# Web Based Security Assessment of Open-Source CAD tools using ZED Attack Proxy (ZAP) scanner

**Abdul Samad Shakir[1], and Bharadwaj R. K. Mantha[1*]**

[1]Department of Civil and Environmental Engineering, College of Engineering, University of Sharjah, Sharjah, United Arab Emirates.
shakirabdul31@gmail.com, rmantha@sharjah.ac.ae

**Abstract –**

**The post-pandemic rise in Computer-Aided Design (CAD) tool adoption has driven a shift toward accessible, cost-effective open-source and "Do-it-yourself" (DIY) CAD tools, increasingly favored by individuals, small and medium sized enterprises (SMEs), and entrepreneurs over proprietary software. Despite their growing popularity, none of the existing studies investigate the security of these open-source CAD tools, particularly when users download and install them on personal devices. This study addresses this key research gap by identifying vulnerabilities of open-source CAD tools used specifically in the Additive Manufacturing (AM) process using the Zed Attack Proxy (ZAP) scanner. A web-based security assessment of five widely used CAD tools was conducted, where 25 unique web-based vulnerabilities were identified. This study provides valuable insights for SMEs operating in the architecture, engineering, construction, and operations (AECO) industry by highlighting the potential risks associated with using open-source CAD tools. These risks include the corruption of critical information on user machines, manipulation and loss of CAD data, and unavailability of CAD tools to operate. Such issues directly impact the core principles of cyber security i.e. confidentiality (C), integrity (I), and availability (A) highlighting the critical need to address these vulnerabilities and ensure secure use of open-source CAD tools.**

**Keywords –**

**Common weakness enumeration (CWE); Cyber-attacks; AECO; CAD model; Application security testing; Web based attacks.**

## 1 Introduction

Computer aided design (CAD) tools are essential in the additive manufacturing (AM)/Three-Dimensional (3D) printing process, serving as the primary tools for designing and verifying models across various industries. They are used to create, modify, and analyse designs, ensuring that the final product meets specific requirements. Recently, there has been an increasing use of CAD tools across various sectors or industries—whether in large organizations within industries like construction, heavy manufacturing, or aerospace, or by smaller businesses and startups looking to design on a smaller scale. CAD tools are also widely used by civil engineering students, general and sub-contractors, architects, small business owners, and freelancers working on projects or prototypes [1].

However, with the growing reliance on CAD tools, a critical question arises about their cybersecurity. Since CAD tools are among the first steps in the additive manufacturing (AM) process, it is important to assess how secure these tools are. According to [2] and [3],, the AM process can be divided into five key phases: CAD, Finite Element Analysis (FEA), slicing, printing, and assembly. While much research has been conducted on the security concerns in later stages of the AM process (such as slicing and printing), there is a significant gap in the literature when it comes to cybersecurity vulnerabilities in the CAD phase, particularly during the tool download and installation process. This study focuses on the initial phase evaluating the security of CAD tools. Each of these phases' present unique cybersecurity risks and understanding the vulnerabilities at the CAD phase is crucial for ensuring the overall security of the AM process. The risks associated with downloading and using CAD tools have not been fully explored, even though this phase is often the first line of attack. The problem lies in the insufficient research and lack of attention to cybersecurity risks, particularly during the CAD tool download and installation phase.

There are several common types of potential cybersecurity attacks that users may encounter during the usage of CAD tools. As per [4], these attack], these attacks can exploit vulnerabilities in both the software and the data it generates or handles. Some of the frequently encountered or potential threats when using CAD tools include: 1) Malware Insertion during CAD tool download; 2) Denial of Service (DoS) attacks when users brose related web pages, making them unavailable

to legitimate users; 3) Social Engineering attacks, where attackers trick users to reveal their credentials and designs; 4) Ransomware attacks targeting CAD tools, where users are locked out or their files are encrypted; 5) Vulnerabilities in the CAD tool itself, which may be exploited by attackers[5].

