**웹 취약점 분석 및 보고**

Web Vulnerability Analysis and Reporting

**보고서 작성자 : 사이버 보안 학과 김유진(10170546)**

2019. 11. 24



내용

[요 약 문 3](#_Toc26516181)

[ **제 목** 3](#_Toc26516182)

[분석 환경 4](#_Toc26516183)

[취약점 분석 툴 4](#_Toc26516184)

[웹 취약점 진단 결과 4](#_Toc26516185)

[취약점 목록 5](#_Toc26516186)

[**OWASP 10(2017 Version)** 5](#_Toc26516187)

[취약점 공격 유형 6](#_Toc26516188)

[상세 분석 결과 9](#_Toc26516189)

[결론 16](#_Toc26516190)

[참고 자료 17](#_Toc26516191)

# 요 약 문

* **제 목**

웹 취약점 분석 및 보고

* **과제의 목적 및 중요성**

E – 나라지표에서 한국인터넷 진흥원(KISA)에서 제공된 침해 사고 건수를 보면 2010 ~ 2018년도까지 웹으로 인해 해킹 건수가 증가하고 있다. 특히, 2014년에는 은닉 사이트 탐지가 47,703건 2015년에는 46,850건에 이르렀다. 최근 분석 2018년도에는 14,754건으로중소 기업 웹 취약점 점검 및 보안조지 강화 등으로 건수가 작년에 비해 67.1% 가 감소 하였지만, 계속해서 웹으로 인한 피해가 발생 하고 있다.

국내 공공기관을 포함한 그 외 기업에서 웹 해킹 관련 솔루션이 잘 되어 있지만, 해킹 공격 기법들의 지능화 ∙ 다양화로 인하여 침해사고 발생이 증가에 따라 침해사고도 지속적으로 증가 하고 있다. 이를 자세히 알 수 있게 전 세계 시장점유율 1위 웹 방화벽(WAPPLES)을 만든 회사인 (주)펜타 시큐리티에서 보도한 2017년도 웹 취약점 트렌드 분석 보고서(Web Application Threat Trend Report, WATT Report)에 주요 공격자 동향 따르면 Cross Site Scripting 33.31% SQL Injection 25.72% Stealth Commanding 24.64% File Uploading 13.32% Directory Traversal이 3.01%으로 국내뿐만 아니라 국외에서도 웹 취약점의 심각성을 알 수 있다.

그러하여 보안의 심각성을 알고 있는 대기업을 물론, 이를 이제야 인식한 중소 기업들이 보안에 대한 심각성을 인지해 투자를 하고 있는 추세이다. 우리는 이러한 트렌드를 따라 인터넷에서 오픈 되어 있는 취약점 사이트를 이용하여 지원한 회사의 기술지원을 하고, 실무경험을 쌓고자 이 프로젝트를 실시했다.

* **기대 효과**

이 프로젝트를 진행 하면서 웹 공격이 어떻게 이루어지며, OWASP 10공격 및 각종 공격에 어떻게 대응을 할 수 있는지 알 수 있으며, 보고서를 작성하여 자신만의 포토 폴리오를 만들 수 있다.

# 분석 환경

|  |  |  |
| --- | --- | --- |
| **PC – Name** | **Windows Version** | **IP** |
| **DESKTOP-VQBKA7M** | **Windows 10 Home**  **(64비트 운영체제, x64 기반 프로세스)** | **Ipv4 : 10.10.10.39** |
| **Ipv6 : fe80::5c3e:2294:df10:2a0a%12(Ipv6)** |
| **프로세스 : Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz 2.40 GHz** |

# 취약점 분석 툴

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Version** | **Download URL** | **Hash(SHA - 256)** |
| SQLmap | 1.3.11.98#dev | <http://sqlmap.org/> | 0FFE63579B4DCE0F5E08B5CCA1C0A44FB482F47E31245410134B49C2693A236D |
| Burp Suit | 2.1.04(64bit) | <https://portswigger.net/burp/communitydownload> | C76A9D21C8B4A6BC39997017BE045502C520A7F765A59A56D959C30440C01CB2 |
| Arachni Scanner | 1.5.1 – 0.5.12 | <https://www.arachni-scanner.com/download/> | ABCAFFCE527EC8AD98C8B7CE93D890629BE16989BCCD1880B17FF293B0D79BAE |

# 웹 취약점 진단 결과

|  |  |  |
| --- | --- | --- |
| No | 이름 | 점수 |
| **1** | **Bitcoin Web Site** | **54** |

# 취약점 목록

웹 관련된 취약점은 공개 된 것이 많은데 이 프로젝트에서는 최근에 공개된 **OWASP(Open Web Application Security Project) 2017**을 이용해서 ∙ 10점–**매우 높음** ∙ 5점–**높음** ∙ 3점-**중간** ∙ 1점–**낮음**으로 점수를 매길 것이다.

