에 제한된 양의 리소스가 있습니다. 그러나 해커는 이 액세스를 악용하여 추가 리소스를 요청합니다. 따라서 시스템 또는 유효한 사용자는 공유 리소스 사용이 거부됩니다.

이러한 유형의 공격은 리소스가 모두 소모되어 실행할 프로세스가 없기 때문에 시스템을 사용할 수 없게 만듭니다.

## **8.4: 예방 조치**

서비스 거부는 범인을 보호하고 조사하기 어려운 공격 유형입니다. 대부분의 해커는 컴퓨터의 IP 주소를 변경하여 범인을 식별하기가 어렵습니다.

라인 증폭 가능성을 방지하려면 다음이 필요하는것음:.

* 에지 라우터에서 브로드 캐스트 기능 취소
* 대기열 대기열 크기 늘리기
* 연결 설정 시간 단축
* Do 공격을 탐지하고 파괴하는 소프트웨어 사용

**의견 :**

서비스 거부로 인해 몇 분 안에 시스템이 중단되었지만 결과는 상당히 컸습니다 (돈과 명성의 범위에 영향을 미침). 해커가 시스템이나 정보에 대한 관리 권한을 얻지 못하거나 기관의 권한을 파괴하려는 경우 해커가 일반적으로 사용하는 기술이며 주소 사기로 인해 해커가 더 쉬워집니다. 탐지 될 염려없이 공격을 수행합니다.

# **제 9 장: 공격 프로세스 요약 및이를 방지하는 방법**

## **9.1: 해커 공격 프로세스 요약**

Stuart McClure의 Hacking Exposed, Joel Scambray 및 George Kurtz에 따르면 공격자는 일반적으로 공격 후 단계를 수행합니다.

### **9.1.1: 대상 인프라에서 정보를 수집**

* **1 단계 : FootPrinting (정보 수집) :**

IP 주소, 후이즈, DNS에 대한 세부 정보를 포함하여 서버 / 기업 / 사용자에 대한 최대 정보를 얻으려는 경우 해커가 수행하는 방식입니다.

지원 도구 : UseNet , search engines (công cụ tìm kiếm) , Edgar Any Unix client, http://www.networksolutions.com/whois, nslookup Is -d , Sam spade, http://www.arin.net/whois, dig.

* **2 단계 : Scanning(프로브 스캔)**

서버의 가장 중요한 정보는 이 단계에서 옵니다. 포트 스캐닝, 운영 체제 식별 ...을 포함하여 서버의 포트를 알고 데이터 라인을 수신합니다.

지원 도구 : fping, icmpenum Ws\_ping ProPack, nmap, SuperScan, fscan nmap, queso, siphon.

* **3 단계 : Enumeration**(**취약점 찾기)**

3 단계는 액세스 권한을 얻는 데 사용할 수있는 잘 보호되지 않은 리소스, 사용자 계정을 찾는 것입니다.

보조 도구 : null sessions, DumpACL, sid2user, OnSite Admin showmount, NAT Legion banner grabbing với telnet, netcat, rpcinfo.

* **4 단계 : Gaining access (액세스 방법 찾기)**

해커는 이제 위의 세 단계에서 얻은 정보를 사용하여 네트워크에 액세스하려고 시도합니다. 여기에 사용 된 방법은 버퍼 오버 플로우 오류를 공격하거나 비밀번호 파일을 검색 및 복호화하거나 무차별 비밀번호를 사용하는 것입니다.

지원 도구 : tcpdump, L0phtcrack readsmb, NAT, legion, tftp, pwdump2 (NT) ttdb, bind, IIS, HTR/ISM.DLL.

* **5 단계 : Escalating privilege(특권 사다리 오르기)**

해커가 계정으로 네트워크에 액세스 할 경우 전체 시스템을 제어하려고 시도합니다. 해커는 관리자 암호를 해독하거나 취약점을 사용하여 권한을 에스컬레이션하려고 합니다. John과 Riper는 널리 사용되는 두 가지 암호 크랙 프로그램입니다.

지원 도구 : L0phtcrack, Ic\_messages, getadmin, sechole.

* **6 단계 : Pilfering (패스가 포함 된 파일이 허점 인 경우 사용)**

다시 한 번 검색 엔진을 사용하여 네트워크에 액세스하는 방법을 찾습니다. 암호 나 기타 안전하지 않은 메커니즘이 포함 된 텍스트 파일은 해커의 대상 일 수 있습니다.

지원 도구 : rhost, LSA Secrets user data, configuration files, Registry.

* **7 단계 : Covering Tracks(추적 삭제)**

필요한 정보를 얻은 후 해커는 추적을 삭제하고 운영 체제의 로그 파일을 삭제하여 관리자가 침입자가 누구인지 감지하거나 알 수 없도록 합니다.

지원 도구 : Zap, Event log GUI, rootkits, file streaming.

