개방형 보안 API를 이용한 취약점 탐지 기법

김성수[O] 김도현 김선혁 홍충선*
경희대학교 컴퓨터공학과
{ mjs0514, doma, kshyuk0605, cshong}@khu.ac.kr

Mechanism for Providing a Vulnerability Detection using Open API

Sung Soo Kim[O] Do Hyun Kim, Seon Hyeok Kim, Choong Seon Hong*
Department of Computer Science and Engineering Kyung Hee University

요 약

현재 웹의 기술이 점점 발전함에 따라 어떤 형태의 서비스라도 웹에서 이용할 수 있는 시대가 오고 있다. 그 중심에 있는 Open API 기술은 여러 서비스가 융합된 형태의 매쉬업 (Mash-up) 서비스가 나타나게 된 요인이다. 이에 따라 국내에서도 많은 종류의 매쉬업 서비스들이 나오고 있지만, 아직 보안에 관련된 Open API를 찾아볼 수 없다. 만약 보안 기능을 Open API로 제공할 수 있다면 서버 관리자들은 이것을 사용해서 쉽게 보안 검사를 할 수 있을 것이다. 이러한 문제를 해결하기 위해 본 논문에서는 Open API를 통해 취약점 탐지 기능을 제공하기 위한 방법을 제안한다.

1. 서 론

현재까지 보안사고 사례들을 살펴보면 몇 가지 공통점이 있다. 특별한 몇 가지 경우를 제외하면, 해커들은 응용 서비스 계층에 존재하는 취약점을 노린다. 예를 들면 AhnLab의 안티 바이러스 프로그램을 업데이트 해주는 업데이트의 서버가 가지는 취약점을 노려서 업데이트 서버에 연결된 모든 클라이언트들에게 악성코드를 배포시킨 사건이 이런 경우에 해당한다. 이처럼 이런 해커들의 공격에 대처하기 위해서라도 반드시 관리자는 여러 보안 이슈들에 대해서 고려해야 한다. 이것은 응용 서비스를 설계하는 단계에서 가장 먼저 고려할 사항으로 보안을 선택해야하는 이유기도 하다. 현재까지 많은 보안 관련 기업 및 연구기관에서 선행 연구를 통해 다수의 플랫폼과 도구들이 개발되었지만, 모든 면에서 시스템의 보안을 완벽하게 보호해주는 도구가 없는 것이 현실이다. 따라서 관리자는 주기적으로 다양한 도구를 가지고 서버의 취약점을 평가하고 관리해야 한다.

취약점 탐지에 대해서 크게 3가지로 분류해 보면, 시스템 취약탐지, 웹 취약점 탐지 그리고 소프트웨어 취약점 탐지가 있다. 첫 번째로 시스템 취약점 탐지는 호스트 PC에 대한 전반적인 취약점을 탐지한다. 두 번째로 웹 취약점 탐지는 웹 사이트와 관련된 취약점을 탐지하며, 마지막으로 소프트웨어 취약점 탐지는 퍼징 기법을 통해 소프트웨어의 취약점을 탐지한다. 퍼징 (Fuzzing)이란 무작위의 데이터를 반복 입력해서 소프트웨어의 조직적인 실패를 유도함으로써 취약점을 찾아내는 것을 말한다. 본 논문에서는 OpenVAS라는 스캐닝 도구를 사용해 시스템 취약탐지와 웹 취약점을 탐지하는 기능을 Open API (Open Application Programming Interface)를 통해 제공하는 방법을 제안한다.

이 논문은 2015년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (R0126-15-1009, ICBMS 플랫폼 간 정보 모델 연동 및 서비스 매쉬업을 위한 스마트 중재 기술 개발) *Dr. CS Hong is the corresponding author

2장에서는 몇 가지 관련연구들에 대해서 설명하고 3장에서는 제안하는 방법의 구조와 절차와 함께 실험에 대해 설명한다. 그 후 끝으로 4장에서 결론 및 향후 계획에 대하여 논한다.

