

# Análise Caixa Preta das Vulnerabilidades de Segurança OWASP Top 10 Web do Sistema MyMobiConf

Arthur Marciano Pires<sup>1</sup>, Thais Regina de Moura Braga Silva<sup>2</sup>

<sup>1</sup>Instituto de Ciências Exatas e Tecnológicas – Universidade Federal de Viçosa (UFV)  
35.690-000 – Florestal – MG – Brasil

{arthur.pires, thais.braga}@ufv.br

**Resumo.** *O MyMobiConf é um sistema de gerenciamento de eventos presenciais e online, que, até o momento, ainda não foi submetido a nenhum processo de análise de segurança. O objetivo deste estudo foi realizar uma avaliação no MyMobiConf buscando por falhas de segurança presentes em aplicações Web, além de identificar a gravidade e compreender os riscos que essas vulnerabilidades podem representar para a integridade do sistema. Foi utilizado como base a lista das dez principais vulnerabilidades em sistemas Web disponibilizada pela OWASP. Os resultados revelaram que o sistema apresenta exposição significativa a falhas de segurança classificadas como de risco médio ou baixo, representando um nível de risco moderado para a segurança da aplicação.*

## 1. Introdução

A expansão da Internet mundial trouxe consigo um grande impacto para a vida de toda a população, influenciando diretamente na forma como as pessoas se comunicam, trabalham e realizam transações. No entanto, essa conectividade também trouxe um aumento substancial na exposição a ameaças cibernéticas, à medida que informações pessoais e empresariais são cada vez mais armazenadas e compartilhadas online.

Conforme a tecnologia avança, torna-se cada vez mais evidente a importância da proteção de sistemas computacionais contra ameaças e ataques cibernéticos. Existem diferentes abordagens para avaliar a segurança de sistemas, sendo as mais conhecidas o teste de caixa branca (*white-box*) e o teste caixa preta (*black-box*). No teste caixa branca o testador possui acesso ao código-fonte da aplicação, sendo capaz de verificar a lógica interna do software e encontrar erros específicos de implementação que levem à problemas de segurança. Já no teste caixa preta o testador examina a segurança do sistema sem conhecer os detalhes internos do código-fonte, avaliando o sistema com base em seu comportamento visível, como um usuário externo faria.

Neste contexto, este trabalho se concentra na avaliação de segurança do MyMobiConf<sup>1</sup>, um sistema de gerenciamento e apoio a participação em eventos, composto por uma aplicação Web e um aplicativo móvel, que pode ser utilizado por muitas pessoas e empresas simultaneamente. Devido ao fato desta aplicação administrar diferentes tipos de informações sensíveis e que não podem ser acessadas indevidamente, como e-mails, senhas e *feedbacks* de usuários, a segurança deste sistema distribuído é de extrema importância. Em vista disso, surge a questão de pesquisa que orienta este estudo: Quantas, quais e o quão graves são as vulnerabilidades de segurança do sistema MyMobiConf?

---

<sup>1</sup><http://www.mymobiconf.caf.ufv.br/>

O objetivo principal deste trabalho, em particular, é realizar uma avaliação de segurança caixa preta utilizando o guia de vulnerabilidades de segurança OWASP Top 10 de 2021<sup>2</sup> da aplicação Web que compõem o sistema MyMobiConf.

O restante deste trabalho está organizado da seguinte forma: a Seção 2 apresenta alguns trabalhos relacionados encontrados na literatura. A Seção 3 apresenta a fundamentação teórica necessária, discutindo mais a fundo sobre o sistema MyMobiConf e os tópicos descritos pelo OWASP Top 10 Web. A Seção 4 descreve o passo a passo das avaliações feitas para cada um destes tópicos. A Seção 5 apresenta uma discussão dos resultados encontrados e, por fim, a Seção 6 apresenta a conclusão e alguns possíveis trabalhos futuros.

## 2. Trabalhos relacionados

O trabalho realizado em [Pereira 2017] segue uma abordagem semelhante ao presente estudo, utilizando a ferramenta de testes automatizados (scanner) OWASP *Zed Attack Proxy* (ZAP), além de outras ferramentas para buscar por vulnerabilidades presentes na aplicação Web testada, que consiste em um portal destinado a promover o acesso e dar maior visibilidade às revistas científica publicadas. A diferença para o atual estudo é que este se aprofunda na análise das vulnerabilidades detectadas pelo scanner, investigando se as falhas identificadas são ou não falsos positivos. Também é importante destacar que mais de um scanner automatizado foi empregado neste trabalho, de forma a consolidar os resultados obtidos e ampliar a cobertura na identificação das vulnerabilidades.

O trabalho em [Franzese 2023] examina as vulnerabilidades do OWASP Top 10 Web de 2021 na plataforma Moodle, uma plataforma de grande porte bastante utilizada por instituições de ensino por todo o mundo. O autor também utiliza a ferramenta automatizada de detecção de vulnerabilidades OWASP ZAP como seu principal recurso analítico. No entanto, este estudo se restringe aos resultados fornecidos pela ferramenta ZAP, sem explorar outras metodologias ou ferramentas automatizadas para identificar vulnerabilidades adicionais. Em contrapartida, o presente trabalho adota uma abordagem mais abrangente, utilizando uma variedade de ferramentas e metodologias na busca por falhas na segurança da aplicação.

