JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 29 No. 7, pp. 109-115, July 2024
https://doi.org/10.9708/jksci.2024.29.07.109

Analysis of Security Vulnerabilities and Personal Resource Exposure Risks in Overleaf

Suzi Kim*, Jiyeon Lee**

*CEO, Studio AABB, Daejeon, Korea
**Professor, School of Computer Science and Engineering, Kyungpook National University, Daegu, Korea

[Abstract]

Overleaf is a cloud-based LaTeX editor, allowing users to easily create and collaborate on documents without the need for separate LaTeX installation or configuration. Thanks to this convenience, users from various fields worldwide are writing, editing, and collaborating on academic papers, reports, and more via web browsers. However, the caching that occurs during the process of converting documents written on Overleaf to PDF format poses risks of exposing sensitive information. This could potentially lead to the exposure of users' work to others, necessitating the implementation of security measures and vigilance to caution against such incidents. This paper delves into an in-depth analysis of Overleaf's security vulnerabilities and proposes various measures to enhance the protection of intellectual property.

▸ **Key words**: Online collaboration tool, cloud security, Overleaf, LaTeX, vulnerability analysis

[요 약]

Overleaf는 클라우드 기반 LaTeX 편집기로, 사용자들이 별도의 LaTeX 설치나 설정 없이도 문서를 쉽게 작성하고 협업할 수 있는 플랫폼이다. 이러한 편의성 덕분에 전 세계 다양한 분야의 사용자들이 웹 브라우저상에서 학술 논문, 보고서 등을 작성하고 편집하며 협업하고 있다. 그러나 Overleaf를 통해 작성된 문서가 PDF로 변환되는 과정에서 발생하는 캐시 저장은 민감 정보 노출의 위험성을 내포하고 있다. 이로 인해 사용자의 저작물이 타인에게 노출될 가능성이 있으며, 이를 방지하기 위해서는 보안 대책과 주의가 필요하다. 이 논문은 Overleaf의 보안 취약성을 심층적으로 분석하고, 저작물 보호를 강화하기 위한 다양한 방안을 탐구하고 제안한다.

▸ **주제어**: 온라인 협업 도구, 클라우드 보안, 오버리프, LaTeX, 취약점 분석

• First Author: Suzi Kim, Corresponding Author: Jiyeon Lee
*Suzi Kim (kim.suzi.89@gmail.com), Studio AABB
**Jiyeon Lee (jiyeon@knu.ac.kr), School of Computer Science and Engineering, Kyungpook National University
• Received: 2024. 06. 03, Revised: 2024. 07. 04, Accepted: 2024. 07. 10.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

Overleaf[1]는 2012년부터 시작된 온라인 LaTeX 편집기로, 전 세계 천오백만 명 이상의 사용자가 웹 브라우저 상에서 학술 논문, 보고서 등의 문서 작성, 편집 및 협업을 진행하고 있다. Overleaf는 복잡한 LaTeX 작업을 웹 기반으로 제공하여 (Fig. 1) 사용자들이 별도의 LaTeX 설치나 환경 설정 과정 없이도 문서를 효율적으로 작성하고 협업할 수 있게 한다. 사용자는 웹 브라우저를 통해 언제 어디서나 접속하여 작업을 진행할 수 있으며, 실시간으로 공동 작업을 할 수 있다. 특히나, 연구자들이 논문 작성 과정을 보다 효율적으로 진행할 수 있도록 도와주는 플랫폼으로, 특히 복수의 저자들이 협업하여 논문을 완성해야 하는 경우에 유용하게 사용된다.

