Cryptographic failures, SQL injection, XSS 이해하기

32217606 이진형

OWASP Top 10 (2021) 중에서 다음 취약점들의 동작 원리 및 방어 기법을 정리하여 보고서로 제출하기 (보고서는, 총 10 페이지 이내로 정리)

* Cryptographic failures란 무엇인지? 이에 대한 방어 기법은?
* Injection 중에서 SQL injection의 동작 원리는? 이에 대한 방어 기법은?
* Injection 중에서 Cross-Site Scripting (XSS)의 동작 원리는? 이에 대한 방어 기법은?

Cryptographic failures는 한국어로 변역하자면 암호화 오류라는 뜻을 가지고 있다. 즉 쉽게 말하면 암호화가 잘 이루어지지 않았다는 뜻이다.

다시 말해, 중요 정보를 저장 및 전송을 할 시 암호화 통신이 이루어지지 않는 경우, 취약한 암호 프로토콜을 사용하는 경우를 Cryptographic failures라고 부른다.

Cryptographic failures의 방어기법은 크게, 중요 정보를 송수신 하는 페이지에 암호화가 통신이 이루어지는지, 취약한 버전의 암호 프로토콜을 사용하고 있지 않은지를 점검해서 판단한다.

많은 웹 어플리케이션들이 신용카드, 개인 식별 정보 및 인증 정보와 같은 중요한 데이터를 제대로 보호하지 않을 때 일어난다. 이 취약점을 공격하는 사람들은 신분을 도용하거나, 신용카드를 사기 치는 등 약하게 보호가 된 데이터를 변경하거나 훔친다. 방어기법은 중요 데이터가 전송, 저장 중일 때 암호화와 같은 보호조치를 가져야 한다. 민감한 정보들이 백업, 저장 공간에 저장될 때 암호화를 시키지 않은 긴 문자들로 저장 되는지 파악해야 하며, 특히 많은 보안적 취약함이 있는 인터넷 트래픽을 조심해야 하고, 전송이 될 때도 내부와 외부에 암호화가 되지 않았는지 살펴보아야 한다. 또한, 오래되거나 취약한 암호 알고리즘을 사용하지는 않았는지, 보안에 취약한 암호키가 생성 되었는지와 키 관리는 적절하게 이루어지고 있는지 키에 관한 문제점 또한 살펴 보아야하며, 헤더가 민감한 데이터나 브라우저 보안지침이 제공되었거나 브라우저에 보내졌을 때 놓치지 않는지에 대해 살펴보아야한다.

SQL Injection은 데이터베이스 관련 어플리케이션에서 발행한다. 공격자가 악의적으로 SQL구문을 주입하여 데이터베이스에 직접 접근하거나 조작한다. 이를 통해 공격자는 표적이 가진 중요한 정보를 무단으로 조회하거나 삭제, 수정한다. 또한 공격자가 신원을 속이고 기존 데이터를 바꿔서 대표적으로 거래 무효화 또는 잔액 변경과 같은 금전적인 부인 문제를 일으키고 시스템의 모든 정보를 완전히 파악하여 데이터를 파괴하거나 데이터베이스 서버의 관리자가 되어 데이터베이스의 정보를 탈취할 수 있다. 특히, 출처가 알 수 없는 데이터가 웹 서버나 데이터베이스에 접근할 수 있거나 SQL쿼리문이 동적으로 생성되었을 시에 SQL Injection 공격이 가능할 수 있다. 가장 대표적인 방법으로 싱글 쿼테이션을 사용한 공격방법이 있다.

SQL Injection의 방어기법은 웹 서버 시큐어 코딩과 웹 방화벽 설치를 하는 방법이 있다. 또한 런타임 어플리케이션 자가 보호 (RASP)를 도입하면 입력값 검증의 정확성과 효율성을 높일 수 있다.

