Towards Continuous Risk Assessment and Conformance Checking of IdM Deployments

Andrea Bisegna*, Roberto Carbone*, Laura Cristiano*, Pietro De Matteis†, Silvio Ranise*‡

*Center for Cybersecurity, Fondazione Bruno Kessler, Trento, Italy
†Co-Innovation Lab, Dedagroup Spa, Trento, Italy
‡Department of Mathematics, University of Trento, Trento, Italy
Emails: {a.bisegna, carbone, l.cristiano, pdematteis, ranise}@fbk.eu

***Abstract*—Ensuring effective threat intelligence sharing, assessing potential risks, and responding to threats remain significant challenges, particularly in complex systems and critical infrastructures. Environmental, Social, and Governance platforms are emerging as comprehensive solutions that integrate cybersecurity with governance principles, enhancing transparency and proactive risk management. However, integrating security tools into platforms that enable conformance checking and continuous risk assessment poses challenges, including automating security workflows and prioritizing vulnerabilities based on severity and exploitability. This paper presents an extended version of Micro-Id-Gym (MIG), an open-source security testing tool for Identity Management (IdM) implementations. The goal of this enhancement is to make MIG easily integrable into platforms for continuous risk assessment and mitigation in complex software supply chains deploying IdM solutions critical to the Zero Trust paradigm. By supporting trustworthy deployments, MIG focuses on conformance testing as a key mechanism to ensure reliability and compliance in multi-entity deployments. The extended version of MIG is designed for seamless integration into Continuous Integration and Continuous Delivery pipelines and has been validated in Open Authorization 2.0 and OpenID Connect deployments.**

***Index Terms*—OIDC, OAuth, Security Testing, Conformance Checking, DevSecOps, Continuous Risk Assessment**

1. Introduction

Effectively sharing threat intelligence, evaluate risk exposure, and react to threats in a timely manner remain significant challenges, particulary in critical infrastructures and complex ecosystems where security incidents can have far-reaching effects. Organizations need to implement stronger systems to identify, evaluate, and reduce risks in real time as cyber threats continue to change. Environmental, Social, and Governance (ESG) [1] platforms are becoming well-known as cutting-edge solutions made to simplify reporting and offer stakeholders and decision-makers useful information. These platforms improve accountability and transparency for a wider audience in addition to assisting companies measure, communicate, and accomplish corporate sustainability goals. Organizations can link cybersecurity with governance principles and ensure a more thorough and proactive approach to risk management by including security risk assessment into ESG platforms.

In today's digital economy, organizations face the challenge of balancing ESG objectives—encompassing environmental, social, and governance goals—with the critical need to increase the cybersecurity posture and data protection.

Moreover, as digital transformation accelerates, securing online services is paramount, as it directly impacts both user experience and overall system security. Organizations must implement stringent authentication and authorization mechanisms to ensure seamless yet secure access to digital platforms and are the base to develop Zero Trust approaches to the security of complex digital ecosystem.

In turn, Zero Trust can be seen as a promising strategy to achieve the goals of Governance in the broader context of ESG adoption. As a consequence, Digital Identity Manangement (IdM) becomes the cornerstone of most cyber resilience strategies that complement Governance. For this reason, IdM protocols play a key role in this process by providing standardized mechanisms for user authentication, authorization, and identity verification. By leveraging third-party authentication, IdM protocols eliminate the need to store authentication credentials within the services they support, offering a secure solution that helps both private and public organizations prevent the misuse or abuse of login credentials and mitigate the risk of data breaches. IdM protocol standards—including Security Assertion Markup Language 2.0 (SAML) [10], OpenID Connect (OIDC) [11], and Open Authorization 2.0 (OAuth) [9]—manage user access requests and provide responses based on the information users provide, enhancing security and interoperability across multiple platforms.

Although many security tools offer specialized functionalities, such as vulnerability scanning, security testing, and intrusion detection, their integration into platforms that enable conformance checking and continuous risk assessment presents significant challenges [16], [17]. These

This work has been partially funded by 1) the project SEcurity and RIghts in the CyberSpace (SERICS) (PE00000014) under the Ministero dell'Università e della Ricerca (MUR) National Recovery and Resilience Plan funded by the European Union—Next- GenerationEU, and 2) the joint laboratory between Fondazione Bruno Kessler and the Italian Government Printing Office and Mint, Italy (Istituto Poligrafico e Zecca dello Stato).

tools are often developed independently, with different architectures, data formats, and operational requirements, making seamless interoperability complex. Additionally, platforms that utilize conformance checking and continuous risk assessment require automated workflows, standardized reporting, and real-time risk assessment capabilities, further complicating the integration process. As a result, achieving a cohesive security framework that effectively leverages multiple tools within such platforms demands robust design, and automation.

