산업융합연구 제21권 제3호 pp. 181-188, 2023
ISSN 2635-8875 / e-ISSN 2672-0124
DOI : https://doi.org/10.22678/JIC.2023.21.3.181

eGovFrame 보안 분석 및 대응 방안에 관한 연구

박중오*
성결대학교 파이데이아학부 조교수

A Study on eGovFrame Security Analysis and Countermeasures

Joong-oh Park*
Assistant Professor, Division of Paideia, Sungkyul University

요 약 전자정부 표준 프레임워크는 국내 정부/공공기관 등 웹 환경 개발을 위한 공통 컴포넌트 재사용, 표준 모듈의 연계와 종속성 해소 등 전반적인 기술을 제공하고 있다. 그러나, 획일화된 개발 환경은 코어 버전에 따른 구버전 업데이트 문제와 해킹이나 컴퓨터 바이러스 등에 의한 개인정보와 기밀정보 유출 가능성이 존재한다. 본 연구는 국내 eGovFrame을 운영하는 웹사이트 중심으로 보안 취약성을 직접 분석한다. 내부 프로그래밍 언어 소스 코드 수준에서 취약점을 분석/분류한 결과, 대표 보안 취약성과 연계되는 5개 항목을 다시 추출할 수 있었다. 이에 대한 대응책으로, 2단계(1, 2차)를 통한 보안 설정과 기능 그리고 보안 정책을 설명한다. 본 연구는 향후 전자정부 프레임워크 보안 기능 개선하고 서비스 활성화에 이바지하고자 한다.

키워드 : 전자정부 프레임워크, 표준 프레임워크, 웹 보안, 스프링, eGovFrame

Abstract The e-Government standard framework provides overall technologies such as reuse of common components for web environment development such as domestic government/public institutions, connection of standard modules, and resolution of dependencies. However, in a standardized development environment, there is a possibility of updating old versions according to core versions and leakage of personal and confidential information due to hacking or computer viruses. This study directly analyzes security vulnerabilities focusing on websites that operate eGovFrame in Korea. As a result of analyzing/classifying vulnerabilities at the internal programming language source code level, five items associated with representative security vulnerabilities could be extracted again. As a countermeasure against this, the security settings and functions through the 2 steps (1st and 2nd steps) and security policy will be explained. This study aims to improve the security function of the e-government framework and contribute to the vitalization of the service.

Key Words : E-government framework, Standard framework, Web security, Spring, eGovFrame

*Corresponding Author : Joong-oh Park(pjo21@naver.com)
Received December 15, 2022
Revised January 13, 2023
Accepted March 20, 2023
Published March 28, 2023

1. 서론

표준 프레임워크는 미리 잘 설계된 서버와 함께 S/W 개발 및 유지보수, 기능 모듈 독립성, 효율적 상호운용성 등 다양한 장점을 제공한다. 국내 “전자정부 표준프레임워크”는 공공·민간 정보화 사업 적용으로 행정, 의료, 복지, 방송, 주민, 교육, 산림, 보험 등 전자정부 등 약 5,324여 개 웹사이트에 표준 프레임워크를 적용했다[1]. 공식 포털을 통해서 서버 구축/개발 환경을 배포하고 있으며 초기 2.7버전 이후 최신 4.0(Spring Framework 5.4.X) 버전을 지원하고 있다[2]. 문제는 수년 내에 개발된 환경이 4.0 버전보다 하위 버전으로 구버전의 웹사이트는 수년 사이 알려진 Log4J, 원격코드 실행, web.xml 등 해킹에 안전하지 않다[3, 4]. 해킹의 주요 원인은 특정 플랫폼 환경에 보안 기능을 의존하여 다양성이 필요한 새로운 보안 기술 적용이 어렵기 때문이다[5]. 최신 버전의 보안 기능은 확장 모듈을 개발자가 직접 수정/개발해야 한다. 결국 새로운 보안 취약점 대응에 많은 인력 및 시간을 투자해야 한다.