As mentioned in [6], open-source CAD tools open source CAD tools are often more vulnerable to cyber-attacks for several reasons which includes: 1) Visibility of source code in the public domain, allowing hackers to inspect, identify, and exploit vulnerabilities; 2) Inconsistent updates and maintenance, despite community support and upkeep; 3) Limited human resources for security management; 4) Fewer security standards compared to paid CAD tools. Despite this, the security concerns related to the initial CAD tool download phase remain under-researched, particularly in the context of open-source CAD tools. The process of a cyberattack typically follows a series of stages. It begins with reconnaissance, where attackers gather information about the target. This is followed by exploitation, where vulnerabilities are leveraged, such as phishing or malware deployment. After gaining access, the attackers may escalate privileges, steal data, or disrupt operations, while attempting to hide their presence [2].

Additionally, a recent study addressed in [3] and [7] indicates a significant increase of 63% in the number of downloads of open-source CAD tools in 2022, further highlighting the growing trend. However, this study focuses specifically on the initial phase of the CAD tool usage process—when users download the tool, as all CAD tools require a download. This paper aims to fill this gap by examining the vulnerabilities during the download and installation phase of CAD tools, which is crucial to securing the overall AM process. The following section describes the journey of a typical user who browses through the internet and visits the website of the CAD tool. It also depicts the journey of an attacker who at the same time could perform malicious activities on the internet.

## 2 Literature Review

In our study, we conducted a structured literature review by searching two academic databases, i.e., IEEE Xplore and Google Scholar. We used specific keywords such as CAD tools, security assessment, open-source CAD vulnerabilities, security in AM, etc., to identify relevant studies. Additionally, we applied inclusion and exclusion criteria to filter relevant papers based on their relevance to web-based security assessments of CAD tools. Upon reviewing selected papers and analyzing their abstracts, we drew inspiration from those that discuss and categorize various taxonomies of cyberattacks in the AM industry, ensuring a comprehensive understanding of the security landscape. In addition, as described above, in the interest of adhering to the page limit, not all references were included in the study**.** To understand the specific gaps in the most recent literature, it is imperative to comprehend the generic process of any user's navigation journey through typical CAD tools. In an objective to make this clear, section 2.1 outlines the overview of the user journey, section 2.2 delves into the specifics of user interaction with the CAD website, and section 2.3 details the attacker's interaction with the CAD website. Finally, section 2.4 provides a summary of the most relevant studies in the domain and identifies the specific research gaps that will be addressed in this study

### 2.1 Overview of the User Journey

Based on the understanding and inferences from the literature review [8], the following elements have been identified as relevant and suitable for our study (as shown in Figure 1). Where 1) user (blue icon) represents a legitimate user engaging with the website of a CAD application, and 2) attacker (black-hat icon) represents a malicious actor attempting to exploit web-based vulnerabilities for them.

### 2.2 User Interaction with the CAD website

First, the website exploration phase, where the user visits the CAD website, typically to explore, read, and download the CAD tool. At this phase, the website can be potentially exposed to countless potential web based cyber threats. Second, CAD tool download phase, where the user downloads the CAD application from the website. This step carries risks such as the possibility of downloading compromised or malicious software. Finally, desktop application phase, where the downloaded tool transitions into a desktop-based application. However, this phase is beyond the scope of our study and is not addressed in our analysis.

### 2.3 Attackers Interaction with the CAD website

The attacker can target two out of 3 phases in the user journey, i.e., the website exploration phase and the CAD tool download phase. These phases present opportunities for exploiting vulnerabilities. Figure 1 highlights several common attack vectors as mentioned by [9]: 1) DDoS (Distributed Denial-of-Service): overloads the website, making it inaccessible to legitimate users, 2) injection attacks: injects malicious code into the website or the download processes 3) phishing: tricks users into providing sensitive information or downloading harmful files, and 4) ransomware: encrypts or locks the user's data or application, demanding payment to restore access.

