디지털산업정보학회 논문지 제11권 제4호-2015년 12월
http://dx.doi.org/10.17662/ksdim.2015.11.4.081

안전한 웹 서버 환경을 위한 시큐어코딩 도구, 웹셸 탐지도구 간의 상호연동 시스템 설계*

김 범 용** · 최 근 창** · 김 준 호*** · 석 상 기****

A Design of Inter-Working System between Secure Coding Tools and Web Shell Detection Tools for Secure Web Server Environments

Kim Bumryong · Choi Keunchang · Kim Joonho · Suk Sangkee

〈Abstract〉

Recently, with the development of the ICT environment, the use of the software is growing rapidly. And the number of the web server software used with a variety of users is also growing. However, There are also various damage cases increased due to a software security vulnerability as software usage is increasing. Especially web shell hacking which abuses software vulnerabilities accounts for a very high percentage. These web server environment damage can induce primary damage such like homepage modification for malware spreading and secondary damage such like privacy. Source code weaknesses checking system is needed during software development stage and operation stage in real-time to prevent software vulnerabilities. Also the system which can detect and determine web shell from checked code in real time is needed.

Therefore, in this paper, we propose the system improving security for web server by detecting web shell attacks which are invisible to existing detection method such as Firewall, IDS/IPS, Web Firewall, Anti-Virus, etc. while satisfying existing secure coding guidelines from development stage to operation stage.

Key Words : Secure Coding, Web Shell, Web Server, Source Code

Ⅰ. 서론

최근 ICT환경의 발전으로 인해 소프트웨어의 사용이 급격히 늘어남과 동시에, 소프트웨어에서 보안취약점(Security Vulnerabilities)으로 인한 다양한 피해사례도 증가하고 있다. 전 세계 인터넷 사용자 수는 2013년 약 27억 명으로 2015년에는 30억 명을 돌파할 것으로 보이며, 이는 전 세계 인구의 약 42%에 해당한다[1]. 인터넷을 이용하는 웹 유저가 증가함에 따라 웹을 겨냥한 해킹 또한 증가하고 있으며, 현재 해킹 공격의

* 이 연구는 서울과학기술대학교 교내연구비의 지원으로 수행되었습니다.
** 숭실대학교 컴퓨터학과 석사과정
*** ㈜이븐스타 보안연구소 이사
**** 서울과학기술대학교 컴퓨터공학과 정교수(교신저자)

89~90% 웹을 겨냥하고 있다. 또 웹 해킹 피해 서버 중 91%에서 웹쉘(Web Shell)이 악용된 흔적이 발견되었다[2]. 실제로 2014년 한 회사의 서버는 웹쉘 업로드를 통해 침투 당하여 홈페이지가 변조되고 악성코드 유포지로 악용되어 접속 PC 6,541대를 감염시킨 사례가 있으며 그 외에도 개인정보 유출과 유출된 개인정보 불법 이용 등 다른 사고로도 이어질 수 있다[3, 4].

이러한 웹쉘의 악용을 막기 위해서는 우선 웹 서버(Web Server)의 응용을 개발할 때 소스코드 보안약점 진단도구를 통해 소프트웨어 개발과정 중 코딩단계(소스코드 구현단계)에서 소스코드의 보안 약점을 제거하고 안전하게 코딩하는 시큐어코딩(Secure coding)을 적용하여 소스코드의 안전성을 확보할 필요가 있다. 더욱이 개발단계에서 시큐어코딩을 적용하였더라도 웹 서버 환경에서 관리자나 해커로부터 변경된 소스코드는 보안취약점을 내포할 수 있기 때문에 시큐어코딩은 개발단계에서 뿐만 아니라 소스코드가 변경될 수 있는 모든 단계에서 적용이 필요하다. 따라서 외부와 커뮤니케이션이 빈번한 웹 서버의 경우에는 언제 어디서 소프트웨어가 변경될지 예측하기 어렵기 때문에, 시큐어코딩 적용에 대한 실시간 모니터링을 실시해야 한다. 또한 웹 서버에 웹쉘과 같은 보안취약점이 발생할 경우 다양한 피해를 야기할 수 있기 때문에 실시간 소스코드 보안약점 진단도구와 상호작용을 통해 소스코드에 대한 안전성을 확보와 더불어 웹쉘에 대한 점검을 함께 병행하여 웹쉘에 대한 안정성 역시 확보해야 한다.