## **OWASP 10(2017 Version)**

|  |  |  |  |
| --- | --- | --- | --- |
| **번호** | **이름** | **위험 지수** | **취약점 유형** |
| **A1** | **인젝션** | **매우 높음(10)** | **1) SQL injection**  **2) CRLF injection**  **3) Command injection** |
| **A2** | **취약한 인증** | **중간(3)** | **1) 쿠키 변조**  **2) 세션 쿠키 타임 아웃**  **3) 다중세션 사용** |
| **A3** | **민감한 데이터 노출** | **높음(4)** | **1) 관리자 페이지 누출**  **2) 중요정보 비 암호화 노출**  **3) 백업 및 시스템 파일 노출**  **4) 부적절한 에러 페이지 노출**  **5.) 검색 엔진에 의한 노출** |
| **A4** | **XML 외부 개체** | **낮음(1)** | **1) injection 공격** |
| **A5** | **취약한 접근 통제** | **높음(5)** | **1) 파일 다운로드**  **2) 파일 업로드**  **3) URL 파라미터 조작**  **4) 자바스크립트 우회** |
| **A6** | **잘못된 보안 구성** | **중간(3)** | **1) 디렉터리 노출**  **2) 불필요한 HTTP Method 사용**  **3) 웹 서버 버전 정보노출**  **4) Missing X – Frame Options**  **5) Access – Control – Allow - Origin** |
| **A7** | **크로스 사이트 스크립팅** | **매우 높음(10)** | **1) XSS**  **2) HTTPOnly 속정 부제** |
| **A8** | **안전하지 않는 역직렬화** | **낮음(1)** | **1) 직렬화 된 객체 변경** |
| **A9** | **알려진 취약점이 있는 구성요소 사용** | **높음(5)** | **1) 컴포넌트 취약점(Spring, Struts 등)** |
| **A10** | **불충분한 로깅 & 모니터링** | **낮음(1)** | **1) 탐지 체계 구축여부** |

# 취약점 공격 유형

* **SQL Injection**

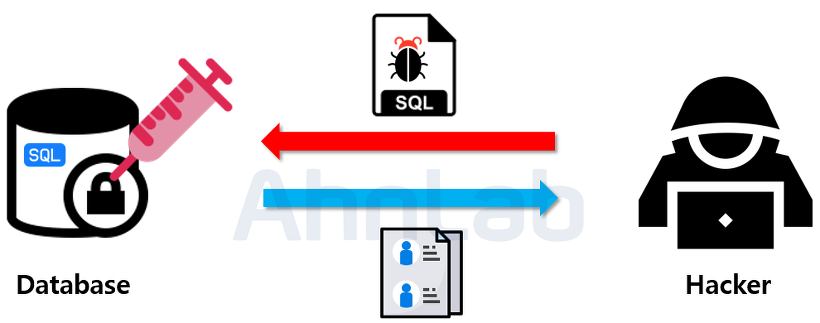


사진 – SQL Injection

: 신뢰할 수 없는 데이터나 명령어나 쿼리문의 일부분으로써, 인터프리터로 보내질 때 발생한다. 공격자의 악의적인 데이터는 예상하지 못한 명령을 실행하거나 적절한 권한 없이 데이터에 접근 하도록 인터프리터를 속이는 공격을 말함.

**→ Union Based SQL Injection**

: 두개의 쿼리문에 대한 결과를 통합해서 하나의 테이블로 보여주게 하는 방법.

**→ Blind SQL Injection(Boolean Based SQL)**

: 데이터 베이스로부터 특정한 값이나 데이터를 전달받지 않고, 단순히 참과 거짓의 정보만 알 수 있는 공격. 만일, 로그인 폼에서 SQL 공격이 가능하다면 서버가 응답하는 **로그인 성공과 로그인 실패 메시지를 이용하여 DB 테이블 정보 외의 정보를 추출 할 수 있음.**

**→ Blind SQL Injection(Time Based SQL)**

: 서버로부터 특정한 응답 대신에 참 혹은 거짓의 응답을 통해 데이터베이스의 정보를 유추하는 기법. 사용하는 함수는 종류에 따라 다르며 **MYSQL의 경우, sleep 와 benchmark가 있음.**