* **8 단계 : Creating Backdoors (다음 침입을 쉽게 준비 할 수 있도록 백도어 생성)**

해커는 "백도어"를 남겨 두는데, 이는 트로이 목마를 설치하거나 새로운 사용자를 만들어 해커가 많은 노력없이 비밀리에 다시 액세스 할 수 있게 하는 메커니즘입니다.

지원 도구: members of wheel, administrators cron, At rc, Startup folder, registry keys, netcat, remote.exe, VNC, BO2K, keystroke loggers, add acct to secadmin mail aliases login, fpnwclnt.dll.

### **9.1.2: 웹 애플리케이션 설문 조사**

설문 조사 방법은 소스 코드를 보고 소스 코드를 볼 수 있는 오류를 활용하는 매우 인기가 있습니다.

오늘날 일부 일반적인 웹 언어에는 ASP (Active Server Pages), CGI (Common Gateway Interface), CFM (ColdFusion Server), HPP (Hypertext Preprocessor)와 같은 많은 오류가 있습니다.

www.google.com을 사용하여 관련 키워드를 검색하여 문제가있는 사이트를 찾으십시오. **allinurl :**을 사용하여 특정 문자열을 검색하기 전에 검색된 웹 페이지에 원하는 문자열이 있어야합니다.

예제9.1.2-1: "allinurl : / advadmin"은 다음 형식의 URL이있는 페이지 만 나열:

http://tentrangweb.com/advadmin.

http://www.google.com에서 파일을 검색 한 다음 웹 사이트에서 검색 할 파일 이름 앞에 단어 **type file:**을 입력하십시오.

예제 9.1.2-2 **:**

+ mdb 파일 (웹 페이지의 비밀번호가 포함 된 파일 인 경우 Access를 사용하여 열려면)을 찾으려면 http://www.google.com으로 이동하여 **type file: mdb**를 입력하십시오.

+ SAM 파일 (Windows NT 암호가 포함 된 파일, L0phtCrack을 사용하여 크랙) 을 찾으려면 http://www.google.com으로 이동하여 **type file: SAM**를 입력하십시오.

### **9.1.3:공격**

해커는 개체를 주의 깊게 수집하고 조사한 후 정보를 얻기 위해 공격을 시작하고 정보를 잘못 입력하거나 제어권을 얻습니다. 침입에 성공하지 못한 경우 그렇다면 Dos는 해커가 시스템을 작동 불가능하게 만드는 마지막 방법입니다.

## **9.2: 예방 조치 요약**

해킹 방지는 웹 프로그래머의 임무가 아니며, 관리자와 사용자의 조합, 지원이 필요합니다. 이러한 요소 중 하나를 누락하면 정보를 도용 할 수 있으며 해커도 네트워크를 제어 할 수 있습니다. 따라서 해커의 공격으로부터 시스템을 보호하기 위해 이 논문은 네트워크 관리자 역할, 응용 프로그램 프로그래머 역할 및 사용자 역할의 세 가지 역할을 제시합니다.

### **9.2.1: 네트워크 관리자와 함께**

* 시스템 관리자는 시스템에서 해당 오브젝트의 우선 순위 레벨을 보호하고 결정하기 위해 가장 중요한 오브젝트를 결정해야합니다.
* 응용 프로그램 구성 : 서버 및 일부 응용 프로그램을 구성 할 때 주의하십시오.
* 시스템에 대한 위협을 식별하는 것은 해당 시스템에서 서비스 및 응용 프로그램의 보안 허점을 식별하는 것입니다. 오류를 감지 할 때 해커가 이전 버전에서 아직 복구되지 않은 응용 프로그램에서 발견 된 취약점을 이용하지 못하도록 최신 소프트웨어로 업데이트해야 합니다.
* 사용 된 소프트웨어의 작동, 중요한 구성 파일 (예 : 기타 / 암호)의 중요성을 이해하고 해싱 코드 (MD5) 사용과 같은 구성 보호 조치를 적용하십시오.
* 몇 가지 도구를 사용하면 로그 파일과 같은 시스템에 대한 유효하지 않은 액세스 활동을 감지 할 수 있습니다.
* 시스템 계정의 권리를 엄격히 통제합니다. 불필요한 경우 루트 권한을 사용하지 마십시오. 시스템에서 사용하지 않는 계정의 경우 암호를 변경하거나 취소해야 합니다.
* 엄격한 비밀번호 관리 :
  + 사용자가 특정 시간 내에 비밀번호를 변경하도록 합니다.
  + 사용자가 비밀번호를 잊어 버린 경우 새 비밀번호를 다시 발행하려면 사용자를 인증하기위한 다른 절차가 필요합니다.