2. 관련 연구

2.1 Open API

Open API는 애플리케이션과 네트워크 사이의 표준화된 인터페이스라고 할 수 있다.[1] Open Mobile Alliance (OMA) 그룹에서는 REpresentational State Transfer (REST) 기반의 Open API를 3rd party 애플리케이션에서 네트워크의 기능을 쉽게 사용할 수 있게 하는 것이라고 정의하고 있다. [2]

2.2 OpenVAS

OpenVAS (Open Vulnerability Assessment System)는 포괄적으로 강력한 취약점 스캐닝과 함께 취약점 관리 솔루션을 가지고 여러 서비스를 제공하는 프레임워크이다. 그림 1에 나와 있듯, OpenVAS는 다음과 같이 GSA (Greenbone Security Assistant), OpenVAS CLI, OpenVAS Manager 그리고 OpenVAS Scanner로 구성된다. [3]

GSA와 CLI는 OMP (OpenVAS Management Protocol) 를 통해 Manager를 구동시키는 애플리케이션이다. GSA는 웹 브라우저로 사용자 인터페이스를 제공하고, CLI는 커맨드로 Manager를 구동시키는 “omp” 툴을 제공한다. 다음으로 Manager는 전체 취약점 관리 솔루션에 취약점 스캐닝을 통합한 중앙 서비스이며 OTP (OpenVAS Transfer Protocol)를 통해 스캐너를 제어한다. 마지막으로 OpenVAS의 핵심인 Scanner는 OpenVAS NVT Feed와 상용 공급 서비스를 통해 매일 업데이트 되는 실제 Network Vulnerability Tests (NVTs)를 수행한다.

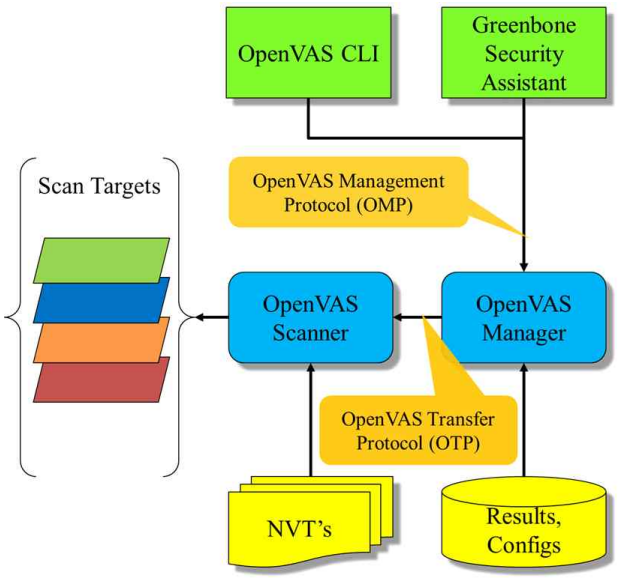

그림 1. OpenVAS Architecture Overview

3. 제안 사항

본 논문에서 제안하는 Open API를 통한 취약점 탐지 기능을 제공하기 위해 OpenVAS라는 취약점 탐지 도구를 사용하고자 한다. 먼저 제안하는 방법에 대한 구조도를 설명하고 절차에 따라 어떻게 동작하는지를 설명한다.

3.1 구조도

다음과 같이 그림 2에는 Open API 서버의 구조도를 보여주고 있다. Application 계층만 외부 계층이고 나머지 계층은 서버의 내부 계층을 나타낸다. 제일 하위 계층인 Library 계층은 오픈소스로 제공되는 보안 라이브러리나 바이너리 같은 보안 도구들이 위치하며, 실제 보안에 대한 프로세스가 돌아가는 계층이다. 그 다음으로 Adaptor 계층은 상위의 Open API 계층과 하위의 Library 계층을 연결해주는 역할을 수행한다. Open API 계층은 API를 제공하기 위한 모듈과 이것을 탑재하기 위한 서버 컨테이너가 있는 계층이다. 마지막으로 Application 계층은 Open API를 통해 애플리케이션을 개발하는 3rd party 애플리케이션들의 계층이라고 할 수 있다.

3.2 동작 순서

가장 먼저 Open API를 통해 스캐닝을 요청하기 위해서는 설계한 URI (Uniform Resource Identifier) 를 통해 config 라는 파라미터에 대한 값을 입력해야 한다. 이 값은 스캐닝 탐지 방식을 Open API를 통해서 설정하기 위해 정의한 입력 변수이다. 이 파라미터를 사용해서 이 방법을 설정할 수 있다. 이처럼 Open API에 대한 요청이 들어오면, Open API 모듈은 이 요청을 내부 어댑터를 통해 라이브러리 계층에 있는 OpenVAS에게 전달한다. OpenVAS에서는 스캐닝을 시작하기 위해서 반드시 Task ID가 필요하다. Task ID는 Config ID와 Target ID를 통해 만들어지는 문자열로 된 식별자이다. 마찬가지로 Config ID는 OpenVAS에서 정의하는 탐지 방식을 구별하기 위한 문자열로 된 식별자이고 이것은 URI로부터 입력받는 파라미터와 대응되는 값이다. 마지막으로 Target ID는 스캐닝하고 싶은 대상의 IP address를 통해 생성된 식별자이며 이것 또한 문자열로 이루진다. 이를 통해 Open API로 취약점 탐지 기능을 사용하기 위해 만든 절차를 그림 3에 나타낸다.

