취약성 탐지 스크립트의 원격 편집기 설계 및 구현

이철호[0]*, 최경희*, 박승규*, 정기현**, 이철원***, 이남훈***
*아주대학교 정보통신전문대학원, **아주대학교 전자공학부, ***국가보안기술연구소
(lchsoft[0], khchoi, sparky, khchung)@madang.ajou.ac.kr, (cheolee, nhlee)@etri.re.kr

Design and Implementation of Remote Script Editor for Vulnerability Scanning

Cheol-Ho Lee[0]*, Kyung-Hee Choi*, Seung-kyu Park*, Gi-Hyun Jung**, Cheol-Won Lee***, Nam-Hoon Lee***
* Graduate School of Information and Communication, Ajou University,
** Division of Electronics Engineering, Ajou University,
*** National Security Research Institute

요 약

본 논문에서는 취약성 탐지 스크립트의 원격 편집기를 설계 및 구현하였다. 원격 편집기는 Windows 계열의 시스템에서 구동되며 스크립트 데이터베이스가 저장되어 있는 서버로부터 스크립트를 인출하여 수정 후 저장하거나 새로운 스크립트를 작성하여 저장한다. 편집기는 스크립트의 작성 및 편집을 위한 편집기능, 서버와의 HTTP 프로토콜 기반 통신 기능, 스크립트의 사용 용도나 스크립트의 경향과 특성을 파악하는 등의 특성분석기능, 스크립트의 유효성 및 문법 검사기능을 가지고 있으며, 원격 편집기의 요청이 있을 때에 이에 응답하기 위하여 스크립트 데이터베이스 서버측에서 CGI(Common Gateway Interface)가 구동된다.

1. 서 론

스크립트(Script)란 다른 프로그램에 의해 번역되어 수행되는 프로그램이나 명령어들의 나열을 말한다. 일반적으로 스크립트 언어들은 C 나 C++과 같은 언어들에 비해 쉬우며 빠르게 작성할 수 있어, 제한된 응용분야의 프로그램들을 만드는데 매우 적당하며, 많은 소프트웨어들은 자신들이 제공하는 기능의 확장 및 사용자의 편의성을 위해서 스크립트 언어를 지원하고 있다.

하지만, 이러한 스크립트들을 관리하는 것은 간단하지 않다. 특히 컴퓨터 시스템이나 네트워크의 취약성을 탐지하는 취약성 탐지 시스템의 일부분으로 작성되는 스크립트는 다양한 검증을 거치고 그들의 특성 및 경향도 파악되어 서버내의 데이터베이스에 보관되고, 설치되어 사용되기 때문이다.

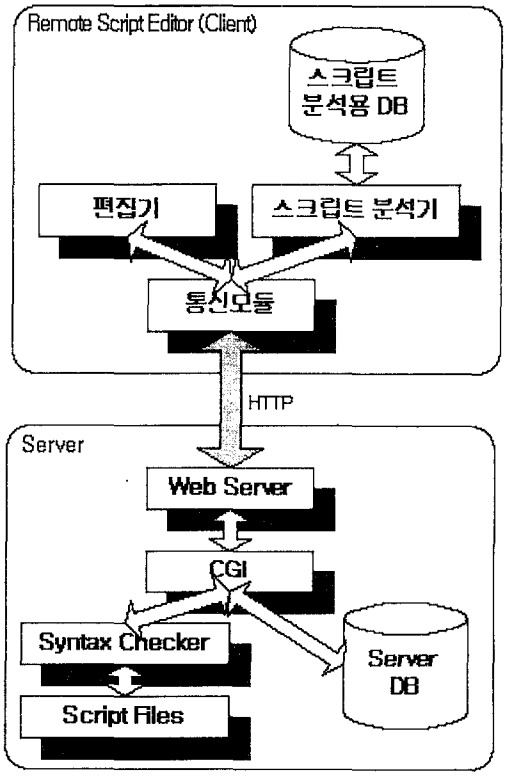
따라서, 본 논문에서는 취약성 탐지 도구에서 사용되는 취약성 탐지 스크립트 관리방법의 어려움을 극복하기위해 클라이언트 시스템에서 HTTP 를 통해 서버 데이터베이스에 설치되어있는 NASL[1] 스크립트를 효과적으로 관리하는 스크립트 편집기를 설계 및 구현하였다. 본 논문의 구성은 다음과 같다. 2 절에서는 스크립트 원격 편집기의 각 구성요소에 대해서 기술하고, 3 절에서는 스크립트 원격 편집기의 구현을 기술하며, 4 절에서는 연구의 결과 분석과 향후 계획에 대해 기술한다.