Já os estudos conduzidos por [Farias et al. 2023] e [Sampaio 2021] apresentam uma análise prática dos impactos provocados pelas vulnerabilidades listadas no OWASP Top 10 Web, contudo o trabalho de [Sampaio 2021] utiliza a lista da OWASP de 2017 enquanto o estudo de [Farias et al. 2023] utiliza a lista da OWASP de 2021. Os autores baseiam suas investigações em aplicações propositalmente vulneráveis, como os laboratórios práticos para testes de penetração fornecidos pela PortSwigger<sup>3</sup>, empresa especializada em segurança de aplicativos Web. Além disso, utilizam o Juice Shop<sup>4</sup>, uma aplicação Web intencionalmente insegura desenvolvida pela OWASP, demonstrando de forma controlada como cada vulnerabilidade pode ser explorada. Por fim, os estudos também sugerem procedimentos de mitigação das falhas de segurança alinhados com os padrões estabelecidos pela fundação OWASP. É importante notar que ambos os trabalhos empregam diversas ferramentas e abordagens com o objetivo de descobrir as vulnerabi-

---

<sup>2</sup><https://owasp.org/www-project-top-ten/>

<sup>3</sup><https://portswigger.net/>

<sup>4</sup><https://juice-shop.herokuapp.com/>

lidades listadas no OWASP Top 10. Contudo, eles optam por utilizar softwares propositalmente vulneráveis para fins de estudo. Em contraste, o presente trabalho tem como objetivo identificar as vulnerabilidades em uma aplicação real, até então não testada, oferecendo uma perspectiva sobre os desafios de segurança em cenários do mundo real.

### 3. Fundamentação Teórica

#### 3.1. MyMobiConf

O MyMobiConf é um sistema projetado para melhorar a experiência de participantes e organizadores de eventos e conferências. O sistema está disponível em duas partes distintas: um aplicativo móvel, destinado aos participantes, e uma aplicação Web, destinada aos organizadores do evento.

O aplicativo móvel do MyMobiConf oferece aos usuários a capacidade de se inscrever em eventos previamente cadastrados, proporcionando acesso completo a programação do evento. Além disso, os participantes têm a oportunidade de interagir em tempo real com os organizadores, seja respondendo a questionários ou fornecendo feedback sobre o evento e suas atividades. Por outro lado, a aplicação Web permite que os organizadores configurem o evento de acordo com suas necessidades específicas, sendo possível criar novas atividades, elaborar questionários, publicar notícias importantes, cadastrar patrocinadores e administradores, além de ser possível visualizar os feedbacks e opiniões dadas pelos participantes do evento em tempo real, auxiliando na melhoria contínua do evento.

Para a implementação da parte móvel do MyMobiConf foi utilizado o Ionic<sup>5</sup>, um framework para desenvolvimento de aplicativos móveis que conta com o uso de algumas linguagens e bibliotecas como Angular<sup>6</sup>, Typescript e HTML. Já a parte Web do sistema foi desenvolvida em Django<sup>7</sup>, um framework para desenvolvimento Web escrito em Python, além da utilização do Javascript, CSS, HTML<sup>8</sup> e do framework Bootstrap<sup>9</sup>.

#### 3.2. OWASP Top 10

A OWASP (*Open Web Application Security Project*) [OWASP 2023] é uma organização internacional sem fins lucrativos que visa melhorar a segurança de software. Todos os projetos, ferramentas e documentos produzidos pela mesma são gratuitos e facilmente acessíveis em seu site, abertos a qualquer pessoa interessada em melhorar a segurança de aplicações Web.

Uma das principais iniciativas da OWASP é o OWASP Top 10, um projeto atualizado regularmente que identifica e lista as dez principais vulnerabilidades de segurança em aplicações Web e Mobile, servindo como um guia para desenvolvedores e profissionais de segurança, ajudando a mitigar ameaças e promover melhores práticas na criação de aplicações seguras. Neste artigo, será utilizada a versão mais recente do OWASP Top 10, lançada em 2021, especificamente para a Web.

Os dez principais riscos de segurança em sistemas Web segundo a OWASP são:

---

<sup>5</sup><https://ionicframework.com/>

<sup>6</sup><https://angular.io/>, <https://www.typescriptlang.org/>

<sup>7</sup><https://www.djangoproject.com/>, <https://www.python.org/>

<sup>8</sup><https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>, <https://www.w3.org/TR/CSS/>

<sup>9</sup><https://getbootstrap.com/>

1. **Controle de acesso quebrado:** essa categoria se refere a situações em que os sistemas não aplicam corretamente políticas de acesso ou autenticação. Controles de acesso corrompidos normalmente dão brecha para que usuários não autorizados tenham acesso a recursos fora das permissões pretendidas. Um exemplo de problema comum é a violação do princípio do privilégio mínimo, ou seja, o usuário só pode realizar as ações previstas para sua categoria.
2. **Falhas de criptografia:** ocorre quando informações confidenciais são mal protegidas e podem ser acessadas por atacantes, comprometendo a privacidade dos dados. De acordo com a OWASP, qualquer falha que resulte em uma exposição de dados sensíveis ou críticos de uma pessoa pode ser considerada uma falha de criptografia. Envolve a quebra do sigilo de dados em repouso (banco de dados ou sistema de arquivos) e em trânsito (servidor ao cliente). Alguns problemas comuns são dados sendo trafegados e armazenados em texto plano, uso de protocolo HTTP ao invés de HTTPS e algoritmos e chaves de criptografia fracos ou inexistentes.
3. **Injeção:** esta vulnerabilidade acontece quando dados não confiáveis são incorporados indevidamente em comandos ou consultas que são executados por um sistema. Isso pode permitir que um atacante insira código malicioso ou comandos diretamente em um aplicativo, explorando assim a falta de validação ou filtragem adequada dos dados de entrada.
4. **Design Inseguro:** o design inseguro trata-se de problemas de segurança que surgem a partir de falhas de Arquitetura e de Design dos sistemas, tornando-os vulneráveis a ataques. Um sistema com design inseguro não pode ser corrigido nem mesmo com a melhor das implementações, já que os devidos controles de segurança não foram feitos. Um exemplo são sistemas que possuem perguntas e respostas para alteração de senhas, o que não pode ser considerado uma evidência de identidade, já que mais de uma pessoa pode saber a resposta. Um problema comum é a ausência de modelagem de ameaças.
5. **Configuração incorreta de segurança:** ocorre quando um sistema é configurado de maneira inadequada, deixando falhas de segurança que podem ser exploradas por atacantes.
6. **Componentes vulneráveis e desatualizados:** se refere ao risco da utilização de softwares desatualizados, como bibliotecas e frameworks, que quando possuem vulnerabilidades oferecem grande risco ao sistema. Muitas vezes a utilização de componentes vulneráveis se dá pela falta de atualizações e um processo automatizado de busca por vulnerabilidades em tais componentes. Para mitigar esse risco, é crucial manter regularmente todos os componentes de software atualizados com as versões mais seguras e corrigidas disponíveis.
7. **Falhas de identificação e autenticação:** muitas aplicações acabam falhando no processo de confirmação da identidade do usuário, bem como em processos de autorização. Alguns problemas comuns são as aplicações que:
  - Permitem ataques de força bruta ou outros ataques automatizados.
  - Permitem a utilização de senhas fracas/conhecidas
  - Não estabelecem um limite de tempo para as sessões dos usuários (permitindo que eles permaneçam autenticados indefinidamente).
8. **Falhas de software e integridade de dados:** esta categoria engloba os riscos relacionados à não verificação da integridade dos dados usados por uma aplicação.