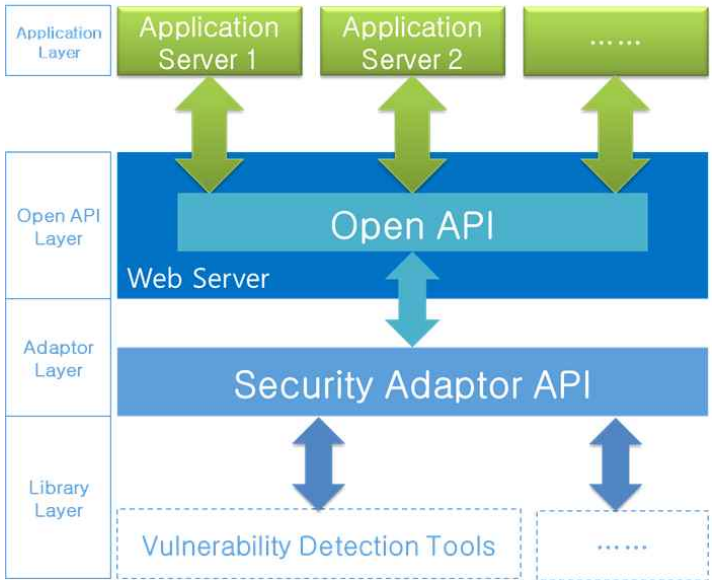

그림 2. Architecture of Open API Server

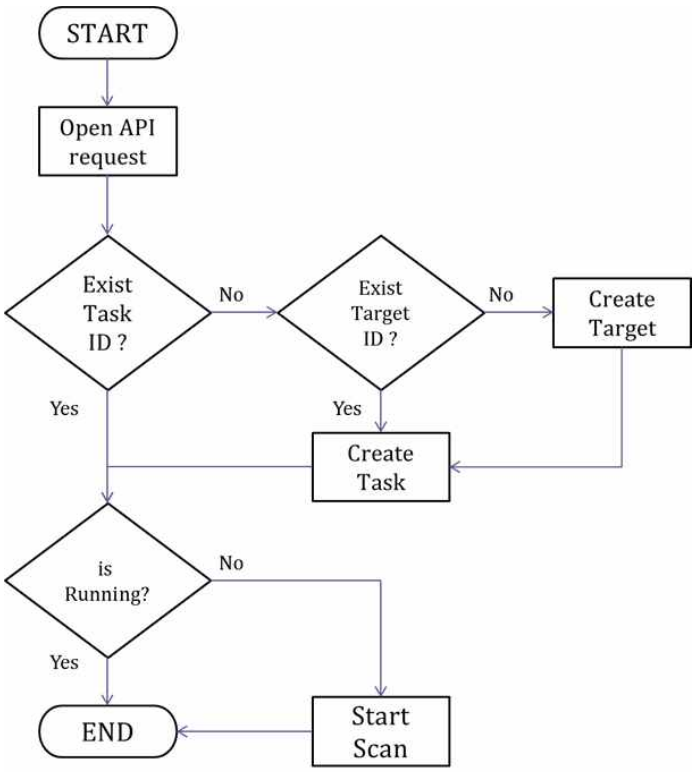

그림 3. Flow Chart for Vulnerability Detection

3.3 구현

구현을 위해 Ununtu 14.04.3 LTS에서 개발환경으로 이클립스에서 자바를 사용했다. 다음과 같은 시나리오를 가지고 제안하는 방법에 맞게 동작하는지 실험했다. 시나리오는 API를 처음 요청한 경우와 처음 요청하지 않는 경우로 진행했으며, 처음 요청하지 않은 경우에는 이전과 다른 새로운 config 값을 입력했을 때와 이전과 같은 config 값을 입력했을 때로 나눠서 실험했다. 다음 그림 4에서 각각의 시나리오에 대한 결과를 나타냈다.