이에 본 논문에서는 소스코드 보안약점 진단도구와 악성코드 탐지도구의 상호연동을 통해 실시간으로 시큐어코딩과 웹 서버의 웹쉘을 탐지를 함께 할 수 있는 시스템의 설계를 제안한다. 2장에서는 시큐어코딩, 웹쉘에 대해 알아본 후 3장에서 본 논문에서 제안하는 시스템에 대해 자세히 서술하고, 4장에서 결론을 맺는다.

Ⅱ. 관련연구

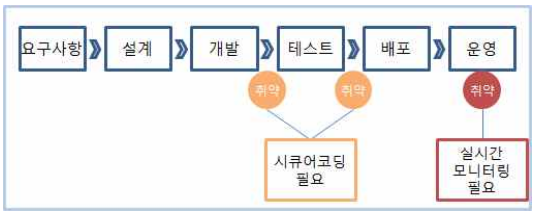
2.1 시큐어코딩

시큐어코딩이란 프로그램 내에 존재할 수 있는 잠재적인 보안약점을 제거하여 보안취약점이 발생하지 않도록 하는 보안 활동이다[5]. 보안약점은 소스코드에 포함되어 공격자에 의해 악용될 소지가 있는 보안 측면의 버그를 의미한다. 소스코드에 포함된 보안약점을 제거하지 않고 배포·운영하게 되면 해당 시스템이 다양한 공격에 노출될 가능성이 높아지며, 이때 사용되는 보안약점은 보안취약점화 되어 공격에 이용되게 된다. 이러한 보안취약점을 이용한 공격행위·공격코드를 익스플로잇(Exploit)이라 한다[6].

시큐어코딩은 소프트웨어의 보안취약점을 근본적으로 해결하기 위한 것으로 개발단계에서 보안을 적용하여 원천적인 보안취약점을 차단할 수 있는 방식이 필요해짐에 따라 등장하였다. 시큐어코딩은 소프트웨어의 개발단계에서 개발자의 실수, 지식부족, 프로그래밍 언어가 가지고 있는 고유한 약점 등으로 인하여 발생할 수 있는 보안취약점을 최소화 하거나 제거하는 것을 통해 개발완료 후 취약점 수정 비용에 비하여 설계, 코딩단계에서 취약점에 대한 수정 시 비용을 대폭 절감할 수 있도록 한다[7].

즉, 개발자가 보안약점으로 인한 보안취약점이 발생하지 않도록 시큐어코딩 규칙 등을 고려하여 기능을 설계 및 구현하고, 안전한 소프트웨어를 만들 수 있도록 하는 것이다. 안전한 소프트웨어를 만들기 위해서는 행정자치부가 발표한 47개 보안약점, 국정원 홈페이지에 명시한 8대 취약점과 전자금융감독규정에서 정의한 가이드라인 등을 기초로 정보화사업을 이용하는 전 분야에 걸쳐 의무 적용이 필요하다[8].

일반적인 소프트웨어와는 다르게 웹 서버의 경우, 개발 및 테스트가 완료된 이후인 운영단계에서도 필



<그림 1> 소스코드 취약점 발생 구간

요에 따라서 빈번한 수정이 발생하게 된다. 따라서 개발단계 이외에도 <그림 1>의 여러 단계 중 운영단계에서 실시간 소스코드 점검을 통해 소스코드 안전진단을 실시할 필요가 있다.

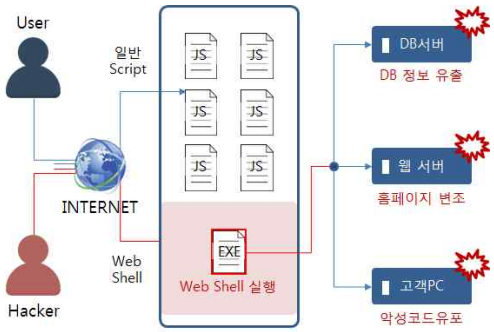
시큐어코딩을 위한 소스코드 진단 방법은 진단 방법에 따라 <표 1>과 같이 구분되며 각각의 특징을 가진다[9].