As falhas de integridade de dados ocorrem quando o sistema não tem capacidade de proteger o código contra violações de integridade (acontece quando o código é obtido de uma fonte não confiável).

9. **Registro de segurança e falhas de monitoramento:** esta categoria visa ajudar a detectar, escalar e responder a violações ativas. Falha na análise de logs acaba impedindo que organizações detectem ataques em tempo hábil para aplicação de controles reativos.
10. **Falsificação de solicitação do lado do servidor:** Falhas de SSRF (*Server-Side Request Forgery*) ocorrem quando uma aplicação Web permite que um invasor manipule URLs para acessar recursos internos não autorizados. Isso acontece quando o servidor não valida adequadamente as URLs fornecidas pelo usuário, permitindo que o invasor faça solicitações em nome do servidor vulnerável podendo resultar em vazamento de informações confidenciais ou em ações não autorizadas em sistemas internos. Um problema comum são aplicações que buscam recursos remotos sem validar o URL fornecido pelo usuário.

## **4. Avaliação de Vulnerabilidades OWASP Top 10 Web no MyMobiConf**

### **4.1. Testes Automatizados - Scanner OWASP ZAP**

Conforme destacado no estudo de [Ferrão and Kreutz 2017], scanners de vulnerabilidades gratuitos e de código aberto nem sempre conseguem detectar todas as vulnerabilidades conforme descrito em seus manuais, enfatizando a importância de empregar múltiplas ferramentas para identificar vulnerabilidades e falhas em sistemas Web de forma abrangente. Neste contexto, este trabalho não se limita a testes automatizados como também realiza testes manuais visando uma maior cobertura das vulnerabilidades presentes no sistema testado.

Os primeiros testes foram realizados utilizando a ferramenta de scan automatizado disponibilizada pela própria OWASP, o OWASP *Zed Attack Proxy* (ZAP). De acordo com o estudo de [Ferrão and Kreutz 2017], esta ferramenta apontou melhores resultados na detecção de vulnerabilidades de alto risco, sendo também o scanner que mais detectou falhas de segurança no trabalho realizado pelos autores. Além de detectar vulnerabilidades Web, o scanner também fornece informações importantes sobre como esses erros podem ser corrigidos e informa em qual categoria do Top 10 a vulnerabilidade encontrada pertence.

As vulnerabilidades encontradas e reportadas pelo OWASP ZAP para a aplicação Web do MyMobiConf são mostradas na tabela 1. O scanner também obteve resultados classificados como informativos. No entanto estes resultados não serão contemplados neste trabalho, de maneira a concentrar o esforço nas vulnerabilidades classificadas como de risco alto, médio e baixo. A ferramenta não detectou nenhuma falha classificada como tendo alto nível de risco. Em contrapartida, para vulnerabilidades de risco médio e baixo o ZAP encontrou quatro falhas de cada classe.

#### **4.1.1. Content Security Policy (CSP) Header Not Set**

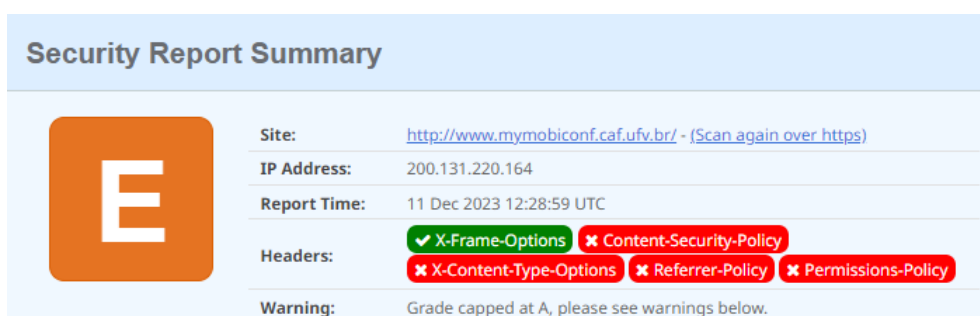
O alerta da vulnerabilidade de risco médio *Content Security Policy (CSP) Header Not Set* indica que o cabeçalho de Política de Segurança de Conteúdo (CSP) não foi configurado

**Tabela 1. Lista de vulnerabilidades detectadas pelo OWASP Zap**

| Vulnerabilidade  | Risco | Categoria no OWASP TOP 10                      |
|--|-------|--|
| Content Security Policy (CSP) Header Not Set                       | Médio | A05: Configuração incorreta de segurança       |
| Hidden File Found  | Médio | A05: Configuração incorreta de segurança       |
| Navegação no Diretório   | Médio | A01: Controle de acesso quebrado               |
| Vulnerable JS Library  | Médio | A06: Componentes vulneráveis e desatualizados  |
| Cookie No HttpOnly Flag  | Baixo | A05: Configuração incorreta de segurança       |
| Cross-Domain Javascript Source File Inclusion                      | Baixo | A08: Falhas de software e integridade de dados |
| Server Leaks Version Information via “Server” HTTP Response Header | Baixo | A05: Configuração incorreta de segurança       |
| X-Content-Type-Options Header Missing                              | Baixo | A05: Configuração incorreta de segurança       |

para a página Web em questão. A CSP é uma medida de segurança que ajuda a prevenir ataques, como *Cross-Site Scripting* em uma página. Cross-Site Scripting é uma vulnerabilidade que ocorre quando um atacante é capaz de inserir scripts maliciosos em uma página Web e executá-los no navegador de outros usuários.