### **9.2.2: 웹 애플리케이션 디자이너와 함께**

* 사용자가 제공 한 데이터가 유효한지 확인: 응용 프로그램에 입력 된 모든 데이터는 <>‘/ ...와 같은 특수 문자를 주의 깊게 확인, 제거 또는 거부해야 합니다.
* 사용자 인증 : 오늘날 많은 응용 프로그램은 sessionID로 사용자의 세션을 관리하지만 세션 관리의 약점으로 인해 해커가 " 세션 관리”. 따라서 세션의 경우 브라우저가 연결을 닫는 즉시 응용 프로그램을 취소해야 합니다.
* 중요 데이터 암호화 : 사용자 이름 / 암호, 신용 카드 등의 중요 정보는 ... 해커가 XSS, SQL 주입과 같이 콘텐츠를 입수하여 사용하지 못하도록 암호화해야 합니다. 또한 전송 중에 회선의 정보가 유실되지 않도록 SSL 방법을 통합하십시오.
* 사용 가능한 소프트웨어 사용 : 현재 프록시 역할을 하는 Appshield와 같은 소프트웨어가 있습니다. 요청에 시스템에 대한 잠재적 인 공격이 포함되어 있음을 발견하면 요청을 거부하고 더 이상 서버로 보내지 않고 자동으로 요청을 취소합니다.
* 권한 설정 : 응용 프로그램의 경우 시스템이 응용 프로그램이 기능을 수행하기에 충분하도록 특정 권한 만 제공해야합니다. 해커가 이 루트를 이용하여 SQL 주입, 버퍼 오버 플로우와 같이 시스템 명령을 실행할 수 있으므로 루트와 같은 최고 권한을 부여하지 마십시오.

### **9.2.3: 웹 애플리케이션 사용자와 함께**

* 언어를 활용할 수 있는 기능이 엄청 나기 때문에 특히 브라우저 기반 클라이언트 언어를 사용할 때 가능한 위험에 대해 웹 사용자에게 경고하십시오. XSS 기술 에서처럼 sessionID.
* 응용 프로그램을 사용한 후에는 쿠키에 저장된 중요한 내용이 삭제되도록 시스템에서 로그 아웃해야 합니다. 해커가 기존 세션 ID를 계속 사용하여 유효한 시스템에 로그인 할 가능성을 피하십시오.
* 계정 관리 : 사용자는 자신의 계정을 보호하는 데 중요한 역할을 알고 있어야합니다. 계정 관리 활동에는 비밀번호 보호, 주기적 비밀번호 변경이 포함됩니다.
* 무단 계정 탐지 : 사용자는 다른 사람이 계정을 이용하여 다른 작업을 수행하지 않도록 활동을 정기적으로 확인해야합니다.

# **제 10 장:**

# **공격 및 정보 보안에 일반적인 소프트웨어 소개**

## **10.1: NMAP**

### **10.1.1: 개요**

Nmap(network mapper)은 원래 고든 라이온(Gordon Lyon)이 작성한 보안 스캐너이다. 이것은 컴퓨터와 서비스를 찾을 때 쓰이며, 네트워크 "지도"를 함께 만든다. 다른 간단한 포트 스캐너들처럼, Nmap은 서비스 탐지 프로토콜로 자신을 광고하지 않는 수동적인 서비스들도 찾아낼 수 있다. 게다가 Nmap은 원격 컴퓨터들의 자세한 정보를 알아낼 수 있다. 이 상세 정보에는 운영 체제, 장치 종류, 운영 시간, 서비스에 쓰이는 소프트웨어 제품, 그 제품의 정확한 버전, 방화벽 기술의 존재와 심지어 근거리 네트워크에서 네트워크 카드의 공급자도 포함한다.

Nmap을 사용해 단일 호스트, 또는 수십만 개의 디바이스와 다수의 서브넷을 포괄하는 광활한 네트워크를 모 니터링 할 수도있다.

Nmap은 리눅스, 윈도, 솔라리스와 맥 OS X를 포함한 BSD, AmigaOS 에서 실행된다. 리눅스가 가장 자주 사용하는 nmap 플랫폼이며, 윈도가 그 두 번째이다.

### **10.1.2:가능**

* 호스트 탐지 - 네트워크상의 컴퓨터들을 확인한다. 예를 들어 ping 응답이나 특정 포트가 열린 컴퓨터들을 나열한다.
* 포트 스캔 - 하나 혹은 그 이상의 대상 컴퓨터들에 열린 포트들을 나열한다.
* 버전 탐지 - 응용 프로그램의 이름과 버전 번호를 확인하기 위해 원격 컴퓨터의 네트워크 서비스에 주의를 기울인다.
* 운영 체제 탐지 - 원격으로 운영 체제와 네트워크 장치의 하드웨어 특성을 확인한다.