본 논문은 현재 국내 표준 프레임워크(eGovFrame)를 중심으로 4.0 버전 전/후로 알려진 보안 취약점을 직접 분석한다. 이후 대표 보안 취약점들을 재분류하고, 자동화된 분석 도구(Tool)와 함께 직접 소스 코드(Source Code)를 분석하였다. 분석 결과, 가장 위험성이 높은 취약성을 대상으로 보안 기능 및 정책 업데이트 등 세부 대응 방안을 설명한다. 본 논문의 구성은 다음과 같다. 2장 관련 연구는 표준 프레임워크과 대표 보안 취약점 현황을 분석한다. 3장은 소스 코드 수준 분석, 4장은 취약점별 대응 방안을 설명하고 5장 결론으로 마친다.

2. 관련 연구

2.1 전자정부 프레임워크 적용 현황 분석

Table 1은 5년 이내 표준 프레임워크 적용 사례(2022년 7월 기준 850건)를 직접 통계 분석한 결과이다[6].

Table 1. E-Government Standard framework status

Year	Websites(number)	Statistics(Ratio)
-	742	87.2%
2018	38	4.4%
2019	21	2.4%
2020	24	2.8%
2021	24	2.8%
2022	1	0.1%

2018년도 기준 표준 프레임워크를 적용한 기관/지자체/기업이 약 12%로 나타났다. 이는 기존 웹사이트(약 87%)가 대부분 3.6 버전(2017년 3월 공개) 이전의 구버전 환경임을 알 수 있다. 문제는 3.6 버전 이전 공통 컴포넌트 보안 모듈은 알려진 보안 취약점이 많고, 시큐어 코딩(Secure Coding) 또한 적용되지 않았다는 점이다. Table 2는 표준 프레임워크 버전 3.6(17년 3월 공개)부터 4.0(22년 3월 공개) 버전까지 공개된 스프링 프레임워크 보안 취약점 CVE(Common Vulnerabilities and Exposures) 항목을 정리한 결과이다[7].

Table 2. Standard framework CVE list

Version	Module	CVE List	Category
3.6	4.1.2	CVE-2014-3625 CVE-2016-5007 CVE-2019-3795	Directory Traversal Attacks, Invalid URL access, Unsafe random function
3.7	4.2.4		
3.8	4.3.16	CVE-2018-1257 CVE-2018-1270 CVE-2018-1275 CVE-2018-15756 CVE-2020-5421	DDOS(Distributed Denial of Service), Remote Code Execution, Session Fixation(Hijacking)
3.9	4.3.22		
3.10	4.3.25		
4.0	5.3.6	CVE-2021-22060 CVE-2021-22096 CVE-2021-22112 CVE-2021-22119 CVE-2022-22950 CVE-2022-22965 CVE-2022-22968	DDOS(Distributed Denial of Service), XSS(Cross Site Scripting), Invalid Authorization and Elevation(Bug)

모듈은 표준 구성에서 정의하는 스프링 프레임워크 코어 버전을 의미한다. 4.X 버전부터 5.3.X 사이 대표 CVE 취약점 유형을 정리하면 5가지(디렉터리/URL 접근, DDOS, 원격 코드 실행, 세션 탈취 및 우회, XSS) 유형으로 분류된다. 최근 Spring4Shell 제로데이 취약점으로 수백만 대 서버 해킹에 대하여 공개 소스 소프트웨어(OSS)에 대한 보안 취약점에 엄격한 테스트, 체계적인 방법론 등 다양한 연구가 진행되고 있다[8-10].

주요 원인은 스프링 프레임워크 내부 RCE 취약점(CVE-2022-22965)이다. 보안 공지에 따르면 JDK(Java Development Kit)의 버전 9 이상을 사용하는 모든 환경이 영향을 받기 때문에 스프링 코어의 버전을 5.3.18, 5.2.20으로 업데이트 권고하고 있다. 표준 프레임워크의 실행환경(업그레이드 가이드)에 따르면 3.6버전 기준으로 4.0까지 5번의 업데이트 과정을 직접 수행하고 테스트해야 한다[10]. 이는, 지속적인 보안 취약점에 빠르게 대응하기 어렵다는 것을 의미한다.

2.2 웹사이트 보안 기능 분석

Table 3은 표준 프레임워크가 적용된 웹사이트와 버전 별 보안 기능을 나타낸다[11-15]. 현재 운영되고 있는 웹사이트를 대상으로 직접 내/외부 적용된 보안 기능을 확인한 결과이다. POST 방식의 데이터 전송을 요구하는 모든 폼 창은 기본적으로 보안 서버를 위한 HTTPS 프로토콜로 전송된다. 기본적으로 로그인 및 인증 서비스를 구현하고 있는데, 각 웹사이트가 구현하는 방식에 차이가 존재했다.