Para confirmar a presença da vulnerabilidade, foi realizado um teste utilizando a ferramenta online SecurityHeaders [SecurityHeaders 2023]. Esta plataforma permite a verificação e análise dos cabeçalhos de segurança de uma página Web. Ao inserir a URL do site, a ferramenta examina os cabeçalhos de segurança HTTP e atribui uma pontuação baseada na adequação das configurações de segurança. Ela verifica se o site implementou diversas práticas de segurança Web, incluindo a Política de Segurança de Conteúdo (CSP). A Figura 1 ilustra o resultado final obtido por meio da ferramenta. Os resultados do teste indicaram que a aplicação foi desenvolvida sem a implementação adequada da CSP, confirmando assim a presença da vulnerabilidade.



**Figura 1. Resultado da ferramenta SecurityHeaders**

#### **4.1.2. *Hidden File Found***

A falha de segurança *Hidden File Found*, encontrada pelo scanner, indica que um arquivo oculto foi identificado como acessível ou disponível durante a análise, podendo vaziar informações confidenciais administrativas ou de configuração que podem ser utilizadas por um atacante mal intencionado.

Durante a investigação das URLs mencionadas pelo scanner, onde supostamente estariam esses arquivos ocultos, o MyMobiConf sempre retornava uma mensagem de erro indicando que a página não foi encontrada. Portanto, esta vulnerabilidade reportada pelo OWASP ZAP parece ser um falso positivo e não representa uma ameaça real.

#### **4.1.3. Navegação no Diretório**

Já a vulnerabilidade de *Navegação no Diretório* indica que quando um servidor Web não está configurado corretamente e permite a listagem de diretórios, os invasores podem explorar essa vulnerabilidade para descobrir informações sensíveis, como scripts ocultos, arquivos de configuração, arquivos de origem de backup, entre outros. Isso pode levar à exposição de informações confidenciais que deveriam estar ocultas aos usuários não autorizados.

No caso do MyMobiConf, adicionar “/static” à URL padrão redireciona o usuário para uma página que expõe informações sensíveis, incluindo códigos CSS e JavaScript usados na aplicação. Este incidente ilustra claramente uma falha em controlar o acesso adequado às informações. Essa vulnerabilidade se enquadra no tópico de “Broken Access Control” do Top 10 da OWASP, já que qualquer um é capaz de acessar o diretório que contém informações que deveriam ser ocultas ao usuário.

#### **4.1.4. *Vulnerable JS Library***

Por fim, a quarta falha de segurança de nível médio encontrada pelo scanner é classificada como *Vulnerable JS Library*, indicando que o sistema do MyMobiConf possui bibliotecas vulneráveis em sua implementação. Esta vulnerabilidade se enquadra na categoria de “Componentes vulneráveis e desatualizados” do Top 10 da OWASP. O ZAP identificou que as bibliotecas do Bootstrap, versão 3.3.7 e do jQuery, versão 1.12.4 são vulneráveis.

Segundo o Banco de dados de vulnerabilidades do Snyk [Snyk Ltd. 2023], o principal banco de dados para vulnerabilidades de código aberto e configurações incorretas na nuvem, versões anteriores ao Bootstrap 3.4.1 são suscetíveis a ataques de *Cross-Site Scripting (XSS)*. O mesmo é válido para o jQuery em versões anteriores à 3.5.0 e posteriores à 1.5.1.

Para confirmar a presença dessas vulnerabilidades, a ferramenta RetireJs [RetireJS 2023], um recurso livre e de código aberto, foi empregada para identificar a presença de versões de bibliotecas conhecidas por apresentarem vulnerabilidades. Este processo foi realizado de maneira eficiente por meio de uma extensão específica para o navegador Firefox. O resultado gerado pela ferramenta, como demonstrado na Figura 2, confirma as falhas nas bibliotecas mencionadas pelo scanner OWASP ZAP.

## Retire.js

☒ Enabled ☐ Show unknown

|           |       |  |
|-----------|-------|--|
| bootstrap | 3.3.7 | Found in <a href="http://www.mymobiconf.caf.ufv.br/static/vendor/bootstrap/js/bootstrap.min.js">http://www.mymobiconf.caf.ufv.br/static/vendor/bootstrap/js/bootstrap.min.js</a> - Vulnerability info: |
|           |       | Medium 28236 XSS in data-template, data-content and data-title properties of tooltip/popover CVE-2019-8331 GHSA-9v3m-8fp8-mj99 <a href="#">[1]</a>   |
|           |       | Medium 20184 XSS in data-target property of scrollspy CVE-2018-14041 GHSA-pj7m-g53m-7638 <a href="#">[1]</a>   |
|           |       | Medium 27044 In Bootstrap before 3.4.0, XSS is possible in the tooltip data-viewport attribute. CVE-2018-20676 GHSA-3mgp-fx93-9xv5 <a href="#">[1]</a>   |
|           |       | Medium 20184 XSS in data-container property of tooltip CVE-2018-14042 <a href="#">[1]</a>  |
|           |       | Medium In Bootstrap before 3.4.0, XSS is possible in the affix configuration target property. CVE-2018-20677 GHSA-ph58-4vrj-w6hr <a href="#">[1]</a>   |
|           |       | Medium XSS is possible in the data-target attribute. CVE-2016-10735 GHSA-4p24-vmcr-4gqj <a href="#">[1]</a>  |
|           |       | Low 72 Bootstrap before 4.0.0 is end-of-life and no longer maintained. <a href="#">[1]</a>   |

Figura 2. Resultado obtido através do uso da extensão retire.js para detecção das vulnerabilidades do Bootstrap versão 3.3.7

### 4.1.5. Server Leaks Version Information

Os alertas da vulnerabilidade *Server Leaks Version Information* ocorreram porque o servidor revela informações sobre sua versão por meio dos cabeçalhos de resposta HTTP. Esta informação pode ser utilizada por atacantes mal intencionados a fim de identificar e explorar vulnerabilidades específicas presentes na versão que o servidor se encontra. Na Figura 3 é possível ver, grifado em verde, que o servidor realmente informa a sua versão no cabeçalho de resposta, confirmando a vulnerabilidade detectada pelo scanner.