In summary, the main challenges [12], [13] in leveraging security tools and integrating them effectively to assess risk and perform conformance checking are:

- Their incorporation into automated workflows, as many tools are not designed for seamless execution in dynamic and continuously evolving environments, requiring adaptability and non-interactive operation.
- Making them able to support automated risk assessment and mitigation, as security tools generate large volumes of raw data that must be contextualized, prioritized based on severity and exploitability, and integrated with governance models for effective mitigation.

In this paper, we present an extended version of Micro-Id-Gym (MIG) [2], an open-source tool designed for security testing of IdM deployments ready to be integrated into a Continuous Integration (CI) and Continuous Delivery (CD) pipeline. The extension has been tested in two different IdM deployments: one based on OAuth and the other on OIDC.

The extended version of MIG addresses the two identified challenges.

The first challenge, ensuring seamless integration into automated workflows, was overcome by embedding MIG within a CI/CD pipeline under the DevSecOps[1] framework. This enables automated security testing at different stages of development, reducing reliance on manual interventions and ensuring a consistent and repeatable testing process. By continuously monitoring the security posture of IdM deployments, MIG facilitates early detection of vulnerabilities, allowing organizations to remediate security gaps before they impact production environments. Additionally, the integration of MIG within CI/CD workflows enhances scalability and operational efficiency, making it easier to enforce security policies.

Concerning the second challenge, MIG automates risk assessment and mitigation by prioritizing vulnerabilities based on their severity and impact. By contextualizing security findings, MIG helps security teams focus on critical threats and optimize response efforts. The reporting mechanism provides detailed mitigation hints, enabling organizations to implement targeted security improvements while ensuring regulatory compliance. This automation reduces manual workload, minimizes human error, and ensures that security measures are efficiently enforced across evolving digital environments.

The paper is structured as follows: Section 2 provides an overview of MIG, how it works, how it was extended to be integrated in a CI/CD pipeline. Section 3 reports two different scenarios to highlight the versatility of the proposed solution. Section 4 concludes the paper and highlights future work.

1. https://www.redhat.com/en/topics/devops/what-is-devsecops

2. Micro-Id-Gym

MIG [3] is a tool designed to assist system administrators and testers in security testing of implementations based on IdM protocols and includes a set of tests. For each test that can be executed in MIG, a security tester defines both a session and a test, as shown in Figure 1. The session functions similarly to a user interface integration test commonly used in web application testing. It consists of a sequence of user actions performed by a browser to generate the HTTP messages necessary for the test, allowing MIG to simulate user interactions and analyze the web application's response. The test, written in MIG-L—a formal and concise declarative language for security testing—comprises a sequence of operations for managing and manipulating HTTP messages and the session. The test also integrates an oracle that automates the criteria for evaluating test results. Designed for versatility, MIG-L is adaptable to all web-based IdM protocols, as its principles and structure are sufficiently generic to support them.

As shown in Figure 1, the test handler is responsible for interpreting and executing tests written in MIG-L. When required, it activates the session handler, which processes the session and replicates user actions within a browser. All HTTP messages pass through a proxy, where they can be intercepted and analyzed. The proxy interaction component facilitates communication with the proxy, ensuring compliance with the test-defined conditions. The reporting component generates detailed output on detected vulnerabilities and recommended security measures, offering valuable insights for security testers and stakeholders to improve cybersecurity risk management. To execute tests, we define three distinct application programming interfaces (APIs) that operate independently of the underlying technology. These APIs—the browser API, proxy API, and session API—support automated user interactions, message interception and manipulation, and test session execution, respectively. They streamline the testing process while enhancing the flexibility and usability of MIG.

MIG [6] includes Burp Suite[2] (Burp) as the proxy, Mozilla Firefox[3] (Firefox) as the browser, and MIG-T[4] as the security testing tool. Both Burp and Firefox are containerized within Docker to ensure a consistent and isolated execution environment.