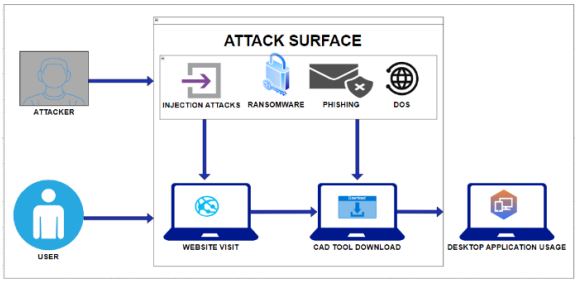


Figure 1: User-Attacker Interaction in CAD environment

## 2.4 Relevant Studies

Several existing studies emphasize the significance of cyber-attacks during different phases of the AM process. For example, in [10], [11], and [12], the authors have highlighted and discussed the taxonomy of potential cyber-attacks in the AM industry. In summary, the following five cyber-critical phases were identified: CAD, Finite Element Analysis (FEA), slicing, printing, testing, and assembly. These phases represent the primary attack vectors highlighted in the literature, providing a general overview of the current state of cybersecurity concerns in the AM industry. All three papers have identified and analyzed the attack phases; however, none of these studies specifically focus on the first attack phase, which is the CAD phase. To elaborate, study [4] examines cyber-attacks in the FEA phase, slicing phase, and printing phase, while Papers [5] and [6] primarily focus on cyber-attacks in the slicing and printing phases.

In addition to the mentioned sources, [13] and [14] emphasize the cybersecurity challenges within the AM process, particularly as it pertains to the design and simulation stages. These stages, which rely heavily on CAD tools, are crucial for ensuring the accuracy and reliability of the final product. Since CAD tools are the first point of contact in the AM process, securing them is essential for protecting the integrity of the entire workflow.

Moreover, as the AM industry continues to embrace digitalization and the integration of advanced technologies such as Artificial Intelligence (AI) and the Internet of Things (IoT), the attack surface for cyber threats expands, especially during the early stages of the design process. The evolution of these technologies brings new challenges in securing CAD tools, which, if compromised, can lead to broader risks across the entire manufacturing chain.

This paper, while drawing on these studies, emphasizes the importance of understanding and mitigating the cybersecurity risks during the CAD phase. Given the increasing reliance on digital tools in the AM industry, ensuring the security of CAD tools is paramount to maintaining the confidentiality, integrity, and availability of the data and models that are central to the manufacturing process.

# 3 Methodology

Figure 2 shows all the steps in the proposed methodology. This methodology outlines a nine-step process along with two decision boxes for conducting a web-based security assessment of CAD tool.

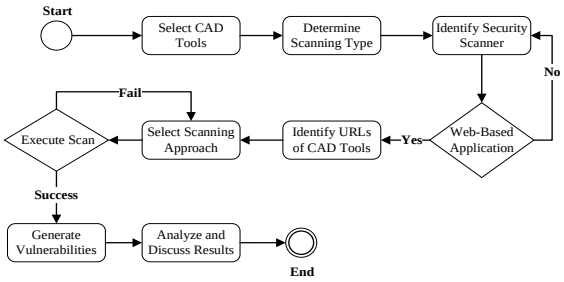


Figure 2: Overview of the proposed methodology

## 3.1 Select CAD Tools

The selection of appropriate CAD tools was the first step in the methodology. According to [15] and [16], CAD systems are classified based on two main criteria: 1) Licensing and 2) Type of computer application. Licensing of the CAD systems refer to their categorization based on commercial and non-commercial use. This study focuses on non-commercial use, which includes freeware and open-source tools. In contrast, commercial CAD systems fall under the closed-source category. These closed-source systems are high-performance platforms equipped with advanced features. According to the same study, the largest providers of commercial CAD systems are Autodesk, CATIA, and solid edge. The second classification is based on the type of application, which can be further divided into desktop-based, web-based, and mobile-based CAD tools. For our study, we have chosen freeware tools, and the specific type of CAD application selected is web-based**.** Therefore, selection of the right CAD tool is crucial, and it is essential to clearly identify the specific type of CAD system that this study intends to evaluate and hence of relevance in the further steps of the methodology.

## 3.2 Determine Scanning Type

It is crucial to understand the three main types of testing that should be conducted when an application goes live. Whether the application is intended for commercial or public use, ensuring its protection against cyberattacks is a top priority. According to [11] and [18], there are three most common types of testing namely static code analysis testing (SAST), dynamic application security testing (DAST), and interactive application security testing (IAST). For our study, we have chosen to

use Dynamic Application Security Testing (DAST). This approach is preferred as it allows for testing the application while it is running, simulating real-world cyberattacks and identifying vulnerabilities that may not be apparent during the development phase. A basic definition of each is provided in the Table 1 for easier comparison.

