MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL GOIANO - CAMPUS URUTAÍ
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

MATHEUS DE SOUZA EL MADI
LUCAS ANTÔNIO DE CASTRO

AVALIAÇÃO DE VULNERABILIDADES EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS

Urutaí, GO
2022

MATHEUS DE SOUZA EL MADI
LUCAS ANTÔNIO DE CASTRO

AVALIAÇÃO DE VULNERABILIDADES EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Sistema de Informação do Instituto Federal Goiano - Campus Urutaí, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Amaury Walbert de Carvalho

Urutaí, GO

2022

Sistema desenvolvido pelo ICMC/USP
Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas - Instituto Federal Goiano

EL48a El Madi, Matheus; Castro, Lucas Antônio
AVALIAÇÃO DE VULNERABILIDADES EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS / Matheus El Madi; Lucas Antônio Castro; orientador Amaury Walbert de Carvalho. -- Urutaí, 2022.
43 p.

TCC (Graduação em BACHARELADO EM SISTEMAS DE INFORMAÇÃO) -- Instituto Federal Goiano, Campus Urutaí, 2022.

1. Vulnerabilidade. 2. Segurança da informação. 3. Ferramentas de detecção. 4. OWASP. I. Walbert de Carvalho, Amaury , orient. II. Título.

Responsável: Johnathan Pereira Alves Diniz - Bibliotecário-Documentalista CRB-1 n°2376

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano, a disponibilizar gratuitamente o documento no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

Identificação da Produção Técnico-Científica

[ ] Tese
[ ] Dissertação
[ ] Monografia – Especialização
[X] TCC - Graduação
[ ] Produto Técnico e Educacional - Tipo: _____
[ ] Artigo Científico
[ ] Capítulo de Livro
[ ] Livro
[ ] Trabalho Apresentado em Evento

Nome Completo do Autor:
Matheus de Souza El Madi Matrícula: 2016101202010276
Lucas Antônio de Castro Matrícula: 201611202010151

Título do Trabalho: AVALIAÇÃO DE VULNERABILIDADES EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS

Restrições de Acesso ao Documento
Documento confidencial: [X] Não [ ] Sim, justifique: _____

Informe a data que poderá ser disponibilizado no RIIF Goiano: 27/07/2022
O documento está sujeito a registro de patente? [ ] Sim [X] Não
O documento pode vir a ser publicado como livro? [ ] Sim [X] Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

1. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

2. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

3. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Urutaí-go, 21/07/2022.
Local Data

Matheus de Souza El Madi Lucas Antônio de Castro Carvalho
Assinatura do Autor e/ou Detentor dos Direitos Autorais

Ciente e de acordo:

Amauri Wallbert de Carvalho
Assinatura do(a) orientador(a)

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Ata nº 72/2022 - DE-UR/CMPURT/IFGOIANO

INSTITUTO FEDERAL GOIANO – CAMPUS URUTAÍ

DIRETORIA / GERÊNCIA DE GRADUAÇÃO E PÓS-GRADUAÇÃO

COORDENAÇÃO DOS CURSOS DA ÁREA DE INFORMÁTICA

CURSO SUPERIOR DE SISTEMAS DE INFORMAÇÃO

ATA DE APRESENTAÇÃO DE TRABALHO DE CURSO

Aos quinze dias do mês de julho de dois mil e vinte e dois, reuniram-se, via videoconferência na plataforma Microsoft Teams, os professores: Amaury Walbert de Carvalho, Mônica Sakuray Pais e Nattane Luiza da Costa, para avaliar o Trabalho de Curso do(s) acadêmico(s): **Matheus de Souza El Madi e Lucas Antônio de Castro** , como requisito necessário para a conclusão do Curso Superior de Sistemas de Informação desta Instituição. O presente TC tem como título: **AVALIAÇÃO DE VULNERABILIDADE EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS,** foi orientado por Amaury Walbert de Carvalho.

Após análise, foram dadas as seguintes notas:

Membros	**Alunos / Notas**	
	Matheus de Souza El Madi	**Lucas Antônio de Castro**
Amaury Walbert de Carvalho	8,5	8,5
Mônica Sakuray Pais	8,8	8,8
Nattane Luiza da Costa	7,7	7,7
MÉDIA FINAL:	**8,3**	**8,3**

OBSERVAÇÕES: _____

_____

Por ser verdade firmamos a presente:

<<Assinado Eletronicamente>>

Amaury Walbert de Carvalho

<<Assinado Eletronicamente>>

Mônica Sakuray Pais

<<Assinado Eletronicamente>>

Nattane Luiza da Costa

Documento assinado eletronicamente por:

- **Monica Sakuray Pais**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/07/2022 14:54:06.
- **Nattane Luiza da Costa**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/07/2022 14:54:00.
- **Amaury Walbert de Carvalho**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/07/2022 14:53:06.

Este documento foi emitido pelo SUAP em 15/07/2022. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 408258
Código de Autenticação: 8a56bc4c1b

INSTITUTO FEDERAL GOIANO
Campus Urutaí
Rodovia Geraldo Silva Nascimento, Km 2,5, Zona Rural, None, None, URUTAI / GO, CEP 75790-000
(64) 3465-1900

MATHEUS DE SOUZA EL MADI
LUCAS ANTÔNIO DE CASTRO

AVALIAÇÃO DE VULNERABILIDADES EM SISTEMAS WEB DE ÓRGÃOS PÚBLICOS DA REGIÃO DA ESTRADA DE FERRO EM GOIÁS

Defendido e aprovado em: 15/07/2022

COMISSÃO EXAMINADORA

Prof. Amaury Walbert de Carvalho
(orientador)

Prof. Mônica Sakuray Pais
Instituto Federal Goiano, GO

Prof. Nattane Luiza da Costa
Instituto Federal Goiano, GO

RESUMO

Com o avanço da tecnologia a vida do ser humano melhorou bastante com a criação de sites webs, mas também dificultou bastante manter segura toda aplicação. Se não tomar os devidos cuidados, cibercriminosos podem roubar informações importantes ou até mesmo excluí-las com intuito de prejudicar o público alvo. Neste trabalho, selecionamos duas ferramentas gratuitas a fim de fazer uma verificação nos sistemas web das prefeituras das cidades da região da estrada de ferro em Goiás e detectar possíveis vulnerabilidades e classificar de acordo com a *Open Web Application Security Project* (OWASP). Essas vulnerabilidades são erros, falhas ou brechas no servidor web permitem o acesso indevido por pessoas mal-intencionadas. As regiões escolhidas da estrada de ferro se devem ao fato de nossa localização atual, onde não foi feito um trabalho correlato com essas regiões. Com isso os resultados apresentados foram bem satisfatórios, mas distintos em relação a vulnerabilidades encontradas. Uma ferramenta demonstrou ser mais agressiva para a detecção enquanto outra ficou no superficial. Com isso incluímos várias tabelas e gráficos para uma melhor visualização dos resultados obtidos, detalhando os principais pontos explorados pelas ferramentas de detecção.

Palavras-chave: Vulnerabilidade, Segurança da informação, Ferramentas de detecção, OWASP.

ABSTRACT

With the advancement of technology, the life of the human being has greatly improved with the creation of web sites, but it also made it quite difficult to keep every application safe. If you do not take proper care, cybercriminals can steal important information or even delete it in order to harm the target audience. In this work we selected two free tools in order to make a check on the web systems of the municipalities of the cities of the railroad region in Goiás and detect potential vulnerabilities and classify according to the Open Web Application Security Project (OWASP). These vulnerabilities are errors, failures or breaches on the web server allow undue access by malicious people. The chosen regions of the railway are due to the fact that our current location, where no correlated work has been done with these regions. With this the results presented were very satisfactory, but distinct in relation to vulnerabilities found. One tool has been shown to be more aggressive for detection while another has been in the superficial. With this we include several tables and graphs for a better visualization of the results obtained, detailing the main points explored by the detection tools.

Keywords: Vulnerability, Information security, Detection tools, OWASP.

LISTA DE ILUSTRAÇÕES

Figura 1. Imagem da ferramenta Skipfish, Imagem produzida pelo autor, de um dos Escaneamentos. 24
Figura 2. Imagem da ferramenta Nikto, Imagem produzida pelo autor, de um dos Escaneamentos. 25

Gráfico 1. Vulnerabilidades encontradas com a ferramenta Skipfish. 27
Gráfico 2. Vulnerabilidades encontradas com a ferramenta Nikto. 28
Gráfico 3. Junção das vulnerabilidades encontradas com as ferramentas Skipfish e Nikto. 28
Gráfico 4. Calor da ferramenta Skipfish. 29
Gráfico 5. Gráfico de barra da ferramenta Skipfish. 29
Gráfico 6. Calor da ferramenta Nikto. 30
Gráfico 7. Gráfico de barra da ferramenta Nikto. 30
Gráfico 8. Calor da junção das ferramentas Skipfish e Nikto. 31
Gráfico 9. Gráfico de barra da junção das ferramentas Skipfish e Nikto. 31
Gráfico 10. Gráfico de barra de vulnerabilidades detectadas em ordem decrescente de acordo com as ferramentas Nikto e Skipfish. 32

LISTA DE TABELAS

Tabela 1. Tabela de informações de identificação dos alvos – Skipfish.....21
Tabela 2. Tabela de informações de identificação dos alvos – Nikto. 21
Tabela 3. Tabela de Resultados da ferramenta - Skipfish. 22
Tabela 4. Tabela de Resultados da ferramenta – Nikto. 23
Tabela 5. Tabela de Vulnerabilidades de acordo com a classificação Owasp, pela ferramenta Skipfish. 26
Tabela 6. Tabela de Vulnerabilidades de acordo com a classificação Owasp, pela ferramenta Nikto. 27
Tabela 7. Tabela de vulnerabilidades detectadas em ordem decrescente de acordo com as ferramentas Nikto e Skipfish. 32
Tabela 8. Tabela de Vulnerabilidades detectadas pela ferramenta Skipfish fora da classificação Owasp. 33
Tabela 9. Tabela de Vulnerabilidades detectadas pela ferramenta Nikto fora da classificação Owasp. 34

SUMÁRIO

1 INTRODUÇÃO 8

2 FUNDAMENTAÇÃO TEÓRICA 9

2.1 Segurança da Informação **10**

2.2 Vulnerabilidades em Sistemas Web **11**

2.3 Ferramentas de Detecção de Vulnerabilidades **13**

2.4 Governo Eletrônico **14**

2.5 HTTP e HTTPS **15**

3 DESENVOLVIMENTO **15**

3.1 Metodologia **15**

3.2 Testes **16**

3.3 Resultados **20**

3.4 Discussão dos Resultados **27**

3.5 Estratégias de Prevenção e Mitigação dos Problemas **37**

4 CONCLUSÃO 40

5 REFERÊNCIAS 41

1 INTRODUÇÃO

A constante evolução tecnológica e a exponencial difusão dos meios de comunicação global proporcionam o surgimento de interesses escusos sobre o processo de manipulação das informações nos sistemas computacionais, em especial, os sistemas web. Diariamente a internet enfrenta novos riscos, como: ataques de segurança aos sistemas computacionais, exploração de vulnerabilidades em sistemas web, propagação de códigos maliciosos, uso de ferramentas de invasão, entre outros.

A proteção dos dados em sistemas web de instituições privadas e públicas em todo o mundo tem se tornado um desafio. Enquanto a evolução tecnológica transforma as formas de comunicação e o comércio, vulnerabilidades têm surgido como limitantes desse desenvolvimento (NAKAMURA; GEUS, 2007). Vazamentos de informações a partir de exploração de vulnerabilidades podem acontecer quando dados confidenciais são expostos na internet, ou seja, quando dados de grande valor que não podem estar disponíveis de forma indiscriminada são acessados de forma irregular e compartilhados na rede. Isso ocorre, em alguns casos, quando a segurança do sistema web não recebe uma boa prática de segurança de dados e com isso o cibercriminoso identifica a falha e, através dela, acessa os dados confidenciais.

A exploração de vulnerabilidades são situações motivadas, em alguns casos, por pessoas mal-intencionadas procurando falhas de segurança em aplicações com o intuito de causar prejuízos ao sistema explorado ou a seus usuários. As falhas podem ocorrer quando uma arquitetura é mal planejada, e os prazos para a entrega de desenvolvimento da aplicação são relativamente curtos para uma maior avaliação dos cenários relacionados à segurança, por muitas vezes não apresentar uma ampla variedade de casos de testes. Com isso, uma forma de evitar as vulnerabilidades em sistema web é estudar e analisar os principais erros cometidos no desenvolvimento de aplicações, configuração de serviços em sistemas Web e realizar testes para identificar possíveis vulnerabilidades no sistema.