### **10.1.3:활용법**

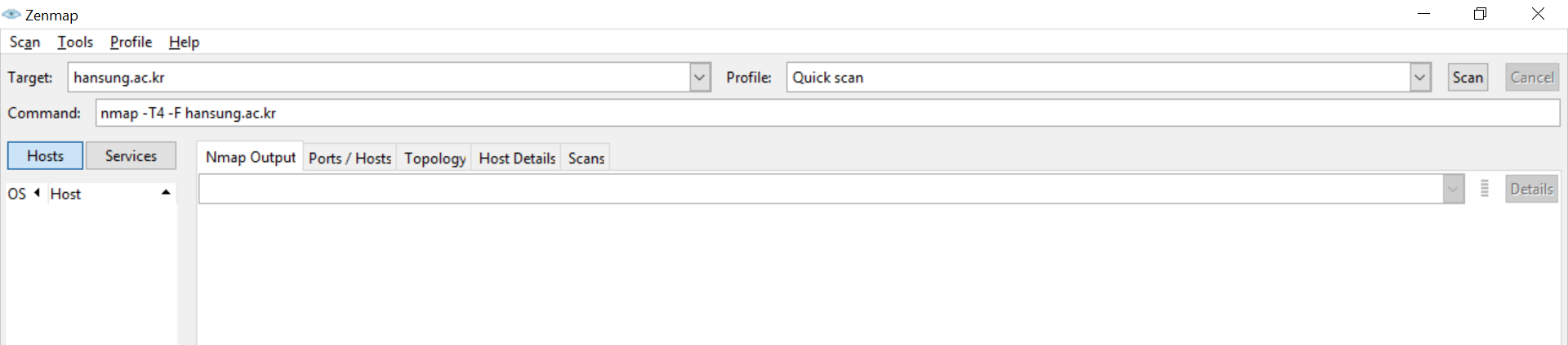


그림 6: nmap의 인터페이스

사용자 친화적 인 인터페이스이므로 nmap을 매우 쉽게 사용할 수 있습니다.

아래 예제는 웹 애플리케이션을 보여줍니다. 여기는 호스트 hansung.ac.kr입니다.

**채우기를 수행:**

* **Target: hansung.ac.kr**
* **Profile: Quick scan**

결과를 얻습니다.

