情報保護學會論文誌
第 22 卷 第 5 號, 2012. 10

보안기능의 무력화 공격을 예방하기 위한 위협분석 기반 소프트웨어 보안 테스팅*

김 동 진,[1†] 정 윤 식,[1] 윤 광 열,[1] 유 해 영,[1] 조 성 제,[1‡] 김 기 연,[2] 이 진 영,[3] 김 홍 근,[3] 이 태 승,[4] 임 재 명,[3] 원 동 호[4]
[1]단국대학교, [2]에이쓰리시큐리티, [3]한국인터넷진흥원, [4]성균관대학교

Threat Analysis based Software Security Testing for preventing the Attacks to Incapacitate Security Features of Information Security Systems*

Dongjin Kim,[1†] Youn-Sik Jeong,[1] Gwangyeul Yun,[1] Haeyoung Yoo,[1] Seong-Je Cho,[1‡] Giyoun Kim,[2] Jinyoung Lee,[3] Hong-Geun Kim,[3] Taeseung Lee,[4] Jae-Myung Lim,[3] Dongho Won[4]
[1]Dankook University, [2]A3SECURITY, [3]Korea Internet Security Agency, [4]Sungkyunkwan University

요 약

정보보안시스템을 무력화하는 공격이 나타남에 따라, 정보보안제품의 취약성을 분석하는 보안 테스팅에 대한 관심이 높아지고 있다. 보안제품 개발의 주요 단계인 침투 테스팅은, 공격자가 악용할 수 있는 취약성을 찾기 위해 컴퓨터 시스템을 실제적으로 테스팅하는 것이다. 침투 테스팅과 같은 보안 테스팅은 대상 시스템에 대한 정보 수집, 가능한 진입점 식별, 침입 시도, 결과 보고 등의 과정을 포함한다. 따라서 취약성 분석 및 보안 테스팅에서 일반성, 재사용성, 효율성을 극대화하는 것이 매우 중요하다. 본 논문에서는, 정보보호제품이 자신의 보안 기능을 무력화하거나 우회하는 공격에 대응할 수 있는 자체보호 기능 및 우회불가성을 제공하는 가를 평가할 수 있는 위협분석 기반의 소프트웨어 보안 테스팅을 제안한다. 위협분석으로 취약성을 식별한 후, 보안 테스팅의 재사용성과 효율성을 개선하기 위해 소프트웨어 모듈과 보안 기능에 따라 테스팅 전략을 수립한다. 제안기법은 위협 분석 및 테스팅 분류, 적절한 보안테스팅 전략 선정, 보안 테스팅으로 구성된다. 사례연구와 보안 테스팅을 통해 제안 기법이 보안 시스템을 체계적으로 평가할 수 있음을 보였다.

ABSTRACT

As attackers try to paralyze information security systems, many researchers have investigated security testing to analyze vulnerabilities of information security products. Penetration testing, a critical step in the development of any secure product, is the practice of testing a computer systems to find vulnerabilities that an attacker could exploit. Security testing like penetration testing includes gathering information about the target before the test, identifying possible entry points, attempting to break in and reporting back the findings. Therefore, to obtain maximum generality, re-usability and efficiency is very useful for efficient security testing and vulnerability hunting activities. In this paper, we propose a threat analysis based software security testing technique for evaluating that the security functionality of target products provides the properties of self-protection and non-bypassability in order to respond to attacks to incapacitate or bypass the security features of the target products. We conduct a security threat analysis to identify vulnerabilities and establish a testing strategy according to software modules and security features/functions of the target products after threat analysis to improve re-usability and efficiency of software security testing. The proposed technique consists of threat analysis and classification, selection of right strategy for security testing, and security testing. We demonstrate our technique can systematically evaluate the strength of security systems by analyzing case studies and performing security tests.

Keywords: Vulnerability, Security Testing, Threat Analysis, Penetration Testing, Self-protection, Non-bypassability

접수일(2012년 9월 11일), 수정일(2012년 10월 18일), 게재확정일(2012년 10월 18일)

* 본 연구는 2012년 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행되었고(2012-0007372), 정보통신산업진흥원의 SW공학 요소기술 연구개발사업의 결과물임을 밝힙니다.

† 주저자, kdjorang@dankook.ac.kr
‡ 교신저자, sjcho@dankook.ac.kr

I. 서 론

소프트웨어(SW) 보안사고가 증가하고 있으며, 한 때는 사이버 공격의 약 75%가 SW의 보안 취약점을 악용한 것으로 나타났다[1]. 이러한 사이버 공격에 따른 경제·사회적 손실과 관심이 급증함에 따라, SW가 오류 및 보안 취약점을 갖지 않고 안전하게 정상으로 동작함을 보장하는 SW 보증에 관한 연구가 활발히 수행되고 있다[2,3,4,5]. 더불어 최근 DDoS (Distributed Denial of Service) 공격과 관련된 악성코드가 SW기반 정보보호제품들의 보안 기능을 무력화 시키고 이를 우회함에 따라, 다른 일반제품들과는 달리 SW기반 정보보호제품의 보안기능이 우회되지는 않는지, 그리고 자체보호 기능이 제공되는지에 대한 평가가 요구되고 있다.

정보보호제품의 평가 및 인증은 국제적으로 공통평가기준(Common Criteria, CC)에 의해 제도화되어 운영되고 있다. 공통평가기준의 취약성 평가 (Vulnerability assessment) 클래스에서 정보보호제품 평가 시, 자체보호(Self-protection) 기능과 우회불가성(Non-bypassability)에 대한 평가가 필요하다.

정보보호제품의 안전성을 평가하는 보안 테스팅 (Security testing)은 일반적인 기능 테스트 (Functional testing) 보다 더 많은 자원과 시간이 필요하다[6,7]. 따라서 SW 제품들의 취약성 평가에서는 공개된 취약점과 기존 보안사고 등을 분석하여 위협을 도출 및 정의하며, 이를 바탕으로 효과적인 테스팅 기법 및 전략을 선정하여 테스팅 비용을 줄이는 것이 필요하다[8,9]. 이러한 과정은 새로운 보안 공격에 대응하기 위해 빈번히 업데이트되고 업데이트 시마다 수행되어야하는 정보보호제품의 취약성 평가에도 필요하다.

일반 SW의 보안 테스팅은 소스코드 상에서의 오류나 입력 데이터에 의한 취약점 존재 여부 및 스트레스 테스팅을 주로 수행하는 반면, SW기반 정보보호제품 테스팅은 자체보호 기능 등을 포함한 침투 테스팅 (Penetration testing)을 수행하기 때문에 보안 테스팅의 방법 및 목적에서 차이가 있다. 하지만 SW기반 정보보호제품에 특화된 취약성 평가를 위한 보안 테스팅 기법 및 절차에 대한 기존 연구는 매우 미비하다. 보안제품 개발의 주요 단계인 침투 테스팅은, 공격자가 악용할 수 있는 취약성을 찾기 위해 컴퓨터 시스템을 실제적으로 테스팅하는 것이다. 침투 테스팅과 같은 보안 테스팅은 대상 시스템에 대한 정보 수집, 가능한 진입점 식별, 침입 시도, 결과 보고 등의 과정을 포함한다. 따라서 보안 테스팅과 취약성 분석 과정에서 일반성, 효율성, 재사용성을 극대화하는 것이 매우 중요하다.