No caso da coleta e tratamento de dados pessoais dos usuários de sistemas web, considerados valiosos na era da informação em que vivemos, há mecanismos que buscam garantir a privacidade e a segurança desses dados, inclusive estabelecendo punições para os casos onde eles são negligenciados. A Lei Geral de Proteção de Dados Pessoais (LGPD) estabelece diretrizes importantes e obrigatórias para a coleta, processamento e armazenamento de dados pessoais e seus objetivos consistem em assegurar o direito à privacidade e à proteção de dados pessoais dos usuários, por meio de práticas transparentes e seguras, garantindo direitos

fundamentais, estabelecendo regras claras sobre o tratamento de dados pessoais, fortalecendo a segurança das relações jurídicas e a confiança, garantindo a livre iniciativa, a livre concorrência e a defesa das relações comerciais e de consumo e também promovendo a concorrência e a livre atividade econômica, inclusive portabilidade de dados.

> A maior parte dos artigos da LGPD (Lei nº 13.709/2018), entrou em vigor no dia 18 de setembro de 2020, e a sua vigência integral se completou no dia 1º de agosto de 2021 (com os arts. 52/54, sobre as sanções administrativas e o procedimento para a sua aplicação), e as mudanças relevantes perceptíveis na prática do tratamento de dados pessoais ocorreram progressivamente, no final de 2020 e durante todo o ano atual, até completar um ano em 18 de setembro de 2021. Além disso, a publicação da LGPD completou três anos no dia 15 de agosto de 2021 (CARDOSO, 2021).

Neste trabalho será realizada uma análise de eventuais vulnerabilidades de segurança em sistemas web em portais. Alguns dos alvos de ataques na rede são os sites governamentais, também conhecidos como e-govs ou governos eletrônicos. Esses portais tem a finalidade de oferecer serviços e informações, para o cidadão, logo, são de suma importância para o governo, pois caso alguma informação seja divulgada de forma errada os prejuízos poderiam ser incalculáveis (SANTOS et al., 2014).

Portanto, se torna extremamente importante a análise de eventuais vulnerabilidades de segurança em sistemas web em portais das prefeituras das cidades que compõem a região da estrada de ferro em Goiás. A hipótese é que será possível encontrar vulnerabilidades em todos os portais das prefeituras das cidades-alvo. Neste trabalho foi utilizado ferramentas automatizadas de verificação de vulnerabilidades para a análise de potenciais vulnerabilidades, entretanto o objetivo não é a comparação das ferramentas, e sim analisar a segurança nos sistemas web.

O trabalho é organizado em capítulos, onde no capítulo 2 evidencia a fundamentação teórica e os trabalhos correlatos, contribuindo para o entendimento e estruturação do trabalho, no capítulo 3 temos o desenvolvimento da pesquisa, sendo composto pela metodologia, resultados, discussão dos resultados e estratégias de prevenção e mitigação dos problemas, pôr fim a conclusão, na qual reúne as considerações finais obtidas na construção deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

O presente capítulo apresenta a fundamentação teórica acerca da temática do trabalho, afim de estabelecer os conceitos em torno da segurança de informação, vulnerabilidade de sistemas web, ferramentas de detecção de vulnerabilidades e governo eletrônico.

2.1 Segurança da Informação

A segurança da informação é definida por Oliveira (2001) como o processo de proteção de informações armazenadas em computadores e redes de processamento de dados. A preocupação com a segurança da informação se iniciou devido os crescentes ciberataques de quadrilhas que tentam roubas dados de informações cadastrais e bancárias, afim de invadir as contas bancárias e sacar o dinheiro (BRASIL, 2018). Além disso, existem constantes ameaças diárias à segurança da informação como: vírus, acessos indevidos, vazamento de informações, hackers, fraudes, erros, entre outros (BRASIL, 2018; VIDAL, 2022).

> A preocupação com a segurança da informação motivou a criação de padrões e normas internacionais. Em 1995, foi criada a norma BS 7799, um padrão britânico que trata da definição de requisitos para um Sistema de Gestão de Segurança da Informação (SGSI), a partir daí foram desenvolvidas várias outras normas e legislações que definem diretrizes para garantir a segurança em um ambiente computacional. A norma mais recente é a ISO/IEC 27001, de 2005 (STEFANINI, 2021; VIDAL, 2022).

A Segurança da Informação tem a finalidade fundamental de proteger a informação referente à sua confidencialidade, integridade, disponibilidade e autenticidade (SUMMERS, 1997; MASCARENHAS NETO et al., 2019).

A confidencialidade se diz respeito a privacidade dos dados da organização. Está relacionada às medidas de segurança para garantir que informações confidenciais e críticas não sejam roubadas por meio de ciberataques, e outras práticas, dos sistemas organizacionais (MASCARENHAS NETO et al., 2019), e garantia de que os dados da empresa não estarão disponíveis nem serão divulgados a indivíduos, entidades ou processos sem autorização (SÊMOLA, 2014). Afim de garantir a confidencialidade, uma empresa necessita da adoção de medidas preventivas como, a definição do acesso às informações apenas a pessoas autorizadas; também é necessário conscientizar toda a equipe para a não violação das regras de segurança, e proteger os computadores contra ciberataques, vírus e espionagem (MACEDO, 2021).

Integridade corresponde à preservação da precisão, consistência e confiabilidade das informações e sistemas pela empresa ao longo dos processos ou de seu ciclo de vida, garantindo que nenhuma interferência externa irá corromper, comprometer ou danificar os dados. É importante que os dados circulam ou sejam armazenados do mesmo modo como foram criados, sem que haja interferência externa para corrompê-los, comprometê-los ou danificá-los (SÊMOLA, 2014; MASCARENHAS NETO et al., 2019).

A disponibilidade, por sua vez, se diz respeito ao tempo e à acessibilidade dos dados e sistemas da empresa, ou seja, se estão disponíveis e podem ser acessados pelos colaboradores, a qualquer momento (SÊMOLA, 2014).

A autenticidade é um pilar tão importante quanto os outros, pois a partir dele que toda informação que foi colocada nas suas condições iniciais e de forma autêntica não pode ser alterada sem permissões e quando precisar sofrer alterações deverá ser informado e o mesmo feito apenas por pessoas capacitadas e autorizadas.

2.2 Vulnerabilidades em Sistemas Web

As vulnerabilidades em aplicações Web são brechas, erros, falhas e falta de filtros em um firewall de aplicação web e no servidor web. As vulnerabilidades mais exploradas nos dias de hoje são as do tipo buffer overflow, que muitas vezes pode dar privilégios de administrador para o invasor, executar códigos maliciosos remotamente, burlar particularidades de cada sistema, ataques de Negação de Serviços (DDoS), e acesso restrito ao sistema e programas (OWASP, 2007; IBM, 2022).

Uma vulnerabilidade pode ser definida como um ponto falho em um sistema que permita a realização e a concretização de um ataque a um sistema computacional (APARECIDO; BELLEZI, 2014).

Dentre as vulnerabilidades existentes há as que afetam sistemas web. Essas são estudadas e listadas por OWASP et al. (2007), sendo organizadas em uma lista com as dez vulnerabilidades mais críticas (OWASP, 2007). A OWASP - *The Open Web Application Security Project* é uma fundação que tem como objetivo melhorar a segurança de softwares, permitindo que indivíduos ou empresas tomem decisões informadas. A classificação TOP 10 da OWASP consiste nos seguintes grupos:

- **Quebra de Controle de Acesso:** 94% das aplicações foram testadas para alguma forma de controle de acesso quebrado. Os 34 CWEs mapeados para Quebra de Controle de Acesso tiveram mais ocorrências em aplicações do que em qualquer outra categoria.
- **Falhas Criptográficas:** anteriormente conhecido como *Exposição de Dados Sensíveis*, que era um sintoma amplo em vez de uma causa raiz. O foco renovado aqui está nas falhas relacionadas à criptografia, que muitas vezes leva à exposição de dados confidenciais ou sistema comprometido.
- **Injeção:** 94% das aplicações foram testadas para alguma forma de injeção com uma taxa de incidência máxima de 19%, uma taxa de incidência média de 3,37% e os 33 CWEs mapeados nesta categoria têm o segundo maior número de ocorrências em aplicações, com 274 mil ocorrências.

- **Design Inseguro:** é uma nova categoria para 2021, com foco em riscos relacionados a falhas de projeto. Se quisermos genuinamente "mover para a esquerda como setor", precisamos de mais modelagem de ameaças, padrões e princípios de design seguros e arquiteturas de referência. Um design inseguro não pode ser corrigido por uma implementação perfeita, pois, por definição, os controles de segurança necessários nunca foram criados para a defesa contra-ataques específicos.
- **Configuração Insegura:** 90% dos aplicativos foram testados para alguma forma de configuração insegura, com uma taxa de incidência média de 4,5% e mais de 208 mil ocorrências de CWEs mapeados para esta categoria de risco.
- **Componente Desatualizado e Vulnerável:** foi anteriormente intitulado "Usar componente com vulnerabilidade conhecida" e é o número 2 na pesquisa da comunidade Top 10, mas também tinha dados suficientes para chegar ao Top 10 por meio de análise de dados. Esta categoria subiu da 9ª posição em 2017 e é um problema conhecido que temos dificuldade em testar e avaliar o risco. É a única categoria a não ter nenhuma Vulnerabilidade e Exposições Comuns (CVEs) mapeada para os CWEs incluídos, portanto, uma exploração padrão e pesos de impacto de 5,0 são considerados em suas pontuações.
- **Falha de Identificação e Autenticação:** era conhecida anteriormente como Falha de Autenticação e caiu da terceira posição para essa, e foram incluídas as CWEs que mais se relacionam com as falhas na identificação. Essa categoria ainda é parte integrante do Top 10, mas a maior disponibilidade de estruturas (frameworks) padronizadas parece estar ajudando a reduzir.
- **Falha na Integridade de Dados e Software:** é uma nova categoria em 2021, focadas em fazer premissas relacionadas a atualização de software, dados críticos, e linhas de CI/CD que não verificam a integridade. Um dos maiores pesos dos dados nessa categoria são CVE/CVSS mapeados para os 10 CWEs nesta categoria. A categoria **Desserialização Insegura** agora faz parte dessa categoria.
- **Monitoramento de Falhas e Registros de Segurança:** anteriormente chamado de Registro e Monitoramentos Insuficientes e foi adicionado pela pesquisa da comunidade de Top 10, ficando em terceiro lugar, passando na 10° posição anterior. Essa categoria foi expandida para incluir um maior número de falhas, sendo um desafio para testar e não está bem representada nos dados de CVE/CVSS. No entanto, falhas nessa categoria podem impactar diretamente a visibilidade, o alerta de incidente e a perícia.

- **Falsificação de Solicitação do Lado do Servidor:** foi adicionada a partir da pesquisa da comunidade, sendo a primeira da classificação. Os dados mostram uma taxa de incidência relativamente baixa com cobertura de teste acima da média, junto com classificações acima da média para potencial de exploração e impacto. Esta categoria representa o cenário em que os membros da comunidade de segurança estão nos dizendo que isso é importante, embora não esteja ilustrado nos dados neste momento.

2.3 Ferramentas de Detecção de Vulnerabilidades

A detecção de vulnerabilidades tem por objetivo verificar a resistência dos sistemas em relação aos métodos de ataque existentes (SOARES, 2014). Visto as descobertas diárias de novas falhas em diferentes sistemas é de importante a realização de auditorias preventivas. Mais especificamente, os testes de invasão podem dar um diagnóstico fidedigno sobre a segurança dos sistemas em questão (SOARES, 2014).

Parte do processo de teste de detecção de vulnerabilidades consiste em verificar se é possível detectar novas vulnerabilidades em um sistema (KIM; SOLOMON, 2014). Porém, antes de iniciar esses testes, é necessário realizar um planejamento, ocasião em que é decidido qual tipo de teste será efetuado.

De acordo com Resende e Barbosa (2013), as ferramentas de detecção de vulnerabilidades são:

- **Scanners:** normalmente esses programas podem ser usados tanto por administradores de rede como por hackers. A finalidade desses softwares é percorrer os principais serviços em uso, exibindo o resultado dos processos e portas em aberto, que possam ser explorados para a prática de invasões. Periodicamente são lançados scanners que detectam as vulnerabilidades mais recentes. É de responsabilidade do administrador de rede, corrigir as vulnerabilidades exibidas na busca dessa ferramenta, antes que os hackers as explorem. Existem vários modelos de Scanners, onde cada um deles executam um scan específico, alguns modelos desses Scanners são: TCP Connect Scan, UDP Scan, TCP Null Scan, TCP Fin Scan e muitos outros;
- **Exploiters:** é um modelo desenvolvido para atacar um sistema localmente, ou remotamente. Ele varia quanto à forma e o poder do ataque, e pode ser constituído por porções de códigos ou uma sequência de comandos. Os Exploiters são preparados para explorar falhas muito específicas e falhas em aplicativos;

- **Exploração de Bugs:** Este é um tipo de ataque em que o Hacker explora uma falha ou também conhecida como um Bug, para realizar o ataque por meio desse erro. Atualmente um dos grandes problemas enfrentados é a exploração de falhas pelos hackers. Devido a isso, muitos programas são atualizados todas as vezes que uma nova falha é descoberta. Porém, nem sempre as falhas são descobertas, o que ocasiona novas vulnerabilidades;
- **Microsoft Incomplete TCP/IP packet:** este ataque explora uma vulnerabilidade. Ele envia pacotes malformados à porta 139 da vítima, fazendo com que os serviços e sistemas dela sejam comprometidos;
- **HP OpenView Node Manager SNMP:** esse ataque é realizado explorando uma vulnerabilidade de uma ferramenta desenvolvida pela HP. Esta ferramenta tem o seu funcionamento comprometido devido a um problema que ocorre no buffer, caso um determinado pedido de GET com tamanho de 136 Bytes for enviado para os serviços web na porta 80 por meio da interface overview 5 CGI. Neste caso o serviço SNMP irá apresentar erros, permitindo a execução arbitrária de códigos por um invasor.