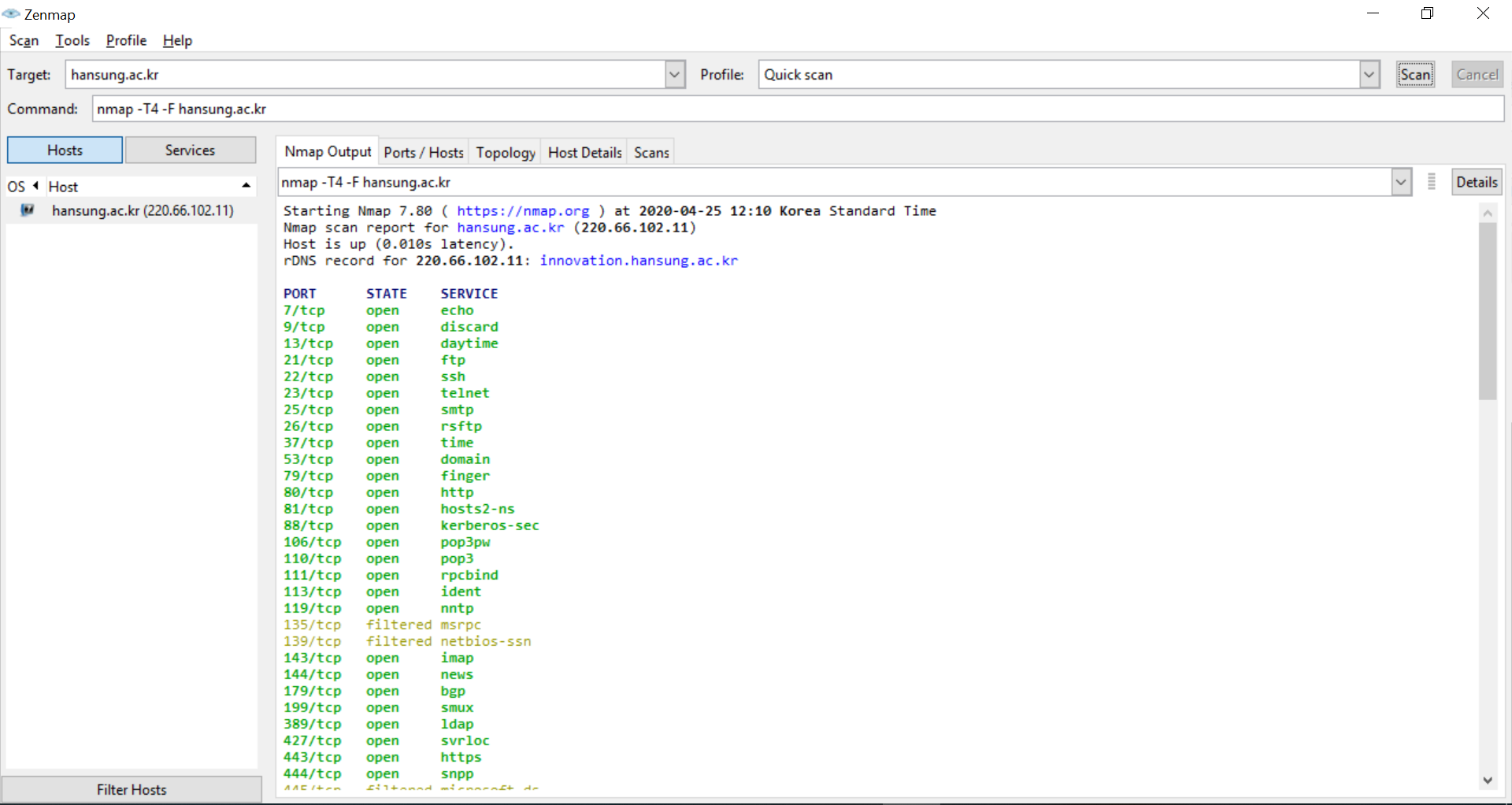


그림 7: nmap의 빠른 스캔 유형 설명

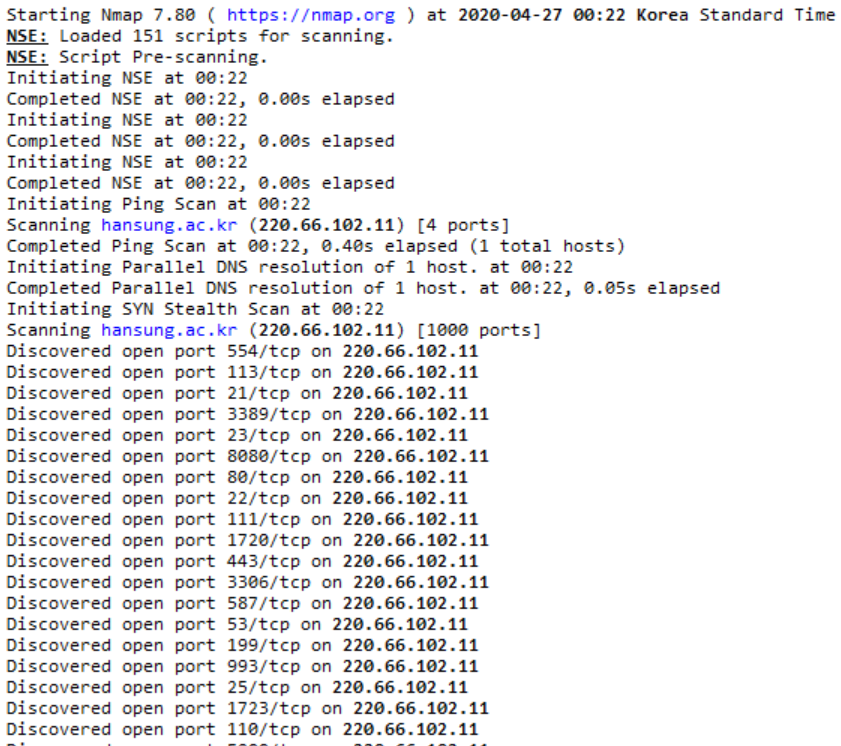


그림 8: nmap 부분 사용시 결과 (nmap output)