Table 1: Three most common scanning types

| SAST | DAST | IAST |
|---|---|---|
| White box-based testing/ Source code-based testing | Black box testing. Conducted while the application is running | Code is analysed for security vulnerabilities while an application is running |

### 3.3 Identify Security Scanner

Security scanners exist in various forms, each designed to identify several types of vulnerabilities, threats, and weaknesses within an environment. There are several other types of security scanners, including: 1) Network Security Scanners, in which they scan for vulnerabilities related to open ports and network configurations.2) Vulnerability Scanners, in which vulnerabilities are identified in systems or networks by comparing them against databases of common weaknesses. 3) Web Application Security Scanners: These specifically detect vulnerabilities within web applications, such as SQL injection or cross-site scripting (XSS). 4) Static Application Security Testing (SAST): These scanners analyze source code or binaries to identify security flaws before the application is run. 5) Dynamic Application Security Testing (DAST) Scanners: These scanners find vulnerabilities by testing applications during runtime, simulating real-world attacks.6) Cloud Security Scanners: These scanners identify vulnerabilities in cloud environments, such as misconfigurations or security gaps in cloud-based services.

Given that the scope of this study is focused on scanning vulnerabilities in web applications, it is important to note that, as stated in [8], web application security scanners simulate the actions of a penetration tester to thoroughly inspect and assess the target application. Among the web application security scanners used in this study, we have selected OWASP ZAP (Zed Attack Proxy), a widely recognized and popular scanner within the security community. There are two primary reasons for choosing this tool which includes: 1) Community Support and Relevance: ZAP is developed, managed, and maintained by the Open Web Application Security Project (OWASP), a prominent organization that also publishes the OWASP Top 10 list of the most critical security risks faced by web applications. According to [9], it is important to note that the OWASP Top 10 serves as a baseline for information security professionals when assessing the security posture of any web application running on the internet. 2) Customizability and Features: The ZAP scanner offers a wide range of customizable features essential for conducting a comprehensive security assessment. These include various scanning techniques, flexible reporting options, and automation capabilities, making it an ideal choice for thorough security evaluations.

### 3.4 Web based application

As the focus of this study is on scanning web applications, it is imperative to utilize tools and methodologies specifically designed for this purpose. The current step acts as a decision box and supports as discussed in the following sections. However, if the application under examination is not web-based, we would need to reassess our approach and move to Step 3.3, where we would identify an appropriate security scanner for non-web applications

### 3.5 Identify URLs of CAD Tools

Once the target application is identified, a crucial step is to determine the URLs associated with the application. This is a key step highlighted in [9], where identifying the URLs (i.e., web pages) to be scanned is considered one of the foundational tasks in conducting a security assessment. Furthermore, the paper describes the general architecture of web application security scanners, noting that the URLs serve as the primary input for the testing process. The list of the main web pages of the five open-source CAD tools that were part of our study is as follows: FreeCAD (https://www.freecad.org/),OpenScad (https://openscad.org/),Blender(https://www.blender.org/), LibreCAD (https://solvespace.com/), and Solverspace (https://librecad.org/).

### 3.6 Select Scanning Approach

There are two primary scanning modes: Active scanning and Passive scanning, both of which are available in the web application security scanner, OWASP ZAP. According to [20], active scanning is considered the more aggressive form of scanning, as it involves actively attacking the application to identify vulnerabilities. During this process, there is a high likelihood that the scan will be visible to the target application team, who may monitor the traffic and observe the different attacks being performed. Typically, active scanning is conducted over private web connections and can be carried out internally. In contrast, passive scanning does not modify the traffic between the scanner and the target application, nor does it execute any

malicious attacks. This mode simply observes the application's behavior without interacting with or altering the system. It is crucial to note that explicit authorization from the application owner is required to conduct active scanning, as it can potentially harm the production environment. For this study's purposes, the ZAP tool has been configured to use passive scanning, and throughout this study, only passive scanning has been performed.

## 3.7 Execute the Scan

Sections 3.7.1 and 3.7.2 outline the prerequisites for using the scanner, along with the appropriate environmental conditions required for its successful operation.

### 3.7.1 Pre-requisites for ZAP proxy 2.15.9

As mentioned in [9], there are 2 prerequisites required to install the tool: 1) Supported operating systems, i.e., Windows, Linux, and macOS. 2) Ensure that the Java Runtime Environment (JRE) is fully patched and maintained.