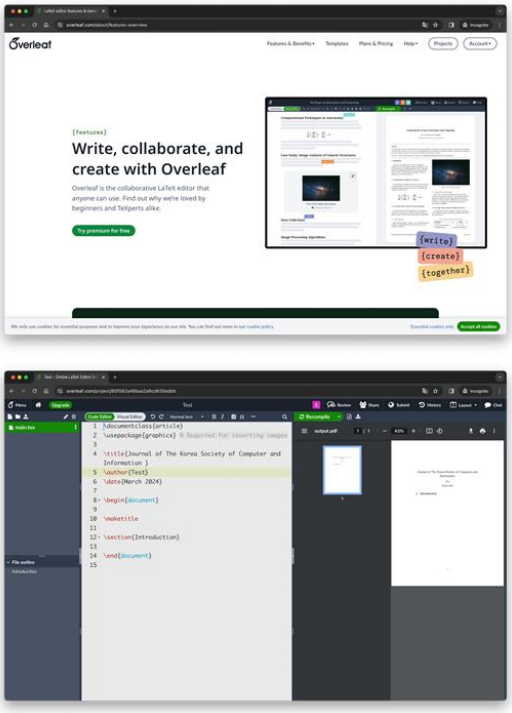


Fig. 1. Overleaf Service Interface

Overleaf에서 제공하는 다양한 기능 중 LaTeX 파일 처리에 해당하는 기본 기능은 크게 편집, 컴파일, 렌더링으로 분류된다. Fig. 2에서 보이는 것처럼 사용자가 편집기에서 LaTeX 문서를 편집한 뒤 컴파일을 요청하면 시스템 내부 컴파일러가 LaTeX 문서를 PDF 파일로 컴파일해 결과물을 자체 서버에 캐싱한다. 캐싱된 PDF 파일의 URL은 렌더러에 보내지며 사용자는 컴파일된 최종 PDF 결과물을 웹 뷰어에서 확인할 수 있다.

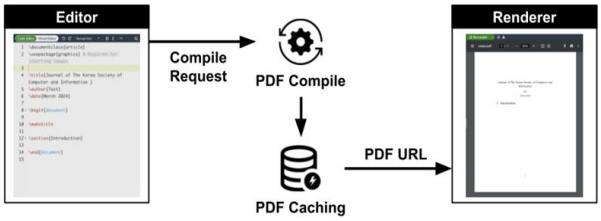


Fig. 2. Overleaf Backend System

이때, 편집에서 렌더링에 이르기까지의 과정이 하나의 장소에서 이뤄지지 않고 HTTP(S) Request에 기반한 통신을 통해 이뤄지기 때문에 통신 과정에서 Overleaf의 취약점이 발견될 수 있다. 본 논문에서는 Overleaf에서 제공하는 편집, 협업, 컴파일, 렌더링 프로세스의 흐름을 분석하고, 이 과정에서 발생할 수 있는 취약점을 조사하였다. 이를 바탕으로 인가받지 않은 사용자가 Overleaf의 PDF 결과물에 접근하는 두 가지 공격 시나리오를 제안한다. 첫 번째 시나리오는 컴파일된 Overleaf의 PDF 파일을 탈취하는 방법이며, 두 번째 시나리오는 세션 하이재킹을 통해 재컴파일 되는 PDF 파일을 탈취하는 방법이다.

두 가지 공격 시나리오를 Overleaf 시스템에서 재현한 결과, 타겟 사용자가 인지하지 못하는 사이에 Overleaf 내 최신 편집 내용을 공격자가 자유자재로 획득할 수 있다는 문제점이 드러났다. 이러한 취약점은 온라인 협업 도구의 신뢰성을 저하시킬 뿐만 아니라, 출판물의 지식 재산권 보호에도 심각한 위협이 된다. 이를 해결하기 위해 Overleaf의 민감 정보 보호와 데이터 보안을 강화하기 위한 세 단계의 대응책을 제안한다.

II. Related Work

LaTeX은 과학 논문을 작성할 때 사용되는 필수적인 조판 시스템이다. LaTeX은 사용자가 문서를 일반 텍스트로 작성하고, LaTeX의 매크로(또는 명령어)를 사용해 서식을 지정한 후, 사용 가능한 컴파일러 중 하나를 사용해 최종 PDF 문서를 생성하는 방식으로 작동한다. 최근 LaTeX이 악성 작업을 수행하는 데 쉽게 악용될 수 있음이 밝혀지면서 LaTeX을 통한 정보 추출, 명령어 및 악성 프로그램 실행, 서비스 거부 등 다양한 공격들이 꾸준히 소개되고 있다[2-6].