사이버 공격기법이 지속적으로 발전하기 때문에, 인공지능 기반 보안 솔루션 도입 또한 생각해볼 수 있다. 이 솔루션은 이상 탐지 기능을 사용하는데, 웹 어플리케이션의 정상적인 트래픽 패턴과 다른 요청들을 식별한다. 이것을 통해 SQL Injection 공격과 관련된 요청들을 차단하고, 자연어 처리 기능을 활용해, 사용자 입력값을 분석하고 SQL Injection을 시도하는 입력값을 식별하게 해준다.

이와 다른 방법으로는 연속적인 모니터링을 통해 공격을 실시간으로 분석해 대응을 하는 방법이 있고, 보안 정책의 자동 최적화로 새로운 공격 기법에 대한 대응 전략을 찾아볼 수 있다.

몇몇 웹 프레임워크에서는 SQL Injection 공격 을 회피할 수 있는 다양한 기능들을 제공하고 있다. PHP는 magic quotes를 제공하는데 POST, GET, COOKIES에서 오는 입력 값에서 4가지 특수문자 ‘, “, /, NULL이 있으면, 자동으로 \을 추가함으로써 SQL Injection공격을 방지해 준다. Struts의 validator는 미리 정의된 규칙에 의해서 입력 데이터 값의 유효성을 검사한다. validator는 SQL Injection 공격에 사용되는 특수문자 들에 대한 규칙의 정의가 잘 되어있지 않다면 SQL Injection 공격에 취약할 수 있으며, 규칙 설정이 복잡하다는 단점이 있긴 하다.

정적 분석 방법도 SQL Injection의 예방 방법이 된다. 정적 분석 방법은 SQL Injection 공격 탐지를 목적으로 하는 것 보다는 사용자 입력 값 유형의 유효성을 검사하는 것에 중점을 두지만, 잠재적으로 SQL Injection 취약점을 감소시키는 역할을 한다. JDBC-Checker는 Java String Analysis라이브러리를 이용하며, 동적으로 생성되는 사용자 입력 값 유형의 유효성을 검사하여 SQL Injection 공격을 방지한다. Wassermann은 Automated reasoning 방법과 결합한 정적 분석 방법을 사용했다. 이 방법은 동적으로 생성되는 SQL 질의에 중복되는 SQL 질의 조건 연산자 가 없다는 가정하에 SQL 질의를 검증한다. Stephen은 웹 어플리케이션에 있는 plain text SQL statement, SQL 질의, execution call을 수집하여 사용자 입력의 유형을 검증할 수 있도록 SQL 질의를 고친다. 이 방법은 SQL Injection 공격을 탐지하고 예방하는 방법이 아닌 취약한 SQL 질의 구문을 사전에 제거하는 방법을 사용해 SQL Injection 공격을 예방한다. 이 방법은 java 언어로 개발된 웹 어플리케이션에만 가능하며, AST와 SQL parser Zql와 같은 특정 라이브러리가 요구된다.

정적 분석 방법과 마찬가지로, 동적 분석을 이용한 방법도 SQL Injection 예방 방법이 된다. 동적 분석 방법은 웹 어플리케이션 외부에 SQL 질의 분석 프로세스를 이용하여 간접적으로 SQL Injection 공격 취약점을 찾는다. 간접적으로 웹 어플리케이션을 분석하기 때문에 정적분석 방법과 다르게 웹 어플리케이션 수정 없이 SQL Injection 공격 취약점을 찾을 수 있다.

정적 분석 방법 중, Sania방법은 클라이언트와 웹 어플리케이션 사이의 정상 SQL 질의와 웹 어플리케이션과 데이터베이스 사이의 정상 SQL 질의를 수집하여, SQL Injeㅇction 공격에 취약한 부분을 찾아 공격 코드를 생성한다. 생성된 공격 코드를 이용하여 공격을 시도하고, 공격으로부터 발생된 SQL 질의를 수집한다. 이렇게 수집된 정상 SQL 질의와 공격 SQL 질의를 parse tree를 이용하여 비교 분석함으로써, 공격의 성공 여부를 확인한다. parse tree를 이용하기 때문에 HTTP 응답 값을 이용하여 성공여부를 확인하는 것보다 정확도가 높다.

