이용 난이도에 따른 취약점 평가 방법[1)]

김동현*, 송주석*
*연세대학교 컴퓨터과학과
e-mail:setflag@emerald.yonsei.ac.kr

Vulnerability assessment by the difficulty of exploitation

Dong-Hyun Kim*, Joo-Seok Song*
*Dept of Computer Science, YonSei University

요약

프로그램들은 밝혀지거나 잠재적인 취약점을 가지고 있으며, 취약점 데이터베이스는 취약점에서 얻어진 정보를 분류하여 저장하고 있다. 그러나 취약점 데이터베이스의 정보만으로는 취약점을 이용하기 위한 난이도에 대한 평가를 할 수가 없다. 본 논문에서는 취약점의 여러 속성의 조합을 통한 이용 난이도에 따른 취약점의 평가 방법을 제시한다. 이용 난이도가 높을수록 그 취약점을 이용한 공격이 어려울 것이며, 이에 따라 공격자의 수준을 나눌 수 있다.

1. 서론

우리가 사용하는 응용 프로그램이나 운영체제는 보안에 관련된 밝혀진 혹은 잠재적인 결점들을 가지고 있다. 취약점으로 알려진 이 결점들은 정보의 비밀성, 무결성, 가용성에 큰 영향을 미치게 된다[1]. 각 국의 CERT (Computer Emergency Response Team), 보안 회사의 게시판 및 운영체제와 응용 프로그램의 개발 회사에서는 이러한 취약점들을 분석, 보고하여 취약점들로 인한 피해를 최소화하려는 노력을 한다. ISS(Internet Security Systems, Inc.), SecurityFocus.com에서는 체계적이고 통계적인 취약점 보고를 위한 취약점 데이터베이스를 운영하고 있으며, NIST (National Institute of Standards and Technology)에서는 산재해있는 취약점 데이터베이스를 일괄적으로 참조할 수 있는 취약점 metabase를 운영하고 있다[2][3][4]. 그러나 이러한 취약점 데이터베이스들은 모두 취약점의 보고를 목적으로 하고 있기 때문에, 간단한 취약점의 내용만을 담고 있다. 이러한 데이터베이스의 내용으로는 각 취약점의 공격자의 능력 수준에 따른 취약성의 이용 가능성을 평가할 수 없다는 문제점이 있다.

본 논문에서는 취약점이 가진 속성들의 조합에 따라 공격자가 각 취약점을 이용하는 난이도에 대한 취약점 평가 방법을 제시한다. 제시하는 평가 방법은 기존의 보고의 목적을 갖는 취약점 데이터베이스의 성격과는 달리 시뮬레이션에 이용될 수 있는 성격을 갖는다. 사이버 공격에 대한 목표 시스템의 피해를 측정하기 위한 시뮬레이션은 공격자에 따라 이용되는 공격 방법이 달라지며, 공격의 결과 또한 달라지게 된다[5]. 전문 해커들은 스크립트 키드들이 할 수 없는 많은 공격을 할 수 있으며, 목표 시스템에 더 큰 영향을 미칠 수 있다. 공격은 일반적으로 시스템의 취약점을 이용하게 된다. 공격자의 수준에 따라서 이용할 수 있는 취약점이 정해지게 되는데, 이용 가능한 취약점은 취약점의 여러 가지 속성으로 그 수준을 정할 수 있다. 어떤 취약점을 이용하여 공격을 하는 것이 어렵다면 그만큼 높은 수준의 공격자만이 그 취약점을 이용하여 공격하는 것이 가능할 것이다. 따라서, 취약점의 이용 난이도에 따라서 이 취약점을 이용할 수 있는 공격자의 수준이 정해지게 된다.

본 논문의 구성은 다음과 같다. 2장에서는 취약점 분석과 사용하는 용어에 대한 간단한 설명을 한다. 3장에서는 본 논문에서 제시할 방법에 사용되는 취약점 속성에 대해서 설명한다. 4장에서는 취약점을 이

1) 본 연구는 한국정보보호진흥원 위탁과제(취약성DB 확대 구축)로 수행되었습니다.