Table 3. Website security features(Analyze)

Version	Website
	Security Features
-	Municipality - https://www.yangju.go.kr/
	Identity Verification(Mobile, Simple, I-PIN, Public Certificate)
3.8	Education - https://www.karts.ac.kr/
	Login Service(ID/PASSWORD)
3.9	Municipality - https://www.huic.co.kr/
	Identity Verification(Mobile, I-PIN) Intranet(ID/PASSWORD)
3.10	Finance/Insurance - https://www.kodit.co.kr/
	Login Service(ID/PASSWORD, Simple, Public Certificate), Keyboard Security(Module)
4.0	Culture/Sports - https://touredu.visitkorea.or.kr/
	Login Service(SSO integration, SNS linkage, etc.)

프레임워크 버전 별 구축 된 대표 웹사이트 조사(무작위 선정) 결과 인증 방식은 5가지 방식(아이디/패스워드, 공인인증서, 휴대폰, 간편인증, 아이핀)으로 구분된다. 버전(3.X 포함)에 상관없이 보안 기능에 특별한 기준이과 일관성이 없다. 문제는 표준 프레임워크를 위한 필수 보안 개발 요구사항이 부재하여, 웹사이트가 아닌 외부 인증 모듈에 의존하고 있다. 표준 프레임워크에 보안 가이드에 따르면 세부적인 보안(API, 암호알고리즘, 세션 등) 기능 구현에 대한 설명이 부재하다. 즉, 직접 보안 기능을 개발해도 내/외부 보안 기능 검증 및 안전성을 보장하지 않는다[16].

사용자 로그인 기능의 경우 프레임워크 하위(Spring Security)에서 제공하는 인증(Authentication) 객체의 안전한 구현 등이 요구된다. 최근 전자서명법 개정안 통과로 기존 공인인증서 이외 간편인증 방식의 비중이 점차 높아지고 있다. 때문에, 표준 프레임워크 보안 가이드는 개발 세부 설명과 외부 보안 모듈에 대한 검증방안을 새로 업데이트가 필요하다고 분석된다. 기존 입력 오류 악용 및 제로데이 피싱, 원격코드 실행 취약점 등 관련 진행중인 연구들은 이를 추적/탐지하기 위한 새로운 서버 구축과 보안 패치의 중요성 및 필요성에 관해 설명하고 있다[17-19].

2.3 핵심 보안 취약점 분석/추출

Table 4는 표준 보안 가이드에서 보안 검증 항목 7개 영역과 세부 항목을 나타낸다[20].

Table 4. Standard security guide vulnerability check

	Security Vulnerabilities
Input data validation	SQL injection, XSS(Cross Site Scripting)
Security function	Invalid Authorization and Settings, Encryption Algorithm, weak Password, Hash Function
Time and Status	TOCTOU Race Condition, Invalid loop or recursive function
Error handling	Information Disclosure, Exception Handling
Code error	Null Pointer Dereferencing, Invalid Resource Release
Encapsulation	Session Data, Debug Code, and Access Directives
API	DNS lookup, Weak API

Table 5는 표준 프레임워크 CVE 취약점 분석 결과 5가지(디렉터리/URL 접근, DDOS, 원격 코드 실행, 세션 탈취 및 우회, XSS)를 비교적 유사한 시큐어 코딩 항목으로 재분류한 결과이다. CVE 위협 수준은 NVD(National Vulnerability Database)에서 제공하는 CVE 데이터베이스의 메타 점수 기준으로 직접 분류한 결과이다[21].

Table 5. Secure coding vulnerability check

Threat level	Base and CVE	Secure Coding
-	Framework Security(Core)	Patch & Setting
1	DDOS	Input Data Validation
2	XSS	
3	Session Hacking	Encapsulation
4	Remote Code Execution	Security Function(Settings)
5	Directory/URL Access	Security Function(Settings)

① 위협 수준 (1) DDOS 공격 : 입력 데이터 검증 유형의 세부 항목인 SQL 삽입 및 XSS 스크립트 공격에 대응해야 한다.

② 위협 수준 (2) XSS : DDOS 점검 항목에 포함되기 때문에 같은 보안 위협으로 정의될 수 있다.