정적 분석 방법의 예시 중, 마지막 방법은 Input flow Analysis와 Input Validation Analysis을 통하여 white-box을 만들고, 이를 이용하여 테스트 입력 데이터를 만들어 SQL Injection 취약성을 찾는 방법이다. 이와 같이 동적 분석 방법은 웹 어플리케이션 수정이 필요 없다는 장점이 있지만, 발견된 웹 어플리케이션 취약점을 개발 자가 직접 수정해야 한다는 점과 미리 정의된 공격코드 이외의 취약점은 찾을 수 없다는 단점이 있다.

정적 분석과 동적 분석을 혼용한 방법 또한 존재하는데, 웹 어플리케이션에 있는 SQL 질의를 정적으로 분석하고 외부에서 발생하는 동적 SQL 질의와 비교 분석하는 방법이다. 이 방법 중 예시 로 AMNESIA, Buehrer,wei를 소개해보겠다.

AMNESIA는 웹 어플리케이션에서 SQL 질의가 실행되는 핫스팟을 찾은 다음 모든 가능한 SQL 질의 들을 생성한다. 이렇게 생성된 정적 SQL 질의와 사용자로부터 받은 동적 SQL 질의를 JSA Library를 이용하여 분석한다.

Buehrer은 웹 어플리케이션의 정적 SQL 질의와 사용자로부터 생성된 동적 SQL 질의를 parse tree를 이용하여 상호 비교 분석함으로써 SQL Injection 공격 여부를 확인한다.

Wei은 웹 어플리케이션에 있는 stored procedure인 정적 SQL 질의와 동적으로 생성되는 동적 SQL 질의를 Control Flow Graph을 만들어 비교 분석함으로써 SQL Injection 공격 여부를 탐지하는 방법을 사용한다.

더욱 자세히 알아보기 위해 관련 논문을 찾아 보던 중, SQL Injection 취약점 사전 진단 시스템이 눈에 들어왔다. 우선, SQL Injection 취약점 사전 진단 시스템은 Tautologies, Union Query, Piggy-Backed Query에 대해 취약점 탐지를 목표로 한다. 사전 진단 시스템의 동작 단계는 대상 서버의 입력 파라미터를 추출하고, 악의적 쿼리를 이용하여 SQL Injection의 취약점을 탐지하는 절차를 밟는다. 본 논문은 이 과정에서, 시스템의 성능평가를 위해 테스트 서버를 구축하였고, SQL Injection 공격을 수행하기 위한 로그인 기능과 게시판 기능또한 구현하였다. 이렇게 구축한 테스트 서버에 공격 쿼리를 이용하여 SQL Injection 사전 진단 시스템을 수행한 후 취약점 여부를 확인하는 과정을 밟았다. 수행결과로, SQL Injection의 공격기법 중 Tautologies, Union Query, Piggy-Backed Query에 대해 사전 진단이 가능해졌다.