용하는 난이도에 대한 취약점 평가 방법을 제시한다. 5장에서는 제안한 평가 방법에 대해서 분석을 한다. 6장에서는 결론과 향후 연구 방향을 제시한다.

2. 취약점 분석

취약점 분석은 많은 취약점들로부터 얻어지는 정보를 분류할 수 있는 방법을 마련하는데 목적이 있다. 이러한 정보들은 침입 탐지의 시그너쳐(signature), 공격자가 취약점을 이용하기 위한 정보 등으로 이루어진다[6]. 각 취약점 데이터베이스들은 취약점들을 분석하여 얻어진 정보들을 적당한 분류 기준을 세워 체계적으로 취약점들을 분류하고 있다. 기본적으로 취약성 분류, 공격자의 공격이 가능한 위치 및 권한, 취약성 결과, 취약한 시스템 등과 같은 정보를 담고 있다. 이러한 정보들을 취약점의 특징, 속성이라 할 수 있다.

본 논문에서 사용할 용어들에 대해서 다음과 같이 정의를 한다.

- vulnerability(취약점) : 공격자에 의해서 악용될 수 있는 프로그램의 결점
- vulnerability type(취약점 유형) : 취약점 분류에 사용되는 취약점의 근본 원인
- attack(공격) : 시스템의 보안 정책을 위배, 침해하는 행위
- exploit : 취약점을 이용하기 위한 공격 방법
- privilege : 특정 시스템에 대한 권한
- location : 공격자의 시스템 상의 공격위치
- Impact : 시스템에 대한 공격의 결과

3. 고려할 취약점 속성

취약점을 이용하기 위한 난이도는 크게 두 가지 측면에서 생각할 수 있다. 첫 번째는 대중성으로써 이것은 취약점이 얼마나 자세하게, 얼마나 많은 정보를 공개하고 있는가를 나타낸다. 취약점에 대한 정보가 많다는 것은 그 취약점에 대해서 자세하게 알 수 있으므로 큰 어려움 없이 취약점을 이용할 수 있다는 것을 나타낸다. 예를 들어, '특정한 길이의 문자열을 입력하라'와 '512 byte 길이의 문자열을 입력하라'라는 것을 생각하면, 당연히 후자가 이용하기가 쉽다. 정보의 양이 많을수록 난이도는 낮아지게 된다. 대중성 측면에서 고려할 수 있는 취약점 속성은 다음과 같다.

- vulnerability type
  - 취약점 유형은 취약점 데이터베이스마다 다르게 정의하고 있다. 취약점 유형은 근본 원인을 설명해주므로, 취약점 유형에 대해서 안다면 그 취약점을 이용할 수 있는 기본적인 원리를 알 수 있다. 같은 유형을 갖는 취약점들은 서로 공통점을 갖게 되므로, 어떤 유형의 취약점들이 많이 보고가 된다면 그 유형에 속하는 취약점들의 대중성은 높아지게 되고, 난이도는 낮아지게 된다.
- exploit 공개
  - 어떤 취약점에 대해서 공격 방법이 알려진다면 그 취약점의 이용 난이도는 급격히 낮아지게 된다. 또한 공격이 가능한 소스코드나 스크립트가 공개되었을 경우도 생각해 볼 수 있다.

두 번째로 단순성을 생각할 수 있다. 취약점을 얼마나 간단하게 이용할 수 있는가를 나타낸다. 예를 들어 같은 결과를 나타내게 하는 방법으로 소스코드를 컴파일해서 실행하는 것보다 쉘 스크립트가 사용하기 더 쉽고, 쉘 스크립트보다 명령어를 사용하는 것이 더 쉽다. 단순성 측면에서 다음과 같은 취약점 속성을 생각할 수 있다.