<표 1> 소프트웨어 보안 취약점 진단 방법 구분

구분	내용	특징
정적 분석	- S/W를 실행하지 않고, 소스코드 수준으로 보안취약점 분석 - S/W 개발과정에서 주로 사용	- S/W 개발 초기에 보안취약점을 발견하여 소프트웨어 수정 비용 절감 - 컴포넌트 간 발생할 수 있는 통합적 보안취약점 발견이 제한적 - 설계, 구조 관점의 보안 취약점은 미 발견 - 오탐, 미탐 존재
동적 분석	- S/W 실행환경에서 보안 취약점 분석 - S/W 시험단계에서 주로 사용	- 소스코드 불필요 - 정확도와 커버리지 향상 - 도구 사용자의 수준에 영향을 받음 - 구조 관점의 보안 취약점은 발견할 수 없음
하이브리드 분석	- 동적분석과 정적분석 방식을 혼용한 방식으로 소스코드 레벨부터 런타임 시 까지 통합분석	

2.2 웹쉘(Web Shell)

웹쉘(Web Shell)이란 웹페이지(Wep Page)의 약어인 '웹(Web)' 과 서버환경에게 명령을 실행하기 위한 인터페이스 역할을 하는 '쉘(Shell)' 의 합성어이며, 웹페이지 상에서 서버에게 명령어를 실행하기 위한 목적으로 만들어진 프로그램이다8].



<그림 2> Web Shell의 공격 예시

웹쉘은 원격 상에서 웹 서버에 명령을 수행할 수 있도록 하기 위해 작성한 웹 스크립트(ASP, JSP, PHP, CGI 파일 등) 형태의 파일이다. 웹 서버의 다양한 보안취약점(웹 어플리케이션 취약점, 웹 서버 취약점) 등을 타깃으로 공격하여 웹 서버 상에 웹쉘을 업로드한 후 일반 웹 브라우저를 이용하여 미리 업로드한 웹쉘을 실행하고 웹쉘이 시스템 명령어를 수행하도록 하므로 네트워크 방화벽 영향을 받지 않고 서버를 제어할 수 있다. 침투한 웹 서버상의 정보를 원격으로 유출 및 변조, 악성코드 유포 등의 불법 행위를 행하는 형태가 많다[10].

<표 2> 백신 탐지 우회기법

탐지 우회기법	내용
짧은 코드	eval, excute 와 같이 정상적인 스크립트 파일 삽입
인코딩	웹 소스를 보호하기 위한 Script Encoder를 활용
문자열 분리	Signature로 이용되는 문자열을 분산 삽입
파일생성	CreateTextFile, Write 와 같이 정상적인 스크립트 메소드를 이용
HTTP 서비스	Web 포트((80:TCP)를 이용하여 제어

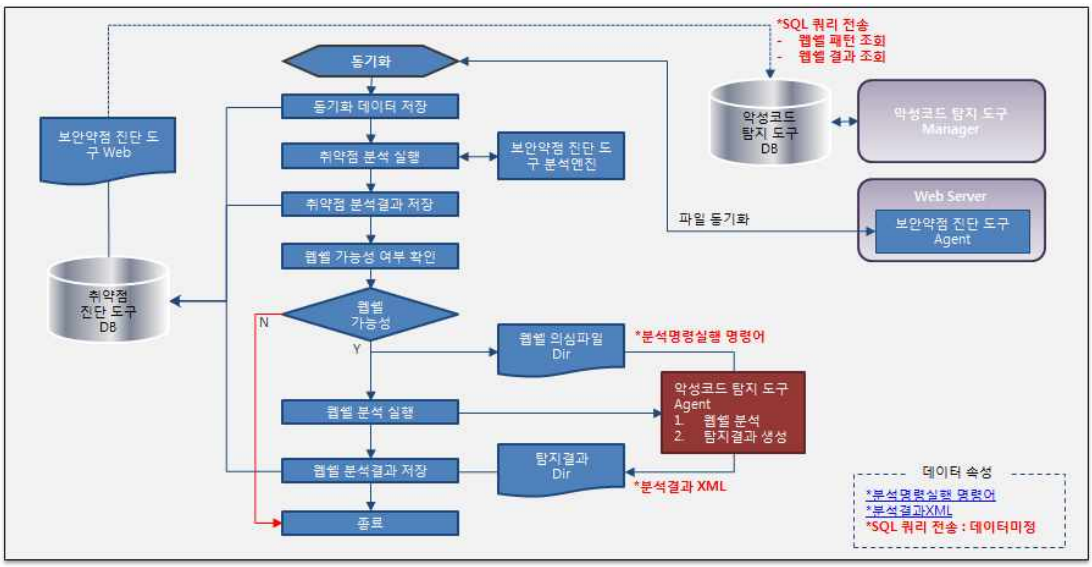
웹쉘은 Firewall, IDS/IPS, Web Firewall, Anti-Virus 등 기존 방법으로는 막기 어려우며, 실제 virustotal에서 웹쉘 백신탐지 결과 19개의 백신 중 14백신에서 미탐지 되었다[11]. 특히 다양한 탐지 우회기법이 적용된 웹쉘은 백신 프로그램으로 탐지가 어려우며, 일반적인 서버 관리자들은 해킹여부를 확인하기 힘들다[12]. 따라서 웹쉘 공격을 막고 피해를 최소화하기 위해서는 전 시스템에 대하여 실시간 빠른 탐지와 선제 대응을 필수로 하여 웹 해킹에 대응하기 위한 특화된 방어가 필요하다[13]. 시큐어코딩을 통해 일정수준까지 방지가 가능하지만 완전한 방지에 어려움이 있으며, 웹쉘 탐지를 위해서는 악성코드 탐지 솔루션의 일종인 웹쉘 탐지 모니터링 도구를 통한 탐지가 필요하다.