2.4 Governo Eletrônico

Governo Eletrônico ou Administração Pública Eletrônica, se refere ao uso da denominada Tecnologia da Informação e Comunicação (TIC) e Tecnologia da Informação, para informar e divulgar serviços ou produtos do Governo à população (MARTINS; RAMOS, 2008). Seu objetivo é prover informações e serviços às pessoas (OLIVEIRA; ELER, 2015). São utilizadas ferramentas eletrônicas que possuem o objetivo de aproximar os órgãos governamentais e os cidadãos, como, por meio de sites, aplicativos ou telefones de serviços.

De acordo com Chahin et al., (2004),

> O conceito de Governo Eletrônico surgiu logo após o ano de 1993, com o lançamento do primeiro sistema de browser, nos Estados Unidos. Na época o então vice-presidente Al Gore, iniciou o primeiro Fórum Mundial de Reinvenção de Governo, onde se propunha o investimento em novas tecnologias nos sistemas de comunicação dos governos, para acompanhar o surgimento da evolução do ciberespaço (CHAHIN et al., 2004).

Dentre os serviços oferecidos pelo Governo Eletrônico estão:

1. Prestação de Contas, acerca da divulgação de informações e dados referentes às despesas públicas, finanças, orçamentos aprovados, licitações em andamento e contratos no geral.
2. Requisições, onde o cidadão pode requerer um tipo específico de serviço ou se queixar sobre um serviço atrasado (BRASIL, 2020).

3. Espaço para Discussão ou Fóruns, espaço para a população debater, opinar, ou propor novas ideias, tornando as decisões mais transparentes (BRASIL, 2020).
4. Cadastro e Serviço Online, na qual o governo utilizando de *softwares* específicos, disponibiliza serviços online para os cidadãos (e.g. Declaração de Imposto de Renda online) (RODRIGUES, 2019).

Diversas políticas e iniciativas foram realizadas até 2016, mas com a publicação da Estratégia de Governança Digital (EGD), foi implantado um novo paradigma de gestão pública e das relações entre o Estado brasileiro e a sociedade (BRASIL, 2020). Desburocratização, modernização do Estado, simplificação de processos, melhoria no acesso à informação pública, transparência, melhoria nos atendimentos e racionalização de gastos públicos são alguns avanços que a política de governança eletrônica e digital proporcionaram (BRASIL, 2020).

2.5 HTTP e HTTPS

HTTP – *Hypertext Transfer Protocol* é o protocolo de segurança básico que envolve hipertexto da *World Wide Web* (WWW), desenvolvido para ser usado na Internet, mas pode ser utilizado em diferentes aplicações cliente/servidor (SILVA; SILVA, 2009; STALLINGS, 2005). Em uma transmissão HTTP, a confiabilidade é feita com o protocolo *Transmission Control Protocol* (TCP), e cada comunicação é tratada de forma independente (SILVA; SILVA, 2009)

HTTPS – *Hypertext Transfer Protocol Secure* é um protocolo HTTP com o *Secure Sockets Layer* (SSL), tem como objetivo garantir a segurança na transmissão de dados, em aplicações, principalmente na área financeira, entre outros dados sigilosos (SILVA; SILVA, 2009). Mesmo sendo a junção de dois protocolos, para o *browser*, o HTTPS é considerado único, assim, quando for chamar a pagina com segurança é preciso utilizar “https://” e para paginas sem segurança utilizar “http://” (SILVA; SILVA, 2009). Páginas com HTTPS são criptografadas e possuem um cadeado fechado na barra de status, caracterizando a segurança das transferências entre browser e o servidor (SILVA; SILVA, 2009).

3 DESENVOLVIMENTO

3.1 Metodologia

Analisamos se os sistemas web dos órgãos públicos da região da estrada de ferro possuem vulnerabilidades e classificamos seus riscos de acordo com a OWASP.

O método utilizado na execução do trabalho foi dividido em cinco etapas essenciais:

i. Levantamento das principais vulnerabilidades existentes em sistemas web;
ii. Seleção das ferramentas: A seleção das ferramentas teve como principal critério a sua disponibilidade, *softwares open source* devido a transparência do seu funcionamento e por estarem abertos ao público sem um custo de sua utilização;
iii. Seleção dos portais web: baseados no critério de avaliação: portais eletrônicos das prefeituras das cidades da região da estrada de ferro em Goiás;
iv. Varredura dos sites: varredura utilizando as ferramentas selecionadas;
v. Análise dos resultados obtidos: análise realizada buscando traçar um paralelo com a classificação das principais vulnerabilidades feita pela Open Web Application Security Project (OWASP), em português, Projeto Aberto de Segurança em Aplicações Web.

As análises desenvolvidas levaram em conta o uso de ferramentas web scanners, que auxiliaram na obtenção dos resultados. *Web scanners* tem como objetivo verificar as redes, aplicações e dispositivos em busca de pontos vulneráveis a ciberataques e erros, além de reportar as alterações em detalhes, categorizar os riscos e sugerir ações corretivas.

Foram selecionadas duas ferramentas: Skipfish e Nikto. Ambas são gratuitas e oferecem um relatório detalhado do ambiente onde são executadas, são de fácil configuração para a varredura, fácil acesso para a varredura dos dados e busca de possíveis vulnerabilidades, além de já virem instaladas junto ao sistema operacional Kali Linux, passo que ajudou na seleção das duas pelo fato de já termos o Kali Linux instalado. Outro ponto relevante e justificativo na seleção das ferramentas foi o fato das duas terem ganhado destaque em todos os trabalhos correlatos que pesquisamos como em COSTA (2017), OLIVEIRA (2019), FERRAO (2018) e que fizeram o uso de ferramentas *scan* com o mesmo intuito.

3.2 Testes

Para testar essas ferramentas executamos uma vez o escaneamento no portal eletrônico da cidade de Ipameri- GO que foi a primeira cidade utilizada nos testes. As ferramentas foram analisadas inicialmente de acordo com suas funcionalidades, seguindo os seguintes critérios: 1. Relatório: O relatório deve detalhar cada vulnerabilidade, além de indicar o local, ou arquivo no sistema onde a mesma aconteceu; 2. Usabilidade: O quão fácil é utilizar a ferramenta; 3. Eficácia: A quantidade de vulnerabilidades detectadas pela ferramenta, considerando a classificação da OWASP.

Analisamos os relatórios gerados levando em consideração o resultado em si, se conseguiu detectar alguma vulnerabilidade, e a utilização da ferramenta na perspectiva da facilidade de uso, para concluir que as mesmas seriam as utilizadas no trabalho.

Após a análise dos resultados do primeiro escaneamento percebemos que ambas as ferramentas conseguiram atender as expectativas e critérios de avaliação para prosseguir com os demais portais. Como dito anteriormente, as duas ferramentas são disponibilizadas pelo sistema operacional Kali Linux, que foi o sistema utilizado para a execução das análises.

Os sites escolhidos como alvo foram órgãos públicos da região da estrada de ferro. Foi realizada uma verificação para confirmar se há um portal da prefeitura de cada cidade, que esteja disponível na web. A região da estrada de ferro do Estado de Goiás compreende aos municípios: Cumari, Goiandira, Catalão, Ipameri, Urutaí, Pires do Rio, Orizona, Vianópolis, Silvânia, Leopoldo de Bulhões, Anápolis, Bonfinópolis, Senador Canedo, Goiânia. Escolhemos essa região devido ao fato de estarmos incluídos nessa lista, nosso Campus fica em Urutaí, e residimos em Ipameri e Pires do Rio, todas as 3 cidades estão inclusas no nosso campo de verificação. Nosso intuito ao escolher essa região é verificar se os portais das prefeituras de cada cidade estão de fato seguras em relação ao vazamento de dados, tendo em vista que existe a LGPD, e que quando ocorre vazamento de dados pessoais podem acarretar alguns problemas para o órgão em específico.

Ambas as ferramentas apresentaram os relatórios dos resultados e do tempo de execução e a sua eficácia de acordo com a quantidade de vulnerabilidades. O processo de escaneamento no sistema web acontece quando se realiza a requisição no servidor web e são listados os arquivos que fazem parte daquela aplicação web que possam estar comprometidos.

Tanto a ferramenta Skipfish quanto a ferramenta Nikto têm formas parecidas de trabalho, ambas rodam via comandos no terminal Linux, onde é especificado o endereço do portal que será executado o escaneamento, o local onde será salvo o relatório com os resultados, e o nome do relatório. Como diferencial o Skipfish salva seu relatório com uma pasta com arquivos, para executar e consultar todo o relatório, além de classificar os níveis das vulnerabilidades encontradas. Já a ferramenta Nikto, é preciso especificar a porta que será rodado o Scan, como padrão ele sempre roda na porta 80, e como relatório, é gerado um único arquivo HTML, com o nome definido ao executar o comando, ele não traz as vulnerabilidades divididas em níveis de riscos.

Em julho de 2021 fizemos uma pesquisa para identificarmos e selecionarmos quais seriam as melhores ferramentas para executar o escaneamento nos portais web, chegamos ao resultado escolhido depois de efetuar pesquisas em trabalhos correlatos e analisando a documentação

fornecida por cada ferramenta, escolhemos então para seguir com o trabalho as ferramentas Skipfish, e Nikto, o sistema operacional utilizado foi o Kali Linux que tivemos a facilidade das duas ferramentas já virem prontas para uso junto com a instalação do SO.

O Skipfish se baseia nos seguintes testes:

1. Falhas de alto risco (possivelmente levando ao comprometimento do sistema)(***High Impact)***:

- Injeção de consulta do lado do servidor (incluindo vetores cegos, parâmetros numéricos).
- Sintaxe explícita do tipo SQL nos parâmetros GET ou POST.
- Injeção de comando shell do lado do servidor (incluindo vetores cegos).
- Injeção de XML / XPath do lado do servidor (incluindo vetores cegos).
- Vulnerabilidades de string de formato.
- Vulnerabilidades de estouro de número inteiro.
- Locais que aceitam HTTP PUT.

2. Falhas de risco médio (possivelmente levando ao comprometimento de dados)(***Medium***):

- Vetores XSS armazenados e refletidos no corpo do documento (suporte mínimo a JS XSS).
  * Vetores XSS armazenados e refletidos via redirecionamentos HTTP.
- Vetores XSS armazenados e refletidos via divisão de cabeçalho HTTP.
- Percurso de diretório / LFI / RFI (incluindo vetores restritos).
- POIs de arquivos variados (fontes do lado do servidor, configurações, etc).
- Script fornecido pelo invasor e vetores de inclusão CSS (armazenados e refletidos).
- Script externo não confiável e vetores de inclusão CSS.
- Problemas de conteúdo misto em recursos de script e CSS (opcional).
- Formulários de senha enviados de ou para páginas não SSL (opcional).
- Tipos MIME incorretos ou ausentes em renderizados.
- Tipos MIME genéricos em renderizados.
- Charsets incorretos ou ausentes em renderizados.
- Informações conflitantes de MIME / charset em renderizados.
- Diretivas de cache ruins nas respostas de configuração de cookies.

3. Problemas de baixo risco (impacto limitado ou baixa especificidade) (***Low***):

- Vetores de desvio de listagem de diretórios.
- Redirecionamento para URLs fornecidos pelo invasor (armazenados e refletidos).

- Conteúdo incorporado fornecido pelo invasor (armazenado e refletido).
- Conteúdo incorporado externo não confiável.
- Conteúdo misto em sub-recursos não programáveis (opcional).
- HTTPS -> submissão HTTP de formulários HTML (opcional).
- Credenciais HTTP em URLs.
- Certificados SSL expirados ou ainda não válidos.
- Formulários HTML sem proteção XSRF.
- Certificados SSL auto assinados.
- O nome do host do certificado SSL não corresponde.
- Diretivas de cache ruins em conteúdo menos sensível.

4. Avisos internos (***Warn***):

- Falha nas tentativas de busca de recursos.
- Limites de rastreamento excedidos.
- Falha nas verificações de comportamento 404.
- Filtragem IPS detectada.
- Variações de resposta inesperadas.
- Nós de rastreamento aparentemente mal classificados.