2. 스크립트 원격 편집기의 구성

[그림 1]은 스크립트 원격 편집기의 구조를 보여 주고 있다. 서버 시스템과의 통신은 서버 시스템이 일반 사용자에게 제공하는 웹서비스, tcp/80 을 이용하여 이루어지며 프로토

[1] Nessus Attack Scripting Language 의 약자로 "Nessus" Project (http://www.nessus.org)에서 제공되는 Security Scanner, "Nessus" 의 스크립트 언어이다.

콜은 HTTP 를 사용하였다. 웹서버에서 가상호스트(Virtual Host)기능을 제공하는 경우에는 일반 사용자와 분리된 별도의 연결채널을 구성할 수도 있다[3].



[그림 1] 스크립트 원격 편집기의 구조

2.1. 편집기 (Editor)

편집기는 취약성 탐지엔진이 구동되는 서버의 취약성 데이터베이스로부터 검색하여 다운로드 받은 스크립트 파일들을 편집하는 기능을 제공한다. 편집기의 가장 큰 특징으로 스크립트 문법에 대한 오류를 최소화하기 위해서 Code Insight[2] 기능을 포함하였다[6].

2.2. 통신모듈 (Communication Module)

통신모듈은 편집기와 서버 시스템 사이의 제반 통신을 제공하며 HTTP 를 사용한다. 원격 편집기는 서버의 웹서버에게 요청을 전송하고 그 결과를 전달 받는다. 스크립트 파일을 서버에 전송할 때는 *HTTP/PUT* 메소드를 사용한다[4].

[2] 특정 언어의 문법환경에 적합한 함수 또는 parameter(명령인자)의 형식을 자동으로 생성하는 기능.

2.3. 스크립트 분석기 (Script Analyzer)

취약성 탐지 스크립트에서 사용되는 함수는 스크립트의 특성을 분류하는데 매우 중요한 단서를 제공한다. 예를 들면, 기 설치된 스크립트와 새로 설치되는 스크립트의 차이점 등을 알아낼 수 있다[6]. 따라서, 스크립트 분석기는 스크립트의 소스를 분석하여 특정한 함수의 호출 여부와 함수의 호출 순서, 스크립트의 성격 분류 등을 찾아내어 스크립트들의 특성 및 경향을 파악하고 취약성 탐지 정책을 수립하는 데 유용한 자료로 사용토록 한다.

2.4. 문법 검사기 (Syntax Checker)

문법 검사기는 사용자가 수정하거나 새로 작성한 스크립트에 대해서 그 문법적 정확성과 그의 실행 가능성 및 유효성을 판단하는데 사용된다. 이 문법 검사기는 "Nessus" Project 에서 제공하는 Nessus 의 인터프리터 Source Code 를 수정하여 제작했다[1]. 이 문법 검사기는 취약성 서버가 구동되는 서버에 위치하며, 원격 편집기에서 사용자가 특정 스크립트에 대해서 문법 검사를 요청하면 서버측 CGI 모듈이 그 요청을 받고 문법 검사기를 실행한다.

2.5. 서버 CGI (Server-Side CGI)

사용자의 요청에 따라서 클라이언트로 스크립트 목록 또는 특정 스크립트를 전송하거나, 클라이언트로부터 받은 스크립트 파일을 서버에 설치하는 등의 기능을 수행한다. 서버측 CGI 는 HTTP 를 통하여 웹서버에 전달되는 원격 편집기의 요청을 실질적으로 처리하는 부분이다. 본 시스템에서의 CGI 는 모두 PHP 를 이용했다[2,3]. HTTP 의 특성상 서버측 CGI 는 익명 사용자가 웹브라우져 등을 통해서 직접 접근할 수 있는 위험성이 있어 이러한 위험성을 제거하고 시스템 관리자(Administrator)의 접근만을 허용하기 위해서 Cookie 를 사용한 인증(Authentication)을 사용하였으며 Password 와 같은 특정 Cookie 값은 MD5 방식으로 암호화 되도록 하였다[5].

3. 스크립트 원격 편집기의 구현

3.1. 구현환경

스크립트 원격 편집기는 Windows 2000 Professional 상에서 Delphi 5.0 으로 구현했으며, 서버 시스템은 Redhat Linux 6.0 상에서 Apache 1.3, PHP 4.1.1, Oracle 8 등으로 구현했다.

3.2. 클라이언트와 서버측 CGI 간의 프로토콜

클라이언트에서 서버로 전달되는 메시지의 구조는 다음

의 표 1 과 같다.