③ 위협 수준 (3, 4) 세션 해킹과 원격코드 실행 : 캡슐화/보안 기능의 세션 데이터와 함께 암호알고리즘, 취약한 패스워드, 잘못된 인증/인가 등에 대응해야 한다.

④ 위협 수준 (5) 디렉터리/URL 접근 : 가장 보안 수준이 낮았지만, 공통보안 항목으로써 대응해야 한다.

표준 프레임워크에서 대표 CVE 취약점과 보안 가이드를 참고하여 가장 위협이 될 수 있는 보안 취약점 분류 5가지 유형을 추출했다. 내부 서버 보안 이외 최우선으로 대응해야 하는 취약점은 입력 데이터 검증의 보안 기능 구현이고, SQL과 XSS 공격과 관련이 가장 높았다.

3. 소스 코드 수준 보안 분석

3.1 자동화 툴 분석 결과

표준 프레임워크 구축의 약 87% 서버가 시큐어 코딩이 적용되지 않은 3.6 이하 버전이다. 실제 웹사이트 코드 분석을 위해 대표 5가지 보안 취약점과 연계되는 보안 취약점을 상세 분석했다. Table 6은 보안 기능 S/W 및 스크립트 기반으로 취약점을 분석한 결과이다. 스캐너를 통한 자동화된 소스 코드 분석은 OWASP ZAP, Burp Suite, NMAP, ModSecurity, Metasploit 등을 활용했다 [22-26].

Table 6. Source code vulnerability analysis(Direct)

Version	Website
	Security Features
-	Municipality - https://www.yangju.go.kr/
	Anti-CSRF, Click jacking, Cookie, File Inclusion, Vulnerable JS library
3.8	Education - https://www.karts.ac.kr/
	Anti-CSRF, Click jacking, X-Frame Header, URL Rewrite(Session), Referrer Exposes(Session), Cookie, File Inclusion, Vulnerable JS library
3.9	Municipality - https://www.huic.co.kr/
	Anti-CSRF, Click jacking, X-Frame Header, File Inclusion, Vulnerable JS library
3.10	Finance/Insurance - https://www.kodit.co.kr/
	Anti-CSRF, Click jacking, X-Frame Header, File Inclusion, Vulnerable JS library
4.0	Culture/Sports - https://touredu.visitkorea.or.kr/
	Path Traversal, Anti-CSRF, Click jacking, X-Frame Header, File Inclusion, Vulnerable JS library

전체 웹사이트에서 공통 취약점으로 다음과 같은 5가지 보안 취약점으로 재분류하고 추출되었다.

① Anti-CSRF : 사용자의 권한을 획득하여 POST 등 중요 전송 데이터 입력/수정할 수 있다. XSS의 일종으로 사용자의 인증된 세션을 악용할 수 있다.

② Click jacking : html 페이지의 특정 영역을 클릭하여 의도하지 않은 악성코드 링크 등 URL의 실행(인증)을 유도할 수 있다.

③ Cookie : 안전하지 않은 쿠키 플래그 설정으로 XSS 공격과 함께 암호화 미적용과 자바스크립트 접근 등 다양한 형태로 중요 파라미터가 노출될 수 있다.

④ File Inclusion : 검증되지 않은 URL 사용은 LFI(로컬)/RFI(원격)로 악성 스크립트를 직접/원격 삽입 공격할 수 있다.

⑤ Vulnerable JS Library : 안전하지 않은 자바스크립트 라이브러리는 보안 취약점에 노출될 수 있다.

앞서 추출된 5가지 대표 취약성과 실제 코드 분석 결과 Anti-CSRF, Cookie, Click jacking 등이 SQL과 XSS 공격의 원인이 되는 악성코드 유형임을 확인했다.

3.2 추가 분석(직접)

SQL, XSS 공격의 원인이 될 수 있는 취약점으로 쿠키, 헤더, URL 등 보안 취약점을 직접 분석했다.

① Path Traversal : 경로 접근 문제로 인한 특정 디렉터리로 접근 및 파일 다운로드를 할 수 있다.

② X-Frame Header : 헤더 보안 미설정으로 인해 클릭재킹, XSS 공격 등에 노출될 수 있다.

③ URL Rewrite(Session) : Rewrite로 인해 현재 통신 연결의 세션 ID를 추적할 수 있다.