Ⅲ. 안전한 웹 서버 환경을 위한 시큐어코딩 도구・웹쉘 탐지도구의 연동 시스템

이 시스템은 소스코드 보안약점 진단도구를 악성코드 탐지도구(웹쉘 탐지 모니터링 도구)와 상호 기능 연동을 통해 소스코드 보안약점 및 웹쉘 파일을 탐지하여 운영되는 웹 서버의 보안성 향상을 목적으로 하며 파일 동기화 단계, 운영 단계, 리포팅으로 구성된다. 파일 동기화 단계는 웹 서버의 파일의 동기화를 통해 소스코드를 점검하고, 운영 단계에서는 소스코드 업로드를 통해 취약점 및 웹쉘을 점검한다. 마지막으로 리포팅에서는 점검결과와 점검에 대한 세부 사항들을 포함하여 리포팅 문서 형태로 제공한다.

3.1 파일 동기화 단계

파일 동기화 단계에서는 기존 소스코드 보안약점 진단도구 웹 서버와 동기화를 수립하여 웹 서버로부터 파일들을 전송 받고 해당 파일들에 대해 소스코드 보안약점 진단도구로 취약점 분석을 진행한다. 취약

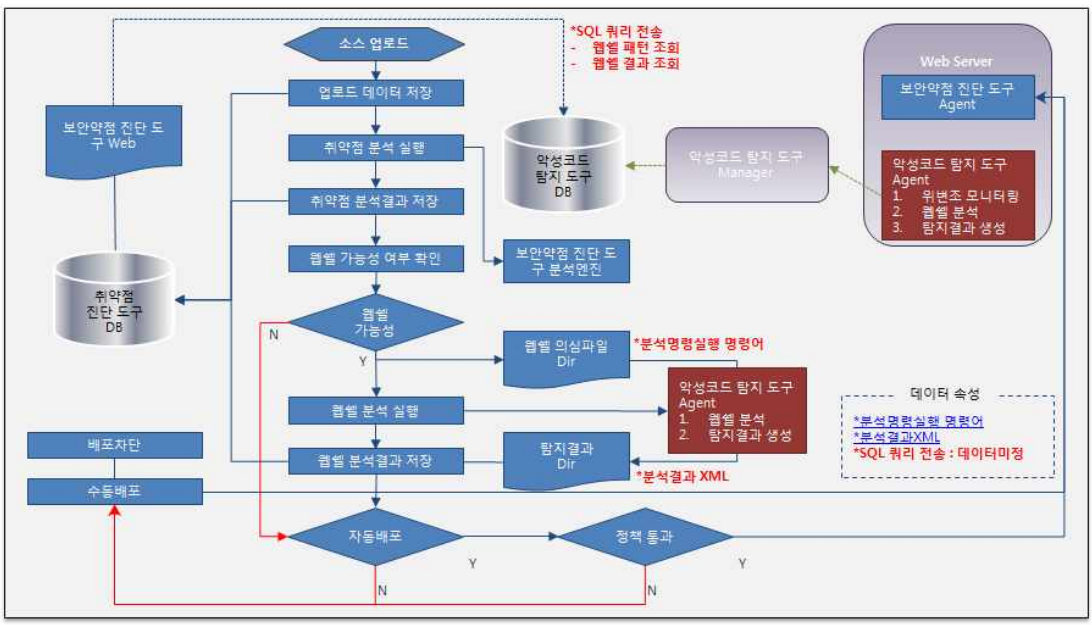


<그림 3> 파일 동기화 단계 Process Flow

점 분석 과정에서는 취약점 DB 및 분석 도구를 활용한다. 취약점 분석 진행 과정 및 결과는 모두 저장하고 관리하여, 차후 분석 가능하도록 제공한다. 취약점 분석 과정을 통해 파일 업로드 취약점, 파일 다운로드 취약점, 운영체제 명령어 삽입 취약점 중 하나 이상이 발견된 파일은 웹쉘 가능성이 존재한다고 판단하고, 분석 명령을 통해 웹쉘 탐지 시스템으로 전송한다. 웹쉘 탐지 시스템은 이에 대한 분석을 실시하고 분석결과를 생성하여 XML 형태로 전송한다. 분석결과는 모두 시큐어코딩 DB에 저장되며, SQL 쿼리를 이용하여 항시 웹쉘 분석의 결과 및 패턴 조회가 가능하며 리포팅 문서로 제공 또한 가능하도록 한다.

3.2 운영 단계

운영 단계에서는 사용자가 소스코드 보안약점 진단 도구에 소스코드를 업로드 하면, 웹쉘 분석을 위해 업로드한 소스코드를 저장 후 취약점 분석을 실행한다. 이때, 사용되는 분석 방법은 파일 동기화 단계와 같이 취약점 DB 및 소스코드 보안약점 분석도구를 활용한다. 소스코드에 대한 취약점 분석 과정을 통해 웹쉘 가능성이 있다고 판단되면, 분석 명령을 통해 웹쉘 탐지 시스템으로 전송하고, 웹쉘 탐지 시스템은 이에 대한 분석을 실시한 후 분석결과 XML 형태로 전송한다. 