### 4.1.6. Cookie sem sinalizador HttpOnly definido

Para falhas de segurança classificadas como nível baixo também foram encontrados quatro problemas.

A primeira vulnerabilidade de baixo risco encontrada foi a de *Cookie sem sinalizador HttpOnly definido*. *HttpOnly* é um marcador adicional incorporado em um cabeçalho de resposta HTTP *Set-Cookie* que é enviado pelo servidor em uma resposta HTTP. Usar o sinalizador *HttpOnly* ao gerar um cookie ajuda a mitigar o risco de o script do lado do cliente acessar o cookie protegido.

Ao analisar a Figura 3, que representa a resposta do servidor a uma tentativa de login no sistema, observa-se o envio de dois cookies para o navegador do usuário.



```
Response
Pretty Raw Hex Render
1 HTTP/1.1 302 Found
2 Date: Wed, 13 Dec 2023 13:30:10 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Location: /control/
5 Content-Length: 0
6 X-Frame-Options: SAMEORIGIN
7 Vary: Cookie
8 Set-Cookie: csrftoken=mTCC3SL9a7wrLWozMdbhADWdXzL6E74JtynnLSy0Du160fVhCBuqDIefkDTv1hRI; expires=Wed, 11 Dec 2024 13:30:10 GMT; Max-Age=31449600; Path=/; SameSite=Lax
9 Set-Cookie: sessionid=yp38t04ysg17yy6erw7ym3t7soch5kbp; expires=Wed, 27 Dec 2023 13:30:10 GMT; HttpOnly; Max-Age=1209600; Path=/; SameSite=Lax
10 Connection: close
11 Content-Type: text/html; charset=utf-8
```

Figura 3. Resposta do servidor a uma tentativa de login

Os cookies em questão são o “sessionid”, utilizado para armazenar um identificador de sessão atribuído ao usuário durante a autenticação, permitindo que a aplicação mantenha informações de estado durante a interação do usuário. Além disso, há o cookie “csrftoken”, um token de segurança empregado para proteger contra ataques de falsificação de solicitações entre sites, conhecidos como CSRF (*Cross-Site Request Forgery*).

Nota-se, ainda na Figura 3. que a flag *HttpOnly* está configurada no cookie “sessionid”, restringindo seu acesso apenas a solicitações HTTP e impedindo que scripts do lado do cliente o acessem. No entanto, a mesma configuração não está aplicada ao cookie “csrftoken”, indicando um possível risco ao sistema, já que scripts maliciosos podem conseguir acessar ou manipular este cookie.

#### 4.1.7. *Cross-domain JavaScript Source File Inclusion*

Outra vulnerabilidade de risco baixo detectada pelo scanner é denominada *Cross-domain JavaScript Source File Inclusion* (*Inclusão de Arquivo de Origem JavaScript entre Domínios*). Esta vulnerabilidade ocorre quando o aplicativo Web carrega arquivos JavaScript de um domínio externo sem a validação adequada, permitindo que um invasor injete códigos maliciosos e potencialmente assuma o controle do aplicativo. A inclusão inadequada de arquivos de origem JavaScript entre domínios pode acarretar problemas significativos, tais como vazamento de dados do usuário ou a execução de códigos JavaScript maliciosos no contexto da aplicação.

No contexto do MyMobiConf, o sistema importa arquivos JavaScript de uma CDN (*Content Delivery Network* ou Rede de Distribuição de Conteúdo), cujo propósito é otimizar o desempenho da entrega de conteúdo na Web, visando melhorar a velocidade de carregamento das páginas.

É importante destacar que a maioria das CDNs oferece algum nível de proteção contra ameaças à segurança do sistema. Diante disso, é razoável concluir que o alerta em questão não representa uma ameaça imediata ao sistema. Pelo contrário, funciona como um aviso para possíveis futuras importações de arquivos JavaScript provenientes de

outros domínios.

#### 4.1.8. *X-Content-Type-Options Header Missing*

Por fim, o OWASP ZAP detectou a presença da vulnerabilidade *X-Content-Type-Options Header Missing*. Esta vulnerabilidade ocorre devido à ausência de configuração do cabeçalho *X-Content-Type-Options*. A falta deste cabeçalho pode criar uma vulnerabilidade conhecida como *MIME Sniffing*. Os navegadores, quando encontram um arquivo com um tipo de conteúdo (*MIME type*) desconhecido ou ausente no cabeçalho *Content-Type*, podem tentar adivinhar o tipo de conteúdo realizando o que é chamado de *sniffing*. Isso pode levar a uma série de ataques e explorações por parte de um invasor. A evidência apresentada na Figura 1 confirma a presença dessa vulnerabilidade na aplicação, já que a mesma não faz uso deste cabeçalho, tal como indicado pelo scanner de segurança.

## 4.2. Testes Manuais

Em seguida foram realizados testes manuais visando encontrar outras vulnerabilidades presentes na lista do Top 10 OWASP que não foram detectadas pelo ZAP Scanner.

Inicialmente, foi realizada uma análise para identificar possíveis falhas de criptografia. Para isso utilizou-se o *Burp Suite* [Suite 2023], um software desenvolvido pela *PortSwigger*, para a realização de testes de segurança em aplicações Web. O *Burp Suite* é capaz de identificar, interceptar, registrar e exibir o tráfego que circula em uma rede, proporcionando uma visão detalhada da comunicação entre o servidor e o cliente.