본 논문에서는 SW기반 정보보호제품의 취약성 평가에 적합한 보안 테스팅 절차에 대해 제안한다. 제안 절차는 보안 테스팅 목적에 적합한 위협 분석 및 분류, 그리고 보안 테스팅 전략 선정 및 보안 테스팅 수행으로 구성된다. 중점적인 연구 목적은 정보보호제품의 자체보호 기능과 우회불가성과 관련된 취약성 평가 및 위협분석 기반 보안 테스팅이다. 이 두 가지 평가 목적별에 따라 정보보호제품 관련 공개 취약점과 공격 사례를 중심으로 위협을 분석하였다. 위협분석 기반 보안 테스팅은 정보보호제품 개발 초기 단계부터, 개발자 및 내부 보안 테스터에 의해서도 수행될 수 있다는 점에서 기존 침투 테스팅과 차이가 있다. 또한, 제안하는 기법은 자체보호 및 우회불가성에 대한 기존 취약점 및 공격 사례를 분석한 결과, 자체보호 기능에 대한 공격의 대상은 대부분 SW 모듈이고, 우회불가성에 대한 공격 대상은 제품의 보안 기능이었다. 이를 기준으로 위협과 각 위협에 해당하는 보안 테스팅 전략을 분류하였으며, 이를 통해 테스팅 기법의 재사용성과 효율성을 높일 수 있다. 기존 취약점 사례를 분석하고 정보보호제품군별 대상을 선정하여 실험을 통해, 본 논문에서 제안한 위협 기반의 보안 테스팅 절차의 활용 가능성, 테스팅의 재사용성 및 효율성을 검증하였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 보안성 및 취약성 평가 방법을 소개하고, 3장에서는 위협 기반 보안 테스팅의 효율성, 본 논문의 주안점에 대해 기술한다. 4장에서는 제안한 보안 테스팅 절차에 대해 서술하고, 5장과 6장에서는 제안 기법의 효율성을 검증한다. 마지막으로 7장에서는 결론 및 향후 연구방향을 제시한다.

II. 취약성 평가

2.1 CVSS, CWSS

SW 제품에 존재하는 보안 취약점(Vulnera-bili-ty) 및 약점(Weakness)을 기준으로 취약성을 평가할 수 있다. CVSS(Common Vulnerability Scoring System)[10]와 CWSS(Common We-ak-

ness Scoring System)[11]는 각각 취약점과 약점의 심각도(severity)를 측정하여 점수화하는 체계로, CVE(Common Vulnerabilities Enumeration)[12]와 CWE(Common Weakness Enumeration)[13]의 일부로 사용된다. CVSS는 이미 발생한 취약점을 대상으로 하기 때문에 제품의 취약성 평가를 위해서는 잠재적인 약점들의 심각도를 점수화하는 CWSS가 취약성 평가에는 더 적합하다.

CWSS가 심각도 점수화를 위해 사용하는 총 18개의 요소들이 공격자가 약점을 악용하기 위해 거쳐야하는 공격표면(Attack surface)을 포함하지만, 실제 취약점 정보를 고려하지는 않는다. CWSS의 공격표면 정보는, 일반적 개념으로 각 제품에 대해 침투 테스팅 등의 보안 테스팅을 수행하기에는 불충분하므로, 보안 테스팅에는 이미 발생한 보안사건에 대한 취약점 정보가 필요하다. 기존연구에서도 유사한 목적 및 기능의 제품 간에 동일 취약점이 나타나는 경우가 있어[8,9], 보안상 안전한 SW 제품을 개발하기 위해서는 관련된 취약점도 함께 고려해야 한다.

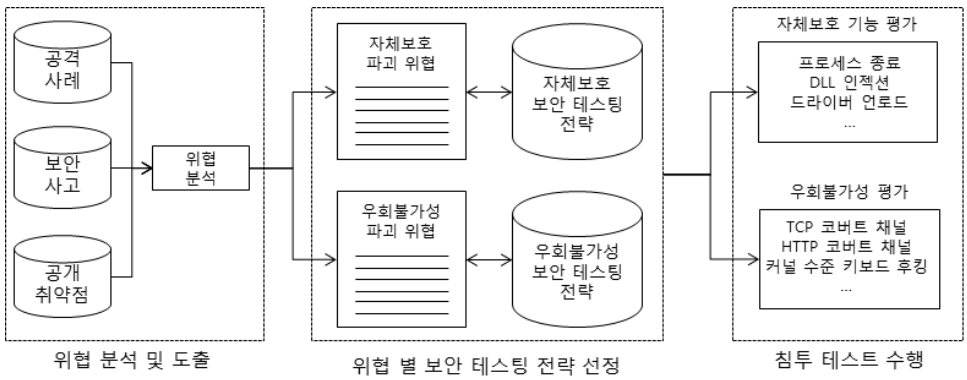
2.2 CC

CC는 정보보호제품에 대한 보안성을 평가 및 보증하기 위한 공통평가기준으로 ISO 15408 표준이다. CC는 1~7단계까지의 보증 등급(Evaluation Assurance Level, EAL)을 제공하며 정보보호제품의 SW 개발 생명주기에 따른 각종 산출물을 평가·인증하는 체계로, '취약성 평가' 클래스를 별도로 제시하고 있다. 특히 CC 3.1 버전에서는 취약성 평가 클래스의 세부 항목들을 '취약성 분석'으로 통합하면서, 취약점 분석에 기반한 침투 테스팅과 같은 보안 테스팅을 더욱 강화하였다. CC 2.3 버전에서는 개발자가 취약점 분석을 수행한 후, 취약점 분석 결과를 바탕으로 침투 테스팅을 수행하고, 평가자는 요구사항을 만족하는지 확인하였다. 하지만 3.1 버전에서는 취약성 분석 및 식별부터 침투 테스팅까지 평가자가 수행하도록 하여, 취약성 평가 결과가 더 높은 객관성 및 신뢰성을 갖도록 하고 있다[14].

공통평가기준에서는 TOE(Target of Evaluat-ion)의 잠재적 취약성을 분석하고 식별하기 위해 평가자로 하여금 공개 영역에 대한 조사를 수행하도록 규정하였다. 이는 공개된 취약점 및 보안사고 등을 통해 평가대상 제품의 위협을 분석하고, 분석된 위협을 근거하여 침투 테스팅을 수행하는 것을 의미한다.

본 논문에서의 보안 테스팅은 CC(3.1 버전)에서의 취약성 평가 대상 중 개발 단계 취약성을 대상으로 한다. 개발 단계 취약성 평가에서는 TSF(TOE Security Function)를 침해하여 TSF의 자체보호(Self-protection)를 파괴하거나, TSF를 우회(Bypassing)하여 우회불가성(Non-bypassability)을 파괴하는 것이 가능한지 확인한다[14].

기존연구에서는 정보보호제품을 평가하기 위해, 자체보호 및 우회불가성 이외에도 제품의 구성요소인 OS, DB, 네트워크 관련된 취약점과 보안상 위험한 소스코드 패턴에 의한 취약점도 고려할 것을 제안하고 있다[15]. 소스코드 수준의 취약점은 구현단계에서 Fortify[16]와 같은 취약점 점검 도구를 사용하여 제거 및 예방할 수 있다[17,18]. 본 연구진은 선행연구로, IT 제품 및 서비스의 구성요소와 관련된 취약점 관리 체계[19,20]에 대해 연구한 적이 있다. 본 연구에서는 선행연구의 확장으로, 소스코드 취약점보다는 자체보호와 우회불가성에 대한 취약성 평가에 초점을 둔다.