해당 결과는 모두 소스코드 보안약점 진단도구 DB에 저장되며, SQL 쿼리를 이용하여 항시 웹쉘의 패턴 및 결과 조회 가능하며 리포팅 문서로 제공 가능하다. 소스코드에 문제가 없다면, 자동배포 여부를 판단하고, 취약점 점검에 대한 정책을 통과한다면 소스코드 보안약점 진단도구 Agent로 전송한다. 웹 서버에서는 실시간으로 위변조 모니터링 및 웹쉘 분석, 탐지결과를 생성하여 웹쉘 Manager로 전송하며, 웹쉘 Manager는 해당 결과를 악성코드 탐지 도구 DB로 전송한다.



<그림 4> 운영 단계 Process Flow

3.3 리포팅

소스코드 보안약점 진단도구 서버에서 기본적으로 기존 시큐어코딩 도구와 마찬가지로 소스코드 보안약점 진단에 대한 리포팅을 제공한다. 또한 기존 시큐터코딩 도구와는 달리 웹쉘 분석 결과 화면을 통해 웹쉘 분석 결과를 확인할 수 있다. 웹쉘 분석 결과에는 프로젝트 명, 웹 쉘 파일명, 탐지된 웹 쉘 패턴 명칭, 해당 웹 쉘 패턴에 대한 설명이 포함된다. 또한 웹쉘 패턴 명칭과 설명은 추후 패턴 DB 갱신 문제를 피하기 위해 웹쉘 탐지도구 매니저 서버에 직접 조회를 하도록 한다. 향후 시스템의 개선을 통해 완성도를 높이고 소프트웨어의 안전한 운영을 보장할 수 있는 시스템이 될 수 있도록 지속적인 연구가 필요하다.

Ⅳ. 결론

시큐어코딩은 개발단계 외의 단계에서 적용이 필요하다. 특히 웹 서버 환경은 외부와 커뮤니케이션이 빈번하고 소스코드의 변경여부를 예측할 수 없기 때문에, 실시간으로 모니터링하여 시큐어코딩을 적용해야 한다.

또한 웹 서버 환경에서 웹쉘과 같은 보안취약점이 발생할 경우 다양한 피해를 야기할 수 있기 때문에 이러한 보안취약점의 발생을 방지하기 위해 실시간으로 소스코드 보안약점 진단도구의 시큐어코딩 적용이 필요하며 시큐어코딩 만으로는 웹쉘에 대한 완전한 방지가 어려우므로 웹쉘 탐지도구와 상호작용을 통해 웹쉘에 대한 점검을 함께 병행하여 웹쉘에 대한 안전성 역시 확보할 수 있도록 하였다. 결론적으로 현재 시행중에 있는 시큐어코딩 지침을 만족하면서 웹 서버 환경의 운영에서 발생할 수 있는 취약점 사전 방지와 실시간 모니터링을 통해 기존에 예측하기 어려운 웹쉘과 같은 취약점 및 해커의 공격을 예방할 수 있도록 하였다.