Checkoway 등은 \input 및 \include 매크로를 사용하여 사용자의 시스템에서 파일을 읽고 그 내용을 포함하는 TeX 파일을 생성하여 컴퓨터에서 민감한 정보를 유출

하는 전략을 소개했다[2-3]. Lacombe 등은 이전의 공격 방법이 데이터 유출 여부를 쉽게 노출시키는 단점을 개선하여, TeX 파일에 표시되는 도난 데이터를 숨기는 테크닉을 추가한 공격 기술을 소개하였으며[4], Overleaf, Papeeria 등의 온라인 협업 편집 환경에서 해당 공격이 유효함을 보였다. 이외에도, 무한 루프를 발생시키는 매크로를 통해 LaTeX 파일에서 서비스 거부 공격이 가능하다는 점이 제기되었으며[5], 온라인에서 다운로드된 LaTeX 실행 파일이나 템플릿을 통한 멀웨어 공격[5-6]의 가능성도 제시되었다.

이와 달리, 본 연구는 LaTeX의 온라인 협업 실행 환경에서 발생 가능한 웹 공격에 대해 연구한다. 온라인 협업 도구는 LaTeX 실행을 위한 필수 프로그램의 설치 없이 쉽게 실행 및 접근할 수 있다는 편리함을 제공하지만, 웹의 특성을 악용한 다양한 공격이 시도될 수 있다는 점에서 보안 관점의 설계가 필수적이다. 본 논문에서는 특히 개인 저작물과 같은 민감 정보를 포함하는 LaTeX 협업 플랫폼에서 URL Path Traversal 공격[7], Cross-site Request Forgery[8] 등 전통적인 웹 공격의 실행 가능성을 제시한다는 점에서 기존 연구와 차별점을 갖는다.

III. Overleaf Output Extract Attack

최근 웹 기반 온라인 협업 툴이 인기를 얻으면서 Overleaf, Papeeria, Authorea 등 LaTeX 컴파일을 지원하는 논문 작성 툴이 증가하고 있는 추세이다. 이들은 구현 정도에 차이가 있으나 URL을 통해 작업 공간에 엑세스하고 유저별 접근 권한 및 세션을 관리한다는 점에서 비슷한 구조를 공유한다.

본 절에서는 LaTeX의 온라인 협업 플랫폼 중 가장 인기있는 Overleaf 서비스에 초점을 맞춰 온라인 협업 과정에서 발생 가능한 취약점을 소개한다. 특히 접근 권한이 없는 악의적인 사용자가 정상 Overleaf 사용자의 작업 공간에서 컴파일된 PDF 문서를 탈취하는 구체적인 공격 방법을 소개한다.

1. Overleaf Overview

Overleaf는 온라인 협업 플랫폼으로 LaTeX 컴파일을 지원하여 전 세계 많은 연구자들에게 인기 있는 논문 작성 협업 툴이다. 사용자가 Overleaf에 가입하기 위해서는 사용자의 개인 이메일 주소, Google, ORCID, IEEE와 같은 서드파티(third-party) 로그인, 또는 소속된 그룹의 SSO(Single Sign-On) 중 원하는 방식을 선택할 수 있다. Overleaf 내부에서는 데이터 처리를 위해 가입 방식과 상관없는 고유한 ID를 매핑해 사용한다. 이와 유사하게 Overleaf에 사용자가 생성한 작업 공간(repository)들은 각각 고유한 프로젝트 ID로 구분되며, 사용자는 유효한 세션 아래서 프로젝트 ID를 통해 개별 문서 편집기에 접근할 수 있다. (Fig. 3)

Overleaf가 제공하는 협업 기능은 사용자가 참여하고 있는 프로젝트에 협업에 참여할 멤버를 해당 프로젝트에 초대(Fig. 4) 하는 방식으로 이뤄진다. 이때 Overleaf는 빠른 탐색을 위해 한 번이라도 협업을 함께했던 계정 및 프로젝트 정보를 서버에 캐싱한다. 협업했던 계정 및 프로젝트의 정보는 Table 1의 URL을 통해 JSON의 형태로 저장되며, 쿼리된 JSON 하위의 ID를 통해 사용자 및 프로젝트의 고유 ID를 획득할 수 있다. (Table 1)