5. Entradas informativas não específicas (***Info***):

- Informações gerais do certificado SSL.
- Mudando significativamente os cookies HTTP.
- Alterando os cabeçalhos do Servidor, Via ou X-....
- Novas 404 assinaturas.
- Recursos que não podem ser acessados.
- Recursos que requerem autenticação HTTP.
- Links quebrados.
- Erros do servidor.
- Todos os links externos não classificados de outra forma (opcional).
- Todos os e-mails externos (opcional).
- Todos os redireccionadores de URL externos (opcional).
- Links para protocolos desconhecidos.
- Campos de formulário que não puderam ser preenchidos automaticamente.

- Formulários de entrada de senha (para força bruta externa).
- Formulários de upload de arquivos.
- Outros formulários HTML (não classificados de outra forma).
- Nomes de arquivos numéricos (para força bruta externa).
- Links fornecidos pelo usuário de outra forma renderizados em uma página.
- Tipo MIME incorreto ou ausente em conteúdo menos significativo.
- Tipo MIME genérico em conteúdo menos significativo.
- Conjunto de caracteres incorreto ou ausente em conteúdo menos significativo.
- Informações conflitantes de MIME/charset em conteúdo menos significativo.
- Convenções de passagem de parâmetros do tipo OGNL.

O método de classificação de cada ferramenta influencia na quantidade de vulnerabilidades mostradas em cada tabela. A ferramenta Nikto apresenta resultados de forma mais simples, sendo exibidos apenas a vulnerabilidade seguida do link e dos locais onde a mesma foi encontrada. O Skipfish, além de encontrar o maior número de vulnerabilidades, apresenta os resultados de forma mais detalhada. A ferramenta faz uma classificação de cores sobre a vulnerabilidade, representada por uma esfera seguida do nome. Em outubro foi feita a seleção dos endereços dos portais das cidades em torno da linha de ferro que iremos executar o experimento de escaneamento, selecionamos 14 cidades para utilizar seus portais.

Começamos os escaneamentos também em outubro, começamos utilizando primeiro a ferramenta Skipfish, o primeiro teste foi executado no portal da cidade de Ipameri, Goiás. Após o primeiro escaneamento e a análise do relatório gerado pela ferramenta, pudemos concluir que poderíamos continuar utilizando-a para os demais portais, rodamos ela em todos os 14 links coletados para as demais cidades. Após a execução e registro do relatório de todos os portais escaneados pelo Skipfish, foi feito o escaneamento utilizando nossa segunda ferramenta Nikto, utilizamos o mesmo método, primeiro rodamos na cidade de Ipameri, analisamos o relatório gerado e posteriormente prosseguimos com o escaneamento nas demais cidades.

3.3 Resultados

Foi obtido para a primeira ferramenta um total de 144 horas e 29 minutos ao somar o total de todo tempo de escaneamento sem interrupção na execução. Para a segunda ferramenta foi obtido um total de 6 horas e 47 minutos ao somar o total de todo o tempo de escaneamento

sem interrupção na execução. No total o experimento levou 151 horas e 17 minutos de trabalho ininterrupto.

As tabelas 1 e 2 ilustram os portais seguidos pelos seus endereços e também a data em que foi executado os scanners pelas ferramentas Skipfish e Nikto. Na coluna Tempo Total é apresentado o tempo de execução dos scanners em cada portal.

Tabela 1. Tabela de informações de identificação dos alvos – Skipfish.

Cidade	Endereço	Data	Tempo Total
Anápolis	www.anapolis.go.gov.br	23/11/2021	16:50:03
Bonfinópolis	www.bonfinopolis.go.gov.br	6/12/2021	0:00:11
Catalão	www.catalao.go.gov.br	11/11/2021	12:16:07
Cumari	www.cumari.go.gov.br	11/11/2021	0:00:12
Goiandira	www.goiandira.go.gov.br	11/11/2021	00:00:16
Goiânia	www.goiania.go.gov.br	10/12/2021	82:55:52
Ipameri	www.ipameri.go.gov.br	26/10/2021	7:07:02
Leopoldo de Bulhões	www.leopoldodebulhoes.go.gov.br	20/11/2021	0:00:11
Orizona	www.orizona.go.gov.br	20/11/2021	0:03:08
Pires do Rio	www.piresdorio.go.gov.br	20/11/2021	0:00:49
Senador Canedo	www.senadorcanedo.go.gov.br	6/12/2021	0:00:01
Silvânia	www.silvania.go.gov.br	20/11/2021	0:00:10
Urutaí	www.urutai.go.gov.br	14/11/2021	25:14:52
Vianópolis	www.vianopolis.go.gov.br	20/11/2021	0:01:02

Tabela 2. Tabela de informações de identificação dos alvos – Nikto.

Cidade	Endereço	Data	Tempo Total
Anápolis	www.anapolis.go.gov.br	15/02/2022	3615 Segundos
Bonfinópolis	www.bonfinopolis.go.gov.br	16/02/2022	138 Segundos
Catalão	www.catalao.go.gov.br	14/02/2022	2012 Segundos
Cumari	www.cumari.go.gov.br	14/02/2022	48 Segundos
Goiandira	www.goiandira.go.gov.br	14/02/2022	50 Segundos
Goiânia	www.goiania.go.gov.br	21/02/2022	3731 Segundos
Ipameri	www.ipameri.go.gov.br	17/01/2022	10201 Segundos
Leopoldo de Bulhões	www.leopoldodebulhoes.go.gov.br	15/02/2022	1 Segundo
Orizona	www.orizona.go.gov.br	15/02/2022	909 Segundos
Pires do Rio	www.piresdorio.go.gov.br	15/02/2022	1127 Segundos
Senador Canedo	www.senadorcanedo.go.gov.br	16/02/2022	190 Segundos
Silvânia	www.silvania.go.gov.br	15/02/2022	50 Segundos
Urutaí	www.urutai.go.gov.br	15/02/2022	45 Segundos
Vianópolis	www.vianopolis.go.gov.br	15/02/2022	1187 Segundos

As tabelas 3 e 4 ilustram o resultado apresentado pelas ferramentas Skipfish, e Nikto, os valores zerados significam que os testes não apresentaram resultados ou a execução não pode ser concluída por mecanismos de segurança presentes nos portais, como políticas específicas de firewall ou adoção de sistemas de detecção/prevenção de intrusão.

Ao observar as tabelas 3 e 4, é possível identificar que o scanner Skipfish apresentou um maior número de detecções de vulnerabilidades em relação ao scanner Nikto. Podemos considerar a presença tanto da porta 80 (porta HTTP) como da porta 443 (porta HTTPS). O scanner Skipfish apresenta um nível de risco que ele considera a ameaça encontrada, entre *Info, Warn, Low, Medium, e High Impact*. O scanner Nikto apresenta apenas as solicitações, erros e descobertas. Alguns dos altos valores nas tabelas em especial do Skipfish podem ser associados a repetições, uma mesma vulnerabilidade contabilizada duas ou mais vezes, uma vez que um mesmo tipo de vulnerabilidade pode ser classificado como de alto e médio risco em páginas idênticas ou quase idênticas, por exemplo.

Tabela 3. Tabela de Resultados da ferramenta - Skipfish.

Cidade	Porta	High	Medium	Low	Warn	Info
Anápolis	Porta 80	0	0	0	2	4
	Porta 443	3	452	325	774	980
Bonfinópolis	Porta 80	0	0	0	0	0
	Porta 443	0	0	0	0	0
Catalão	Porta 80	110	162	20	389	388
	Porta 443	3	302	57	16	1270
Cumari	Porta 80	0	0	0	0	5
	Porta 443	0	0	0	1	0
Goiandira	Porta 80	0	0	0	0	6
	Porta 443	0	0	0	1	0
Goiânia	Porta 80	0	6	127	23	208
	Porta 443	0	519	1556	217	1375
Ipameri	Porta 80	0	36	18	78	33
	Porta 443	1	70	30	145	198
Leopoldo de Bulhões	Porta 80	0	0	0	0	0
	Porta 443	0	0	0	0	0
Orizona	Porta 80	0	14	18	81	16
	Porta 443	0	1	2	35	4
Pires do Rio	Porta 80	0	0	0	0	3
	Porta 443	0	0	1	0	7
Senador Canedo	Porta 80	0	0	0	0	3
	Porta 443	0	0	0	0	0
Silvânia	Porta 80	0	0	0	0	5
	Porta 443	0	0	0	0	0
Urutaí	Porta 80	1	31	8	159	2212

	Porta 443	0	20	22	176	46
Vianópolis	Porta 80	0	0	0	0	3
	Porta 443	0	0	1	1	9

Tabela 4. Tabela de Resultados da ferramenta – Nikto.

Cidade	Solicitações	Erros	Descobertas
Anápolis	8.051	0	21
Bonfinópolis	82	19	4
Catalão	8.874	15	14
Cumari	318	20	7
Goiandira	318	20	7
Goiânia	8.055	0	19
Ipameri	9.685	14	24
Leopoldo de Bulhões	0	0	0
Orizona	405	21	5
Pires do Rio	809	19	7
Senador Canedo	314	18	7
Silvânia	318	20	7
Urutaí	318	20	7
Vianópolis	841	20	7

As figuras 1 e 2 representam a interface do relatório de cada uma das ferramentas. A ferramenta Skipfish na figura 1 traz uma classificação de acordo com o nível da ameaça encontrada, e um resumo de onde foi localizada tal ameaça, além da data e tempo de escaneamento. O número de vulnerabilidades de um dado tipo é exibido na frente da categoria do mesmo. A ferramenta Nikto mostrada na figura 2 traz um resumo das descobertas seguido do número total encontrado, e também a data e tempo de escaneamento.

É considerada a seguinte classificação para a ferramenta Skipfish:

- Vermelha: Vulnerabilidade de alto impacto.
- Laranja: Vulnerabilidade de risco médio.
- Azul: Vulnerabilidade de baixo risco.
- Cinza: Aviso.
- Verde: Informação.