Currently, security testers perform tests on a Deployment Under Test (DUT) using MIG-T through its Graphical User Interface (GUI). However, integrating MIG into a CI/CD pipeline requires an alternative approach, as automated execution and effective test result management are essential for seamless integration and automation.

To address this limitation and align with the ongoing development of MIG [5], MIG has been adapted to meet the following requirements: *(i)* simulating user actions within the DUT as defined in the session, and *(ii)* executing tests on the HTTP messages intercepted by the proxy.

2. https://portswigger.net/burp/communitydownload
3. https://www.mozilla.org/en-US/firefox/
4. https://github.com/stfbk/mig-t

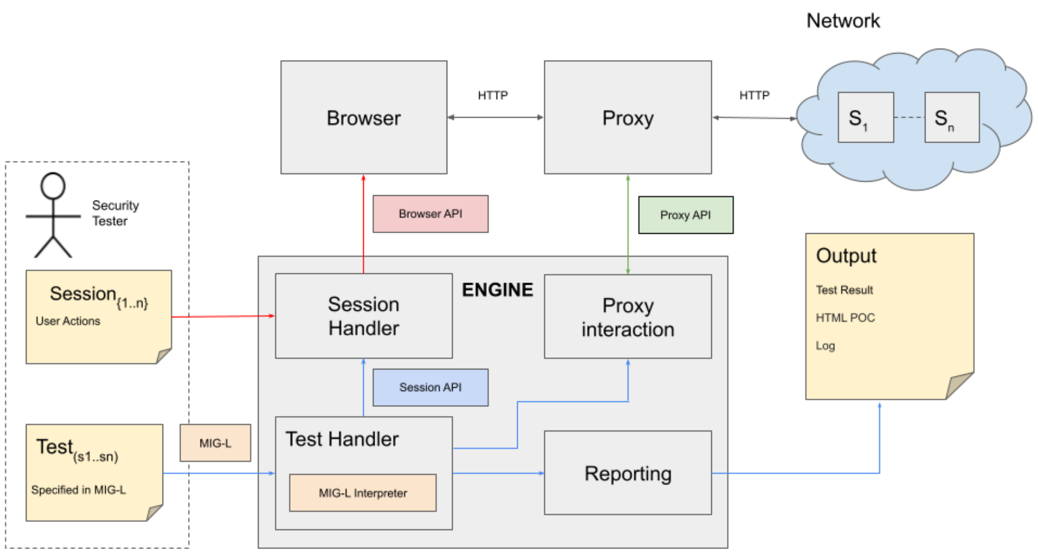

Figure 1. High Level Architecture of MIG [6].

To achieve this, MIG has been enhanced with the necessary capabilities to operate in headless mode—without any GUI—allowing all components, including Mozilla Firefox, Burp, and MIG-T, to run seamlessly.

Running Mozilla Firefox in headless mode was achieved by setting the environment variable `MOZ_HEADLESS` to 1, following the approach described in [7]. By adding `ENV MOZ_HEADLESS=1` to the Dockerfile, Mozilla Firefox runs without launching a visible window, optimizing system resources while enabling seamless web interactions and tests required by MIG-T. This approach eliminates the overhead of managing a full browser instance, ensuring efficient execution within the testing environment.

Regarding Burp, it can be executed in headless mode using the `-Djava.awt.headless=true` option. According to Oracle's technical resources on Java SE, this Java Virtual Machine (JVM) parameter disables the graphical components of Java applications, allowing them to run without a display [8]. By incorporating this headless option into the Dockerfile, we ensured that Burp runs efficiently in such environments.

Regarding MIG-T, we introduced two APIs to facilitate test execution and result retrieval without requiring interaction with the MIG-T GUI. Specifically, we implemented:

- `/execute` An HTTP POST request that validates and executes test by verifying its syntax. The `onlyValidate` parameter can be used (e.g., `/execute?onlyValidate=true`), allowing the test to be validated without execution.
- `/result` An HTTP GET request that checks whether the test execution is complete and retrieves the results. MIG's provides detailed reports, offering developers and security teams rapid feedback and valuable insights into security vulnerabilities, and recommended mitigations to facilitate risk assessment and prioritization of fixes.

Figure 2 illustrates a GitHub Workflow integrating MIG within a CI/CD pipeline for automated security testing. The process follows these steps:

1. Action. A developer initiates the process by performing an action on the Entity Under Test (EUT) repository. The action could be, for instance, pushing new code into repository or submitting a pull request on the main branch.