[그림 1] 정보보호제품을 위한 위협 기반 보안 테스팅 절차

III. 위협 기반 SW 보안 테스팅

보안 테스팅을 수행하기 위해서, 테스팅 대상과 동일 혹은 유사 제품에 존재하는 기 공개된 취약점 정보, 유사 제품을 대상으로 한 공격 사례를 분석하는 것이 도움이 된다. 이전에 공개된 취약점과 공격 사례 분석을 활용한 보안 테스팅 기법의 유효성은 다양한 사례를 통해 확인되었다. 기존 연구[8,9]에서는 멀티미디어 재생프로그램의 보안 테스팅을 위해 기존 취약점들을 분석하여 효율적인 퍼징(Fuzzing) 전략을 도출하고 이를 검증하였다. 또한 특정 멀티미디어 재생프로그램의 취약점 분석에 사용되었던 공격 데이터를 다른 유형의 재생프로그램에 적용한 결과 다량의 유사 취약점을 발견하기도 하였다. 또한 [21] 연구에서는 웹 애플리케이션의 취약성 평가를 위해, 기존 공격 사례와 취약점 분석하여 보안 테스팅을 수행하였으며, 'Directory Traversal' 등의 취약점을 발견하였다.

최근 유명 온라인 게임의 계정탈취에 악용된 취약점은, 기존 다른 온라인 게임에서 사용되었던 유사한 공격방법에 의해 악용되었던 것으로 알려졌다[22]. 이는 보안제품의 취약성 평가 시에 기존 공격 사례가 적절히 분석되지 않았음을 의미하며, 기존 공격 사례를 분석하는 것이 테스팅에 필요함을 보여준다.

하지만 기 공개된 취약점 정보와 기존 공격 사례는 보안사고 후에 얻을 수 있는 대응 관점의 정보이므로, 예방 관점을 고려해야 하는 위협(Threat) 기반 보안 테스팅에 그대로 적용하는 것에 한계가 있다. SANS 자료(SANS Institute Reading Room Site)[23]에 의하면, 위협은 어떤 서비스나 아이템에 대해 변조·파괴·가로막기 등의 행위로 귀결될 수 있는 무언가(해커, 절도, 우연한 사고, 바이러스 등)이다. 위협 및 위험 평가의 목적은 기능성과 사용성(Usability)을 제공하면서 동시에 비밀성, 무결성, 가용성을 극대화하도록 권고하는 것이다. Eldan Software Systems사[24]에 의하면 실질적인 위협분석(Practical Threat Analysis, PTA)은 시스템과 애플리케이션을 안전하게 해 주는 가장 유익하며 비용-효과적인 위험 평가 방법론 및 도구이다. 실질적 위협분석 과정의 역할은 시스템 취약점을 식별, 시스템 자원을 사상, 위협의 위험을 평가, 효과적인 위험 완화 계획을 정의하는 것이다.

이를 반영하여 본 논문에서는 위협 기반 보안 테스팅을 위해 정보보호제품 평가 목적과 기능에 따라 위협을 분류·식별하고, 분류 결과에 맞게 테스팅 전략을 선정하는 과정을 도입한다. 또한, 대상 SW제품의 이전 버전에 대한 공개 취약점 정보나 유사 SW제품에 대한 공개 취약점 정보, 유사 SW제품에 알려진 공격 사례 등을 보안 테스팅에 접목한다. 공개 취약점 분석 및 기존 공격 사례를 통한 위협 기반 보안 테스팅 절차는 공통적인 장점을 갖는다. 유사한 기능의 SW 제품들은 동일한 위협에 노출되어 있기 때문에 위협 기반의 보안 테스팅 기법은 재사용성도 높고 또 효율성도 향상시킨다. 이러한 장점은 SW기반 정보보호제품의 취약성 평가에도 해당된다. 정보보호제품은 동일한 핵심 기능을 갖는 제품이 매우 다양하게 존재하는데, 기능이 유사하기 때문에 구현 기법도 유사하다. 또한 새로운 보안 공격에 대응하기 위해 자주 업데이트되기 때문에 취약성 평가도 매우 빈번하게 수행된다. 이러한 특징을 반영하여, 본 논문에서는 재사용성과 효율성을 고려한 위협 기반 보안 테스팅 절차를 제안한다.

IV. 정보보호제품 보안 테스팅 절차

본 논문에서 제안하는 위협 기반 보안 테스팅 절차에서는 취약성 평가 대상 제품과 동일 혹은 유사 제품에서 발견된 공개된 취약점과 기존 보안 사고에서의 공격 사례를 분석하여, 위협을 정의한다. 그리고 분석 결과를 자체보호 평가와 우회불가성 평가 목적에 적합하도록 분류한 후, 분류 결과를 기준으로 보안 테스팅 전략을 선정한다. [그림 1]은 제안하는 보안 테스팅 절차를 나타낸다.

4.1 위협 분석 및 도출

공개된 취약점은 미국 NIST(National Institute of Standards and Technology)에서 운영하는 국가 취약점 DB인 NVD(National Vulnerability Database)[25]를 중심으로 수집하였고, 기존 공격 사례와 보안 사고는 언론 매체와 정보보호 관련 기관의 보안 권고 사항, 악성코드 분석정보 등을 통해 수집하였다.

4.1.1 자체보호 기능에 대한 위협

자체보호 기능에 대한 평가는 정보보호제품이 외부 공격으로부터 자신을 보호하는 기능이 있는지를 검증하는 것이다. 자체보호 기능에 대한 공격은 대부분 제품의 구현 및 실행 상의 구조나 특징을 대상으로 한

〔표 1〕 정보보호제품 대한 위협: (a) 자체보호 기능에 대한 위협, (b) 우회불가성에 대한 위협

위협 분류		위협 대상 제품	설명
(a)	프로세스 기반	보안 기능 제공을 위해 프로세스 기반한 제품	프로세스를 강제 종료시킴으로써 제품 기능 무력화
	멀티쓰레드 기반	보안 기능 제공을 위해 쓰레드 기반한 제품	쓰레드를 강제 종료시킴으로써 제품 기능 무력화
	DLL 사용	보안기능을 DLL로 구현하여 적재하는 제품	DLL 변조를 통해 악성코드 실행 및 제품 기능 무력화
	드라이버 사용	시스템 수준의 보안기능 제공을 위해 드라이버를 사용하는 제품	드라이버를 언로드 하여 제품 기능 무력화
	서비스 실행	윈도우 상에서 서비스로 등록되어 실행되는 제품	서비스 실행 강제 종료 및 서비스 실행 파일 변조를 통해 제품 기능 무력화
	SSDT	윈도우에서 실행되는 모든 제품	SSDT 후킹을 통해 악성 코드 은닉 및 정보보호 제품 무력화
	업데이트 수행	새롭게 보고된 악성코드/행위를 탐지하기 위해 원격 서버와 업데이트하는 제품	업데이트 서버 주소 변조 및 업데이트 프로세스 강제 종료를 통해 업데이트 차단
	레지스트리 및 설정파일 사용	제품의 기능설정을 위한 설정 저장을 저장하는 제품	주요 보안기능 설정의 변조를 통해 제품 기능 무력화
	BHO 및 ActiveX 사용	웹브라우저를 통해 인터넷 사용과 관련된 보안기능을 제공하는 제품	ActiveX 실행 차단을 통해 제품 기능 무력화
(b)	패턴 기반 악성코드 탐지 우회	시그니처(signature) 및 패턴 기반으로 악성코드 탐지하는 제품	다양한 인코딩 기법을 통해 패턴을 숨겨서 우회
	행위 기반 악성코드 탐지 우회	프로세스, 파일, 레지스트리 등 시스템 상태 변화를 감지하여 악성코드 탐지하는 제품	탐지 시스템임을 감지하고 악성행위를 하지 않도록 하여 우회
	TCP 패킷 필터링 우회	내부 시스템 보호를 위해 TCP 패킷 필터링 기능을 수행하는 제품	TCP 프로토콜 중 데이터를 은닉할 수 있는 공간을 이용하여 우회
	HTTP 패킷 필터링 우회	웹 기반 공격 탐지 및 내부 시스템 보호를 위해 HTTP 패킷 필터링 기능을 수행하는 제품	HTTP 프로토콜 중 데이터를 은닉할 수 있는 공간을 이용하여 우회
	웹브라우저 기반 키보드 보안 우회	웹브라우저 수준에서 키보드 입력 정보의 유출을 차단하는 제품	DOM 스니핑을 통한 우회
	시스템 수준 키보드 보안 우회	키보드 입력을 시스템 수준에서 유출을 차단하는 제품	메시지 후킹을 통한 우회
	자료유출 방지 우회	중요 정보가 시스템 외부로 유출되는 것을 차단하는 제품	이동식 디스크를 하드디스크 폴더로 연결하여 우회