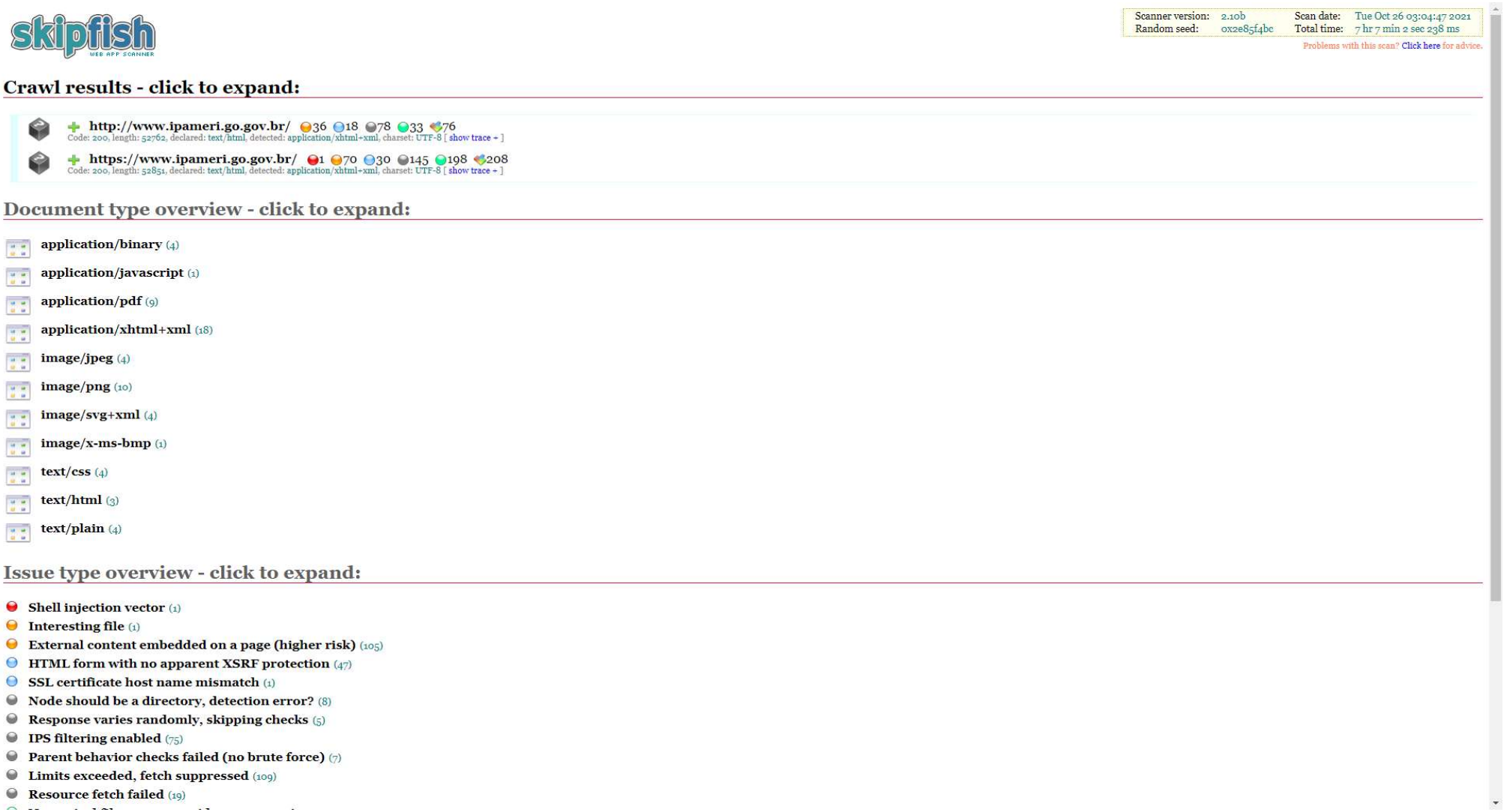
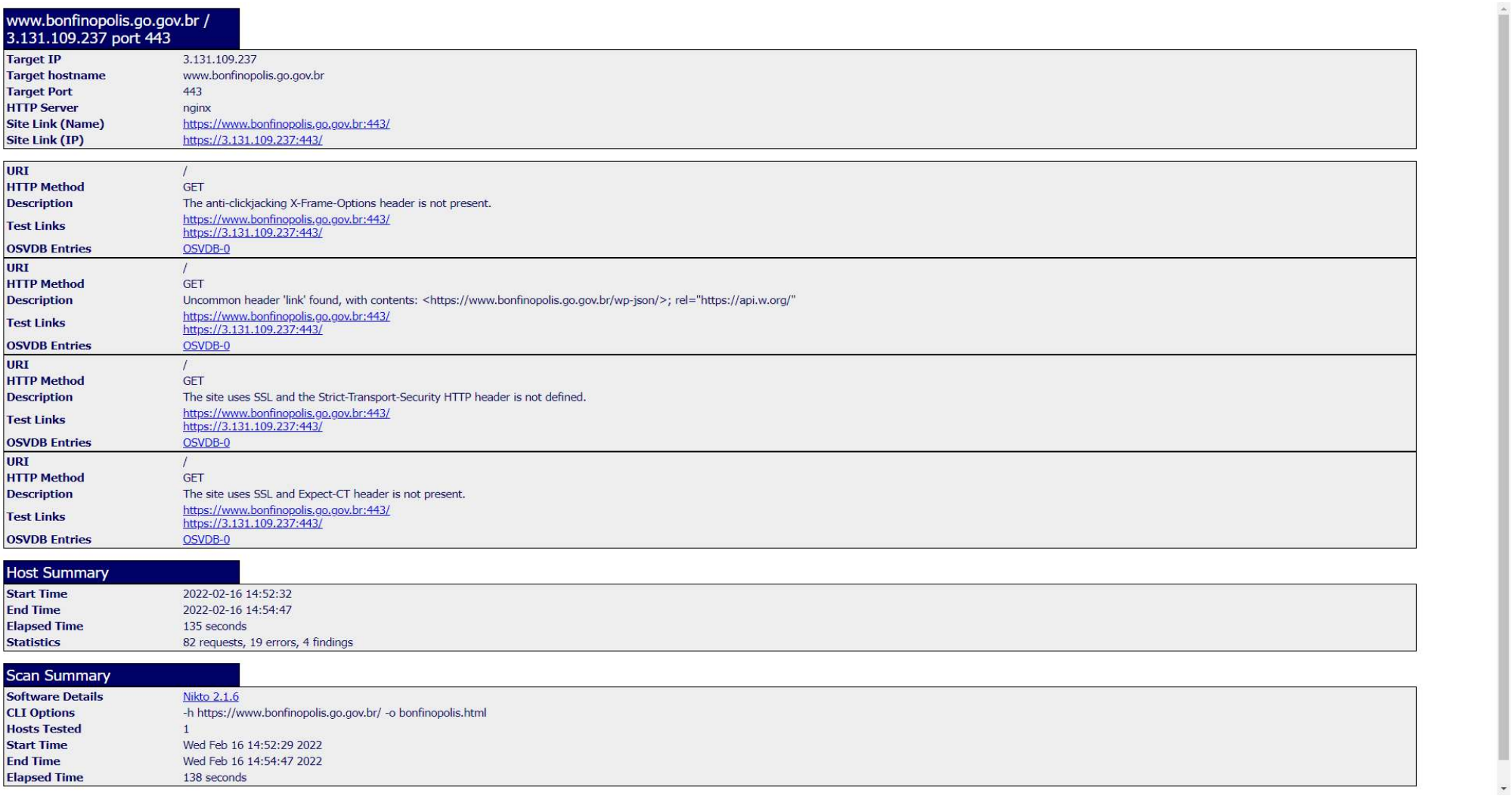


Figura 1. Imagem da ferramenta Skipfish, Imagem produzida pelo autor, de um dos Escaneamentos.



www.bonfinopolis.go.gov.br / 3.131.109.237 port 443

Target IP	3.131.109.237
Target hostname	www.bonfinopolis.go.gov.br
Target Port	443
HTTP Server	nginx
Site Link (Name)	https://www.bonfinopolis.go.gov.br:443/
Site Link (IP)	https://3.131.109.237:443/

URI	/
HTTP Method	GET
Description	The anti-clickjacking X-Frame-Options header is not present.
Test Links	https://www.bonfinopolis.go.gov.br:443/ https://3.131.109.237:443/
OSVDB Entries	OSVDB-0

URI	/
HTTP Method	GET
Description	Uncommon header 'link' found, with contents: <https://www.bonfinopolis.go.gov.br/wp-json/>; rel="https://api.w.org/"
Test Links	https://www.bonfinopolis.go.gov.br:443/ https://3.131.109.237:443/
OSVDB Entries	OSVDB-0

URI	/
HTTP Method	GET
Description	The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
Test Links	https://www.bonfinopolis.go.gov.br:443/ https://3.131.109.237:443/
OSVDB Entries	OSVDB-0

URI	/
HTTP Method	GET
Description	The site uses SSL and Expect-CT header is not present.
Test Links	https://www.bonfinopolis.go.gov.br:443/ https://3.131.109.237:443/
OSVDB Entries	OSVDB-0

Host Summary

Start Time	2022-02-16 14:52:32
End Time	2022-02-16 14:54:47
Elapsed Time	135 seconds
Statistics	82 requests, 19 errors, 4 findings

Scan Summary

Software Details	Nikto 2.1.6
CLI Options	-h https://www.bonfinopolis.go.gov.br/ -o bonfinopolis.html
Hosts Tested	1
Start Time	Wed Feb 16 14:52:29 2022
End Time	Wed Feb 16 14:54:47 2022
Elapsed Time	138 seconds

Figura 2. Imagem da ferramenta Nikto, Imagem produzida pelo autor, de um dos Escaneamentos.

As tabelas 5 e 6 apresentam a distribuição das vulnerabilidades encontradas pelos 2 scanners em relação com a classificação da Owasp, levando em conta uma classificação das 10 vulnerabilidades mais graves recorrentes em sistemas web. Para fazer essa relação, pegamos o nome de identificação da vulnerabilidade por cada ferramenta e pesquisamos entre as classificações fornecidas pelo próprio site da Owasp.

A utilização da sigla "A" seguida do número e ano se refere a cada uma das vulnerabilidades descritas pela Owasp:

- A01:2021 – Controle de acesso quebrado
- A02:2021 – Falhas criptográficas
- A03:2021 – Injeção
- A04:2021 – Design inseguro
- A05:2021 – Configuração incorreta de segurança
- A06:2021 – Componentes Vulneráveis e Desatualizados
- A07:2021 – Falhas de Identificação e Autenticação
- A08:2021 - Falhas de integridade de software e dados
- A09:2021 – Falhas de registro e monitoramento de segurança
- A10:2021 – Falsificação de solicitação do lado do servidor (SSRF)

Tabela 5. Tabela de Vulnerabilidades de acordo com a classificação Owasp, pela ferramenta Skipfish.

Cidade	A01:2021	A02:2021	A03:2021	A04:2021	A05:2021	A06:2021	A07:2021	A08:2021	A09:2021	A10:2021
Anápolis	0	0	0	0	2	0	0	0	0	0
Bonfinópolis	0	0	0	0	0	0	0	0	0	0
Catalão	0	0	113	0	0	0	0	56	0	0
Cumari	0	0	0	0	0	0	0	0	0	0
Goiandira	0	0	0	0	0	0	0	0	0	0
Goiânia	0	0	0	0	0	0	0	0	0	0
Ipameri	0	0	1	0	0	0	1	0	0	0
Leopoldo de Bulhões	0	0	0	0	0	0	0	0	0	0
Orizona	0	0	0	0	0	0	0	0	0	0
Pires do Rio	0	0	0	0	0	0	0	0	0	0
Senador Canedo	0	0	0	0	0	0	0	0	0	0
Silvânia	0	0	0	0	0	0	0	0	0	0
Urutaí	0	0	1	0	1024	0	1	0	0	0
Vianópolis	0	0	0	0	0	0	0	0	0	0

Tabela 6. Tabela de Vulnerabilidades de acordo com a classificação Owasp, pela ferramenta Nikto.

Cidade	A01:2021	A02:2021	A03:2021	A04:2021	A05:2021	A06:2021	A07:2021	A08:2021	A09:2021	A10:2021
Anápolis	0	1	0	0	3	0	0	0	0	0
Bonfinópolis	0	1	0	0	0	0	0	0	0	0
Catalão	1	0	1	0	0	0	1	0	0	0
Cumari	0	0	0	0	0	0	0	0	0	0
Goiandira	0	1	0	0	0	0	0	0	0	0
Goiânia	0	0	0	0	0	0	0	0	0	1
Ipameri	1	2	0	0	1	0	1	0	0	0
Leopoldo de Bulhões	0	0	0	0	0	0	0	0	0	0
Orizona	0	1	0	0	0	0	0	0	0	0
Pires do Rio	0	1	0	0	0	0	1	0	0	1
Senador Canedo	0	1	0	0	0	0	0	0	0	0
Silvânia	0	1	0	0	0	0	0	0	0	0
Urutaí	0	1	0	0	0	0	0	0	0	0
Vianópolis	0	1	0	0	0	0	1	0	0	0

3.4 Discussão dos Resultados

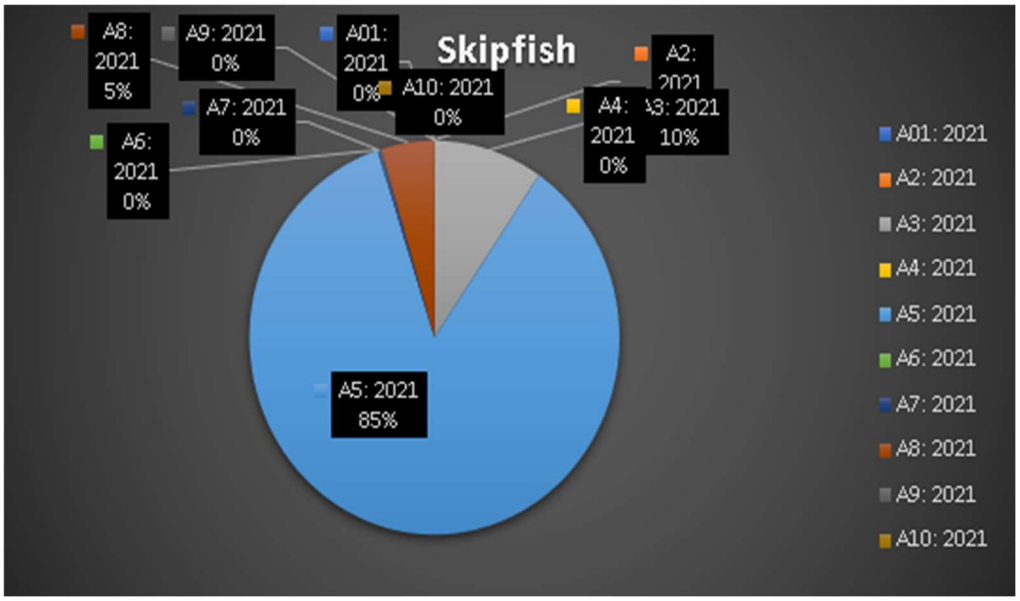


Gráfico 1. Vulnerabilidades encontradas com a ferramenta Skipfish.

O Gráfico 01 contém os resultados obtidos pela ferramenta Skipfish. Nele pode-se observar que a maior parte das vulnerabilidades encontradas se encontra na categoria A5 que é

Security Misconfiguration seguidos pelos demais A3 Injection e A8 Software and data integrity failures.