④ Referrer Exposes(Session) : 요청된 페이지의 링크 이전의 주소를 나타내는 헤더를 조작할 수 있다.

HTML 헤더와 Referer 등 기본적인 서버 보안 설정의 부재하고, POST 전송 방식으로 전송하는 대부분의 HTML 폼(Form)이 시큐어 코딩이 적용되지 않았다. 표준 프레임워크의 버전과 상관없이 웹사이트의 보안 패치 관리가 전반적으로 미흡한 것으로 나타났다.

4. 표준 프레임워크 보안 대응 방안

4.1 보안관리 정책

국내 공공기관 웹사이트는 암호화, 비밀번호 찾기 기능 등 보안성 점검(HTTPS, 개인정보)을 지속하여 수행해 왔다[27]. 보안 서버 구축 및 확대는 실제 십수 년 전 이전부터 도입되었고, 비밀번호 찾기 기능은 서버 보안성보다는 개인정보 관리 차원의 영역에서 추가되었다. 행정기관

또는 공공기관 정보시스템 구축·운영 지침 제8조(보안성 검토 및 보안관리)와 제7장 검사 및 운영 내용에 따르면 웹사이트 보안관리의 명확한 검사 주기 및 범위가 정의되어 있지 않다[28]. 또한 전자정부 웹사이트 품질관리 지침에 따르면 관리 항목 신뢰성의 개인정보 노출 여부 점검 이외에는 추가 보안점검 계획이 존재하지 않는다[29].

즉, 기존 시행해온 웹사이트 보안성 점검의 범위는 극히 제한적이기 때문에, 보안점검에 대한 새로운 변화가 필요한 실정이다. Table 7은 본 연구에서 제안하는 표준 프레임워크의 보안 정책들을 나타낸다.

Table 7. Standard Framework Security Policy

Security Policy	Description
Verification Cycle	Initial server Construction(1st) 1st inspection Branch(continuous) Year including 2nd(once)
Verification Scope	1st : Apply/Update Standard Framework(3.6 or higher)
	2nd : OWASP TOP 10, Ministry of Public Administration and Security S/W Vulnerability Checklist, etc. Source Code Check/report submission
1st	Server Security settings 5 items, etc. Automation
2st	Source Code level 11 items, etc. Automated, Additional Inspection(human)
Submit	Quarterly(2 times), in case of emergency Notice(Additional)

① 검증 주기 : 초기 프레임워크 기반으로 서버 구축과 함께 반드시 1차 검증(보안 설정)을 수행해야 한다. 2차 검증은 SSL 인증서의 최소 유효기간인 1년을 기준으로 연 1회 검사한다. 이외 중요한 보안 공지(이슈)가 되었을 때, 즉시 보안성 점검을 수행한다.

② 검증 범위 : 현실성을 고려하였을 때, 최상위 기관에서 전체 웹사이트 점검을 단독 수행은 어렵다. 실제로 많은 시간과 인력이 필요로 하므로 점검 과정을 분할했다. 1, 2차로 검증 범위를 세분화한다.

③ 1차 검증 : 표준 프레임워크 적용 유무와 내부 코어 버전을 확인하여 서버 보안 설정 5개와 내부 서버 코어 버전(보안 모듈)을 확인한다. 기본적으로 보안 패치 상태를 점검하는 모듈로 개발하여 배포한다.

④ 2차 검증 : 보안관리자는 소스 코드 수준 보안 취약점 항목 11개를 자체 검사한다. 스캐너 검사 이후 사람이 직접 분석 확인하는 과정으로 수행된다. 보안관리자는 보고서 분석 결과를 주기적으로 제출한다.

1차 검증 기능 수준에서 보안 설정 검사(완전 자동화), 2차 검증 기능 수준에서 소스 코드 검증(반 자동화) 순서로 새로 개발하여 배포해야 한다. 소스 코드 검증의 경우 자동점검 솔루션 수행 이후 사람이 직접 확인해야 하는 작업이 필요하므로, 반 자동화로 정의한다.

4.2 기술적 보안 대응(1, 2차 검증)

표준 프레임워크 기관은 보안 공지를 통해 새로운 업데이트/검증 확장 모듈을 배포한다. Table 8은 표준 프레임워크의 개발 환경(JAVA + WAS(아파치 서버, 탐캣)) 기준 1차 보안점검 항목을 나타낸다.