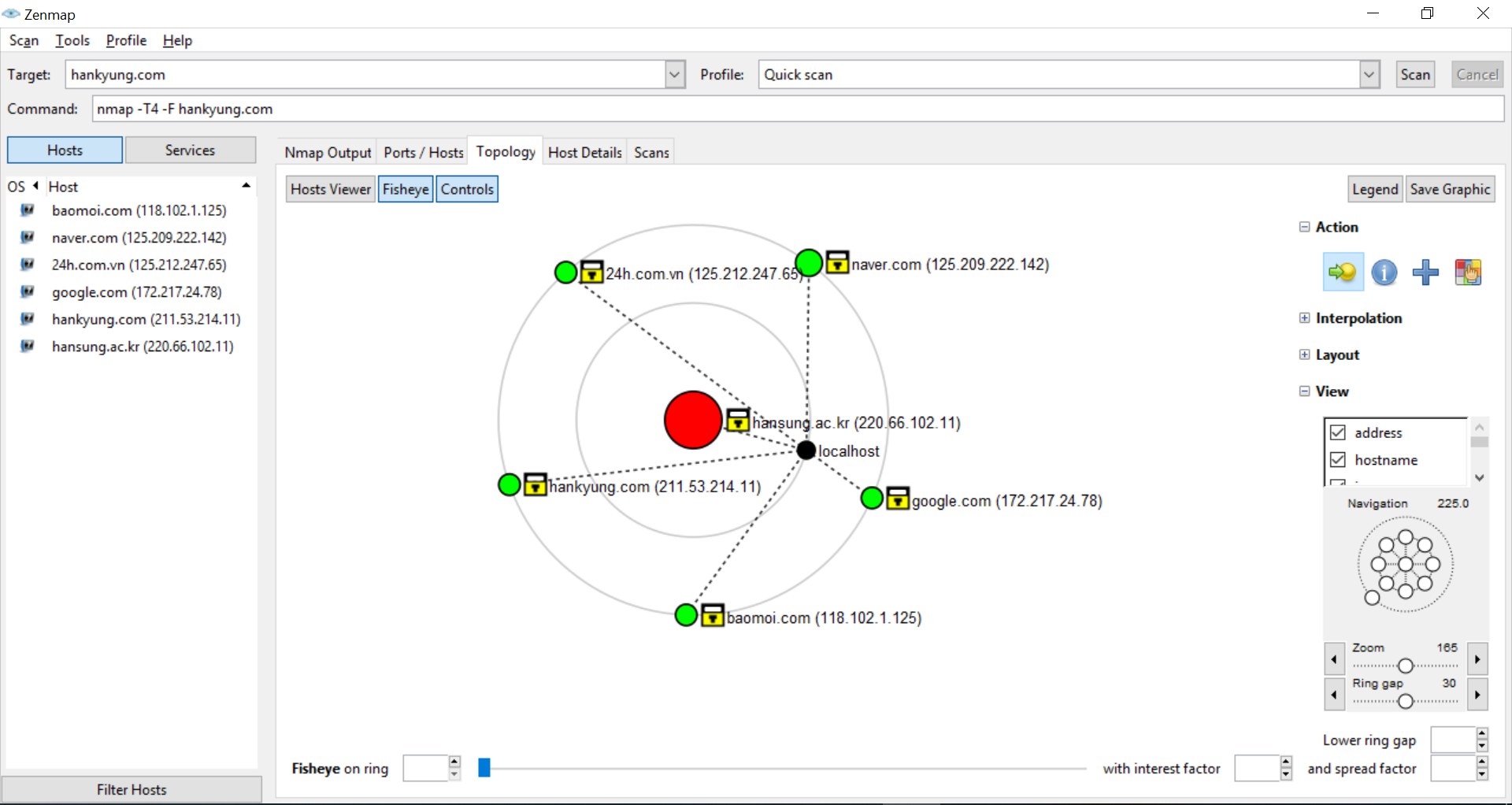


그림 9: 검사중인 호스트의 이미지

많은 기능이 있지만 nmap이 객체의 정확한 매개 변수를 결정하지 않는 경우가 있습니다. 위의 경우와 같이 Nmap은 호스트 소프트웨어에 대한 정보를 식별하지 않습니다.

**Nmap의 장단점:**

|  |  |
| --- | --- |
| 장점 | 단점 |
| -커뮤니티가 지속적으로 개발  -기업이 보안의 차이를 찾을 수 있도록 지원 | -해커가 악용 할 수 있음  -빠른 DNS 조회 |

## **10.2: SQLMAP**

SQL 인젝션 취약점은 오랫동안 사용되어 왔으며 우리에게 매우 친숙합니다. 그러나이 오류는 오래되었지만 여전히 모든 웹 사이트에 위험합니다. 그 이유는 웹 사이트의 데이터 처리 코드를 작성하는 부주의 한 프로세스와 비교하기 때문입니다.

이 오류는 침입자가 서버가 로그인하지 않고 데이터를 반환하도록 요청할 수 있습니다. 관리자 계정 정보를 해커에게 보내달라고하면 어떻게 생각하십니까? 웹 사이트 전체가이 해커의 게임과 함께 가고 있습니까?

여기서는 SQLMap을 사용하여 SQL 주입 취약점을 찾습니다.

### **10.2.1: 개요**

SQLMap은 SQL 관련 취약점을 빠르게 악용 할 수있는 도구입니다. 이 도구는이 SQL 악용 배열에서 가장 강력한 무기로 간주됩니다.

SQL 침투 테스트에 적합한 많은 기능과 데이터베이스에서 마커를 가져와 데이터베이스에서 데이터를 가져 오는 다양한 자동 분기 도구가 있습니다. 기본 파일 시스템에 대한 액세스를 지원하고 대역 외 연결을 통해 운영 체제에서 명령을 실행하십시오.

즉, 일단 액세스하면 인터넷을 통해 데이터를 조작 할 수 있습니다.

Kali Linux 또는 Back Track 5 운영 체제에는 SQLMap이 내장되어 있습니다. 그러나 Windows의 경우이를 사용하려면 Python 및 SQLMAP을 설치해야 합니다. SQLMap은 Python으로 작성되었으며 컴퓨터에서 Python 라이브러리를 실행해야하기 때문입니다.