- exploit type
  - 공격 방법은 크게 프로그램 언어의 소스코드 형태와 명령어 조합의 두 가지로 나눌 수 있다. 필요에 따라서 컴파일을 하거나, 특정 프로그램이 필요한 전자의 경우보다 그런 것이 필요 없는 후자가 공격 방법이 더 간단하다.
- location and privilege
  - 어떤 취약점을 이용하기 위해서는 다음의 3가지 정도의 위치와 권한의 관계가 필요하다.

remote & user : 공격자는 목표 시스템과 다른 시스템에 있으며, 현재 일반 사용자의 권한이다.
remote & root : 공격자는 목표 시스템과 다른 시스템에 있으며, 현재 루트의 권한이다.
local & user : 공격자는 목표 시스템에 있으며, 현재 일반 사용자의 권한이다.

취약점을 이용하기 위한 단순성은 차례대로 낮아지게 된다.

4. 제안하는 평가 방법

취약점의 이용 난이도는 대중성과 단순성 값의 합으로 나타낸다. 값이 클수록 난이도는 낮아지고, 값이 작을수록 난이도는 높아지게 된다. 대중성과 단순성은 3장에서 살펴본 취약점 속성에 따라 값을 얻게 된다.

4.1 대중성(popularity)

$$P(popularity) = VT + ED + \alpha$$

- VT(vulnerability type)

vulnerability type	value
buffer overflow (input validation error)	5
race condition	3
access validation	2
other type	1

· ED(exploit 공개)

공개 수준	value
비공개	0
일반적 공개	3
소스코드 수준의 공개	7

· α(가중치)

이 값은 일반적인 취약점보다 더 많은, 더 자세한 정보가 공개될 경우 $0<\alpha\leq5$의 값으로 정한다. 일반적으로 CERT의 권고문보다는 사고노트나 기술문서가 자세한 정보를 기록하고 있다. 취약점에 해당하는 사고노트나 기술문서가 있을 경우 가중치를 부여하게 된다. 가중치의 값은 공개되는 정보의 수준에 따라 정한다.

4.2 단순성(simplicity)

S(simplicity) = ET + LP - β

· ET(exploit type)

exploit type	value
소스코드(특정 조건 필요)	1
소스코드(조건 불필요)	3
명령어 조합 & GUI 프로그램	5

특정 조건이 필요하지 않은 경우는 쉘 스크립트나 c 코드처럼 시스템에서 일반적으로 사용되는 소스코드 유형을 나타낸다. 명령어 조합의 경우는 윈도우 환경에서 사용할 수 있는 GUI 프로그램까지를 포함한다.

· LP(location & privilege)

location & privilege	value
remote & user	6
remote & root	4
local & user	1

· β(페널티)

이 값은 공격자가 공격을 할 때, 직접적으로 느낄 수 있는 번거로움의 정도를 $0<\beta\leq3$의 값으로 정한다. exploit 소스코드가 특정한 헤더파일을 요구하거나, 일정 수준 이상의 시간을 요하는 경우를 들 수 있다.

4.3 취약점 이용 난이도

DVE(Difficulty of Vulnerability Exploitation) = P + S

취약점 이용 난이도는 대중성 P와 단순성 S로 값을 정할 수 있으며, $0\leq DVE\leq28$의 값을 가질 수 있다. 값이 높을수록 난이도는 낮은 것이며, 값이 낮을수록 난이도는 높아서 취약점을 이용하기가 어렵게 된다.

5. 분석 및 고찰

5.1 UNICODE 취약점

unicode관련 취약점은 작년에 "확장 UNICODE character를 이용한 MicroSoft IIS 4.0/5.0 웹서버 내부 명령어 실행"이란 제목의 권고문으로 발표되었다[7].

· 대중성

VT = 5 (Input validation error)[4]
ED = 7 (공격이 가능한 정도의 공격방법 공개)[8]
α = 3 (사고 노트에 자세한 정보 공개)[8]
P = VT(5) + ED(7) + α(3) = 15

· 단순성

ET = 5 (명령어 조합)
LP = 6 (remote & user)
β = 2 (명령어 조합이 길이가 길고 복잡하다.)
S = ET(5) + LP(6) - β(2) = 8

· 이용 난이도

DVE = P(15) + S(8) = 23

5.2 Solaris sadmind의 버퍼오버플로우 취약점

sadmind관련 취약점은 작년에 "Solaris sadmind의 버퍼오버플로우 취약점"이란 제목의 권고문으로 발표되었다[9].