구분	메시지 구조
HTTP/GET	http://서버주소/script_editor/main.php? command=COMMAND& wparam=WPARAM& lparam=LPARAM
HTTP Cookie	user_id=ID password=MD5(PASSWORD)

[표 1] 클라이언트에서 서버로 전달되는 메시지 구조

다음의 표 2 는 사용 가능한 COMMAND 의 종류와 그에 따른 WPARAM, LPARAM 의 값을 표시한다.

COMMAND	WPARAM	LPARAM
COMMAND_LOGIN	없음	없음
COMMAND_LIST_SCRIPT	없음	없음
COMMAND_GET_SCRIPT	파일명	없음
COMMAND_PARSE_SCRIPT	파일명	없음
COMMAND_DELETE_SCRIPT	파일명	없음
COMMAND_INSTALL_SCRIPT	파일명	위험도

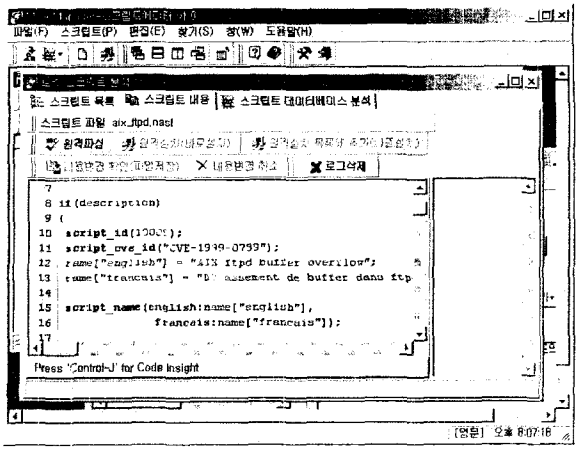
[표 2] 클라이언트에서 서버로 전달되는 COMMAND 의 종류

CGI 모듈은 원격 편집기로부터 요청을 받으면 우선 HTTP Cookie 에 포함된 user_id, password 를 사용하여 서버시스템의 관리자 권한이 있는지 검사하여 COMMAND, WPARAM, LPARAM 값을 분석한 후 각 COMMAND 에 따르는 작업을 수행한다. 실행 결과는 다시 편집기로 전달되며, 그 때 사용되는 메시지의 구조는 다음의 [표 3] 와 같다.

구분	메시지 구조
HTTP Cookie	RESULT=SUCCESS 또는 RESULT=FAILURE 또는 RESULT=QUERY
HTTP Body	응답 문자열이 포함됨. (Plain Text)

[표 3] 서버에서 클라이언트로 전달되는 메시지 구조

위와 같은 응답을 받은 편집기는 HTTP Cookie 의 RESULT 값을 조사하여 자신이 요청한 작업의 성공여부를 판단하며 사용자에게는 HTTP Body 에 포함된 문자열을 보여준다. 서버에서 돌아오는 메시지의 HTTP Body 를 적절히 활용해서 스크립트 목록을 다운로드 받는 것과 특정 스크립트를 다운로드 받는 등의 기능을 구현했다. [그림 2]는 원격 편집기의 실행모습을 보여주고 있다.



[그림 2] 스크립트 원격 편집기의 실행모습

4. 결론 및 향후과제

본 논문에서는 원격 취약성 탐지 스크립트의 원격 편집기를 HTTP 기반으로 구현하였다. 편집기의 스크립트 특성분석 기능으로 인하여 스크립트의 분석 및 취약성 데이터베이스와 연동이 가능하여 스크립트의 관리가 매우 수월해졌다. 하지만, HTTP/PUT 메소드를 사용하면서 발생하는 보안의 취약성과 취약성 탐지 시스템이 구동되고 있는 동안 본 원격 스크립트 편집기를 통해서 스크립트에 변화를 가할 경우 생기는 동기화 문제 등을 해결하여 실용성을 높이는 작업을 진행중이다.

5. 참고문헌

[1] Renaud Deraison, "The Nessus Project", http://www.nessus.org.
[2] The PHP Group, "PHP Manual", http://www.php.net.
[3] The Apache Software Foundation, "Apache HTTP Server Project", http://www.apache.org.
[4] RFC 2616, "Hypertext Transfer Protocol -- HTTP/1.1", Network Working Group, 1999.
[5] RFC 2617, "HTTP Authentication: Basic and Digest Access Authentication", Network Working Group, 1999.
[6] Steve Teixeira, Xavier Pacheco, "Delphi5 Developer's Guide", SAMS, 1999.