O teste, como mostrado na Figura 4, comprova que as credenciais da aplicação são enviadas através do método POST via HTTP, e como mostrado nos trechos sublinhados de vermelho, os dados são transportados em texto plano pelo canal de comunicação sem criptografia. Um atacante então poderia interceptar o canal e comprometer as informações transportadas. Especificamente, esta vulnerabilidade permitiria a interceptação de uma requisição de login por parte do atacante, obtendo informações de nome de usuário e senha dos usuários que realizam login no sistema.



```
1 POST /login HTTP/1.1
2 Host: www.mymobiconf.caf.ufv.br
3 Content-Length: 130
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://www.mymobiconf.caf.ufv.br
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36
9 Accept:
10 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://www.mymobiconf.caf.ufv.br/
12 Accept-Encoding: gzip, deflate
13 Accept-Language: en-US,en;q=0.9
14 Cookie: _ga=GA1.1.2005347466.1701092717; _ga_3GKTCB3HHS=GS1.1.1701710538.2.1.1701710546.0.0.0; csrftoken=
15 mTCC3SL9a7wrLWozMdbbADWdXzL6E74JtynnLSy0Du160fVhCBuqDIefkDTv1hRI
16 Connection: close
17 csrfmiddlewaretoken=0mTwfEYF0q6Zm4KPQK5FF1LDXvRjQJ6j71EhXELkhNBEpnxG8o0I63FkzZIdTT3&user=Arthur+Marciano&pass=arthur&login=Entrar
```

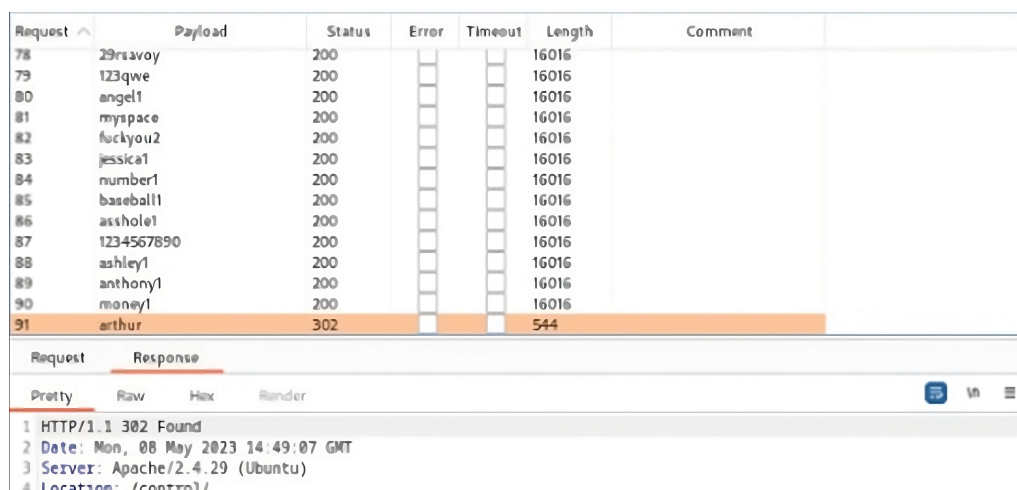
Figura 4. Análise do tráfego de rede com o Burp Suite

Também foram realizados testes manuais com o objetivo de detectar falhas de identificação e autenticação. A primeira vulnerabilidade identificada é que o sistema per-

mite a utilização de senhas fracas, com a única restrição sendo o comprimento mínimo de 6 caracteres.

Outro problema presente no MyMobiConf é que os tempos limite da sessão do sistema não estão definidos corretamente. Isso significa que um usuário pode permanecer autenticado indefinidamente, mesmo após fechar a guia do navegador, representando um risco substancial, especialmente em cenários de computadores públicos. Um invasor pode utilizar o mesmo navegador uma hora depois, e o usuário ainda estará autenticado, o que pode ser explorado de maneira maliciosa.

Por fim, o sistema não implementa medidas automatizadas de proteção contra ameaças ou tentativas de preenchimento de credenciais por força bruta. Um exemplo que ilustra isso é a tentativa de descoberta de senha por ataque de força bruta utilizando o *Burp Suite*. No exemplo da Figura 5, utilizou-se uma pequena *wordlist* com diferentes senhas a serem testadas, de modo que a última senha seria a correta. Ao analisar a imagem, observa-se que, quando o atacante utiliza a senha correta, o sistema retorna uma mensagem com status e tamanho diferentes das demais, evidenciando que a aplicação é, de fato, vulnerável a ataques de força bruta.



| Request | Payload    | Status | Error | Timeout | Length | Comment |
|---------|------------|--------|-------|---------|--------|---------|
| 78      | 29rsavoy   | 200    |       |         | 16016  |         |
| 79      | 123qwe     | 200    |       |         | 16016  |         |
| 80      | angel1     | 200    |       |         | 16016  |         |
| 81      | myspace    | 200    |       |         | 16016  |         |
| 82      | fuckyou2   | 200    |       |         | 16016  |         |
| 83      | jessica1   | 200    |       |         | 16016  |         |
| 84      | number1    | 200    |       |         | 16016  |         |
| 85      | baseball1  | 200    |       |         | 16016  |         |
| 86      | asshole1   | 200    |       |         | 16016  |         |
| 87      | 1234567890 | 200    |       |         | 16016  |         |
| 88      | ashley1    | 200    |       |         | 16016  |         |
| 89      | anthony1   | 200    |       |         | 16016  |         |
| 90      | money1     | 200    |       |         | 16016  |         |
| 91      | arthur     | 302    |       |         | 544    |         |

| Request | Response                            |
|---------|-------------------------------------|
| 1       | HTTP/1.1 302 Found                  |
| 2       | Date: Mon, 08 May 2023 14:49:07 GMT |
| 3       | Server: Apache/2.4.29 (Ubuntu)      |
| 4       | Location: /control/                 |

Figura 5. Ataque de força bruta ao sistema do MyMobiConf

### 4.3. Vulnerabilidades não encontradas

Para as vulnerabilidades 3, 4, 9 e 10, não foram identificados problemas no sistema MyMobiConf durante a avaliação de segurança.