* **Cross Site Scripting**

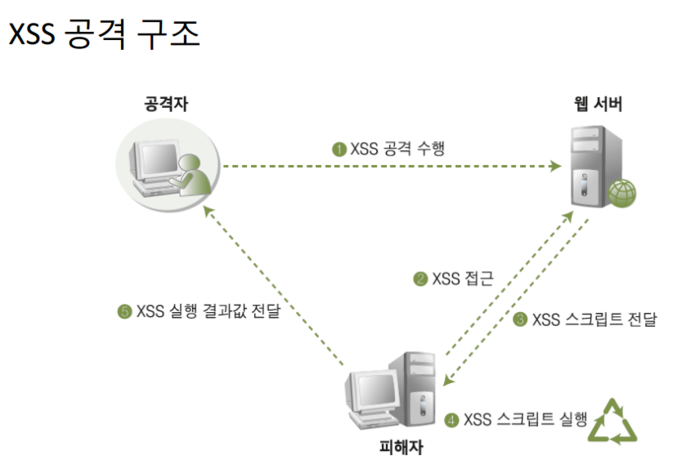


사진 – Cross Site Scripting

: 애플리케이션이 신뢰할 수 없는 데이터를 가져와 적절한 검증이나 제한 없이 웹 브라우저로 보낼 때 발생하며, 공격자가 피해자의 브라우저에 스크립트를 실행하여 사용자 세션 탈취, 웹 사이트 변조, 악의적인 사이트로 이동을 하게 만드는 공격을 말함.

**→ Reflected XSS**

: 사용자의 입력 값이 웹 화면에 바로 출력되는 취약점으로, 입력 parameter에 스크립트 삽입을 통해 취약점을 확인 할 수 있음.

**→ Stored XSS**

: 사용자의 입력 데이터가 데이터 베이스에 저장되고 데이터 베이스에서 해당 값을 추출할 때 발생하는 공격으로 주로, 게시판 글쓰기 폼에서 취약점이 많이 발견됨.

**→ DOM XSS**

: 원본 클라이언트 측에서 사용하는 희생자의 브라우저에서 DOM환경을 수정한 결과 공격이 실행되는 공격. 페이지 자체(HTTP 응답은 변경되지 않음)가 변경 되지 않지만 DOM 환경에서 발생한 악의적인 변경으로 페이지에 포함 된 클라이언트 측 코드가 다르게 실행되는 것을 말함.

* **Broken Authentication**

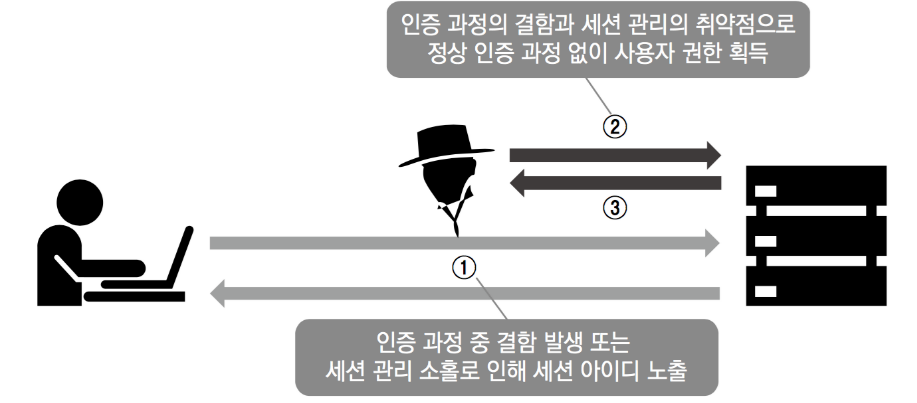
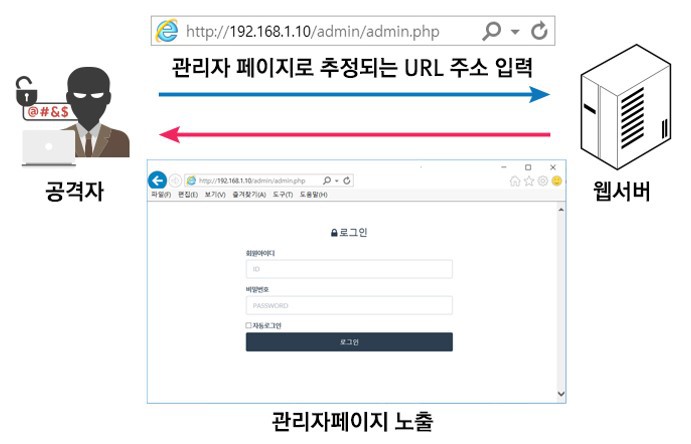


사진 – 취약한 인증

: 인증과 세션 관리와 관련된 애플리케이션 기능은 정확하게 구현되어 있지 않아, 공격자가 패스워드, 키 또는 세션 토큰을 해킹하거나 다른 구현 취약점을 공격하여 다른 사용자 계정을 일시적 또는 영구적으로 탈취하는 공격.