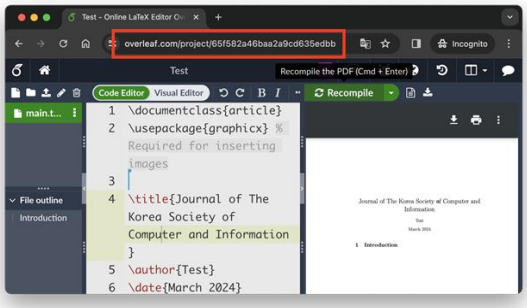


Fig. 3. Overleaf Project ID Example

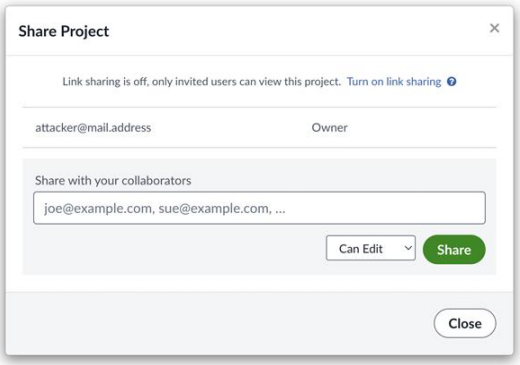


Fig. 4. Overleaf Collaboration Invitation

Table 1. Collaboration Information Query

URL	https://www.overleaf.com/user/contacts

Result JSON Format	{
	"contacts": [{"id": "ID1",
	"email": "ID1@email.com",
	"first_name": "id1_first_name",
	"last_name": "id1_last_name",
	"type": "user"},
	{"id": "ID2",
	"email": "ID2@email.com",
	"first_name": "id2_first_name",
	"last_name": "id2_last_name",
	"type": "user"},]
	}

URL	https://www.overleaf.com/user/projects

Result JSON Format	{
	"projects":[{"_id":"project_id1",
	"name":"project 1 name",
	"accessLevel":"owner"},
	
	{"_id":"project_id 2",
	"name":"project 2 name",
	"accessLevel":"readWrite"},
	
	{"_id":"project_id 3",
	"name":"project 2 name",
	"accessLevel":"readOnly"},]
	}

Overleaf에서 제공하는 다양한 기능 중 LaTeX 파일 처리에 해당하는 기본 기능은 크게 편집, 컴파일, 렌더링으로 분류된다. Fig. 2에서 보이는 것처럼 사용자가 편집기에서 LaTeX 문서를 편집한 뒤 컴파일을 요청하면 시스템 내부 컴파일러가 LaTeX 문서를 PDF 파일로 컴파일해 결과물을 자체 서버에 캐싱한다. 캐싱된 PDF 파일의 URL은 렌더러에 보내져 사용자가 컴파일된 최종 PDF 결과물을 웹 뷰어에서 확인할 수 있다. 이때, PDF 뷰어 영역은 iframe 태그를 통해 Overleaf 호스트에 임베딩된다. 즉, 편집에서 렌더링에 이르기까지의 과정이 하나의 장소에서 이뤄지지 않고 HTTPS에 기반한 통신을 통해 이루어진다.

2. Attack Vectors and Procedures

Overleaf의 렌더링 구조에서 알 수 있듯이 Overleaf에서는 웹 기반 뷰어를 제공하기 위해 사용자의 저작물에 고유 URL이 생성되어 매칭된다. 이는 해당 URL을 통해 캐싱된 PDF 파일이 공격자에게 노출될 수 있는 위험성을 잠재한다. 본 절에서는 URL Path Traversal 공격을 통해 인가받지 않은 사용자가 Overleaf로 컴파일된 아웃풋 파일에 접근 가능한 시나리오를 소개한다.

URL Path Traversal 공격[7]은 특정 파일을 가리키는 URL path의 값을 유추하여 서버에 저장된 파일에 접근하는 공격 기법으로, 이를 적용하기 위해서 Overleaf URL의 path 구성에 대한 분석이 필요하다. 먼저, Overleaf에서 생성하는 URL의 path 파라미터는 Table 2와 같이 구성된다. URL을 통한 비인가 리소스의 접근을 위해 네 개의 ID (user_id, project_id, build_id, server_id) 정보가 필요함을 파악할 수 있다.