### 3.7.2 Environment setup

There are three main steps that include: 1) Cloning a Virtual Machine: The primary host machine to conduct this study was completed on Windows VM. 2) Download ZAP Proxy**:** Visit the OWASP ZAP website to download the latest version compatible with your operating system. 3) Install ZAP Proxy**:** Run the downloaded installer on the Windows VM and follow the installation prompts.

## 3.8 Generate Vulnerabilities

The active scanning technique performed on all five URLs was successful in identifying vulnerabilities across all five CAD tools examined. The results are displayed in Table 2. High (H), which represents critical vulnerabilities that require immediate attention; Medium (M), indicating moderate risks that should be addressed promptly; Low (L), referring to less severe issues that may not immediately impact security; and Informational (I), which are minor details that do not pose a direct threat. These levels are inherited from the scanner. Additionally, the ZAP Proxy scanner provides a breakdown of vulnerabilities based on their severity, using data from its own database. The vulnerabilities from the ZAP proxy are categorized into four levels of severity in decreasing order (as shown Table 2): High (H), Medium (M), Low (L), and Informational (I) which are inherited from the scanner.

It is important to note that OWASP ZAP utilizes its own vulnerability assessment database, meaning that the scanning algorithm and rules are defined within the ZAP scanner itself. The scanner identifies vulnerabilities based on predefined patterns. Similarly, the scanner references classifications from the Open Web Application Security Project (OWASP) and the Common Weakness Enumeration (CWE), which is maintained by MITRE.org. These are community-driven resources that catalog common software and hardware weaknesses. According to CWE.mitre.org, which is a not-for-profit organization that addresses challenges in cybersecurity, develops comprehensive cybersecurity frameworks, and provides resources to help organizations identify, defend, and respond to cyberattacks. A weakness is defined as a condition in software, firmware, hardware, or a service that, under certain conditions, could lead to vulnerabilities. These vulnerabilities were detected by the ZAP scanner, each of which is associated with a CWE ID, as explained in Section 4 and 4.1

Table 2: Breakdown of Vulnerabilities by Severity

| Tool Name | H | M | L | I | Sub Total |
|---|---|---|---|---|---|
| Free Cad | 1 | 5 | 2 | 5 | 13 |
| OpenScad | 0 | 6 | 4 | 5 | 15 |
| Blender | 0 | 2 | 3 | 5 | 10 |
| LibreCad | 0 | 3 | 1 | 3 | 10 |
| SolverSpace | 0 | 4 | 5 | 0 | 9 |

### 3.8.1 Risk Vs Confidence matrix

According to [9] and [10], the ZAP Proxy scanner further categorizes detected vulnerabilities using two parameters: Risk and Confidence. Risk can be defined as the potential consequences of data damage caused by a cyberattack. As noted by [21], risk is calculated in Equation 1.

$$\text{Risk} = \text{Threat} + \text{Vulnerability} \quad (1)$$

The level of Confidence is determined by the maliciousness of the Indicator of Compromise (IOC), which may include elements such as Internet Protocol (IP) addresses, email addresses, domains, URLs. The ZAP Proxy scanner provides both risk and confidence levels when listing vulnerabilities [11]. Based on this data, a heat map (as shown in Figure 4) is generated for the five CAD tools analyzed in this study. In the heat map, Confidence is represented along the X-axis, and Risk is represented along the Y-axis. The risk levels are categorized as Informational, Low, Medium, and High, in ascending order of severity. The X-axis is divided into three categories corresponding to low, medium, and high confidence levels, arranged in increasing order of confidence.

## 3.9 Analyze and Discuss Results

The scan was conducted independently for each of the five CAD tools. From these independent scans, a total