Table 8. Server internal security(Setting)

Vulnerability	Security Countermeasures
Anti-CSRF	Referer Verification Security Token
Clickjacking	Script Tag Filtering X-Frame Header
X-Frame Header	X-Frame-Options
Referer Exposes(Session)	Referer Verification

① Anti-CSRF : 각 요청 페이지의 헤더(Header)의 Referer를 체크하고, 서버 내부 스프링 보안의 애플리케이션 속성에 CSRF token을 설정하여 사용한다. 일반적으로 서버 설정으로 도메인 검사와 랜덤한 보안 토큰을 구현하여 사용한다.

② Click jacking : Iframe 등 안전하지 않은 태그 사용을 검사하고, Content-Security-Policy 표준 리소스 정책을 제한하여 사용한다. (2가지 모두 필수 : 호환성) 헤더 보안 설정을 강화하기 위해 Helmet 모듈을 추가 설치하고, 내부 Click jacking 방어를 위한 Frameguard를 추가 활성화한다.

③ X-Frame Header : X-Frame-Options 보안 설정을 통해, 주요 보안 취약점인 XSS 공격에 대해 쿠키와 세션 등을 안전한지 검사한다. 앞서 설치된 Helmet 모듈의 CSP(Content Security Policy)와 헤더 옵션들을 추가 활성화한다.

④ Referer Exposes(Session) : 세션 내부 참조가 외부로 노출되는지 검사한다. Anti-CSRF와 같이 Header Referer 검사를 추가 적용한다.

주요 대표 4가지 항목은 서버 보안 설정으로 해결할 수 있는 항목들이다. 서버 관리자가 직접 내부 웹 서버 설정

을 변경하는 방식보다, 표준 프레임워크의 코어 보안 모듈을 업데이트하는 방식이 적절하다. 문제는 아파치 서버에서 제공하는 기본 보안 설정은 항목이 다양하지 않다. XSS 공격이 선행되었다고 가정하면, 도메인 검사 우회나 토큰 변조/위조가 가능하다.

가장 안전한 방법은 로그인 사용자가 설정한 패스워드 또는 OTP(One Time Password)를 추가 검증하여 2채널 인증을 적용하는 방법이다. 현재 네이버, 다음 등 대부분 포털이 새로운 사용자 로그인(도메인 접근)에 대하여 대부분 메일 인증, 또는 모바일 OTP를 추가 사용하고 있다. 서버 보안 설정을 강화하는 것만으로는 안전성 강화에 한계가 있다는 것이다. Table 9는 표준 프레임워크의 개발환경 기준 2차 보안점검 항목을 나타낸다.

Table 9. Implement source code level security

Vulnerability	Security Countermeasures
Cookie	Cookie(Set) Security Session
File Inclusion	URL Verification Input Data Verification
Vulnerable JS	JAVA Version Update JAVA Script File Update
Path Traversal	Input Data Verification Directory Options
URLRewrite(Session)	Redirect URL Verification White List(Domain)

① Cookie : 세션 해킹에 따라 노출될 수 있는 쿠키를 보안 설정(Http Only, secure)하고, 쿠키 이외 대체 수단으로써 안전한 세션(Http Session)을 구현하여 사용한다. 쿠키의 경우 반드시 세션 id만 저장하고, 실제 값을 저장하지 않도록 해야 한다. 세션의 경우 중요 페이지에서 반드시 체크하고, 필요 없는 페이지의 경우 세션을 false(비활성화) 구현할 수 있다. (필수 검사 : 암호화, 유효기간 설정 여부)

② File Inclusion : 중요 접근 디렉터리와 URL을 필터링하고, 업로드 파일을 검사(크기 및 개수, 종류, 이름/경로, 권한 수정)한다. 웹 서버 폴더에 할당된 저장공간을 차단/제한하여 관리하는 것이 가장 안전하다. (필수 : 확장자 우회 방지, 아파치 웹 서버 내 htaccess 파일 무결성 검사)

③ Vulnerable JS : 표준 프레임워크에 적절한 최신 자바 JDK 버전인지 검사한다. 이외 자체 개발한 자바스크립트와 jQuery, Node 등 추가 외부 라이브러리 등의 취약한 버전 유무를 확인하여 자바스크립트의 안전성(XSS 필터 등)을 점검한다.