다. 따라서 자체보호 기능을 평가하기 위한 위협 분석 및 보안 테스팅을 위해 보안 제품들의 모듈 및 실행 상의 구조나 특징을 파악할 필요가 있다. MS 윈도우 시스템에서 구동되는 대부분의 SW기반 정보보호제품들의 구조나 특징은 다음과 같다. 첫 번째로, 모든 제품은 프로세스 및 다중 쓰레드(Multi-thread) 기반으로 구동되며, 윈도우의 특성상 DLL(Dynamic Link Library)로 구현된다. 두 번째, 시스템 수준의 기능 제공을 위해 드라이버를 사용한다. 세 번째, 지속적인 감시를 위해 서비스로 수행된다. 네 번째, 새롭게 보고된 악성코드/행위를 탐지하기 위해 원격 서버를 통해 주기적으로 업데이트를 수행한다. 다섯 번째, 제품의 다양한 기능 설정을 위해 레지스트리 및 각 제품별 설정파일을 생성·사용한다. 여섯 번째, 윈도우의 특성상 모든 애플리케이션들 커널 수준 실행을 위해 SSDT(System Service Dispatch Table)[1] 를 참조한다. 마지막으로 인터넷 보안 제품들은 대부분의 경우 웹브라우저에 BHO(Browser Helper Object)나, ActiveX를 추가한다. 이런 특징과 기존 공격 사례를 바탕으로 위협을 분석 및 도출한 결과는 〔표 1〕의 (a)와 같다.

〔표 1〕의 (a) 위협 분류를 SW 기반의 대표적인 정보보호제품군과 사상하면 〔표 2〕의 (a)와 같다. 모든 주요 제품군에 동일한 위협 분류들이 중복되어 포함되었는데, 제안하는 위협을 기준으로 보안 테스팅 전략을 제시한다면 테스팅 기법의 재사용성이 개선될 수 있음을 의미한다. 〔표 2〕의 사상 결과는 보안 제품의 구성 모듈 및 구현 기법 및 제품이 포함하는 보안기능에 따라 달라질 수 있다.

1) SSDT(System Service Dispatch Table): Windows환경에서 커널 수준에서 실행되는 Native API들의 주소 테이블

〔표 2〕 대표 보안제품군과 제안 위협 분류 사상 결과: (a) 자체보호 평가를 위한 분류, (b) 우회불가성 평가를 위한 분류

분류		주요 SW 기반 보안 제품군				
		안티 바이러스	방화벽	인터넷 보안	키보드 보안	자료유출 방지
(a)	프로세스 기반	O	O	O	O	O
	멀티쓰레드 기반	O	O	O	O	O
	DLL 사용	O	O	O	O	O
	드라이버 사용	O	O		O	O
	서비스 실행	O	O			
	SSDT	O	O	O	O	O
	업데이트 수행	O	O	O	O	
	레지스트리 및 설정파일 사용	O	O	O	O	O
	BHO 및 ActiveX 사용			O	O	
(b)	패턴 기반 악성코드 탐지 우회	O				
	TCP 패킷 필터링 우회		O			
	HTTP 패킷 필터링 우회		O	O		
	웹브라우저 키보드 보안 우회			O	O	
	시스템 수준 키보드 보안 우회				O	
	자료유출 방지 우회					O

4.1.2 우회불가성 위협

우회불가성에 대한 평가는 정보보호제품의 보안 기능이 우회되는지 여부를 확인하는 것이다. 예를 들면, 이미 알려진 악성코드패턴이 포함된 패킷을 방화벽이 정상적으로 차단하는지 검증하는 것이다. 따라서 보안 기능별로 위협을 도출하는 것이 효과적이다. 하지만 최근에는 단일 보안 제품이 다양한 보안기능을 포함하는 경우가 많고, 단순히 보안 기능을 기준으로 위협을 도출하면 보안 테스팅 기법을 도출하기 어렵기 때문에 보안 기능을 더 세부적으로 분류하였다. 예를 들면, TCP 패킷 필터링과 HTTP 패킷 필터링 제품은 모두 방화벽 제품이지만 보안 테스팅 기법이 서로 다르다. 우회불가성에 대한 관련 위협을 세부적인 보안 기능별로 도출한 결과는 〔표 1〕의 (b)와 같다.

〔표 2〕의 (b)는 〔표 1〕의 (b) 위협 분류를 대표적인 정보보호제품군과 사상한 결과이다. 행위 기반 악성코드 탐지는 일반적으로 악성코드 분석 시스템에서 사용되기 때문에 제외하였다. 〔표 2〕의 (a)와 비교하여 서로 다른 제품군에 공통적으로 포함되지는 못하지만 하나의 제품군 안에서 서로 다른 제품에 동일하게 재사용 가능하다.

4.2 위협별 보안 테스팅 기법

자체보호 기능 평가를 위한 테스팅 기법과 우회불가성 평가를 위한 테스팅 기법은 서로 다르다. 자체보호 기능 평가를 위한 보안 테스트는 보안 제품을 직접

〔표 3〕 자체보호 기능 평가를 위한 위협기반 보안 테스팅 전략

위협 분류	위협기반 보안 테스팅 전략
프로세스 기반	프로세스 종료 관련 API (NtsuspendProcess(), TerminateProcess(), ZwTerminateProcess(), WinStationTerminatedProcess()) 기반 강제 프로세스 종료
	프로세스 종료 관련 메시지 (WM_CLOSE, WM_QUIT) 전달 기반 강제 프로세스 종료
멀티쓰레드 기반	쓰레드 종료 관련 API (EndTask(), SendMessage(), SuspendThread(), CreateRemoteThread(), SetThreadContext(), TerminateThread(), ZwTerminateThread()) 기반 강제 쓰레드/프로세스 종료
DLL 사용	DLL 파일 변조
	DLL 인젝션 - 관련 API(SetWindowsHookEX(), SetWinEvenHook()) 기반, IAT(Import Address Table) 수정 기반
드라이버 사용	드라이버 언로드 및 삭제
서비스 실행	서비스 실행 파일 변조
SSDT	SSDT 후킹
업데이트 수행	업데이트 프로세스 강제 종료
	hosts 파일 변조를 통한 업데이트 차단
레지스트리 및 설정파일 사용	중요 보안기능 설정 값 변조
BHO 및 ActiveX 사용	웹브라우저에서 설정 중 추가기능 관리에서 사용안함 설정
	ActiveX 실행 파일 변조

공격하여 자체보호 기능에 대한 가용성 및 무결성을 훼손시킬 수 있는지 확인하는 것이다. 예를 들면, 보안제품에서 프로세스 강제 종료에 대해 자신을 보호하는 것이 가능한지 확인한다. 하지만 우회불가성 테스트는 보안제품을 직접 공격하는 것이 아니라 구동 중인 보안 기능을 우회할 수 있는지를 평가하는 것이다.