XSS는 공격자가 상대방 브라우저에 스크립트를 실행할 수 있게 하여 웹 사이트를 변조, 악의적 컨텐츠를 삽입하거나 피싱 공격 또한 가능하게 한다. XSS공격은 스크립트 언어 그리고 취약한 코드들이 공격대상이 되는데, JavaScript, VBScript, ActiveX, HTML, Flash, interactive bulletin boards, custom error pages, search engines CGI scripts 들이 있다. 이때 XSS가 위험한 이유는 공격 기법 자체가 HTML과 Script를 이용하는 것이기 때문에 쉽게 공격이 가능한 코드를 제작할 수 있다는 것과, 이렇게 제작된 코드 대부분이 홈페이지에 손쉽게 올릴 수 있기 때문이다. XSS는 Stored XSS와 Reflected XSS 이렇게 크게 두가지로 나뉜다고 한다. Stored XSS는 공격자가 게시물에 악성 스크립트를 삽입하는 것으로 시작한다. 이때, 사용자가 게시물을 클릭하게 되면 공격자의 자바스크립트가 포함된 응답이 전송되게 되는데, 공격자는 이때 사용자의 쿠키, 세션 등 원하는 정보를 얻게 된다. Reflected XSS는 URL의 CGI 인자에 Script Code를 삽입하는 것인데, 공격자가 이메일에 특정한 웹 페이지 링크를 보내는 것으로 시작한다. 사용자가 만일 그 링크를 클릭하게 되면 웹페이지에 대한 링크 URL에 삽입된 스크립트 코드가 실행되게 되며 웹 페이지의 내용이 변경되게 된다. Reflected XSS 공격은 스크립트를 저장하기 위한 웹 사이트가 필요하지 않다는 특징이 있고, 그 이유는 조작된 링크 주소가 클릭될 때 링크에 대한 웹페이지가 로드 되어 그 스크립트 코드가 실행되기 때문이다.

XSS의 방어기법은 우선 중요한 정보를 쿠키에 저장을 하지 않는 방법이 있다. 또한, Script code에 사용되는 특수문자에 대해 정확히 필터링을 해야 한다. 사용자가 입력할 수 없는 문자들은 필터링을 시켜 XSS 공격에 필요한 특수문자들의 사용을 방지하는 것이다. 다른 방법으로는 게시판에서 HTML 포맷의 입력을 불가능하게 하는 것이다. 그 외의 여러 방법들을 나열해 보자면, 이메일 링크 같은 형식을 바로 클릭하지 말고, 주소창에 URL을 직접 입력하여 들어가는 방법이다. 이것은 XSS 공격에 가장 기본적인 대응 방법이다. 또한, 인터넷의 패치를 가장 최신의 것으로 업데이트 해 자신의 보안 취약점을 없애고, 인터넷 옵션에 개인 정보 등급을 높이게 해서 불필요한 쿠키값을 보내지 않게 하는 것과, 웹 메일 옵션에서javascript 허용여부등을 허용하지 않는 것으로 선택해 XSS공격을 예방할 수 있다.