④ Path Traversal : 중요 경로 접근에 대해 제한(설정)하고, Path를 조작할 수 있는 특수문자 필터링 검사한다. 이외 프로그래밍 언어에서 시스템 함수(API)의 원격 실행을 차단한다. 또한 특정 접근할 수 있는 디렉터리 내 파일 권한을 제한/분리할 수 있다. 이 부분은 1차 점검 과정에서 서버 보안 설정의 스크립트에 포함하도록 한다.

⑤ URL Rewrite(Session) : Rewrite 모듈의 활성화 유무를 검사하고, 조작된 세션 ID를 이용하는 특정 URL Redirect를 차단한다. 이 부분은 1차 점검 과정에서 모듈 활성화에 대한 점검을 스크립트에 포함하도록 한다.

2차 점검 과정에서 5가지 취약점 중 3가지 항목은 반드시 소스 코드 수준의 점검이 필요하다. 웹사이트 내 전체 소스 코드를 대상으로 검사하려면 자동화된 스캐너가 필수이다. 이외 취약점에 대해 사람이 직접 분석해야 한다. 이에 따라 기관 측은 점검할 수 있는 공용 스캐너와 점검 스크립트를 업데이트로 배포해야 한다. 이외 플랫폼 종속 문제를 해결하려면 서버 내부 스크립트 형태 배포가 적절하다. 서버 개발자는 공지 확인을 통해 스크립트를 내려받아 소스 코드를 직접 검수하고, 대응에 필요한 코드를 수정하고 서버를 업데이트해야 한다. Table 10은 기존 표준 프레임워크와 제안한 정책/기술적 대응의 비교를 간략히 나타낸다.

Table 10. Comparison of security policy/technical countermeasures

	Current	Proposal
Policy	Security Update Advisory	Security Update required
	Limited inspection Scope	Subdivision and expansion of inspection Scope
	Inspection interval not Specified	Definition of maintenance intervals
Technique	Difficulty in comprehensive website Investigation	Deploy/manage Security verification module
	Lack of check item Definition	1st – Automated by scanner method
		Secondary inspection – direct inspection (semi-automated)

5. 결론

국내 주요 공공 웹사이트는 기관 주관하에 세부 보안 점검 및 진단 도구가 부재하여, 노후화된 기관 홈페이지의 경우 전반적으로 다양한 해킹 공격들에 대해 무방비로 노출되어 있음을 확인했다. 본 연구는 웹사이트 기반이 되는 표준 프레임워크(eGovFrame)를 대상으로 보안 취약점을 분석했다. 구버전부터 3.8, 3.9, 3.10, 4.0 등 버전별 대표 취약성 5가지 분석 결과, 자동화된 스캐너 수준에서도 대부분 취약점이 발견했다. 따라서 앞으로 기관에서 보안 기능으로 개발해야 할 주요 보안 정책에서 보안 강화를 위한 절차를 정의하였고, 5가지 보안 취약점에 대한 상세 보안 기능 개발/대응책을 설명했다.

기존 보안점검 서비스 및 다양한 보안 가이드라인들은 초기 서버 구축 또는 웹사이트 리뉴얼(Renewal) 과정에서 도움이 될 수 있다. 그러나 현실적인 보안점검과 취약점 대응을 위해서는 선행 과제로써 보안 정책의 변화와 함께 취약점 분석 공통(표준)도구 등을 개발하여 배포해야 할 것이다. 특히, 자동화된 보안점검 도구의 배포는 기존 보안 업체/기관의 보안점검 서비스 이용을 활성화할 것이다. 향후 연구로서 공공기관 웹사이트 취약점 분석 범위를 확대(분야별, 이용률)하고 다양한 취약점에 대한 대응 방안을 추가 분석할 계획이다.

REFERENCES

[1] Standard Framework Portal. (2022). Introduction of the standard framework (applied cases – achievements), Retrieved from https://www.egovframe.go.kr/

[2] Standard Framework Portal. (2022). Download (Development Environment - Release Notes), Retrieved from https://www.egovframe.go.kr/

[3] Canitano, G. (2022). Development of framework for Attack/Defense Capture the Flag competition (Doctoral dissertation, Politecnico di Torino).

[4] Shcherbakov, M., Balliu, M., & Staicu, C. A. (2023). Silent spring: Prototype pollution leads to remote code execution in node. js. In USENIX Security Symposium 2023.