· 대중성

VT = 5 (Buffer Overflow)[4]
ED = 7 (소스코드 수준의 공개)[3]
α = 2 (사고 노트에 자세한 정보 공개)[10]
P = VT(5) + ED(7) + α(2) = 14

· 단순성

ET = 3 (c언어 소스코드 존재)[3]
LP = 6 (remote & user)
β = 2 (소스코드를 컴파일 해야한다.)
S = ET(3) + LP(6) - β(2) = 7

· 이용 난이도

DVE = P(14) + S(7) = 21

5.3 이용 난이도에 따른 취약점 평가

위의 두 취약점은 23, 21이라는 비교적 낮은 이용

난이도를 보인다. 이용 난이도가 낮다는 것은 공격방법이 쉬우므로 많은 공격이 이루어질 수 있음을 나타낸다. 위의 두 취약점은 현재 많은 공격에 사용되고 있음을 CERT의 통계 및 사고노트에서 확인 할 수 있다.

6. 결론 및 향후 연구

본 논문에서는 이용 난이도에 따른 취약점 평가 방법을 제시하였다. 이용 난이도는 취약점 이용에 대한 대중성과 단순성의 값을 정하여 계산하였다. 대중성과 단순성은 취약점의 여러 속성의 조합으로 이루어진다. 난이도가 낮을수록 취약점에 대한 이용 빈도가 높아져, 이 취약점을 이용한 많은 공격이 이루어짐을 알 수 있다.

향후 연구에서는 취약점 이용 난이도와 공격자의 수준을 연관시키는 부분의 연구가 필요하다. 또한, 대중성의 경우 시간에 따라 증가하는 경향을 보이지만, 어느 시점을 지나면 다시 감소하는 경향을 보인다. 이러한 취약점의 발견이후 시간 경과에 따른 난이도 변화에 대한 연구도 고려되어야 할 것이다.

참고문헌

[1] Iaakso M., Takanen A., Röning J., "The vulnerability process: a tiger team approach to resolving vulnerability cases", In proceedings of the 11th FIRST Conference on Computer Security Incident Handling and Response. Brisbane. 13-18 June, 1999.

[2] Internet Security Systems, Inc. X-Force database <http://xforce.iss.net/>

[3] SecurityFocus.com, <http://www.securityfocus.com/>

[4] ICAT Metabase <http://icat.nist.gov/>

[5] F. B. Cohen, "Simulating Cyber Attacks, Defenses, and Consequences," Available at http://all.net/journal/ntb/ simulate/simulate.html, May 13, 1999.

[6] M. Bishop, ""Vulnerabilities Analysis," Proceedings of the Recent Advances in Intrusion Detection pp. 125-136 Sep, 1999

[7] CERTCC-KR 권고문 KA-2000-039 : 확장 UNICODE character를 이용한 MicroSoft IIS 4.0/5.0 웹서버 내부 명령어 실행. Available at http://www.certcc.or.kr/advisory/ka2000/ka2000-039.txt

[8] 사고노트 CERTCC-KR-IN-01-005 : 미국-중국 사이버공격에 따른 국내 IIS 웹서버 해킹증가. Available at http://www.certcc.or.kr/paper/incident_note/2001/in2001_005.html

[9] CERTCC-KR 권고문 KA-2000-03 : Solaris sadmind Buffer Overflow. Available at http://www.certcc.or.kr/advisory/ka2000/ka2000-003.txt

[10] 사고노트 CERTCC-KR-IN-01-006 : 솔라리스 sadmind Worm 확산. Available at http://www.certcc.or.kr/paper/incident_note/2001/in2001_006.pdf