### **10.2.2:가능**

* 데이터베이스 관리 시스템 MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, HSQLDB 및 Informix를 완벽하게 지원합니다.
* 부울 기반 블라인드, 시간 기반 블라인드, 오류 기반, UNION 쿼리 기반, 누적 쿼리 및 대역 외의 6 가지 SQL 주입 기술을 완벽하게 지원합니다.
* SQL 인젝션을 통해 전송하지 않고 데이터베이스에 직접 연결할 수 있습니다. 연결할 DBMS 로그인 정보, IP 주소, 포트 및 데이터베이스 이름 만 입력하면 됩니다.
* 데이터베이스에서 사용자, 비밀번호, 권한, 역할, 데이터베이스, 테이블 및 열을 나열하도록 지원합니다.
* 사용자 비밀번호의 해시를 해독하는 데 도움이 됩니다. 즉, 원하는 모든 사용자의 암호를 해독하고 암호를 얻을 수 있습니다.
* 원하는 ㅎ대로 데이터베이스를 백업 (백업) 할 수 있습니다.
* 입력에 성공하면 데이터베이스를 완전히 제어 할 수 있습니다.
* 데이터베이스에 파일을 매우 빠르게 다운로드하여 업로드 할 수 있습니다.

### **10.2.3:활용법**



그림 10: sqlmap

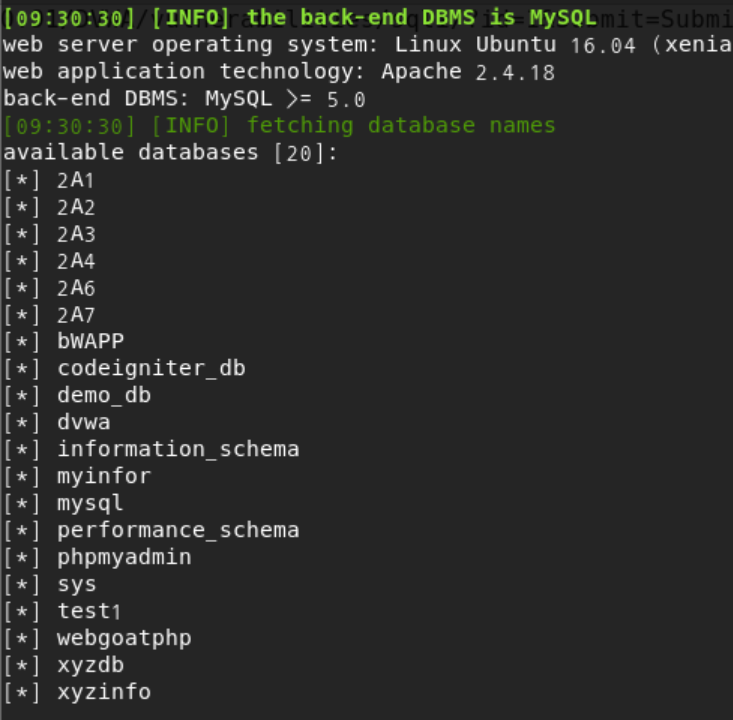
먼저 sqlmap는 시작하기위한 부분이고

-u는 공격타겠부분입니다.

--cookie는 쿠키값을 그래로 사항하기위해 프록시에서 얻어온 정보를 그대로 입력합니다.

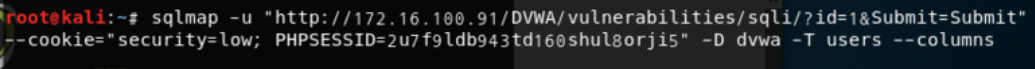
--dbs는 데이터베이스의 정보를 보여달라는 명령어입니다.





위의 명령어를 통해 얻어온 데이터베이스 이름듭니다.

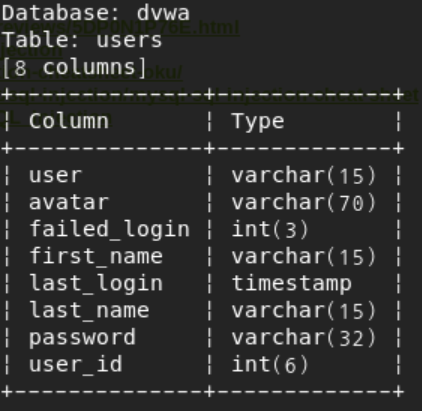
이제 특정 데이터베이스의 테이블을 알아봅니다.



-D: 데이터베이스를 지정합니다.

-T: 테이블을 지정합니다.

--columns: 그 안의 컬럼 정보들을 얻어답니다.

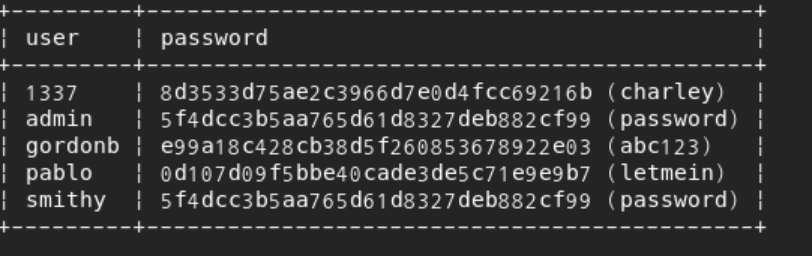


이베 컬럼도 알았으니 컬럼의 값 들도 알아봐야 할 것입니다.



-C: 컬럼을 지정합니다

--dump: 안의값들을 얻어답니다.