Em relação à vulnerabilidade de injeção (número três na lista OWASP 2021), foram realizados testes manuais de modo a tentar inserir uma instrução SQL maliciosa no formulário de login do sistema. Para isso, foram passadas instruções SQL na entrada do usuário na tentativa de burlar as consultas SQL realizadas e obter acesso não autorizado ao sistema. Além disso, o scanner OWASP ZAP também realizou diversos testes visando encontrar falhas de injeção de código. Os resultados não apontaram nenhum problema relacionado a essa vulnerabilidade, indicando que o MyMobiConf implementa as devidas medidas para evitar a ocorrência deste tipo de falha.

Quanto à vulnerabilidade de design inseguro (número quatro na lista OWASP 2021), inclui riscos como a falta de controle de acesso adequado e falta de proteção de

dados sensíveis. Em vista disso, constatou-se que os riscos associados a essa vulnerabilidade também foram abordados em outras vulnerabilidades analisadas anteriormente. Portanto, não foi conduzida uma análise detalhada dessa categoria da lista OWASP Top 10 de 2021.

No caso da falha de segurança número nove, relacionada a registros de segurança e falhas de monitoramento, uma análise aprofundada não foi possível devido à indisponibilidade de acesso aos registros gerados pelo sistema MyMobiConf. Sem a possibilidade de examinar os logs, não foi viável realizar uma análise para identificar atividades suspeitas ou possíveis violações de segurança no sistema.

Finalmente, foi conduzido um teste para identificar possíveis vulnerabilidades relacionadas às falhas de falsificação de solicitação do lado do servidor (SSRF). Para isso utilizou-se o Wapiti [Wapiti 2023], uma ferramenta de varredura de segurança automatizada projetada para identificar vulnerabilidades em aplicações Web. A ferramenta envia várias solicitações e analisa as respostas das solicitações em busca de padrões que indiquem a presença da vulnerabilidade de SSRF. Os resultados do teste conduzido com o Wapiti não indicaram problemas relacionados à falsificação de solicitação do lado do servidor. Isso significa que, com base na análise realizada pela ferramenta, não foram encontradas vulnerabilidades significativas ou explorações potenciais associadas a SSRF no aplicativo da Web testado.

## **5. Discussão dos Resultados**

A seção anterior apresentou o passo a passo de como foi feito o processo de avaliação das vulnerabilidades no aplicativo Web do sistema MyMobiConf. Já esta seção se propõe a consolidar os resultados apresentados, proporcionando uma análise mais profunda sobre a quantidade, tipos e gravidade dos problemas de segurança identificados.

A presença de vulnerabilidades em seis das dez principais categorias sugere que o sistema, embora não esteja completamente desprotegido, é relativamente inseguro. Esta constatação indica a necessidade de aprimoramentos na segurança, uma vez que cada categoria representa uma potencial porta de entrada para ameaças cibernéticas.

Uma observação positiva nos resultados é a ausência de alertas sobre vulnerabilidades consideradas graves pelo scanner automatizado da OWASP. No entanto, o estudo destaca que, embora não existam vulnerabilidades classificadas como graves, várias foram categorizadas como de nível médio. Estas, apesar de não representarem um perigo imediato, podem comprometer a integridade e a segurança do MyMobiConf se não forem devidamente tratadas. A gravidade dessas vulnerabilidades médias deve ser considerada com seriedade, uma vez que ainda podem proporcionar oportunidades para exploração por parte de atacantes.

Além disso, deve-se dar destaque ao fato do sistema utilizar o protocolo HTTP ao invés de HTTPS e não realizar a criptografia dos dados transmitidos entre o servidor e o usuário. Conforme alerta a OWASP, falhas de criptografia ocupam a segunda posição no ranking de riscos de segurança para aplicativos Web, ganhando destaque a cada atualização da lista do "Top ten". Diante disso, essa vulnerabilidade pode ser considerada como a mais crítica dentre todas as questões identificadas, representando um ponto de vulnerabilidade significativo para a aplicação.

Finalmente, para consolidar os resultados obtidos até então, foram utilizadas mais duas ferramentas de varredura automatizada: o Wapiti (detalhado na seção 4.2) e o Skipfish [Skipfish 2023]. O Skipfish é uma ferramenta de análise automatizada de vulnerabilidades em sistemas Web, de código aberto, que realiza a identificação de arquivos por meio de um processo de *crawl recursivo*. Além disso, gera um relatório final que apresenta as vulnerabilidades identificadas e informações cruciais. Segundo o estudo de [Ferrão and Kreutz 2017], o Skipfish teve o segundo melhor desempenho da detecção de vulnerabilidade de risco alto, ficando atrás apenas do OWASP ZAP, também utilizado anteriormente.

Após a execução destes dois outros scanners, é possível concluir que o sistema do MyMobiConf não apresenta vulnerabilidades de alto risco, uma vez que, assim como o OWASP ZAP, nem o Skipfish nem o Wapiti identificaram problemas de segurança dessa magnitude. Além disso, os resultados obtidos por estas ferramentas destacaram vulnerabilidades de nível médio e baixo bem semelhantes às vulnerabilidades detectadas pelo OWASP ZAP. O Wapiti, por exemplo, concentrou-se principalmente em questões relacionadas à ausência de cabeçalhos de segurança, um padrão que também foi reconhecido pelo scanner da OWASP. Por outro lado, o Skipfish apontou vulnerabilidades associadas a conteúdo externo desatualizado incorporado nas páginas do sistema, uma preocupação alinhada com as detecções do ZAP, classificadas como *Vulnerable JS Library*. Esta consistência nos resultados fortalece a confiabilidade das conclusões sobre a análise de segurança do MyMobiConf.

## 6. Conclusão e Trabalhos Futuros

Este trabalho apresentou um estudo abrangente em relação à segurança do MyMobiConf, conduzindo uma análise completa das principais vulnerabilidades presentes em aplicações Web atualmente. Ao abordar diversos conceitos essenciais relacionados à segurança de software, o estudo enfatizou a importância de manter um aplicativo protegido contra potenciais ameaças. A análise da aplicação seguiu a abordagem da OWASP, utilizando a lista das dez principais vulnerabilidades em aplicações Web como referência. A intenção foi cobrir de forma abrangente todos os itens dessa lista, visando identificar e documentar as vulnerabilidades associadas a cada um deles.