* **Broken Access Control**



* : 인증된 사용자가 수행할 수 있는 것에 대한 제한이 제대로 적용되지 않은 것을 공격자가 이용하여 사용자 데이터 수정, 액세스 권한 변경 등을 할 수 있는 공격을 말함.

# 상세 분석 결과

|  |  |
| --- | --- |
| A1 – SQL Injection | |
| **시나리오** | **툴을 이용한 공격으로 웹 페이지에서 사용 되는 데이터 베이스를 찾을 수 있으며, 그 외 취약점도 찾을 수 있음.** |
| **위험 지수** | **10** |
| **공격 구문** | **[1] sqlmap.py -u http://aspnet.testsparker.com/Products.aspx?pid=4 --batch --dbs** |
| **[2] sqlmap.py -u http://aspnet.testsparker.com/Products.aspx?pid=4 --batch -D (db name) -–tables** |
| **(사용 명령어)**  **--batch (사용자의 입력을 요구하지 않고, 기본값으로 진행)**  **-- dbs(데이터 베이스 목록)**  **-D (데이터 베이스 이름 지정)**  **-- tables (지정한 데이터 베이스의 테이블 목록)** |
| **결과** | 사진 – SQLmap 실행 후 공격 구문 입력  **- Anaconda Prompt을 관리자 권한으로 실행함.**  **- sqlma.py가 있는 경로로 이동하여 프로그램 실행함.**    사진 – 툴로 발견된 6개의 db  **- 총 6개의 데이터 베이스를 발견함.**  **- ASPtate ,Master ,Model ,Msdb ,Tempdb ,testsparke**를 툴에서 발견 함. |

|  |  |
| --- | --- |
| A1 – SQL Injection(Blind Injection) | |
| URL | http://aspnet.testsparker.com/ |
| 위험 지수 | **10** |
| 공격 URL | http://aspnet.testsparker.com//blog/vTjxEEiT'));%20waitfor%20delay%20'0:0:5'-- |
| 결과 | - F12 개발자 도구를 통해 공격이 통하는지 확인.  사진 – 네트워크를 통해 공격이 성공 한 것을 확인.  - vTjxEEiT'));%20waitfor%20delay%20'0:0:5— 경로에서 ‘5.22’초 동안 URL 요청이 지연 된 것을 알 수 있음.  = 사이트에서 사용되는 DB는 MSSQL인 것을 알 수 있음. |

|  |  |
| --- | --- |
| A3 - XSS | |
| URL | http://aspnet.testsparker.com/About.aspx?hello=visitor |
| 위험지수 | **10** |
| 공격 URL | http://aspnet.testsparker.com/About.aspx?hello=visitor%22%3E%3Cscript%3Ealert(1);%3C/script%3E |
| 결과 | 사진 6 – GET에 공격이 들어 간 것을 확인  사진 – 실행 결과  사진 – xss 공격 명령어 삽입  - URL창에 기본 공격인 ‘alert’문을 입력.  - 헤더에서 전달 메소드 **GET**에서 값을 전달 받는 과정에서 악성 스크립트 문자열을 필터링이 이루어 지지 않아 ‘alert’ 스크립트 공격이 성공 된 것을 알 수 있음. |

|  |  |
| --- | --- |
| A3 – XSS( HTTPOnly 설정 여부 ) | |
| 결과 | 스크린샷이(가) 표시된 사진  자동 생성된 설명  - XSS를 방지 하기 위해서는 Response 헤더에 set – cookie에 HTTPOnly 속성을 있어야 안전하는데 개발자 도구 – 네트워크를 통해 Default.aspx에 있는 set – cookie에는 HTTPOnly가 없는 것을 확인 함.  사진 7 – 전달 되는 쿠키 값 확인  HTTPOnly X |