2. Trigger Pipeline. The action triggers the pipeline, executing the job on a runner machine. The pipeline can also be triggered manually.

3. Retrieve Images. The first step in the GitHub Job[5] is to retrieve the necessary Docker images[6] to set up the execution environment for running the tests. The Docker images include:

- The EUT image used for testing verification.
- The Reference Environment (Ref. Env.) image includes all the necessary entities defined by the IdM protocol.
- The MIG image includes MIG-T, Firefox and Burp.

4. Build and Deploy. On the runner machine, the Docker images are deployed, and the following containers are instantiated:

- The EUT and Ref. Env. are federated and deployed in the same container.
- MIG is deployed in a separate container.

5. Perform Tests. MIG executes tests on the deployed EUT container. The collected tests are organized in a test plan and available in MIG repository[7].

6. Save Results as Artifacts. After the testing is completed, the results are collected and stored as artifacts

5. https://docs.github.com/en/actions
6. https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-an-image/
7. https://github.com/stfbk/mig

on GitHub for sharing. The output includes detailed information on the tests performed, identified vulnerabilities or compliance issues, and suggested mitigations.

7. Data Transfer. Once MIG results are stored as artifacts, the data can be transferred via its APIs. This can be done on demand or integrated as the final step in the workflow.

3. Case Studies

Two use cases are presented to demonstrate the versatility of the proposed solution. Below, we describe Scenario 1 (S1), OAuth for Enterprise deployment; and Scenario 2 (S2), the SPID/CIE OIDC deployment in Developers Italia.

S1: OAuth for Enterprise Deployment

This scenario is based on a case study from the Co-Innovation Lab CLEANSE[8], which focuses on enhancing security automation within enterprise OAuth deployments. The primary objective is to integrate security testing into the Security Development Lifecycle (SDL) by automating security assessments through a CI/CD pipeline.

In this scenario, the EUT acts as the OAuth Client, while the Ref. Env. is Keycloak[9], an open-source identity and access management system that provides an OAuth protocol implementation. The security assessment was conducted using MIG, which was integrated into the CI/CD GitHub pipeline to perform automated security testing and compliance verification.

As part of the security evaluation, a predefined set of tests tailored for OAuth was executed to analyze the client implementation and ensure adherence to best practices outlined in OAuth specifications. The security analysis uncovered a misconfiguration in the Authorization request, specifically related to the Proof Key for Code Exchange (PKCE)[10] mechanism:

- Missing PKCE implementation: The absence of the `code_challenge` and `code_verifier` parameters in the Authorization request indicated that PKCE was not implemented. PKCE is mandatory for clients using the Authorization Code flow, as it prevents Cross-Site Request Forgery (CSRF) and authorization code injection attacks, thereby strengthening the security of OAuth-based authentication flows. Failure to implement PKCE significantly increases the risk of authorization code interception and replay attacks, which could lead to unauthorized access, session hijacking, and potential data breaches.

Following the identification of this misconfiguration, the results were shared with the development team, along with recommended mitigation strategies also returned by MIG. As a result, the identified issue was successfully fixed, improving the security posture of the OAuth client and ensuring compliance with OAuth security best practices.

8. https://www.deda.group/deda/innovazione/co-innovation-lab
9. https://www.keycloak.org/
10. https://www.rfc-editor.org/info/rfc7636

This use case demonstrates how MIG enables early detection and mitigation of security misconfigurations, ensuring automated compliance verification and strengthening security within enterprise OAuth deployments.

S2: SPID/CIE OIDC deployment in Developers Italia

As part of a long-term collaboration with the Italian Government Printing Office and Mint (Istituto Poligrafico e Zecca dello Stato), we developed a set of compliance and security tests for SPID/CIE OIDC [14] deployments. This national digital identity solution is based on the OIDC standard and leverages the Italian electronic identity card (CIE). The tests are available on GitHub[11], offering resources to the community of developers who design and develop code for Italian digital public services.

The goal of this scenario is to verify the compliance of the OpenID Provider (OP) and the Relying Party (RP) of the SPID/CIE OIDC Federation SDK deployment[12], with the SPID/CIE OIDC specifications [14] by using MIG in a CI/CD pipeline and executing a set of tests available in GitHub[13].