자체보호 기능에 대한 위협과 보안 테스팅 전략을 사상한 결과는 〔표 3〕과 같다. 자체보호 기능에 대한 테스팅 전략은 Anti-Malware Test Lab의 Self-protection test〔26〕의 테스팅 기법들을 바탕으로 위협별 재분류한 것이다. 최근 정보보호제품은 프로세스 및 쓰레드 기반으로 실행되기 때문에 프로세스/쓰레드 종료 관련 API를 사용해서 강제로 프로세스 종료가 가능한지 확인해야 한다. 시스템 드라이버를 사용하는 경우, 드라이버가 언로드(Unload)되면 정상적으로 보안기능을 수행할 수 없기 때문에 언로드 가능한지 확인해야 하고, DLL을 사용할 경우 다양한 DLL 인젝션(Injection) 기법이 존재하기 때문에 DLL 인젝션이 가능한지 검증해야한다. 또한 서비스로 등록 후 실행되는 경우에는 서비스 실행 파일 변조 가능 여부를 확인해야하고, 레지스트리 및 설정파일을 통해 중요 설정 값을 저장하는 경우, 악의적으로 설정 값을 변경하여도 정상 수행되는지 등을 확인해야한다. 그리고 SSDT 후킹(Hooking)을 통해 프로세스 및 쓰레드가 강제 종료되는지를 확인해야 한다.

자체보호 기능 평가기법은, 과거부터 최근까지의 Windows 계열 플랫폼에 계속해서 적용되고 있는 프로세스, 쓰레드, DLL, SSDT와 같은 플랫폼의 필수 구성요소 및 주요 특성을 기반으로 하고 있다. 실제로 DLL 인젝션 및 SSDT 후킹 등의 기법은 가장 최신 운영체제인 Windows 7에서도 사용 가능한 기법이다. 또한 DLL 인젝션 및 SSDT 후킹 등과 관련된 대부분의 보안 테스팅 기법은 정보보호제품에서 사용할 수 있는 방법이기 때문에 〔표 3〕의 보안 테스팅 기법들은 플랫폼 의존성이 높지 않다.

우회불가성에 대한 위협과 이에 대한 보안 테스팅 전략의 사상 결과는 〔표 4〕와 같으며, 대부분의 우회불가성 평가 기법들은 보안제품 자체의 분석기능을 우회하는 것이므로 플랫폼이나 수행 환경에 대한 의존성이 낮다. 안티바이러스와 같은 패턴 기반의 악성코드 탐지 제품은 파일의 구조를 알아보기 어렵게 인코딩하는 패킹(Packing)〔27〕, 난독화〔28〕 또는 분석 대상 파일의 구조 분석 실패 유도〔29〕 기법에 의해 우회되는지 확인해야 한다. 네트워크 패킷 필터링 기반 방화벽 제품들은 TCP 은닉채널〔30〕과 HTTP 은닉채널〔31〕 기술로 우회되는지 확인해야 한다. 클라이언트 허니팟과 같이 악성코드의 행위에 의해 발생하는 시스템의 변화를 통해 악성코드를 탐지하는 행위 기반 탐지 시스템은 지연시간, 사용자 상호작용, 시스템 환경 등 악성코드가 원하는 조건이 만족되는 경우에만 공격을 수행하는 논리폭탄(Logic-bomb)〔32〕에 의해 우회되지는 여부를 확인해야 한다(나머지 보안 테스팅 기법은 〔표 3〕, 〔표 4〕를 참조 바람). 정보보호제품의 취약성 평가를 위한 보안 테스팅 기법은 다양하지만, 〔표 3〕과 〔표 4〕에서는 대표적인 테스팅 기법들만 보여 준다.

〔표 4〕 우회불가성 평가를 위한 위협기반 보안 테스팅 전략

위협 분류	보안 테스팅 전략
패턴 기반 악성코드 탐지 우회	패킹, 난독화, 파일구조 분석 실패 유도 기반 우회
행위 기반 악성코드 탐지 우회	논리폭탄 기반 우회
TCP 패킷 필터링 우회	TCP syn flag 은닉 채널 기반 우회
	TCP fin flag 은닉 채널 기반 우회
	TCP ack flag 은닉 채널 기반 우회
	TCP urgent flag 은닉 채널 기반 우회
HTTP 패킷 필터링 우회	Http 은닉 채널 기반 우회
웹브라우저 수준 키보드 보안 우회	웹브라우저 DOM 스니핑을 통한 우회
시스템 수준 키보드 보안 우회	메시지 후킹을 통한 우회
자료유출 방지 우회	이동식 디스크를 하드디스크 폴더로 연결하여 우회
	VMware를 이용한 우회

V. 사례 분석

본 논문에서 제안한 보안 테스팅 절차를 사용하여 테스팅의 효율성과 재사용성을 개선할 수 있는지 검증하기 위해, 대표적인 기존 공개된 취약점과 테스팅 사례를 분석한다. 취약점 정보는 NVD를 통해서 수집하였고, NVD의 취약점 설명 정보가 부족한 경우에는 다른 취약점 DB인 SecurityFocus〔33〕에서 추가적으로 취약점 정보를 수집하였다.

〔표 5〕 SSDT 후킹 위협에 대한 자체보호 기능 관련 취약점

CVE 식별자	제품명	CVE 식별자	제품명	CVE 식별자	제품명
2008-1735	BitDefender Antivirus 2008 20080118 이하 버전	2008-1736	Comodo Firewall Pro 3.0 이하 버전	2008-1737	Sophos Anti-Virus 7.0.5, 7.x
2008-1738	Rising Antivirus 2008 20.38.20 이하 버전	2008-0365	CORE FORCE 0.95.172 미만 버전	2008-0366	CORE FORCE 0.95.172 미만 버전
2007-5086	Kaspersky Anti-Virus, Internet Security 7.0	2007-5039	Ghost Security Suite beta 1.110	2007-5040	Ghost Security Suite alpha 1.200
2007-5041	G DATA InternetSecurity 2007	2007-5042	Outpost Firewall Pro 4.0.1025.7828	2007-5043	Kaspersky Internet Security 7.0.0.125
2007-5044	ZoneAlarm Pro 7.0.362.000	2007-5047	Norton Internet Security 2008 15.0.0.60	2007-4967	Online Armor Personal Firewall 2.0.1.215
2007-4968	Privatefirewall 5.0.14.2	2007-4969	Process Monitor 1.22	2007-4970	ProcessGuard 3.410
2007-4971	ProSecurity 1.40 Beta 2	2007-4972	RegMon 7.04	2007-2083	ZoneAlarm Pro before 7.0.302.000
2007-1793	Norton Personal Firewall 2006 9.1.0.33, 9.1.1.7	2006-7160	Outpost Firewall PRO 4.0	2007-0708	Comodo Firewall Pro before 2.4.16.174
2007-0709	Comodo Firewall Pro 2.4.16.174	2006-5153	Sunbelt Kerio Personal Firewall 4.3.268	2006-3074	Kaspersky Internet Security, Anti-Virus 6.0, 7.0

5.1 자체보호 관련 사례

NVD에 등록된 취약점 중 정보보호제품의 자체보호 관련 위협 중 SSDT와 관련된 취약점은 〔표 5〕과 같다. 〔표 5〕의 모든 취약점은 SSDT에서 커널 수준 API인 Native API의 호출 주소를 변조하여 Windows Native API들을 후킹하고, 이를 통해 시스템 및 정보보호제품의 가용성을 훼손하거나 권한상승을 시도하는 위험성 높은 취약점들이다. 취약점이 존재하는 제품은 안티바이러스, 방화벽, 시스템 모니터링 도구 등으로 다양하고, 후킹을 시도하는 API들도 서로 다르다. 하지만 SSDT를 통해 Native API를 후킹하는 공격방법은 동일하다. SSDT는 Windows 운영체제의 구성요소로 SSDT 후킹 위협에 대한 보안제품의 자체보호 기능은 보안제품 자체에서 지원해야한다. 〔표 5〕과 같이, 27개의 제품에서 SSDT 후킹 위협에 의한 자체보호 기능 훼손이 가능하다는 것은 Windows 환경의 보안 제품 평가 시, SSDT 후킹과 관련된 자체보호 기능 평가가 필요하다는 것을 의미한다. 이에 SSDT 후킹 위협에 대한 일반화된 테스팅 전략을 정의하여 재사용성을 높이는 것이 필요하다. 본 논문에서 자체보호 기능 평가를 위해 모듈별 위협을 기준으로 분류한 테스팅 전략이 정보보호제품의 종류와 보안 기능에 상관없이 재사용 가능하고 효율적임을 보인다.