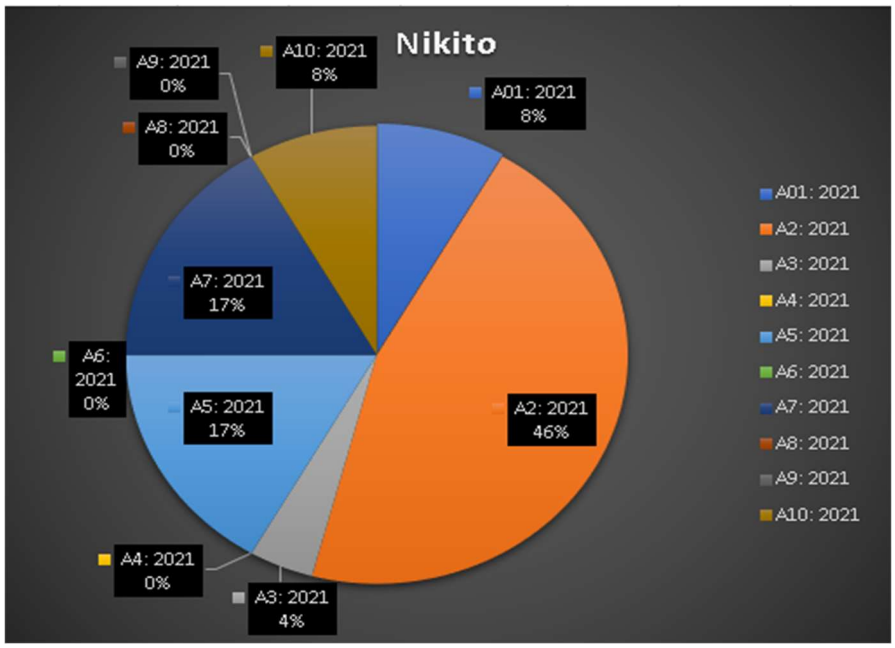


Gráfico 2. Vulnerabilidades encontradas com a ferramenta Nikto.

No gráfico 2 tem como referência os resultados obtidos através da ferramenta Nikto, nele podemos observar que a maior parte das vulnerabilidades encontradas podem ser classificadas na categoria A2 que é Cryptographic failures seguidos pelos demais A7 Identification and Authentication Failures e A8 Software and data integrity failures.

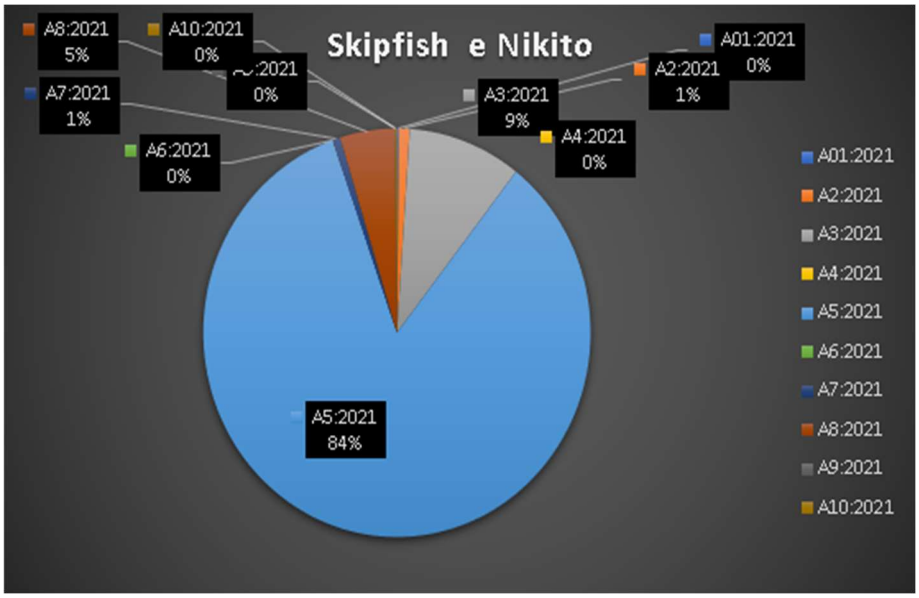


Gráfico 3. Junção das vulnerabilidades encontradas com as ferramentas Skipfish e Nikto.

No gráfico 3 tem como referência os resultados obtidos através das ferramentas Skipfish e Nikto em conjunto, nele podemos observar que a maior parte das vulnerabilidades encontradas se encontra na categoria A5 que é *Security Misconfiguration* seguidos pelos demais *A3 Injection* e *A8 Software and data integrity failures*.

Cidade	A01:2021	A02:2021	A03:2021	A04:2021	A05:2021	A06:2021	A07:2021	A08:2021	A09:2021	A10:2021
Anápolis	0	0	0	0	3	0	0	0	0	0
Bonfinópolis	0	0	0	0	0	0	0	0	0	0
Catalão	0	0	113	0	0	0	0	56	0	0
Cumari	0	0	0	0	0	0	0	0	0	0
Goiandira	0	0	0	0	0	0	0	0	0	0
Goiânia	0	0	0	0	0	0	0	0	0	0
Ipameri	0	0	1	0	1	0	1	0	0	0
Leopoldo de Bulhões	0	0	0	0	0	0	0	0	0	0
Orizona	0	0	0	0	0	0	1	0	0	0
Pires do Rio	0	0	0	0	0	0	0	0	0	0
Senador Canedo	0	0	0	0	0	0	0	0	0	0
Silvânia	0	0	0	0	0	0	0	0	0	0
Urutaí	0	0	1	0	1024	0	1	0	0	0
Vianópolis	0	0	0	0	0	0	1	0	0	0

Gráfico 4. Calor da ferramenta Skipfish.

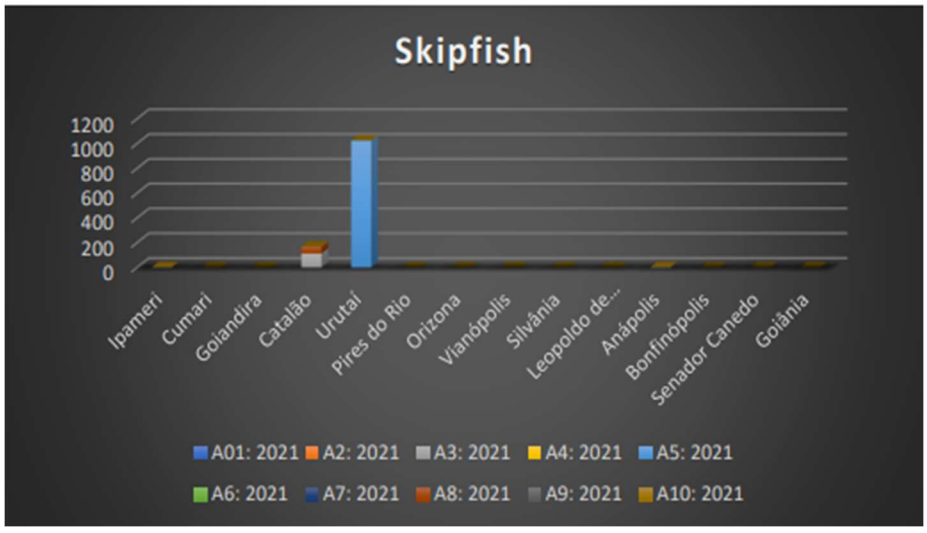


Gráfico 5. Gráfico de barra da ferramenta Skipfish.

No gráfico 4 e 5 da ferramenta Skipfish, podemos observar que a vulnerabilidade que mais aparece é A5:2021 *Security Misconfiguration*. Na cidade de Urutaí é a cidade onde foi encontrado o maior número de vulnerabilidades. Seguido por Catalão na A03:2021 *Injection*.

Cidade	A01:2021	A02:2021	A03:2021	A04:2021	A05:2021	A06:2021	A07:2021	A08:2021	A09:2021	A10:2021
Anápolis	0	1	0	0	3	0	0	0	0	0
Bonfinópolis	0	1	0	0	0	0	0	0	0	0
Catalão	1	0	1	0	0	0	1	0	0	0
Cumari	0	0	0	0	0	0	0	0	0	0
Goiandira	0	1	0	0	0	0	0	0	0	0
Goiânia	0	0	0	0	0	0	0	0	0	1
Ipameri	1	2	0	0	1	0	1	0	0	0
Leopoldo de Bulhões	0	0	0	0	0	0	0	0	0	0
Orizona	0	1	0	0	0	0	0	0	0	0
Pires do Rio	0	1	0	0	0	0	1	0	0	1
Senador Canedo	0	1	0	0	0	0	0	0	0	0
Silvânia	0	1	0	0	0	0	0	0	0	0
Urutaí	0	1	0	0	0	0	0	0	0	0
Vianópolis	0	1	0	0	0	0	1	0	0	0

Gráfico 6. Calor da ferramenta Nikto.

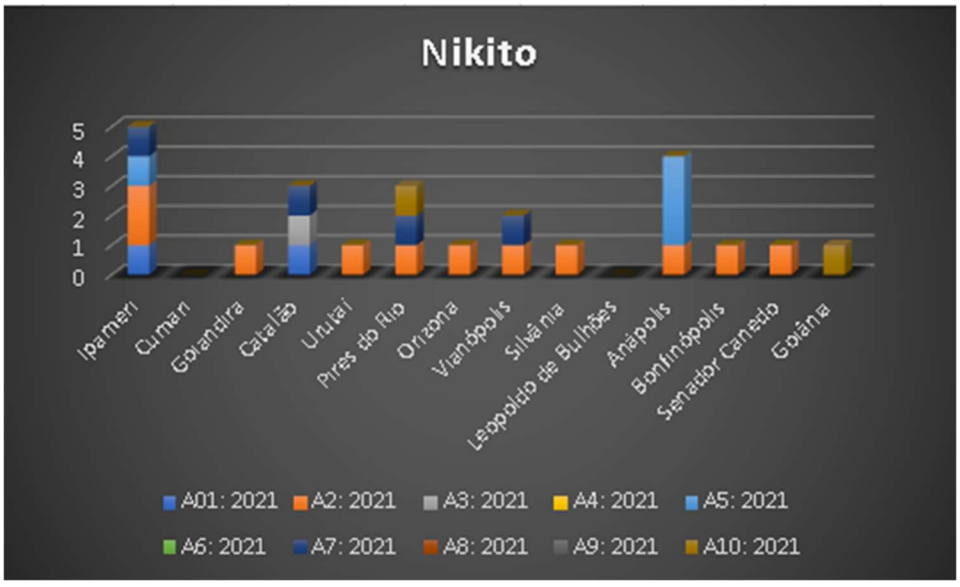


Gráfico 7. Gráfico de barra da ferramenta Nikto.

No gráfico 6 e 7 da ferramenta Nikto ficou bem distribuído as vulnerabilidades com um pequeno foco no A5:2021 Security Misconfiguration, na cidade de Anápolis. Outras cidades também ficaram próximas, como: Ipameri, Catalão e Urutaí com suas respectivas vulnerabilidades mostradas no gráfico 6.

Cidade	A01:2021	A02:2021	A03:2021	A04:2021	A05:2021	A06:2021	A07:2021	A08:2021	A09:2021	A10:2021
Anápolis	0	1	0	0	3	0	0	0	0	0
Bonfinópolis	0	1	0	0	0	0	0	0	0	0
Catalão	1	0	114	0	0	0	1	56	0	0
Cumari	0	0	0	0	0	0	0	0	0	0
Goiandira	0	1	0	0	0	0	0	0	0	0
Goiânia	0	0	0	0	0	0	0	0	0	1
Ipameri	1	2	1	0	1	0	2	0	0	0
Leopoldo de Bulhões	0	0	0	0	0	0	0	0	0	0
Orizona	0	1	0	0	0	0	1	0	0	0
Pires do Rio	0	1	0	0	0	0	1	0	0	1
Senador Canedo	0	1	0	0	0	0	0	0	0	0
Silvânia	0	1	0	0	0	0	0	0	0	0
Urutaí	0	1	1	0	1024	0	1	0	0	0
Vianópolis	0	1	0	0	0	0	2	0	0	0

Gráfico 8. Calor da junção das ferramentas Skipfish e Nikto.

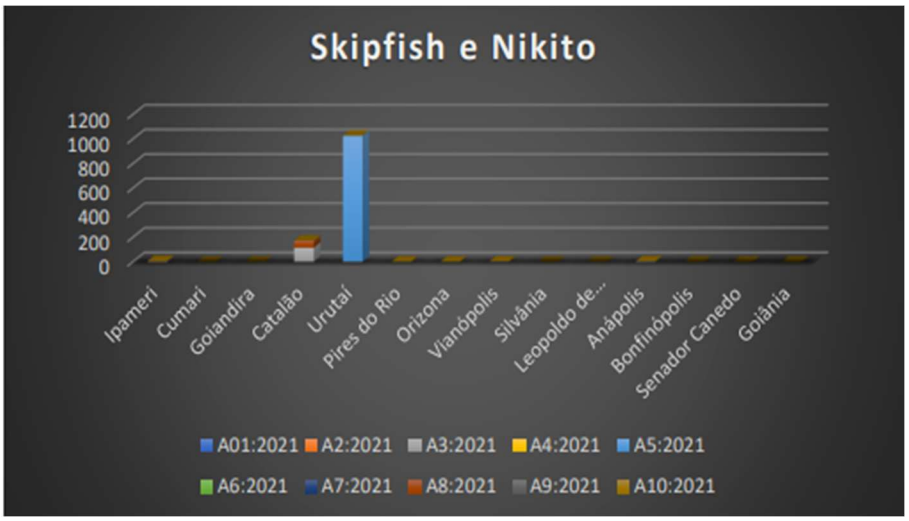


Gráfico 9. Gráfico de barra da junção das ferramentas Skipfish e Nikto.