In this scenario, the Ref. Env. contains the EUT, configured as either an OP and a RP. Additionally, the Trust Anchor (TA) is included in the Ref. Env. to ensure the integrity and trustworthiness of the authentication process.

After running the tests for the OP, two issues were identified in the Authorization request:

- The `client_id` and `response_type` parameters could be edited and removed, contrary to the requirement that they should be present both as query parameters and inside the JSON Web Token (JWT) request object. This vulnerability weakens the integrity of the Authorization request, potentially allowing attackers to manipulate request parameters, bypass access controls, and gain unauthorized access to protected resources.
- The presence and correctness of the `scope` parameter in the URL were not properly validated, despite the requirement that it must be sent as query parameter and inside the JWT `request` object, with both values being the same. This mismatch occurred because the JWT `request` object suppressed the URL values. Improper validation of the `scope` parameter can lead to privilege escalation, where an attacker could request broader permissions than intended, resulting in unauthorized access to sensitive data or functionalities.

Regarding the RP test results, one issue was identified during the analysis:

- The `iss` parameter was removed from the authentication response without triggering any error at the RP. Despite editing the `iss` parameter with an arbitrary value, the RP failed to validate it correctly, which resulted in a security vulnerability. The lack of proper validation for the `iss`

11. https://github.com/stfbk/mig/tree/master/testplans/spid-cie-oidc/
12. https://github.com/italia/spid-cie-oidc-django
13. https://github.com/stfbk/mig/tree/master/testplans/spid-cie-oidc/implementations/spid-cie-oidc-django

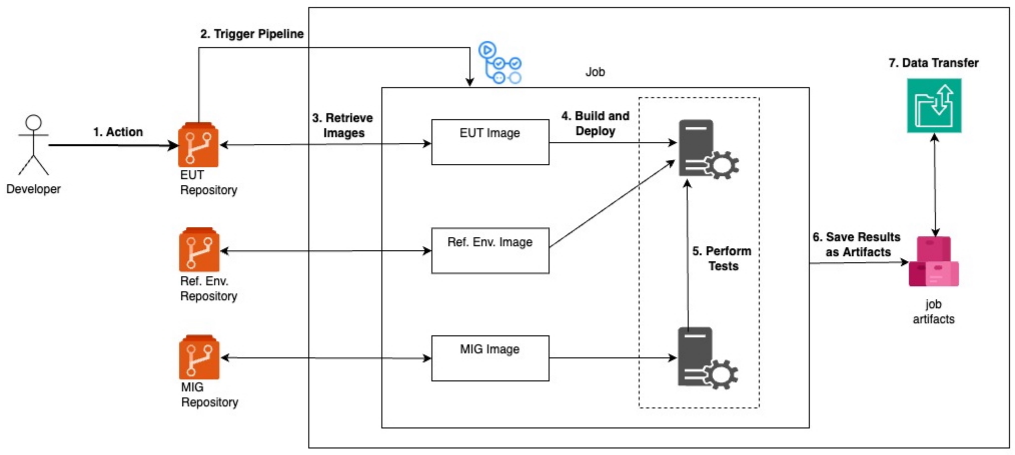

Figure 2. GitHub Workflow integrating MIG for automated security testing in a CI/CD pipeline.

parameter leads to a mix-up attack [15], where an attacker can exploit a malicious OP to impersonate a legitimate server and trick the RP into accepting invalid authentication data. This attack allows to steal authorization codes or access tokens. Once in possession of these, the attacker can gain unauthorized access to the victim's resources, potentially leading to severe privacy breaches or data theft. This issue underlines the critical need for proper validation of all authentication parameters, including the `iss` parameter, to ensure that authentication flows cannot be manipulated by malicious actors. Failure to validate the `iss` parameter properly creates a high-severity vulnerability, exposing authentication flows to manipulation, unauthorized access, and potential data breaches.

All reported issues were addressed by the developers, leading to an improved security posture for the implementations.

4. Conclusions and Future Work

This paper presents an extended version of MIG, an open-source security testing tool designed for IdM deployments and suitable for integration into platforms that support conformance checking and continuous risk assessment. Aligned with DevSecOps principles, MIG enables security assessments for IdM deployments in a CI/CD pipeline by automating testing processes and providing tailored mitigation hints. The transition of MIG from a GUI-based application to a CI/CD-compatible platform enhances scalability and accessibility, allowing for faster, more consistent, and repeatable analyses.