5.2 우회불가성 관련 사례

2012년 3월 NVD에 등록된 CVE-2012-1419부터 CVE-2012-1463까지 총 45개는, 악성코드가 삽입된 특정 포맷의 파일을 정보보호제품이 탐지하지 못하는 취약점이다. 파일포맷은 ELF, PE와 같은 실행파일 포맷과 TAR, ZIP과 같은 압축파일 포맷 등이고, 취약성이 존재하는 정보보호제품들은 안티바이러스를 포함하여 파일 구조를 분석하고 악성코드의 존재여부를 패턴기반으로 확인하는 제품들이다(구체적인 파일 포맷과 제품명은 CVE 웹사이트를 참고바람). 파일 구조 변조를 통해 정보보호제품의 보안 기능을 우회한 것이기 때문에, 취약성이 존재하는 제품들은 우회불가성이 지원되지 않는다고 볼 수 있다. 즉 패턴기반 악성코드 탐지 기능을 가진 정보보호제품의 파일 구조 분석 과정이 실패하도록 유도하여 우회한 사례에 해당한다. 특히, 〔표 6〕과 같이 다수의 제품에 동일한 취약점이 존재하는 경우가 대부분으로, 하나의 취약점이 평균 약 7.9개(약 8개)의 정보보호제품에 존재하며, CVE-2012-1459의 경우 최대 35개의 제품에 공통적으로 존재한다. 이를 통해, 본 논문에서 우회불가성 평가를 위해 제안한 위협 및 보안 테스팅 방법이 동일한 보안기능 제품의 평가를 위해 반복해서 사용될 수 있음을 알 수 있다. 즉, 제안한 보안 테스팅 방법의 재사용성이 개선될 수 있음을 의미한다.

〔표 6〕 패턴기반 악성코드 탐지 제품과 관련된 취약점과 각 취약점이 공통적으로 존재하는 제품 개수

CVE 식별자	제품개수	CVE 식별자	제품개수	CVE 식별자	제품개수	CVE 식별자	제품개수
2012-1419	2	2012-1420	11	2012-1421	4	2012-1422	4
2012-1423	11	2012-1424	6	2012-1425	15	2012-1426	6
2012-1427	3	2012-1428	3	2012-1429	9	2012-1430	9
2012-1431	10	2012-1432	4	2012-1433	5	2012-1434	4
2012-1435	5	2012-1436	5	2012-1437	1	2012-1438	2
2012-1439	3	2012-1440	3	2012-1441	1	2012-1442	10
2012-1443	33	2012-1444	2	2012-1445	3	2012-1446	12
2012-1447	2	2012-1448	4	2012-1449	2	2012-1450	3
2012-1451	2	2012-1452	3	2012-1453	12	2012-1454	5
2012-1455	2	2012-1456	20	2012-1457	29	2012-1458	2
2012-1459	35	2012-1460	8	2012-1461	20	2012-1462	12
2012-1463	12						

VI. 실험 및 검증

제안 기법이 정보보호제품의 취약성 검증에 실제 적용할 수 있음을 확인하기 위해, 국내 정보보호제품을 대상으로 실험하였다. 실험 결과는 2011년 9월에 수행한 결과이며 실험 제품은 2011년 9월 당시 최신 버전을 사용하였고, 그 당시 발견한 취약성들은 2012년 8월 현재 대부분 패치(Patch)된 상태이다. 방화벽 제품의 우회불가성 평가를 위한 보안 테스팅은 Ubuntu 10.10에서 수행되었고, 이를 제외한 모든 보안 테스팅은 Windows XP Professional SP3에서 수행되었다. 보안 테스팅 결과, 취약성이 발견된 제품과 각 제품에 실시한 테스팅 전략은 〔표 7〕과 같다.

자체보호 평가를 위한 보안 테스팅은 안티바이러스 제품을 대상으로 수행하였으며, 기본 프로세스 관리 도구와 본 연구진이 직접 제작한 공격 도구를 사용하였다. 대표적으로 프로세스 및 쓰레드 기반 보안 제품의 자체보호 기능을 테스트하기 위해서 가장 먼저 정보보호 제품의 정상 수행을 확인하고, 최신 패치를 적용하였다. 그리고 프로세스 모니터링 도구인 Process Explorer〔34〕를 통해 정보보호 제품의 프로세스 이름과 PID(Process ID)를 획득하고, 프로세스 및 스레드를 강제 종료 시킬 수 있는 다양한 공격 기법을 선택적으로 실행할 수 있도록 직접 제작한 테스팅 도구를 이용하여 TOE를 공격하고, 다시 모니터링 도구를 통해 정보보호 제품의 프로세스가 종료되는지 확인하였다.

실험 결과 *백신제품1*과 *백신제품2*에서 취약성이 발견되었다. *백신제품1*은 본 논문에서 제안한 대부분의 자체보호 보안 테스팅에 취약하였다. 실험을 수행한 시점에 다른 대부분의 안티바이러스 제품들도 포함하고 있던 프로세스 및 쓰레드 강제 종료에 대한 자체보호 기능이 모두 무력화되었다. 또한 업데이트 서버의 도메인 이름에 대한 IP 주소를 임의로 변경하여 hosts 파일에 기록한 결과, 악성코드 패턴의 업데이트가 불가능하였다. *백신제품2*는 드라이버 삭제 공격에 대한 취약성이 발견되었다. 드라이버 삭제를 위한 실험에서는, 시스템에 로드된 드라이버를 삭제하는 기능이 있는 마이크로소프트의 Windows Sysinternals Suite〔35〕를 사용하였다. *백신제품2*에서 설치한 파일 시스템 관련 드라이버를 삭제하자 악성코드 탐지 기능, 무결성 검증 기능, 실시간 시스템 감시 기능 등 안티바이러스의 기능 중 대부분이 무력화 되었다.