그렇게 얻은 정보입니다.

# **제 3 부: 결론**

## **I: 달성 된 문제**

웹 응용 프로그램은 빠른 속도로 개발되므로 공격 기술과 웹 응용 프로그램 보안도 높은 수준으로 개발됩니다. 웹 응용 프로그램 해킹 및 보안은 많은 지식과 전문 지식이 필요한 매우 광범위한 주제입니다. 논문과 기술의 짧은 시간으로 인해,이 논문에서 제시하는 부분은 기본 지식이며 여전히 제한적입니다. 그러나 컴퓨터 과학 교사들의 헌신적 인 도움으로 이 논문을 완성했습니다. 이 논문을 직접 작성했기 때문에 오랫동안 익숙하지 않은 웹 응용 프로그램과 관련된 전문 개념을 더 잘 이해했습니다.

초기 요청 "공격 기술 및 웹 응용 프로그램 보안에 대한 연구"에 따르면, 논문은 다음과 같은 내용을 달성했습니다.

* 웹 응용 프로그램의 기본 사항을 배웁니다.
* 기술을 포함한 웹 애플리케이션 공격 기술을 배웁니다.
  + URL, 숨겨진 양식 변수, 쿠키 및 HTTP 헤더와 같은 매개 변수 매개 변수 조작
  + SQL 쿼리 삽입
  + 하이재킹 사용자 세션
  + 오버 플로우 버퍼
  + 서비스 거부
* 공격 기법을 통해 네트워크 관리자, 웹 응용 프로그램 디자이너 및 사용자가 보안 조치를 제공합니다.

## **II: 논문의 개발 방향**

대학 논문에서 기본 논문은 요구 사항을 충족했습니다. 그러나 제한된 문서와 시간으로 인해 결과는 매우 완만합니다. 향후 가능하다면 논문은 다음과 같은 추가 내용을 개발하려고 노력할 것입니다.

* 보다 수준 높은 웹 응용 프로그램 보안을 제공하는 공격 기술에 대해 자세히 알아보십시오.
* 웹 응용 프로그램 수준뿐만 아니라 네트워크 및 서비스의 보안을 개발하기 위해 보안에 대해 더 깊이 배웁니다.
* 공격에 대비 한 효과적인 웹 응용 프로그램 보안 도구 또는 방법을 개발하십시오.

# **참고 문헌**

**[1]Hacking Exposed, Stuart McClure, Joel Scambray, George Kurtz**

[2] [**https://www.researchgate.net/figure/Three-tier-architecture-BULLET-Presentation-layer-MindteckTMS-BULLET-BusinessLogic\_fig6\_318905882**](https://www.researchgate.net/figure/Three-tier-architecture-BULLET-Presentation-layer-MindteckTMS-BULLET-BusinessLogic_fig6_318905882)

**[3]RFC2617, J.Franks, P. Hallam-Baker, J.Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart – 06/1999**

###### **[4]**[**http://www.sqlsecurity.com/**](http://www.sqlsecurity.com/)

###### **[5]**[**http://www.nextgenss.com/papers/**](http://www.nextgenss.com/papers/)

###### **[6]**[**http://www.owasp.org/**](http://www.owasp.org/)

###### **[7]**[**http://www.4guysfromrolla.com/webtech/**](http://www.4guysfromrolla.com/webtech/)

###### **[8]**[**http://www.guardent.com/**](http://www.guardent.com/)

###### **[9]**[**http://www.idefense.com/**](http://www.idefense.com/)

###### **[10]**[**http://www.jmu.edu/computing/info-security/engineering/issues/**](http://www.jmu.edu/computing/info-security/engineering/issues/)

###### **[11]**[**http://www.microsoft.com/technet/support/**](http://www.microsoft.com/technet/support/)

###### **[12]**[**http://www.microsoft.com/technet/security/**](http://www.microsoft.com/technet/security/)

###### **[13]**[**http://community.whitehatsec.com/**](http://community.whitehatsec.com/)

###### **[14]**[**http://www.codeproject.com/**](http://www.codeproject.com/)

**[15]https://ko.wikipedia.org/**

**[16]**[**https://namu.wiki/w/nmap**](https://namu.wiki/w/nmap)**/**

**[17]**[**https://www.stationx.net/nmap-cheat-sheet/**](https://www.stationx.net/nmap-cheat-sheet/)

**[18]**[**http://sqlmap.org/**](http://sqlmap.org/)

**[19]** ETHICAL HACKING 1-5 by EC-Council

[20] [http://vietbao.vn/Vi-tinh-Vien-thong/Tim-hieu-ve-cac-ung-dung Web/65082882/229/](http://vietbao.vn/Vi-tinh-Vien-thong/Tim-hieu-ve-cac-ung-dung%20Web/65082882/229/)

[21]https://vovanhai.wordpress.com/web/jsp/mo-hinh-mvc-va-1-vi-dụ-ứng-dụng /