Os resultados apresentados indicam que o sistema MyMobiConf possui falhas significativas em termos de segurança, possuindo problemas que poderiam ser explorados por atacantes mal intencionados, e que, portanto, devem receber os devidos cuidados para sua mitigação.

Como continuidade ao trabalho aqui apresentado, a condução de uma análise de segurança abrangente do aplicativo móvel do MyMobiConf, usando como base de orientação a lista da OWASP Top 10 para aplicativos móveis, pode ser feita. Além desta análise, uma perspectiva futura seria a realização de um estudo aprofundado sobre como as vulnerabilidades identificadas podem ser mitigadas de maneira eficaz. Por fim, outra opção seria a exploração de abordagens adicionais de teste, como os testes de caixa-branca. Essa expansão nos métodos de avaliação de segurança permitirá uma visão mais completa e abrangente, identificando possíveis pontos de vulnerabilidade que podem não ter sido identificados neste trabalho.

## Referências

- Farias, G. d. M. et al. (2023). Análise prática e estudo dos impactos provocados pelas principais vulnerabilidades do owasp top 10 2021. Trabalho de Conclusão de Curso (Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia do Piauí.
- Ferrão, I. and Kreutz, D. (2017). Segurança na web: análise black-box de scanners de vulnerabilidades. In *Anais da I Escola Regional de Engenharia de Software*, pages 137–144, Porto Alegre, RS, Brasil. SBC.
- Franzese, L. F. (2023). Um estudo das vulnerabilidades do owasp top 10 na plataforma moodle. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Federal de Uberlândia.
- OWASP (2023). Open web application security project. <https://owasp.org/>. Acesso em: 5 de dezembro de 2023.
- Pereira, P. H. A. (2017). Testes de segurança em aplicação web. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Faculdade de Computação, Instituto de Ciências Exatas e Naturais, Universidade Federal do Pará.
- RetireJS (2023). Retire.js. <https://retirejs.github.io/retire.js/>. Acesso em: 10 de setembro de 2023.
- Sampaio, F. F. (2021). Uma análise prática das principais vulnerabilidades em aplicações web baseado no top 10 owasp. Trabalho de Conclusão de Curso (Graduação em Redes de Computadores) - Universidade Federal do Ceará, Campus de Quixadá.
- SecurityHeaders (2023). Disponível em <https://securityheaders.com/>. Acesso em: 15 outubro 2023.
- Skipfish (2023). Web application security scanner. <https://gitlab.com/kalilinux/packages/skipfish/>. Acesso em: 13 de novembro de 2023.
- Snyk Ltd. (2023). Snyk - open source security platform. Disponível em <https://security.snyk.io/>. Acesso em: 23 outubro 2023.
- Suite, B. (2023). Web security testing software. Disponível em <https://portswigger.net/burp>. Acesso em: 23 outubro 2023.
- Wapiti (2023). The web application security scanner. <https://wapiti-scanner.github.io/>. Acesso em: 13 de novembro de 2023.

## Anexo

### Tutorial passo a passo da execução dos testes

#### Kali Linux

A maior parte dos testes foram realizados utilizando o Kali Linux, que é uma distribuição Linux projetada para realização de testes de segurança. O Kali Linux oferece uma ampla gama de ferramentas de segurança e testes de penetração.

- **Link oficial do Kali Linux:** <https://www.kali.org/>

No site oficial, você encontrará informações detalhadas sobre o sistema operacional, documentação, downloads, e recursos adicionais relacionados ao Kali Linux.

#### Coleta de Informações

Primeiramente foi feita uma análise de quais software o MyMobiConf utiliza. Para isto foi utilizada a ferramenta **Wappalyzer** que é uma extensão de navegador que identifica as tecnologias e softwares utilizados em um site.

- **Link oficial do Wappalyzerx:** <https://www.wappalyzer.com/>

Em seguida foi utilizado o **Dirb (Directory Brute-Forcer)**, que é uma ferramenta de teste de penetração incluída no Kali Linux. O principal objetivo do Dirb é descobrir diretórios e arquivos escondidos em um servidor web que podem ser alvos potenciais para ataques ou explorações.

- **Documentação e informações sobre o dirb:** <https://www.kali.org/tools/dirb/>

#### Ferramentas utilizadas

Todos os scanners automatizados utilizados estão disponíveis no Kali Linux. Para executá-los basta seguir os passos indicados em suas páginas:

- **OWASP Zap:** Site oficial: <https://www.zaproxy.org/>  
Para executar um scan basta abrir a ferramenta e clicar em "automated scan" e inserir no link da aplicação.
- **Wapiti:** Site oficial: <https://wapiti-scanner.github.io/>  
Para executar um scan simples basta inserir o seguinte comando no terminal do Kali Linux: `wapiti -u http://target/`
- **Skipfish:** <https://www.kali.org/tools/skipfish/>  
Para executar um scan simples utiliza-se o seguinte comando no terminal do Kali Linux: `skipfish -o skip http://target/`

Outra ferramenta muito utilizada foi o **Burp Suite**. O Burp permite interceptar e modificar a comunicação entre o navegador web, além de permitir ataques de força bruta, comparação entre solicitações HTTP, entre muitas outras funcionalidades. A ferramenta também está disponível no Kali Linux e possui uma versão grátis e uma paga (com mais funcionalidades).

- **Site oficial do Burp Suite :**<https://portswigger.net/burp>

Para confirmação das vulnerabilidades obtidas foram utilizadas outras várias ferramentas listadas a seguir:

- **Security Headers:** ferramenta para análise e verificação dos cabeçalhos de segurança configurados no sistema.  
Link oficial: <https://securityheaders.com/>
- **Banco de dados de vulnerabilidades do Snyk:** útil para confirmar a presença de possíveis vulnerabilidades presentes na versão dos softwares utilizados.  
Link oficial: <https://security.snyk.io/>