우회불가성 평가 대상으로는 자료유출 방지, 방화벽, 키보드 보안 제품을 사용하였다. 방화벽을 평가하기 위해서, HTTP 및 TCP 은닉 채널을 사용하여 방화벽이 설치된 시스템에 커맨드 명령을 전송하고, 결과 데이터를 전송 받는지 확인하는 과정을 통해 패킷 필터링 기능이 우회되는지 실험하였다. 그 결과, 인터넷 뱅킹 시에 실행되는 *방화벽제품1*의 필터링 기능이 모두 우회되었다. 인터넷 뱅킹 시에 자동 실행되어 키보드 후킹 및 입력 데이터 유출을 방지하는 *키보드보안제품1*이 메시지 후킹을 통해 우회되는지 확인하기 위해 마이크로소프트에서 제공하는 SPY++를 사용하여 키보드 관련 메시지를 후킹하였고, 이를 통해 키보드 입력 데이터를 스니핑할 수 있었다. 금융 웹사이트 이용 시 키보드 입력 데이터에 대한 스니핑 위협으로부터 자유롭고자 사용되는 *웹사이트 가상키보드*를 통해 입력된 데이터도 웹 브라우저의 DOM(Document Object Model) 스니핑을 통해 확인할 수 있었다. 시스템 내부의 정보가 이동식디스크 등을 통해 외부로 유출되는 것을 방기하기 위한 *자료유출방지제*

〔표 7〕 제안 기법 검증을 위한 보안 테스팅 대상 제품 및 전략

제품군	제품명	보안 테스팅 전략
안티바이러스	백신제품1	프로세스 종료 관련 API 기반 강제 프로세스 종료 (NtsuspendProcess(), TerminateProcess(), ZwTerminateProcess(), WinStationTerminatedProcess())
		프로세스 종료 관련 메시지 전달 기반 강제 프로세스 종료 (SendMessage()를 이용하여 WM_CLOSE, WM_QUIT 메시지 전달)
		쓰레드 종료 관련 API 기반 강제 쓰레드/프로세스 종료 (EndTask(), SendMessage(), SuspendThread(), CreateRemoteThread(), SetThreadContext(), TerminateThread(), ZwTerminateThread())
		DLL 인젝션
		DLL 변조(기존 DLL 수정)
		서비스 실행 파일 변조(레지스트리 수정)
		hosts 파일 변조를 통한 업데이트 차단
	백신제품2	드라이버 삭제
방화벽	방화벽제품1 (Ubuntu 버전)	HTTP 은닉 채널을 기반 우회
		TCP syn flag 은닉 채널 기반 우회
		TCP fin flag 은닉 채널 기반 우회
		TCP ack flag 은닉 채널 기반 우회
		TCP urgent flag 은닉 채널 기반 우회
키보드 보안	키보드보안제품1	메시지 후킹을 통한 우회
	웹사이트 가상키보드	웹브라우저 DOM 스니핑을 통한 우회
자료유출 방지	자료유출방지제품1	이동식 디스크를 하드디스크 폴더로 연결하여 우회

*품1*은 Windows의 '컴퓨터 관리' 도구의 '비어 있는 NTFS 폴더에 탑재' 설정을 통해 시스템 내부에 위치한 폴더와 연결하는 기법을 통해 보안기능이 우회되었다.

5장의 사례 분석과 6장의 실험을 통해, 본 논문에서 제안하는 취약성 평가 절차 및 위협 기반 보안 테스팅이 타당함을 확인하였다.

VII. 결론 및 향후연구

3.3 DDoS 등 최근의 대규모 보안 사고에서 정보보호제품의 무력화가 시도됨에 따라, 무력화 공격에 악용되는 정보보호제품의 취약성에 대한 평가가 중요시되고 있다. 정보보호제품의 취약성 평가를 위한 보안 테스팅 기법 중의 침투 테스팅은 완성된 제품을 대상으로 정보수집, 가능한 공격표면 식별, 침입시도, 결과보고 등을 수행해야 하므로 많은 시간과 자원이 필요하다. 이러한 문제를 개선하기 위해, 본 논문에서는 재사용성과 효율성을 고려한 취약점 분석, 위협 분류, 테스팅 전략 선정, 위협기반 보안 테스팅 기법에 대해 연구하였다.

또한, 본 논문에서는 정보보호제품의 무력화 공격에 대응하기 위해, 자체보호 기능과 우회불가성 평가를 위한 효과적인 보안 테스팅 절차를 제안하였다. 제안 절차는 보안 테스팅 목적에 적합한 위협 분석 및 분류, 테스팅 전략 선정, 위협 기반 보안 테스팅 수행으로 구성된다. 제안한 보안 테스팅 기법은 위협 분석을 통해 SW 모듈 및 제품의 보안 기능 별로 분류하여 재사용성을 높였다. 제안 기법의 타당성을 검증하기 위해, 정보보호제품들의 기존 취약점 사례들을 분석하였고 정보보호제품을 대상으로 보안 테스팅을 수행하였다.

향후, 다양한 정보보호제품에 대해 체계적인 위협 분석과 보안 테스팅 기법에 대해 연구하고, 정보보호제품의 평가·인증 강화를 위한 다른 테스팅 기준 및 방법에 대해서 연구할 계획이다.

참고문헌

〔1〕 Gartner, "Now is the time for security at Application Level," Dec. 2005.

〔2〕 G. McGraw, "Software assurance for security," IEEE Computer, vol. 32, pp. 103-105, Apr. 1999.

〔3〕 G. McGraw and B. Potter, "Software Security Testing," IEEE Security and

Privacy, Vol.2, pp.81-85, Sep. 2004.
〔4〕 B. Arkin, S. Stender and G. McGraw, "Software penetration testing," IEEE Security & Privacy, vol.3, pp. 84-87, Jan. 2005.
〔5〕 D.P. Gilliam , T.L. Wolfe, J.S. Sherif and M. Bishop, "Software Security Checklist for the Software Life Cycle," Proceedings of the Twelfth International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 243, Jun. 2003.
〔6〕 X. Zhang, L. Shao and J. Zheng, "A Novel Method of Software Vulnerability Detection based on Fuzzing Technique," Proceedings of the 2008 International Conference on Apperceiving Computing and Intelligence Analysis : ICACIA 2008, pp. 270-273, Dec. 2008.
〔7〕 V. Ganesh, T. Leek and M. Rinard, "Taint-based directed whitebox fuzzing," Proceedings of the 2009 International Conference on Software Engineering : ICSE 2009, pp. 474-484, May 2009.
〔8〕 D. Thiel, "Exposing Vulnerabilities in Media Software," Tech. Rep., BlackHat USA, Jul. 2007.
〔9〕 D. Kim and S. Cho, "Fuzzing-based Vulnerability Analysis for Multimedia Players," Journal of KIISE : Computing Practices and Letters, vol.17, no.2, Feb. 2011.(in Korean)
〔10〕 CVSS(Common Vulnerability Scoring System) home page: http://www.first.org/cvss
〔11〕 CWSS(Common Weakness Scoring System) home page: http://cwe.mitre.org/cwss/
〔12〕 CVE(Common Vulnerabilities Enumeration) home page: http://cve.mitre.org/
〔13〕 CWE(Common Weakness Enumeration) home page: http://cwe.mitre.org/
〔14〕 Common Criteria, "Common Criteria for Information Technology Security Evaluation- Part 3: Security assurance components, Version 3.1," Sep. 2007.
〔15〕 K. Bang, I. Kim, J. Lee, J. Lee and J. Choi, "Classification Criteria and Application Methodology for Evaluating IT Security Products," Journal of Korea Knowledge Information Technology Society, vol.6, no.5, Oct. 2011. (in K-orean)
〔16〕 Fortify Software, Inc. home page: http://www.fortify.com
〔17〕 J.A. Kupsch, B.P. Miller, E. Heymann and E. Cesar, "First Principles Vulnerability Assessment," Proceedings of the 2010 ACM workshop on Cloud computing security workshop, pp. 87-92, Oct. 2010.
〔18〕 Y. Son, "A Study on Software Vulnerability of Programming Languages Interoperability," Proceedings of the Advanced Computer Science and Information Technology Communications in Computer and Information Science, vol.195, pp. 123-131, Sep. 2011.
〔19〕 D. Kim and S. Cho, "An Analysis of Domestic and Foreign Security Vulnerability Management Systems based on a National Vulnerability Database," Journal of Internet and Information Security, vol.1, no.2, pp. 130-147, Nov. 2010. (in Korean)
〔20〕 D. Kim, D. Seo, W. Yi and S. Cho, "An Efficient Vulnerability Management System for Utilization of New Information Technologies-related Security Vulnerabilities," Proceedings of the 37th KIISE Fall Conference, vol.37, no.2(B), pp. 66-71, Nov. 2010. (in Ko- rean)
〔21〕 G. Kim and S. Cho, "Fuzzing of Web Application Server Using Known Vulnerability Information and Its Verification," Proceedings of the KIISE Korea Computer Congress 2011, vol.38, no.1(B), pp. 181-184, Jun. 2011. (in Korean)