Table 2. Cached PDF File Query

Request URL	https://compiles.overleafusercontent.com/zone/{***zone***}/project/{***project_id***}/user/{***user_id***}/build/{***build_id***}/output/output.pdf?compileGroup=standard&clsiserverid={***server_id***}&enable_pdf_caching=true	
Parameter Description	zone	5th element of the array that separates *server_id* by the delimiter '-'
	project_id	Project ID
	user_id	User ID
	build_id	Compile Build ID
	server_id	Compile Server ID

이중, user_id와 project_id는 사용자 계정 및 작업 디렉터리 삭제 시까지 변하지 않는 값으로, 앞서 설명한 JSON 쿼리를 통해 과거의 협업 사용자 ID 및 프로젝트의 ID를 손쉽게 조회할 수 있다. 뿐만 아니라 무작위 파일에 대한 협업을 요청하는 피싱 메일을 통한 스캠 공격을 통해서도 임의의 사용자의 고유 정보를 획득할 수 있다. 컴파일 정보인 build_id와 server_id 정보의 경우 HTTPS 헤더에 노출되고 있다. 이러한 정보 획득을 바탕으로 컴파일된 PDF 파일 접근을 위해 다음의 두 가지 공격 시나리오를 소개한다.

2.1 Compiled PDF Exfiltration Attack

컴파일된 PDF 탈취 공격 (Fig. 5)은 패킷 스니핑을 통해 build_id 및 server_id를 획득하고, 유효한 URL을 생성하여 리소스에 접근한다. Fig. 6는 사용자가 Overleaf 사이트의 컴파일 버튼을 눌렀을 때, 전송되는 패킷의 일부를 보여준다. HTTP 헤더에는 리소스 접근에 필요한 build_id 및 server_id 정보가 포함되는 것을 확인할 수 있다. 이 정보는 컴파일 버튼 클릭 뿐만 아니라 다운로드 클릭 등 다수의 이벤트 발생 시 노출되며, 두 서버 간의 빈번한 통신은 네트워크 공격을 더욱 용이하게 만든다.

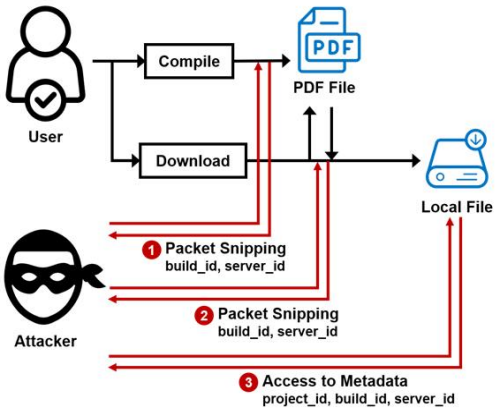


Fig. 5. Overall Procedure of Compiled PDF Exfiltration Attack

Fig. 6. Compile Request Header

build_id 및 server_id는 다운로드 된 PDF 파일의 메타 정보 확인을 통해서도 획득 가능하다. Fig. 7은 Overleaf로 컴파일된 PDF 파일의 메타 정보를 보여준다. URL의 파라미터 속성에서 project_id, build_id, server_id가 표시되고 있는 것을 확인할 수 있다. 이 방법은 특히 공용 컴퓨터 또는 클라우드 환경에서 파일의 접근 권한에 대한 취약점이 존재할 경우 유효성을 갖는다.

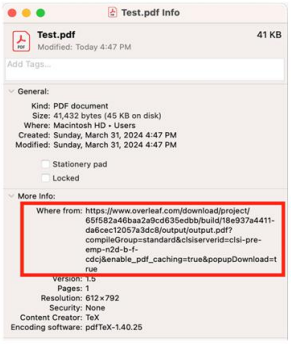


Fig. 7. Compile Data Exfiltration from Download URL

2.2 PDF Recompilation Attack

다음으로 사용자의 세션 하이재킹을 통한 PDF 재컴파일 공격 (Fig. 8)을 소개한다. Cross-site Request Forgery(CSRF)[8]는 인증된 사용자(예: 로그인 성공)가 웹 애플리케이션에 특정 요청을 보내도록 유도하는 공격으로 이를 적용하면 공격자는 Overleaf의 컴파일 요청을 위조할 수 있다.