참고문헌

[1] 한국인터넷진흥원, “2014 한국인터넷백서,” 2015, p. 494.

[2] 한국인터넷진흥원, “2008년 5월 인터넷침해사고 동향 및 분석 월보,” 2008, p. 22.

[3] 조성규・전문석, “개인정보 보호를 위한 조직구성과 관리체계에 관한 표준화 모델링,” 디지털산업정보학회지, 8권, 3호, 2012, pp. 33-39.

[4] 서우석・전문석, “보안이벤트 사이의 상관분석 기법을 이용한 조기위험경보시스템의 설계,” 디지털산업정보학회지, 8권, 1호, 2012, pp. 65-72.

[5] Da-Hye Jung, Jin-Young Choi, Song-Hee Lee, “Nuclear-related Software analysis based on secure coding.,” Journal of the Korea Institute of Information Security and Cryptology, 23권, 2호, 2013, pp. 243-250.

[6] 김준호, “소스코드 보안약점 탐지를 위한 정적도구의 활용과 기호실행 엔진을 활용한 개선 방안 연구,” 숭실대학교, 서울, 2015, p. 5.

[7] 박우열・양일권・손창환・한경숙・표창우, “PHP 시큐어 코딩 규칙의 개발을 위한 보안 취약점 분석,” 2014 한국정보과학회 제 41 회 정기총회 및 동계학술발표회, 2014, pp. 750-752.

[8] 안준선・이은영・창병모, “SW 개발보안을 위한 보안약점 표준목록 연구,” 정보보호학회지, 25권, 1호, 2015, pp. 7-17.

[9] 김정숙, “소프트웨어 보안을 위한 시큐어 코딩,” 한국콘텐츠학회지, 11권, 4호, 2013, pp. 56-60.

[10] Byung-Ha Choi, Kyung-San Cho, “An Improved Detecting Scheme of Malicious Codes using HTTP Outbound Traffic,” Journal of The Korea Society of Computer and Information, 14권, 9호, 2009, pp. 47-54.

[11] 박남열・김용민・노봉남, “우회기법을 이용하는 악성코드 행위기반 탐지 방법,” 정보보호학회논문지, 16권, 3호, 2006, pp. 17-28.

[12] 허준호・홍명호・이정민・서경룡, “근거리 통신망에서의 DDoS 봇넷 탐지 시스템 구현,” 멀티미디어학회논문지, 16권, 6호, 2013, pp. 678-688.

[13] 손유승・남길현・고승철, “스피어 피싱 대응을 위한 관리적 보안대책에 의한 접근,” 한국정보통신학회논문지, 17권, 12호, 2013, pp. 2753-2762.

■ 저자소개 ■

김 범 용
Kim Bumryong

2014년 9월~현재 숭실대학교 컴퓨터학과 석사과정
2014년 8월 국가평생교육진흥원 컴퓨터공학(공학사)

관심분야 : 핀테크, RFID, 시큐어코딩
E-mail : gflawer@ssu.ac.kr

최 근 창
Choi Keunchang

2014년 9월~현재 숭실대학교 컴퓨터학과 석사과정
2014년 8월 국가평생교육진흥원 컴퓨터공학(공학사)

관심분야 : 핀테크, IoT, 시큐어코딩
E-mail : muziag@ssu.ac.kr

김 준 호
Kim Joonho

2011년 7월~현재 ㈜이븐스타 보안연구소 / 이사
2015년 2월 숭실대학교 정보보안학과(석사)
1991년 2월 서강대학교 물리학과(이학사)

관심분야 : 취약점 분석, 시큐어코딩, 보안
E-mail : muziag@ssu.ac.kr

석 상 기
Suk Sangkee

1982년 7월~현재 서울과학기술대학교 컴퓨터공학과 정교수
1993년 홍익대학교 (이학박사)

관심분야 : 데이터베이스 및 설계, 객체/분산 데이터베이스
E-mail : sksuk@seoultech.ac.kr

논문접수일: 2015년 11월 15일
수 정 일: 2015년 11월 27일
게재확정일: 2015년 12월 4일