3.4 성능 평가

제안하는 방법에 대해 성능을 평가하기 위해서 Open API 를 통해 들어오는 동시 요청 수에 따라 처리하는 지연 시간이 얼마큼 증가하는지 측정한 평균값을 그림 5를 통해서 나타냈다. 기존에 존재하는 Task에 대해서는 평균적으로 4~5개의 요청이 들어와도 처리하는 시간이 크게 달라지지는 않았지만 반면에, 기존에 존재하지 않는 Task들은 영향을 끼치지 않는 동시 요청 수가 좀 더 적은 것을 알 수 있었다. 결과적으로 처리 지연시간은 동시 요청 수가 증가 할수록 지수 적으로 증가했다.

4. 결 론

본 논문에서는 Open API를 통해 취약점 탐지 기능을 사용할 수 있게 했다. 제안하는 방법을 통해 손쉽게 3rd party에서 사용해서 취약점 탐지 기능을 개발할 수 있도록 했다. 이것은 공개된 API 특성상 한명이 아닌 다수의 사용자에게 편의성을 제공하는 효과를 기대할 수 있다.

향후 계획으로는 취약점 탐지 도구뿐만 아니라 다른 형태의 보안 도구들까지 모두 확장할 계획이며, Open API를 제공하는 서버 자체에서 발생할 수 있는 보안 이슈 등에 대해서 연구할 계획이다.

5. 참고 문헌

[1] Open API, “https://en.wikipedia.org/wiki/open_API”

[2] Sunhwan Lim, Changsup Keum, “Method of application driven Qos service in open service platform based on RESTful web services,” IEEE Conference on Information and Communication Technology Convergence (ICTC 14), pp.632-633, October 2014.

[3] OpenVAS, “http://openvas.org/index.html”

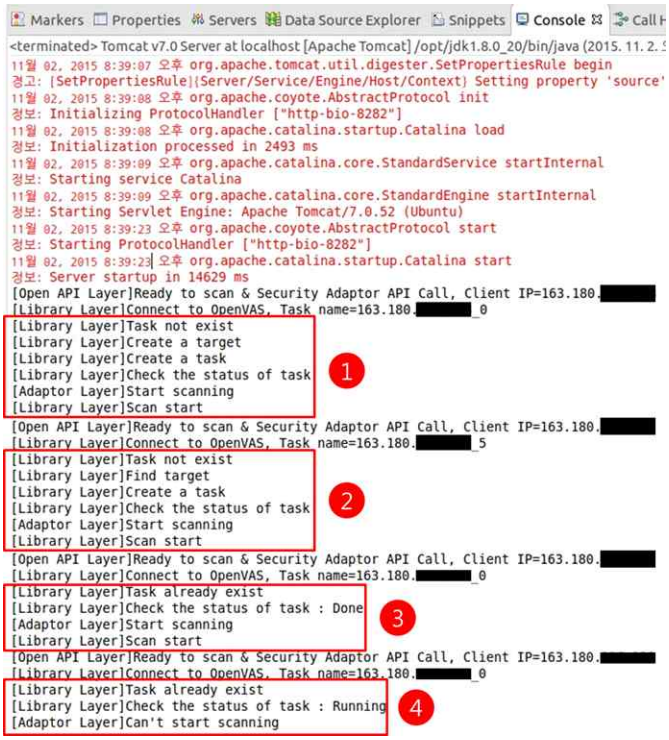

그림 4. Implementation result of mechanism for vulnerability detection

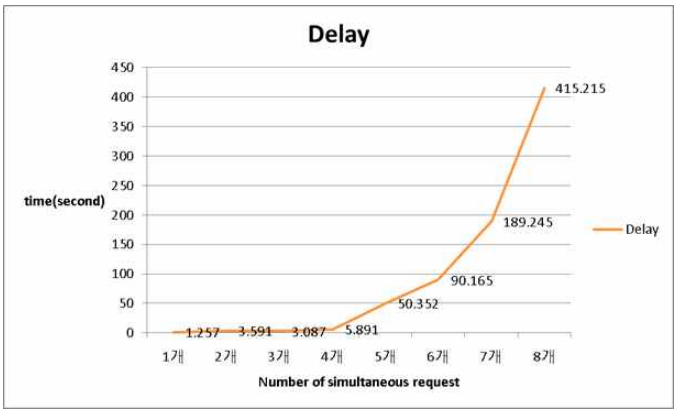

그림 5. Process delay according to number of simultaneous request