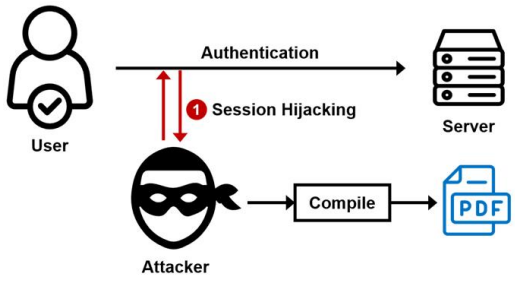


Fig. 8. Overall Procedure of Recompilation Attack

Fig. 9은 session_id를 통해 컴파일 request을 위조하는 과정을 보여준다. 이를 위해서 project_id와 session_id 값이 필요하며, session_id는 로그인된 사용자의 세션에 할당되는 값으로 서버 응답 시에 쿠키에 저장된다. 이를 이용해서 공격자는 로그인된 사용자의 세션을 탈취하여 컴파일을 시도할 수 있으며, 성공 시 build_id와 server_id가 새로 갱신되기 때문에 컴파일 된 리소스로부터 id 값을 획득하지 않아도 공격이 가능하다. 뿐만 아니라, 공격자가 은밀하고 지속적으로 컴파일을 시도할 수 있기 때문에 더욱 능동적으로 리소스를 탈취할 수 있다는 장점을 갖는다.

Fig. 9. Forged Compile Request

IV. Evaluation

본 절에서는 제시한 공격 방법이 Overleaf로 컴파일된 PDF 출력물에 접근 가능한지 유무를 파악하고 결과를 논

의한다. 실험을 위해 Overleaf에 가입된 세 개의 계정을 사용했으며 한 개의 계정은 공격자로, 다른 두 개의 계정은 공격 대상으로 설정했다. 두 공격 대상 계정은 각각 무료 플랜과 유료 구독 플랜 계정으로, 플랜의 종류와 관계없이 공격자 계정이 공격 대상 계정이 소유한 정보의 탈취 가능 여부를 실험했다. 제시한 공격은 네트워크 공격자를 가정하며, 로컬 네트워크 또는 공용 WiFi 환경 등 공격 대상자가 통신하는 패킷 모니터링이 가능한 환경에서 공격 실행이 가능하다. 제시한 두 공격 모두 URL을 기반으로 공격이 실행되기 때문에 추가적인 악성코드의 실행이 요구되지 않으며 실행 환경에도 거의 영향을 받지 않는다. 모든 실험은 macOS 13.0, Google Chrome 125.0에서 진행되었다.

첫 번째 실험으로 패킷 모니터링을 통해 획득한 컴파일 서버의 URL과 다운로드 된 파일의 메타 정보에서 획득한 Overleaf 메인 서버의 URL 각각에 대해 컴파일된 PDF 파일에 접근 가능 여부를 파악한다. 실험 결과, 컴파일 서버는 획득한 URL을 통해 비인가 리소스의 액세스가 가능했으며, 무료 및 유료 구독 플랜의 종류와 상관없이 타겟 사용자의 리소스를 획득할 수 있었다. 이와 반대로 Overleaf의 메인 서버에 캐싱되는 리소스의 경우, 메타 정보로 획득한 URL을 통해 접근이 불가능했다. (Fig. 10) 이를 통해 컴파일 서버에 더 약한 액세스 컨트롤이 적용되어 있음을 알 수 있다. 두 경우 모두 캐싱한 파일을 서버에 보관하는 시간은 최대 약 1시간 정도로, PDF 파일에 접근하는 공격이 충분히 보장되는 시간이었다.