|  |  |
| --- | --- |
| A4 – Broken Access Control | |
| URL | http://aspnet.testsparker.com/Help.aspx |
| 위험지수 | **5** |
| 공격 URL | http://aspnet.testsparker.com/Help.aspx?item=../windows/win.ini |
| http://aspnet.testsparker.com/Help.aspx?item=../../../windows/win.ini |
| 결과 | - 처음 시도한 ?item=../windows/win.ini공격에 에러 메시지가 뜨면서 win.ini가 있는 ‘C:\Websites\aspnet.testsparker\windows\win.ini’ 파일을 찾을 수 없다고 함.  - ‘ ../ ’의 계속적으로 입력하여 ../../../windows/win.ini에서 성공적으로 내용을 호출함. |

|  |  |
| --- | --- |
| A8 – CSRF(Cross Site Request Forgery) | |
| 시나리오 | 공격자가 게시물에 악성 스크립트를 삽입해 권리자 권한으로 실행되어 사이트에 피해를 줄 수 있다. |
| 위험지수 | **5** |
| 결과 | - http://aspnet.testsparker.com/guestbook.aspx에 있는 게시판에 취약점이  있는 것을 추측함.  스크린샷이(가) 표시된 사진  자동 생성된 설명  - 결과, ‘alert’문이 실행되면서 확인 후 , 작성한 스크립트가 보이지 않고 작성자만 보여 게시판에 취약점이 있는 것으로 판단함. |

# 결론

OWASP 10의 취약점 기준으로 모의 해킹 사이트를 분석한 결과, 일단 A1 – SQL Injection에서는 잘 알려져 있는 기법이나, 분석 툴을 이용해서 제일 먼저 발견되었다. 그리고 A3 – XSS와 A4 – Broken Access Control, A5 – Security Misconfiguration, A8 – CSRF(Cross Site Request Forgery)등 취약점을 수동으로 발견하기는 조금 어렸지만 웹 사이트를 사용하는 사용자 모르게 취약점들이 많은 것을 알 수 있었다. 물론, 지금 현재로서 SQL Injection이 가장 많은 취약점이 발견되기 때문에 그것을 중점으로 많이 하고 있지만, 그 외의 취약점이 어떻게 변형이 되어 공격 해올지 모른다.

따라서, 이번 프로젝트를 통해 웹 취약점에 대한 연구와 시큐어 코딩을 계속 발전을 해야 한다고 생각하며, 취약점 분석을 할 수 있는 사이트와 코드들이 해외에서는 많이 제공하지만, 국내에서는 제공 되는 것이 별로 없다. 바라는 것이지만 국내에서도 웹 취약점에 대해 모의 사이트나, 보고서 등을 제공 하면 국내에서의 취약점 분석을 바라는 이들에게 좋을 거 같다.

# 참고 자료

* 통계표 참고

<http://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1363>

* 툴

<https://xn--ex3bt1ov9l.kr/301>

<https://blog.naver.com/PostView.nhn?blogId=ndb796&logNo=221409941106&parentCategoryNo=&categoryNo=91&viewDate=&isShowPopularPosts=false&from=postView>

* 공격 기법

<https://m.blog.naver.com/PostView.nhn?blogId=heungmusoft&logNo=220731901661&proxyReferer=https%3A%2F%2Fwww.google.com%2F&view=img_1>

<https://berr-my.tistory.com/entry/XSS-%EA%B3%B5%EA%B2%A9>

* 보고서 작성

<https://aonenetworks.tistory.com/203>

<https://prof.ysu.ac.kr/pds_update/%EC%B7%A8%EC%95%BD%EC%A0%90%EB%B6%84%EC%84%9D%EB%B3%B4%EA%B3%A0%EC%84%9C.pdf>

[file:///C:/Users/rladb/Downloads/WWQ\_Report\_Sample[1]%20(3).PDF](file:///C:\Users\rladb\Downloads\WWQ_Report_Sample%5b1%5d%20(3).PDF)

<https://gmlwjd9405.github.io/2019/01/28/http-header-types.html>

<http://blog.naver.com/PostView.nhn?blogId=devotaku&logNo=220629958882>

<https://noirstar.tistory.com/264>

<https://b1a423.tistory.com/36>

<https://hacking-beginner.tistory.com/184>

<https://itstory.tk/entry/CSRF-%EA%B3%B5%EA%B2%A9%EC%9D%B4%EB%9E%80-%EA%B7%B8%EB%A6%AC%EA%B3%A0-CSRF-%EB%B0%A9%EC%96%B4-%EB%B0%A9%EB%B2%95>

* 그 외

<http://aspnet.testsparker.com/Request.aspx?r=/statics/download/>