The extended version of MIG improves integration with various security frameworks, accelerates needs analysis for addressing high-severity vulnerabilities, and enables efficient mitigation procedures. By combining vulnerability detection with targeted mitigation hints, MIG strengthens cybersecurity practices.

For future work, we plan to implement an alternative architecture where the Ref. Env. and MIG are deployed in separate containers and provide MIG as a service.

References

[1] T.-T. Li, K. Wang, T. Sueyoshi, and D. D. Wang, "ESG: Research Progress and Future Prospects," *Sustainability*, vol. 13, no. 21, p. 11663, October 2021.

[2] A. Bisegna, R. Carbone, I. Martini, V. Odorizzi, G. Pellizzari, and S. Ranise, "Micro-Id-Gym: Identity Management Workouts with Container-Based Microservices," *International Journal of Information Security and Cybercrime*, vol. 8, no. 1, pp. 45–50, 2019.

[3] A. Bisegna, M. Bitussi, R. Carbone, L. Compagna, A. Sudhodanan, and S. Ranise, "CSRFing the SSO waves: Security testing of SSO-based account linking process," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Vienna, Austria, 2024, pp. 139–154, doi: 10.1109/EuroSP60621.2024.00016.

[4] A. Bisegna, R. Carbone, G. Pellizzari, and S. Ranise, "Micro-Id-Gym: A Flexible Tool for Pentesting Identity Management Protocols in the Wild and in the Laboratory," in *Proceedings of the International Workshop on Emerging Technologies for Authorization and Authentication*, Springer, 2020, pp. 71–89.

[5] A. Bisegna, R. Carbone, and S. Ranise, "Integrating a Pentesting Tool for IdM Protocols in a Continuous Delivery Pipeline," in *Proceedings of the International Workshop on Emerging Technologies for Authorization and Authentication*, Springer, 2021, pp. 94–110, doi: 10.1007/978-3-030-93747-8_7.

[6] A. Bisegna, M. Bitussi, R. Carbone, and S. Ranise, "Enhancing Security Testing for Identity Management Implementations: Introducing Micro-Id-Gym Language and Micro-Id-Gym Testing Tool," *IEEE Security and Privacy*, vol. 22, no. 6, pp. 50–61, Nov.-Dec. 2024, doi: 10.1109/MSEC.2024.3450277.

[7] A. Perunicic, "Running Selenium with Headless Firefox," *Intoli*, July 2017. [Online]. Available:https://intoli.com/blog/running-selenium-with-headless-firefox/.

[8] R. Ananiev and A. Redko, "Using Headless Mode in the Java SE Platform," *Oracle*, June 2006. [Online]. Available: https://www.oracle.com/technical-resources/articles/javase/headless.html.

[9] D. Hardt, V. Torstensson, and A. Parecki, "OAuth 2.1," Internet-Draft draft-ietf-oauth-v2-1-11, Internet Engineering Task Force, Jan. 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-oauth-v2-1-11.

[10] OASIS Security Services Technical Committee, "SAML 2.0 Technical Overview," OASIS, [Online]. Available: https://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html.

[11] OpenID Foundation, "How OpenID Connect Works," [Online]. Available: https://openid.net/developers/how-connect-works/.

[12] Lacework, "4 Reasons Why SecOps Is Still Pretty Difficult," [Online]. Available: https://www.lacework.com/blog/esg-4-reasons-why-secops-is-still-pretty-difficult.

[13] Anvilogic, “ESG Report: Security Operations Challenges and Priorities,” [Online]. Available: https://www.anvilogic.com/report-esg.

[14] Agenzia per l’Italia Digitale (AgID), “SPID CIE OIDC Documentation,” [Online]. Available: https://docs.italia.it/italia/spid/spid-cie-oidc-docs/it/versione-corrente/index.html.

[15] D. Fett, “Mix-Up Revisited: New and Improved Attacks on Federated Login,” [Online]. Available: https://danielfett.de/2020/05/04/mix-up-revisited/.

[16] R. Bischofberger and E. Duss, “SAML Raider—SAML2 Burp Extension,” [Online]. Available: https://github.com/SAMLRaider/SAMLRaider.

[17] R. Thaqi, K. Vishi, and B. Rexha, “Enhancing Burp Suite with Machine Learning Extension for Vulnerability Assessment of Web Applications,” *Journal of Applied Security Research*, vol. 18, no. 4, pp. 789–807, 2023.