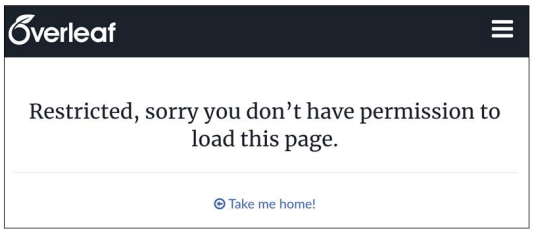


Fig. 10. Result From Accessing the URL Obtained via Metadata

세션 하이재킹을 통한 Overleaf 리소스 탈취 공격 역시 project_id 및 session_id 정보를 통해 성공적으로 유효한 request를 전송 가능함을 확인할 수 있었다. 한편, Overleaf의 편집 화면은 사용자가 컴파일 버튼 클릭 시 뷰어 화면이 1회 새로고침 되는 특징을 갖는다. 해당 공격으로 인한 사용자 뷰어의 영향을 관찰한 결과, 사용자 화면에 아무런 변화가 일어나지 않는 것을 확인할 수 있었다. 이는 사용자가 공격 시행 여부를 감지하기 어렵게 만들며, 제안된 공격이 더욱 견고하게 시행 가능함을 나타낸다.

V. Defense Strategy

지금까지 LaTeX의 온라인 협업 툴로 잘 알려진 Overleaf 플랫폼의 취약점을 소개했다. 본 논문에서 소개한 PDF 탈취 공격과 PDF 재컴파일 공격은 전통적인 웹 공격을 기반으로 하기 때문에, 리소스 캐싱과 URL 기반 접근 등을 제공하는 온라인 협업 플랫폼에서도 이러한 취약점이 존재할 수 있다. 특히, 웹 개발 시 오픈소스가 빈번히 활용되고 개발 트렌드가 공유되기 때문에 유사한 서비스에서도 동일한 수준의 피해가 발생할 것으로 예상된다. 본 절에서는 LaTeX의 온라인 협업 툴을 안전하게 구현하기 위한 방안 제시한다.

1. Strengthening Access Control

본 논문에서 제시한 공격은 Overleaf가 관리하는 컴파일 서버의 약한 액세스 컨트롤로 인해 발생한다. 실험 결과로 알 수 있듯이 컴파일 서버는 메인 서버와 다르게 Path Traversal을 통해 리소스 접근이 가능하다. 따라서, 패킷 스니핑을 통해 획득할 수 있는 ID 값 이외의 인증 필드를 추가하여 보안성을 높일 수 있다. 예를 들어, 서버 단에서 Referer[9] 확인 및 시크릿 토큰을 활용[10-11]하여 request 출처를 체크하거나, 또는 Content Security Policy[12], Cross-Origin Resource Sharing[13] 등 최신 브라우저가 제공하는 보안 기술을 적용하여 허용되지 않은 JavaScript 코드의 URL 접근을 방어할 수 있다.

2. Implementing Encryption for Cache Files

다음으로 캐싱 파일에 암호화를 적용하여 비인가 사용자로부터 캐싱 스토리지의 리소스를 보호하는 방법에 대해 논의한다. SecureC2Edit[14]은 온라인 LaTeX 편집기 자체를 신뢰할 수 없는 환경을 가정하여 해당 클라우드 서버에 리소스 저장 시 암호화를 수행하는 프레임워크를 제안했다. 이를 확장하여 URL 기반의 파일 액세스를 차단하고 HTTPS의 Handshake 과정에서 공유된 암호화 키를 이용해서 파일 액세스가 가능하도록 수정한다면 캐싱된 PDF를 보호할 수 있다.

3. Preventing File Creation via WebGL

컴파일 서버에 보안 메커니즘 적용이 어려운 경우, WebGL을 기반으로 한 PDF 렌더링이 해법이 될 수 있다. 이 방법은 파일 생성 자체를 불필요하게 하며, 리소스를 컴파일 서버에 따로 캐싱하지 않아도 되기 때문에 안정성을 높일 수 있다. Sejda[15], Xodo[16], Jumpshare[17],

DocFly[18] 등 다수의 온라인 PDF 파일 뷰어는 WebGL 기반 렌더링을 채택하고 있으며, 캔버스 요소에 각 콘텐츠를 텍스트, 도형 등의 요소로 표현할 수 있기 때문에 콘텐츠 탐색 및 하이라이트 등의 처리에도 편리성을 제공할 수 있다.