〔22〕 보안뉴스, "디아블로3 사용자 계정 탈취용 악성파일 국내 등장," 호애진, 2012년 06월 - web site: http://www.boannews.com/media/view.asp?idx=31582
〔23〕 SANS Institute Reading Room web page: http://www.sans.org/reading_room
〔24〕 "Practical Threat Analysis for Information Security Experts," web page: http://www.ptatechnologies.com
〔25〕 NVD(National Vulnerability Database) home page: http://nvd.nist.gov/
〔26〕 Anti-Malware Test Lab. home page: http://www.anti-malware-test.com/?q=taxonomy/term/16
〔27〕 C.S. Collberg and C. Thomborson, "Watermarking, tamper-proofing, and obfuscation - tools for software protection," IEEE Transactions on Software Engineering, vol.28, pp. 735-746, Aug. 2002.
〔28〕 M. Kim, J. Lee, H. Chang, S. Cho, Y. Park, M. Park and P.A. Wilsey, "Design and Performance Evaluation of Binary Code Packing for Protecting Embedded Software against Reverse Engineering," Proceedings of the IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing 2010, pp. 80-86, May 2010.
〔29〕 S. Jana and V. Shmatikov, "Abusing file processing in malware detectors for fun and profit," Proceedings of the IEEE Symposium on Security and Privacy 2012, pp. 80-94, May 2012.
〔30〕 M. Bauer, "New covert channels in HTTP: adding unwitting Web browsers to anonymity sets," Proceedings of the 2003 ACM workshop on Privacy in the electronic society, pp. 72-78, Oct. 2003.
〔31〕 S. J. Murdoch and S. Lewis, "Embe- dding covert channels into TCP/IP," Proceedings of the 7th international conference on Information Hiding 2005, pp. 247-261, Jul. 2005.
〔32〕 C. Seifert, P. Komisarczuk and I. Welch, "True Positive Cost Curve: A Cost-Based Evaluation Method for High-Interaction Client Honeypots," in Proceedings of the 2009 Third International Conference on Emerging Security Information, Systems and Technologies, pp. 63-69, Jun. 2009.
〔33〕 SecurityFocus home page: http://www.securityfocus.com/
〔34〕 Process Explorer home page: http://technet.microsoft.com/en-us/sysinternals/bb896653.aspx
〔35〕 Windows Sysinternals Suite home page: http://technet.microsoft.com/en-us/sysinternals/bb842062

〈著者紹介〉

김 동 진 (Dongjin Kim) 학생회원
2009년: 단국대학교 컴퓨터과학과 졸업
2011년: 단국대학교 컴퓨터과학과 공학석사
2011년 3월~현재: 단국대학교 컴퓨터과학과 박사과정
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 보안 테스팅, 시스템소프트웨어 등

정 윤 식 (Youn-Sik Jeong) 학생회원
2012년: 단국대학교 컴퓨터과학과 졸업
2012년 3월~현재: 단국대학교 컴퓨터과학과 석사과정
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 시스템소프트웨어 등

윤 광 열 (Gwangyeul Yun) 학생회원
2009년: 단국대학교 컴퓨터과학과 졸업
2011년: 단국대학교 컴퓨터과학과 공학석사
2011년 3월~현재: 단국대학교 컴퓨터과학과 박사과정
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 소프트웨어공학, 웹공학, 비용이익분석 등

유 해 영 (Haeyong Yoo) 정회원
1979년: 단국대학교 수학과 졸업
1981년: 단국대학교 수학과 이학석사
1994년: 아주대학교 컴퓨터공학과 공학박사
1983년 3월~현재: 단국대학교 소프트웨어학과 교수
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 소프트웨어공학, 웹공학, 서비스 및 콘텐츠 정책 등

조 성 제 (Seong-Je Cho) 종신회원
1989년: 서울대학교 컴퓨터공학과 졸업
1991년: 서울대학교 컴퓨터공학과 공학석사
1996년: 서울대학교 컴퓨터공학과 공학박사
2001년: 미국 University of California, Irvine 객원연구원
2009년: 미국 University of Cincinnati 객원연구원
1997년 3월~현재 : 단국대학교 소프트웨어학과 교수
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 시스템소프트웨어, 실시간스케줄링, 임베디드 소프트웨어 등

김 기 연 (Gi-Youn Kim) 정회원
2010년: 단국대학교 컴퓨터과학과 졸업
2012년: 단국대학교 컴퓨터과학과 공학석사
2012년 4월~현재: 에이쓰리시큐리티 기술보안팀
〈관심분야〉 컴퓨터보안, 소프트웨어 보증, 시스템소프트웨어 등

〈著者紹介〉

이 진 영 (Jinyoung Lee) 정회원
2009년: 단국대학교 컴퓨터과학과 졸업
2012년: 단국대학교 컴퓨터과학과 공학석사
2011년 12월~현재: 한국인터넷진흥원
〈관심분야〉 컴퓨터보안, 난독화, 소프트웨어 취약점 검증, 공통평가기준 등

김 홍 근 (Hong-Geun Kim) 종신회원
1985년: 서울대학교 컴퓨터공학과 졸업
1987년: 서울대학교 컴퓨터공학과 공학석사
1994년: 서울대학교 컴퓨터공학과 공학박사
1994년~1996년: 한국전산원
1996년~2009년: 한국정보보호진흥원
2009년 7월~현재: 한국인터넷진흥원
〈관심분야〉 컴퓨터 보안, 시스템 보안취약점 관리 등

이 태 승 (Taeseung Lee) 정회원
1994년: 광운대학교 전자계산학과 졸업
1996년: 포항공과대학교 전자계산학과 공학석사
1996년~2001년: 삼성전자
2002년~2009년: 한국정보보호진흥원
2009년~현재: 한국인터넷진흥원
〈관심분야〉 컴퓨터 보안, 정보보호제품 보안성 평가 등

임 재 명 (Jae-Myung Lim) 종신회원
1981년 2월: 한양대학교 전자공학과 졸업
1983년 9월: 한양대학교 전자공학과 석사
2008년 2월: 항공대학교 통신정보공학과 박사
2000년 11월~현재: 한국인터넷진흥원 공공정보보호단장
〈관심분야〉 SW보안약점, 정보화역기능(해킹, 웜, 스팸), CIP보호, 정보보호평가 등

원 동 호 (Dongho Won) 종신회원
1976년~1988년: 성균관대학교 전자공학과 (공학사, 공학석사, 공학박사)
1978년~1980년: 한국전자통신연구원 전임연구원
1985년~1986년: 일본 동경공업대 객원연구원
1988년~2003년: 성균관대학교 교학처장, 전기전자 및 컴퓨터공학부장, 정보통신대학원장, 정보통신기술연구소장, 연구처장
1996년~1998년: 국무총리실 정보화추진위원회 자문위원
2002년~2003년: 한국정보보호학회장
현재: 성균관대학교 정보통신공학부 교수, 한국정보보호학회 명예회장, 정보통신부지정 정보보호인증기술연구센터 센터장, IT보안성평가연구회 위원장
〈관심분야〉 암호이론, 정보이론, 정보보호