of 54 vulnerabilities were identified, with only 25 vulnerabilities being unique. The list of unique vulnerabilities are as follows: V1) Cross-site scripting, V2) Header not set, V3) Hidden file found, V4) Missing anti-clickjacking header, V5).htmaccess information leak, V6) Absence of anti-csrf tokens, V7) Strict-transport security header not set, V8) X-content options header missing, V9) Modern web application, V10) Re-examine cache control directives, V11) Security policy header not set, V12) User agent fuzzer, V13) User controllable HTML element (XSS), V14) Wildcard Directive, V15) style-src unsafe-inline, V16) Cross-domain misconfiguration, V17) Vulnerable JS library, V18) Cross-domain JavaScript source file inclusion, V19) Timestamp disclosure – Unix, 20) CSP: Header & Meta, V21) Information disclosure – suspicious comments, V22) Retrieved from Cache, 23) Charset Mismatch, V24) Application error disclosure, V25) Information disclosure – debug error messages. Figure 3 below displays a 5-set Venn diagram illustrating the unique vulnerabilities identified across the five CAD tools. Notably, only one vulnerability (V2) was found in all five CAD tools, while two vulnerabilities (V4 and V7) were found in four tools, and four vulnerabilities (V9, V10, V21, and V19) were found in three tools. This highlights the significant overlap among these CAD tools. In contrast, the study also identified distinct vulnerabilities unique to certain CAD tools: 8 distinct vulnerabilities for Free Cad, 4 distinct vulnerabilities for Solverspace, 3 distinct vulnerabilities for OpenScad, 1 distinct vulnerability for Blender, and none for Libre CAD. According to Figure 3, the most frequent vulnerabilities that were repeated are shown in Table 3.

The analysis of the results was conducted for the following scenarios: 1) The detected vulnerability with the highest severity, and 2) The significance of the most frequent vulnerabilities.

### 3.9.1 High Risk Vulnerabilities

The Free CAD tool was identified with a high-severity vulnerability classified as CWE-79, which pertains to Improper Neutralization of Input During Web Page Generation (Cross-Site Scripting). This particular vulnerability was detected exclusively within this tool. It typically arises when untrusted data is allowed to enter a web application, often via a web request. According to [12] and [13], during the process of page generation, the victim's web browser executes the injected malicious code within the context of the web server's domain, thus violating the web browser's same-origin policy. This policy is designed to prevent scripts from one domain from accessing or modifying data in another domain. According to [22], exploitation of this vulnerability could enable an attacker to perform a range of malicious activities, including, but not limited to, the theft of sensitive application data such as session cookies and other confidential information.2) Identification of Input Vectors: Understanding all potential areas where untrusted input may enter the application, such as parameters, arguments, cookies, network data, environment variables, reverse DNS lookups, query results, request headers, URL components, email data, files, filenames, databases, and external systems providing data to the application. This is a critical aspect of overall architectural design.3) Server-Side Validation: Ensuring that all input validation checks are also duplicated on the server side, providing an additional layer of protection.4) Use of Security Appliances: Implementing a Layer 7 firewall and a web application firewall (WAF) capable of detecting and mitigating attacks targeting this vulnerability [15] [16].

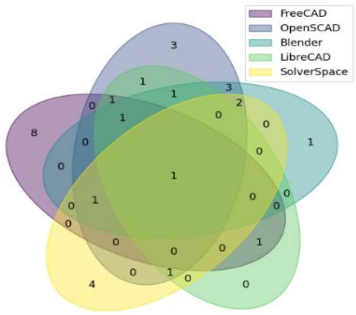


Figure 3: Vulnerability across CAD tools

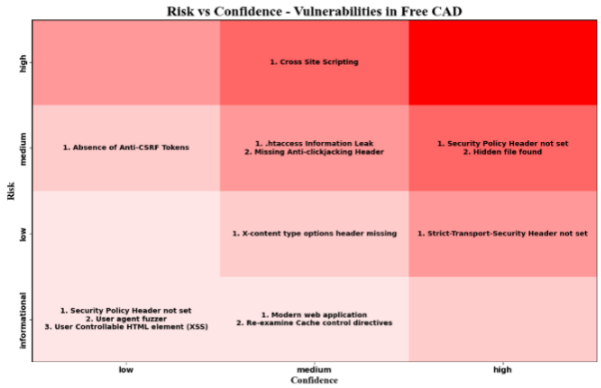


Figure 4: HeatMap for FREECAD tool

Table 3: Most Frequent Vulnerabilities

| Name of Vulnerability | Frequency |
|---|---|
| Header not set | 5 |
| Missing anti-clickjacking header | 4 |
| Strict transport security header not set | 4 |
| Re-examine cache control directives | 3 |
| Information disclosure – suspicious comments | 3 |
| Modern web application | 3 |
| Timestamp disclosure – Unix | 3 |