[5] Kim, S. S. (2020). [Diagnosis] E-Government Standard Framework, 'JAVA Only' have to change. Retrieved from https://www.comworld.co.kr/

[6] Standard Framework Portal. (2022). Introduction of the standard framework (applied cases – technical support details), https://www.egovframe.go.kr/

[7] NVD. (2022). Spring Framework CVE® List, Retrieved from https://www.cve.org/

[8] Krcert. (2022). Spring Java Framework Security Update Advisory, https://www.krcert.or.kr/

[9] Mohamed, H. M., & El-Gayar, O. (2022). Security Vulnerability Impact on Open Source: A Social Media Exploration. (AMCIS 2022 TREOS)

[10] Bai, S., Bøe, E. B., & Hegland-Antonsen, R. C. (2022). Efficiently Weaponizing Vulnerabilities and Automating Vulnerability Hunting (Bachelor's thesis, NTNU).

[11] YANGJU CITY. (2022). Yangju City Hall - website. Retrieved from https://www.yangju.go.kr/

[12] Korea National University of Arts. (2022). Korea National University of Arts - website. https://www.karts.ac.kr/

[13] Hanam Urban Innovation Corporation. (2022). Hanam Urban Innovation Corporation - website. https://www.huic.co.kr/

[14] KDIT. (2022). Korea Credit Guarantee Fund, Retrieved from https://www.kodit.co.kr/

[15] KTO. (2022). Korea Tourism Information - E-learning. Retrieved from https://touredu.visitkorea.or.kr/

[16] Im, Y. G. (2022). Expanded to 11 types of private certifications on public sites...Added Hana Bank and Dream certifications. Retrieved from https://zdnet.co.kr/

[17] Ashouri, M. (2019). Practical dynamic taint tracking for exploiting input sanitization error in java applications. In Australasian Conference on Information Security and Privacy, 494-513. Springer, Cham.
DOI : 10.1007/978-3-030-21548-4_27

[18] Ponta, S. E., Plate, H., & Sabetta, A. (2020). Detection, assessment and mitigation of vulnerabilities in open source dependencies.

Empirical Software Engineering, 25(5), 3175-3215. DOI : 10.1007/s10664-020-09830-x

[19] Jung, B.-M., Jang, J.-Y., & Choi, C.-J. (2019). Countermeasure of an Application Attack Scenario Using Spring Server Remote Code Execution Vulnerability (CVE-2018-1270). The Journal of the Korea Institute of Electronic Communication Sciences, 14(2), 303-308. DOI : 10.13067/JKIECS.2019.14.2.303

[20] Standard Framework Portal. (2020). Standard Framework Security Development Guide for E-Government SW Developers and Operators, Retrieved from https://www.egovframe.go.kr/

[21] NIST. (2020). National Institute of Standards and Technology - CVE Record Metadata, Retrieved from https://csrc.nist.gov/

[22] ZAP. (2022). OWASP ZAP(OWASP Zed Attack Proxy), Retrieved from https://www.zaproxy.org/

[23] PortSwigger. (2022). Burp Suite - Application Security Testing Software, Retrieved from https://portswigger.net/burp

[24] NMAP. (2022). Nmap Security Scanner, Retrieved from https://nmap.org/

[25] ModSecurity. (2022). SpiderLabs - ModSecurity, Retrieved from https://www.modsecurity.org/

[26] Metasploit. (2022). Metasploit - Penetration Testing Software, Retrieved from https://www.metasploit.com/

[27] Im, M, C. (2021). Personal Information Commission, major public institution website security check… "See HTTPS applied", Retrieved from https://www.ajunews.com/

[28] Ministry of Public Administration and Security. (2022). Guidelines for establishment and operation of information systems for administrative and public institutions, Retrieved from https://www.law.go.kr/

[29] Ministry of Public Administration and Security. (2021). Guidelines for Quality Management of E-Government Websites, Retrieved from https://www.law.go.kr/

박 중 오(Park, Joong Oh) [정회원]

- 2000년 7월 : 성결대학교 컴퓨터공학과 졸업
- 2003년 3월 : 명지대학교 전자계산교육 석사
- 2011년 8월 : 숭실대학교 컴퓨터공학 박사
- 2016년 3월~현재 : 성결대학교 조교수
- 관심분야 : Network security, Cryptography, PKI
- E-Mail : pjo21@naver.com