Nos Gráficos 08 e 09 estão os resultados agrupados das duas ferramentas, Skipfish e Nikto, onde é possível observar que a maior vulnerabilidade é a A5:2021 *Security Misconfiguration*, com foco na cidade de Urutaí como a mais vulnerável, seguido por Catalão com vulnerabilidade A3:2021 *injection* e A8:2021 *Software and Data Integrity Failures*.

Tabela 7. Tabela de vulnerabilidades detectadas em ordem decrescente de acordo com as ferramentas Nikto e Skipfish.

Cidades	Vulnerabilidades
Urutaí	1027
Catalão	172
Ipameri	7
Anápolis	4
Pires do Rio	3
Vianópolis	3
Orizona	2
Goiandira	1
Silvânia	1
Bonfinópolis	1
Senador Canedo	1
Goiânia	1
Cumari	0
Leopoldo de Bulhões	0

Gráfico 10. Gráfico de barra de vulnerabilidades detectadas em ordem decrescente de acordo com as ferramentas Nikto e Skipfish.

A tabela 7 e o gráfico 10 nos mostra em ordem decrescente as cidades mais vulneráveis de acordo com as ferramentas Nikto e Skipfish. Começando com a Cidade de Urutaí, seguida por Catalão e Ipameri.

As duas ferramentas detectaram mais vulnerabilidades do que aqueles presentes na tabela da Owasp, ou vulnerabilidades que não estão presentes nelas, mas as mesmas não foram contabilizadas pois o foco do trabalho é usar a classificação proposta pela Owasp. As tabelas 8 e 9 ilustram as vulnerabilidades detectadas pelas ferramentas Skipfish e Nikto, mas que não estão presentes na Owasp.

Tabela 8. Tabela de Vulnerabilidades detectadas pela ferramenta Skipfish fora da classificação Owasp.

Cidade	Vulnerabilidade
Anápolis	*Query injection vector, Server-side XML injection vector, External content embedded on a page (higher risk), XSS vector in document body, HTML form with no apparent XSRF protection, External content embedded on a page (lower risk), Node should be a directory, detection error?, Response varies randomly, skipping checks, IPS filtering enabled, Parent behavior checks failed (no brute force), Limits exceeded, fetch suppressed, Numerical filename - consider enumerating, Incorrect or missing charset (low risk), Password entry form - consider brute-force, Unknown form field (can't autocomplete), Hidden files / directories, Server error triggered, Resource not directly accessible, New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen, New HTTP cookie added, SSL certificate issuer information.*
Bonfinópolis	*0*
Catalão	*Query injection vector, Shell injection vector, Directory traversal / file inclusion possible, Interesting file, Incorrect or missing charset (higher risk), External content embedded on a page (higher risk), XSS vector in document body.*
Cumari	*Incorrect or missing charset (low risk), New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen.*
Goiandira	*Incorrect or missing charset (low risk), New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen, New HTTP cookie added.*
Goiânia	*Interesting file, External content embedded on a page (higher risk), Signature match detected, HTML form with no apparent XSRF protection, External content embedded on a page (lower risk), Response varies randomly, skipping checks, Parent behavior checks failed (no brute force), Limits exceeded, fetch suppressed, Numerical filename - consider enumerating, Incorrect or missing charset (low risk), Incorrect or missing MIME type (low risk), Password entry form - consider brute-force, HTML form (not classified otherwise), Unknown form field (can't autocomplete), Hidden files / directories, Server error triggered, HTTP authentication required, Resource not directly accessible, New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen, New HTTP cookie added, SSL certificate issuer information.*
Ipameri	*Interesting file, External content embedded on a page, HTML form with no apparent XSRF protection, Node should be a directory, detection error?, Response varies randomly, skipping checks, IPS filtering enabled, Parent behavior checks failed (no*

	brute force), Limits exceeded, fetch suppressed, Numerical filename - consider enumerating, Incorrect or missing charset (low risk), Incorrect or missing MIME type (low risk), HTML form (not classified otherwise), Unknown form field (can't autocomplete), Hidden files / directories, Server error triggered, Resource not directly accessible, New 404 signature seen, New 'X-' header value seen, New 'Server' header value seen, New HTTP cookie added, SSL certificate issuer information.*
Leopoldo de Bulhões	*0*
Orizona	*External content embedded on a page (higher risk), XSS vector in document body, HTML form with no apparent XSRF protection, External content embedded on a page (lower risk), Node should be a directory, detection error?, Limits exceeded, fetch suppressed, Numerical filename - consider enumerating, Password entry form - consider brute-force, Unknown form field (can't autocomplete), New 404 signature seen, New 'Server' header value seen, New HTTP cookie added, SSL certificate issuer information.*
Pires do Rio	*SSL certificate host name mismatch, Incorrect or missing charset (low risk), New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen, SSL certificate issuer information.*
Senador Canedo	*Incorrect or missing charset (low risk), New 404 signature seen, New 'Server' header value seen.*
Silvânia	*Incorrect or missing charset (low risk), New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen.*
Urutaí	*Interesting server message, Interesting file, External content embedded on a page (higher risk), Signature match detected, HTML form with no apparent XSRF protection, External content embedded on a page (lower risk), Self-signed SSL certificate, HTTP credentials seen in URLs, Node should be a directory, detection error?, IPS filtering enabled, Limits exceeded, fetch suppressed, Numerical filename - consider enumerating, Incorrect or missing charset (low risk), Generic MIME used (low risk), Incorrect or missing MIME type (low risk), HTML form (not classified otherwise), Unknown form field (can't autocomplete), Hidden files / directories, Resource not directly accessible, New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen.*
Vianópolis	*Parent behavior checks failed (no brute force), Incorrect or missing charset (low risk), Hidden files / directories, New 404 signature seen, New 'X-*' header value seen, New 'Server' header value seen, SSL certificate issuer information.*

Tabela 9. Tabela de Vulnerabilidades detectadas pela ferramenta Nikto fora da classificação Owasp.

Cidade	**Vulnerabilidade**
Anápolis	*The anti-clickjacking X-Frame-Options header is not present. Uncommon header 'link' found, with contents. Uncommon header 'x-redirect-by' found, with contents: WordPress. Entry '/wp-admin/' in robots.txt returned a non-forbidden or redirect HTTP code (302). "robots.txt" contém 2 entradas que devem ser visualizadas manualmente. Uncommon header 'x-fastcgi-cache' found, with contents: BYPASS. The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack. This might be interesting. Server may leak inodes via ETags, header found with file /transito/, inode: 3440791, size: 3002, mtime: Tue Feb 8 17:37:38 2022. Coccoon from Apache-XML project reveals file system path*

	in error messages. This WordPress script reveals the installed version. License file found may identify site software. A Wordpress installation was found. Cookie wordpress_test_cookie created without the httponly flag.
Bonfinópolis	*The anti-clickjacking X-Frame-Options header is not present. Uncommon header 'link' found, with contents. The site uses SSL and Expect-CT header is not present.*
Catalão	*The anti-clickjacking X-Frame-Options header is not present. The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. The site uses SSL and the Strict-Transport-Security HTTP header is not defined. Securecontrolpanel: Web Server Control Panel. Webmail: Web based mail package installed. Cpanel: Web-based control panel. Img-sys: Default image directory should not allow directory listing. Webmail/lib/emailreader_execute_on_each_page.inc.php: This might be interesting... has been seen in web logs from an unknown scanner. /controlpanel/: Admin login page/section found.*
Cumari	*The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. Ncommon header 'report-to' found, with contents. Uncommon header 'alt-svc' found, with contents: h3=":443"; ma=86400, h3-29=":443"; ma=86400. Uncommon header 'nel' found, with contents: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}. The site uses SSL and the Strict-Transport-Security HTTP header is not defined. Expect-CT is not enforced, upon receiving an invalid Certificate Transparency Log, the connection will not be dropped. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.*
Goiandira	*The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. Uncommon header 'alt-svc' found, with contents: h3=":443"; ma=86400, h3-29=":443"; ma=86400. Uncommon header 'report-to' found, with contents: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?s=KZ%2FmE16YluwzzNYzZWqwGDaE2EAuI0iRvV0i5fkrOJC%2BqQerLz8TW4mobCuqO5JOQpmLMOdcK68Su0CKjsX11j7wDr%2B76kPwgXwZ42CTjH7qVqfZaEfWJcGKAMdYXSdPOP0Bc6WdY9oOWQ%3D%3D"}],"group":"cf-nel","max_age":604800}. Uncommon header 'nel' found, with contents: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}. Expect-CT is not enforced, upon receiving an invalid Certificate Transparency Log, the connection will not be dropped. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.*
Goiânia	*Uncommon header 'x-fastcgi-cache' found, with contents: HIT. Uncommon header 'x-wp-cache' found, with contents: HIT. Uncommon header 'link' found, with contents. The site uses SSL and Expect-CT header is not present. Uncommon header 'x-redirect-by' found, with contents: WordPress. "robots.txt" contains 2 entries which should be manually viewed. The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack. Server is using a wildcard certificate: *.goiania.go.gov.br. Retrieved x-powered-by header: ASP.NET. Uncommon header 'x-elasticpress-query' found, with contents: true. This might be interesting. This might be interesting... potential country code (Ghana). This WordPress script reveals the installed version. License file found may identify site software. A Wordpress installation was found. wordpress_test_cookie created without the httponly flag. CardDAV file may contain server info, per RFC-5785. See http://tools.ietf.org/html/rfc6764.*

Ipameri	*The anti-clickjacking X-Frame-Options header is not present. The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. The site uses SSL and Expect-CT header is not present. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack. Vários arquivos de índice encontrados. IlohaMail 0.8.10 contains an XSS vulnerability. Previous versions contain other non-descript vulnerabilities. Web Server Control Panel. Web based mail package installed. Mailman was found on the server. Web-based control panel. This might be interesting. Isso pode ser interessante... foi visto em logs da web de um scanner desconhecido. Arquivo PHP padrão do Monkey Http Daemon encontrado. Admin login page/section found. X-XSS-Protection header has been set to disable XSS Protection. There is unlikely to be a good reason for this.*
Leopoldo de Bulhões	*0*
Orizona	*The anti-clickjacking X-Frame-Options header is not present. The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. The site uses SSL and the Strict-Transport-Security HTTP header is not defined. The site uses SSL and Expect-CT header is not present.*
Pires do Rio	*The anti-clickjacking X-Frame-Options header is not present. Uncommon header 'x-redirect-by' found, with contents: WordPress. The site uses SSL and Expect-CT header is not present. "robots.txt" contains 3 entries which should be manually viewed.*
Senador Canedo	*The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. Uncommon header 'alt-svc' found, with contents: h3=":443"; ma=86400, h3-29=":443"; ma=86400. Uncommon header 'report-to' found, with contents: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?s=wx%2FDGChz3ZJj7lLg5qvDC2BIA692W9oT8Se2Zdk8wBPkrIPv7bEiGabWonNLDg1mLXSOyuqQXav7REkfm%2FeUdjp9t96%2Fjt14sV3T1F32uHEMkd2uuKgti3WdAfyMgLkDYwt2WrnIr3GIow7yEIQ%3D"}],"group":"cf-nel","max_age":604800}. Uncommon header 'nel' found, with contents: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}. Expect-CT is not enforced, upon receiving an invalid Certificate Transparency Log, the connection will not be dropped. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.*
Silvânia	*The anti-clickjacking X-Frame-Options header is not present. The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. The site uses SSL and the Strict-Transport-Security HTTP header is not defined. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.*
Urutaí	*The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS. Uncommon header 'alt-svc' found, with contents: h3=":443"; ma=86400, h3-29=":443"; ma=86400. Uncommon header 'nel' found, with contents: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}. Uncommon header 'report-to' found, with contents: {"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?s=SQSJQTDviv6ZN6GmSHDBEqlqbZZA9UIC72bN78G%2F%2BSkPCtb4kKjvftuCQa99LDCOLRPF9NlM7wavh9WvfdMw7GLOtRyYWAoZayqJDnPy%2BpAk%2Fw9VFyd4Mi1h5zddacmUZ9YYzr%2F9Ug%3D%3D"}],"group":"cf-nel","max_age":604800}. Expect-CT*

	is not enforced, upon receiving an invalid Certificate Transparency Log, the connection will not be dropped. The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
Vianópolis	*Retrieved access-control-allow-origin header. The anti-clickjacking X-Frame-Options header is not present. Uncommon header 'x-redirect-by' found, with contents: WordPress. The site uses SSL and Expect-CT header is not present. Uncommon header 'link' found, with contents.*

3.5 Estratégias de Prevenção e Mitigação dos Problemas

Vamos abordar aqui as principais vulnerabilidades encontradas, falar um pouco sobre cada uma para tirar as dúvidas sobre o que é cada vulnerabilidade em destaque de acordo com a OWASP e também maneiras de prevenções para que se atentem aos riscos de deixar aplicações webs vulneráveis.