### 3.9.2 Most Frequent Vulnerabilities

According to Table 3, the "Header not set" vulnerability (CWE 693) is present in all 5 CAD tools. Additionally, the "Missing anti-clickjacking header" (CWE 1021) is present in 4 CAD tools, and the "Strict Transport Security header not set" (CWE 319) is also present in 4 out of 5 CAD tools. This vulnerability represents a subset of the broader Content Security Policy (CSP) framework, which functions as an additional layer of security designed to detect and mitigate critical threats, particularly Cross-Site Scripting (XSS) and data injection attacks. By injecting CSP headers from the server, the browser becomes equipped to protect the user from malicious dynamic content that may be loaded onto the page they are currently visiting [9]. It is important to recognize that this vulnerability is also a component of XSS, and we can conclude that all five of these CAD tools are highly susceptible to Cross-Site Scripting attacks. The vulnerability related to the absence of an anti-clickjacking header has been detected across all CAD tools, except Blender [17]. Clickjacking is a type of cyberattack wherein users are deceptively manipulated into clicking on elements they did not intend to interact with. This is typically achieved by superimposing a transparent layer over legitimate content, causing the user's click to activate hidden or malicious elements controlled by the attacker [22]. One effective countermeasure against such attacks is the implementation of an anti-clickjacking header in the HTTP response. The HTTP Strict Transport Security (HSTS) vulnerability is another significant issue identified in all CAD tools, except Libre CAD. In these tools, sensitive or security-critical data is transmitted in cleartext over communication channels, making it susceptible to interception by unauthorized actors. According to [18], this compromise violates the *Confidentiality* component of the CIA triad, as anyone gaining access to the network channel can read the transmitted information. To mitigate this type of attack, it is essential to encrypt the data using robust protocols prior to transmission, ensuring SSL/TLS encryption is implemented throughout the entire session, from login to logout.

## 4 Conclusions and Future work

This study contributes to the web-based security evaluation of five widely used CAD tools (i.e., Free Cad, OpenScad, Blender, Libre Cad, and Solverspace) and identifies 25 unique vulnerabilities (numbered from V1 to V25) with the help of ZAP proxy scanner. The proposed methodology however is generic and applicable to examine other web-based open-source CAD tools as well. The vulnerabilities observed through the analysis in this study range in severity, from highly risky to informational, and likelihood of occurrence, from the most probable to the least probable. The most critical vulnerability, V1, was detected in the Free Cad tool and is considered a high-risk issue. That is CWE 79, which pertains to Improper Neutralization of Input During Web Page Generation, Cross-Site Scripting, common in web-based attacks.

It is crucial to address these vulnerabilities, as they compromise the tools' adherence to fundamental security principles and expose end-users to a wide range of threats, from data theft to sophisticated attacks such as clickjacking and session hijacking. These findings are particularly relevant for AECO industry professionals, who increasingly rely on CAD tools for design and collaboration, as addressing these vulnerabilities ensures the security of sensitive project data and reduces risks to increasingly prevalent digital project workflows.
Additionally, future work also seeks to draw broader, more applicable inferences about the vulnerabilities and corresponding countermeasures on specifically web-based open-source CAD tools in general, which is the focus of this study. As part of this effort, AI-driven security tools will be explored to enhance the detection, classification, and mitigation of vulnerabilities in these CAD applications.

One of the key limitations of this study is that the results were generated using the ZAP proxy scanner. Potential variations can be observed if other scanners are utilized on these CAD tools, the proposed methodology and the results observed still largely will remain applicable. That is, the authors cannot guarantee that similar results would be obtained under different conditions, such as: 1) using a different web scanner, 2) applying a different scanning method, or 3) conducting the study at a different time. Vulnerabilities may be addressed, or new ones may emerge, potentially altering the results even if the same scanner is used due to the dynamic nature of website traffic [23]. As part of an ongoing effort, the authors are working to address the identified limitations of this study. Future work aims to generate more comprehensive results that provide deeper insights into the selected CAD tools. Additionally, future work also seeks to draw broader, more applicable inferences about the vulnerabilities and corresponding counter measures on specifically web based open-source CAD tools in general which is the focus of this study.

## 5 References

[1] Sun, Y., Liu, K., Li, Y., & Zhang, D. (2022). Towards an Open-Source Industry CAD: A Review of System Development Methods. *Tehnički vjesnik*, *29*(6), 2127-2136.
[2] Belikovetsky, S., Yampolskiy, M., Toh,
[3] J., Gatlin, J., & Elovici, Y. (2017). dr0wned–