그 외, 자세한 XSS 방어 기법을 알아보기 위해 논문을 탐색해본 결과 흥미로운 방어 기법을 발견했다. 이 논문에서는 XSS의 취약점 스캔 도구를 비교 분석하여 취약점 탐지율이 높은 스캔 옵션을 발견한 것인데, 이러한 취약점 스캔 도구의 자산 수집 기능의 한계점을 해결하기 위해 자산 수집 도구 또한 연계하여 취약점 탐지 범위를 늘린 프레임워크를 제안한 것이다. 본 내용을 자세히 말해보자면, 우선 XSS 취약점 스캔 도구의 분석 결과는 XSpear 도구가 효율성과 정확성 측면에서 우수하다고 언급했다. 이때 앞서 언급했듯이, XSS취약점 스캔 도구들은 주로 서브도메인이나 파라미터 수집 기능이 제한적이거나 불편함을 갖고 있다. 가장 효율성이 좋다고 언급한 XSpear 도구는 엔드포인트와 파라미터 수집 기능이 존재하지 않으며, Dalfox 도구의 경우 Wordlist에 존재하지 않을 경우 효과적이지 않고, 직접 엔드 포인트를 수집할 수 없다. 또한, XSSer,XSStrike 도구는 상위 URL에 종속 된 범위에서만 엔드포인트 수집을 지원하기 때문에 와일드 카드 형태의 도메인에서는 서브도메인의 수집이 불가능하다고 한다. 이처럼 XSS 취약점 스캔 도구들은 취약점 탐지에만 치우친 나머지, 자산 정보 수집의 한계점이 존재한다. 반면에 취약점 탐지를 위해 신뢰성이 있는 자산 정보를 최대한으로 수집하는 것이 XSS 취약점 스캔 도구의 효용성과 밀접한 역할이 있어 자산 정보는 중요한 역할을 하고 있으며, 이러한 자산 정보적 한계를 보완한 효과적인 XSS 취약점 탐지 프레임워크를 제안한 것이다. 본 논문에서 제안하는 프레임워크는 자산 수집 도구를 사용하여 와일드 카드 형태의 도메인으로부터 모든 서브 도메인을 추출하고, 수집된 도메인의 유효성을 판단한다. 유효성이 검증된 도메인에서 파라미터가 포함된 엔드포인트를 수집하고, 이를 XSS 취약점 스캔 도구에 전달하여 공격 벡터를 분석한다. 제안하는 프레임워크는 다수의 단계로 이루어져 있으며, 우선은 Subfinder 도구를 사용하여 지정된 와일드 카드 형태의 도메인으로부터 서브도메인을 추출해야 한다. 해당 과정은 도메인의 수가 큰 경우 효율적으로 서브도메인을 식별하는데 중요한 역할을 한다. 2번째로는, Httpx 도구를 통해 수집된 서브도메인에서 동작하는 웹 서버의 활성화 여부를 통해 유효성을 검증한다. 해당 과정은 존재하지 않거나 비활성화된 도메인을 판별하는 과정으로 효과적인 자원 사용과 분석의 정확도를 높일 수 있다. 세번째로, Katana 도구를 사용하여 추출된 서브도메인으로부터 파라미터가 포함된 엔드포인트를 수집한다. 해당 과정을 통해 XSS 취약점 탐지에 필수적인 정보를 제공한다. 마지막으로, 3번 절차를 통해 생성된 데이터를 XSpear 도구에 전달하여 XSS 취약점을 분석한다. 취약점 탐지율을 높이면서도 시스템 가용성에 미치는 영향을 최소화하는 스캔 옵션을 사용함으로써 XSS 취약점 스캔 도구의 한계점을 해결하고, 효과적인 XSS 취약점 탐지가 가능한 것이다. 결론적으로 요약하자면, 수동 분석의 한계를 해결할 수 있는 대표적인 사안 중 하나인 XSS 취약점 스캔 도구는 특징과 성능이 서로 다르며 도구의 정확한 이해가 없는 경우 자동화 도구의 효율성을 극대화할 수 없고, 가용성 침해의 위험성 또한 높다. 그렇기 때문에 이 논문에서는, 대표적인 XSS 취약점 스캔 도구를 비교 분석해 가장 우수한 Xspear 도구를 중심으로 자산 수집 도구와 연계하여 XSS 취약점 스캔의 효율성을 극대화하는 효과적인 탐지 프레임워크를 제안하였다. 제안하는 프레임워크를 통해 기존 취약점 스캔 도구의 자산 수집에 대한 한계점을 자산 수집 도구와 연계함으로써 해결하였다는 XSS 취약점 해결 방안이 있다.

유도진. (2023). 중국 해킹그룹 샤오치잉 사이버 무기체계 대응방안 연구: SQL Injection 및 OSINT기반 Known Vulnerability 공격. 한국산학기술학회 논문지, 24(6), 267-273, 10.5762/KAIS.2023.24.6.267

홍성혁. (2013). XSS 공격과 대응방안. 디지털융복합연구, 11(12), 327-332.

서홍준, 이정우, 윤성민, 이승환, 박연준, 배민준, 김주원.(2024).XSS 취약점 스캔 도구 분석을 통한 효과적인 탐지 프레임워크 제안.한국통신학회 학술대회논문집,(),1328-1329.

박영민, 이대원. (2023). SQL Injection 취약점 사전 진단 시스템 개발. Proceedings of KIIT Conference,