VI. Conclusion

클라우드 기반 온라인 협업 도구의 이용이 증가함에 따라 사용자들은 보다 많은 개인 정보를 플랫폼에 노출시키게 되며, 이로 인해 온라인 협업 도구의 보안 문제와 개인 정보 보호에 대한 우려가 높아지고 있다. 본 연구에서는 Overleaf의 보안 취약성과 개인 리소스 노출 위험을 분석하고, 사용자 ID, 프로젝트 ID, 그리고 캐싱된 PDF와 같은 민감한 정보를 보호하기 위한 대응책을 제시했다. 이러한 연구 결과는 온라인 협업 환경에서의 보안 문제에 대한 이해를 높이고, 보다 견고한 보안 프로토콜을 제시하여 신뢰도 높은 온라인 협업 도구 개발에 도움이 될 것으로 기대된다.

REFERENCES

[1] Overleaf, https://www.overleaf.com

[2] S. Checkoway, H. Shacham, E. Rescorla, "Are Text-Only Data Formats Safe? Or, Use This LATEX Class File to Pwn Your Computer", USENIX conference on Large-scale exploits and emergent threats, 2010. DOI: 10.1145/2046707.2046735

[3] S. Checkoway, H. Shacham, E. Rescorla, "Don't take latex files from strangers", login USENIX Mag, Vol. 35, 2010.

[4] G. Lacombe, K. Masalygina, A. Tahiri, C. Adam, and C. Lauradoux, "Can You Accept LaTeX Files from Strangers? Ten Years Later", arXiv, 2021. DOI: 10.48550/arXiv.2102.00856

[5] K.A. McMillan, "A Platform Independent Computer Virus", pp. 38, 1994.

[6] U. Wermuth, "Transparent file I/O using the original TEX program and the plain TEX format", TUGboat, Vol. 43, 2022.

[7] Path Traversal, OWASP, https://owasp.org/www-community/attacks/Path_Traversal

[8] Cross Site Request Forgery (CSRF), OWASP, https://owasp.org/www-community/attacks/csrf

[9] Referer, MDN Web Docs, https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referer

[10] Z. Mao, N. Li, and I. Molloy, "Defeating Cross-Site Request Forgery Attacks with Browser-Enforced Authenticity Protection", Financial Cryptography and Data Security, pp. 238-255, 2009. DOI: 10.1007/978-3-642-03549-4_15

[11] M. Zhou, P. Bisht, and V. N. Venkatakrishnan, "Strengthening XSRF Defenses for Legacy Web Applications Using Whitebox Analysis and Transformation", Information Systems Security, pp. 96-110, 2010. DOI: 10.1007/978-3-642-17714-9_8

[12] Content Security Policy (CSP), MDN Web Docs, https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

[13] Cross-Origin Resource Sharing (CORS), MDN Web Docs, https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

[14] A. Shashank, and K. A. Pradeep, "SecureC2Edit: A Framework for Secure Collaborative and Concurrent Document Editing", IEEE Transactions on Dependable and Secure Computing, 2023. DOI: 10.1109/TDSC.2023.3302810

[15] Sejda, https://www.sejda.com/

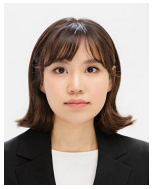
[16] Xodo, https://xodo.com/pdf-viewer

[17] Jumpshare, https://jumpshare.com/

[18] DocFly, https://www.docfly.com/

Authors

Suzi Kim received the B.S., M.S., and Ph.D. degrees in the School of Computing at KAIST in 2012, 2015 and 2022, respectively. Dr. Kim worked at Wētā Digital from 2022 to 2023 and is now the CEO at Studio AABB, collaborating with Wētā FX to drive forward advancements in computing technology.



Jiyeon Lee received the B.S. degree in Computer Science from Dankook University, and the M.S. and Ph.D. degrees in School of Computing from KAIST, in 2014 and 2021, respectively. She is currently a professor in the Department of Computer Engineering at Kyungpook National University, with research interests focused on information and web security.