A01:2021 – Broken Access Control - O controle de acesso impõe a política de forma que os usuários não possam agir fora de suas permissões pretendidas. As falhas geralmente levam à divulgação, modificação ou destruição de informações não autorizadas de todos os dados ou à execução de uma função comercial fora dos limites do usuário.

Foram encontradas 2 vulnerabilidades através da ferramenta Nikto.

Prevenções: O invasor não pode modificar a verificação ou os metadados do controle de acesso. Implemente mecanismos de controle de acesso uma vez e reutilize-os em todo o aplicativo, inclusive minimizando o uso do Cross-Origin Resource Sharing (CORS). Desative a listagem de diretórios do servidor web e certifique-se de que os metadados do arquivo (por exemplo, .git) e os arquivos de backup não estejam presentes nas raízes da web. Registre falhas de controle de acesso, alerte os administradores quando apropriado (por exemplo, falhas repetidas), OWASP (2021).

A02:2021 – Cryptographic Failures - Determinar as necessidades de proteção dos dados em trânsito e em repouso. Por exemplo, senhas, números de cartão de crédito, registros de saúde, informações pessoais e segredos comerciais exigem proteção extra, principalmente se esses dados estiverem sob leis de privacidade.

Foram detectadas 11 vulnerabilidades através da ferramenta Nikito.

Prevenções: Classifique os dados processados, armazenados ou transmitidos por um aplicativo. Identifique quais dados são confidenciais de acordo com as leis de privacidade, requisitos regulatórios ou necessidades de negócios, não armazene dados confidenciais desnecessariamente. Descarte-o o mais rápido possível ou use tokenização compatível com PCI

DSS, certifique-se de criptografar todos os dados confidenciais. Garantir que algoritmos, protocolos e chaves padrão atualizados e fortes estejam em vigor; use o gerenciamento de chaves adequado, Desabilite o armazenamento em cache para respostas que contenham dados confidenciais, Aplique os controles de segurança necessários de acordo com a classificação dos dados. OWASP (2021).

A03:2021 – Injection.

Através da ferramenta Skipfish foram descobertas 115 vulnerabilidades nos sites das 14 cidades que foram escaneadas e apenas uma detecção através da ferramenta Nikito dentre as mesmas.

Prevenções: Manter os dados separados de comandos e consultas, usar uma API segura, que evite totalmente o uso do interpretador, forneça uma interface parametrizada ou migre para Object Relational Mapping Tools (ORMs). Usar validação de entrada positiva do lado do servidor. OWASP (2021).

A05:2021 – Security Misconfiguration - Falta de proteção de segurança apropriada em qualquer parte da pilha de aplicativos ou permissões configuradas incorretamente em serviços de nuvem. Recursos desnecessários são ativados ou instalados (por exemplo, portas, serviços, páginas, contas ou privilégios desnecessários).

Foram detectadas 1028 vulnerabilidades através da ferramenta Skipfish e 4 na ferramenta Nikto.

Prevenções: Remova ou não instale recursos e estruturas não utilizados, uma arquitetura de aplicativo segmentada fornece separação eficaz e segura entre componentes ou locatários, com segmentação, envio de diretivas de segurança para clientes, OWASP (2021).

A07:2021 – Falhas de Identificação e Autenticação - A confirmação da identidade do usuário, autenticação e gerenciamento de sessão é fundamental para proteger contra ataques relacionados à autenticação.

Dentro dessa categoria de vulnerabilidade, a ferramenta Nikto e Skipfish detectaram 4 vulnerabilidades.

Prevenções: implemente a autenticação multifator para evitar ataques automatizados de preenchimento de credenciais, força bruta e reutilização de credenciais roubadas, não envie ou implante com credenciais padrão, principalmente para usuários administradores, implemente verificações de senhas fracas, como testar senhas novas ou alteradas na lista das 10.000 piores senhas. Registre todas as falhas e alerte os administradores quando forem detectados preenchimento de credenciais, força bruta ou outros ataques, use um gerenciador de sessão

integrado, seguro e do lado do servidor que gera um novo ID de sessão aleatório com alta entropia após o login, OWASP (2021).

A08:2021 - Falhas de integridade de software e dados - As falhas de integridade de software e dados estão relacionadas a código e infraestrutura que não protegem contra violações de integridade. Muitos aplicativos agora incluem a funcionalidade de atualização automática, em que as atualizações são baixadas sem verificação de integridade suficiente e aplicadas ao aplicativo anteriormente confiável. Os invasores podem fazer upload de suas próprias atualizações para serem distribuídas e executadas em todas as instalações.

A ferramenta Skipfish detectou 56 vulnerabilidades enquanto a ferramenta Nikto não encontrou nenhuma.

Prevenções Use assinaturas digitais ou mecanismos semelhantes para verificar se o software ou os dados são da fonte esperada e não foram alterados, garanta que bibliotecas e dependências, como npm ou Maven, estejam consumindo repositórios confiáveis. OWASP (2021).

A10:2021 – Falsificação de solicitação do lado do servidor (SSRF) - Ocorrem sempre que um aplicativo da Web está buscando um recurso remoto sem validar a URL fornecida pelo usuário. Permite que um invasor force o aplicativo a enviar uma solicitação criada para um destino inesperado, mesmo quando protegido por um firewall, VPN ou outro tipo de lista de controle de acesso à rede (ACL).

Foram detectadas 2 vulnerabilidades através da ferramenta Nikto.

Prevenções Aplique políticas de firewall “negar por padrão” ou regras de controle de acesso à rede para bloquear todo o tráfego de intranet, exceto o essencial, higienize e valide todos os dados de entrada fornecidos pelo cliente, aplique o esquema de URL, a porta e o destino com uma lista de permissões positiva, não envie respostas brutas aos clientes, desabilitar redirecionamentos HTTP, esteja ciente da consistência do URL para evitar ataques como religação de DNS, OWASP (2021).

Como observamos ao longo do trabalho desenvolvido a vulnerabilidade que se tem mais destaque é a A05:2021 *Security Misconfiguration*. O impacto causado por ignorar essa vulnerabilidade pode ser catastrófico. Podem haver perda de dados parcial, modificação de dados, comprometimento do sistema por completo e ter um alto custo para recuperação dos dados.

4 CONCLUSÃO

Neste trabalho realizamos testes de vulnerabilidades em sites da região da estrada de ferro, com isso usamos duas ferramentas para auxiliar: Nikto e Skipfish. É importante lembrar que o objetivo não é a comparação das ferramentas, e sim analisar a segurança nos sistemas web.

O scanner Skipfish apresentou um maior número de detecções de vulnerabilidades em relação ao scanner Nikto, que apresenta os seus resultados de forma mais simples, sendo exibidos apenas a vulnerabilidade seguida do link e dos locais onde a mesma foi encontrada. Por meio da ferramenta Nikto foi possível descobrir vulnerabilidades logo na primeira cidade escaneada, mas os resultados não mais simples e minimalistas em relação à outra ferramenta. Já a plataforma Skipfish faz uma classificação de cores sobre a vulnerabilidade e com mais detalhes em relação à ferramenta Nikto.

Com o estudo das ferramentas e as vulnerabilidades encontradas concluímos que é verdadeira a hipótese de que os sites da região da estrada de ferro não estão totalmente seguros e propícios a ataques. Os dados coletados durante a pesquisa mostram claramente quais as cidades mais vulneráveis e que precisam de atenção para mitigar esses problemas.

A intenção do estudo e dos testes foi demonstrar algumas ferramentas para teste de segurança da informação e abordar os temas principais referentes à segurança da informação e como evitar possíveis ataques ou fraudes em sistemas web. Com isso observamos que a maioria dos sites das regiões da estrada de ferro possui algum tipo de vulnerabilidade que podem ser prejudiciais. Sendo assim a utilização das ferramentas é de suma importância como vimos no decorrer do trabalho, com testes profundos para possíveis falhas de segurança. Com isso para trabalho futuro poderia detalhar mais as vulnerabilidades encontradas, procurar melhores ferramentas com intuito de fazer uma verificação total para poder minimizar essas vulnerabilidades, evitando assim problemas futuros como perda de dados, roubo, danos à aplicação web, transtornos financeiros para a empresa. Já que essas informações são dados sigilosos e indivíduos não autorizados não podem acessar. E também relacionar toda nossa pesquisa com a LGPD já que todo controle e segurança da informação é dada a partir dela e visto que quando ocorre um vazamento de dados pode ocorrer punições aos órgãos responsáveis.

5 REFERÊNCIAS

BRASIL. **Crimes cibernéticos** / 2ª Câmara de Coordenação e Revisão, Criminal. – Brasília: MPF, 2018.

BRASIL. **Governo Eletrônico**. Linha do tempo - Governo Eletrônico. Ministério da Economia. 2020.

CARDOSO, O. V. **Um Ano da Lei Geral de Proteção de Dados Pessoais.** Jusbrasil, 2021. Disponível em: <https://ovcardoso.jusbrasil.com.br/artigos/1282908252/um-ano-da-lei-geral-de-protecao-de-dados-pessoais>. Acesso em: 20 set. 2021.

CHAHIN, A. et al. **E-gov.br:** a próxima revolução brasileira. São Paulo: Prentice Hall, p.380, 2004.

IBM Community. **If you're looking for a developerWorks forum** — Don't panic! Disponivel em: <https://community.ibm.com/community/user/legacy?lang=en> Acesso em: 13 jun. 2022.

KIM, D; SOLOMON, M. G. **Fundamentos de Segurança de Sistemas de Informação**. Rio de Janeiro: LTC, 2014.

LGPD - **Lei Geral da Proteção de Dados Pessoais.** SEBRAE, 2018. Disponível em: <https://www.sebrae.com.br/sites/PortalSebrae/canais_adicionais/conheca_lgpd>. Acesso em: 20 ago. 2021.

MACEDO, M. **Introdução à Segurança da Informação Corporativa**. Secretaria de Administração. Governo do estado de Pernambuco, 2021.

MARTINS, D. A.; RAMOS, A. S. M. **Conceitos de Governo Eletrônico e Governança Eletrônica**: Confrontação e Complementaridade. EnAPG - Encontro de Administração Pública e Governança. Salvador, BA, 2008.

MASCARENHAS NETO, et al. **Segurança da Informação:** Uma visão sistêmica para implantação em organizações. Editora da UFPB, 2019.

NAKAMURA, E. T.; GEUS, P. L. de. **Segurança de redes em ambientes cooperativos**. 1. ed. São Paulo: Novatec, 2007.

OLIVEIRA, A. D. A.; ELER, M. M. **Acessibilidade em Governo Eletrônico**: um estudo sobre a aplicação de padrões web em sítios gov.br. Universidade de São Paulo, 2015.

OLIVEIRA, W. **Segurança da informação**: técnicas e soluções. São Paulo: Atlas, 2001.
O que é scanner de vulnerabilidade, importância e como integrar? . GAT-InfoSec, São Paulo, 2022.

OWASP Foundation. **As 10 vulnerabilidades de segurança mais críticas em aplicações WEB**. OWASP TOP 10. 2007.

OWASP, P. et al. OWASP Top 10 - 2013: **Os dez riscos de segurança mais críticos em aplicações web**. OWASP Top 10, p. 23, 2013.

RESENDE, J.; BARBOSA, R. **Segurança de rede implementando técnicas de segurança.** Monografia (Especialização em Redes de Computadores). Universidade Federal de Juiz de Fora, Juiz de Fora, 2013.

RODRIGUES, K. S. B. **Gestão da informação pública: uma análise dos portais de transparência pública da região sudeste**. Trabalho de Conclusão (Bacharel em Ciência e Economia) Universidade Federal de Alfenas. Varginha/MG, 2019.

SÊMOLA, M. **Gestão da segurança da informação:** uma visão executiva. 2 ed. Rio de Janeiro: Elsevier, 2014.

SILVA, A. L. S.; SILVA, R. C. M. F. Protocolo HTTP x Protocolo HTTPS. **Núcleo**, v.6, n.1, abr. 2009.

SOARES, R. **Auditoria Teste de Invasão(Pentest)** – Planejamento, Preparação e Execução. Blog SegInfo. Disponível em: <http://www.seginfo.com.br/auditoria-teste-de-invasaopentest-planejamentopreparacao-e-execucao/>. Acesso em: 13 jun. 2022.

STALLINGS, W. **Redes e sistemas de comunicação de dados**: teorias e aplicações corporativas. 5 ed. Rio de Janeiro: Elsevier, 2005.

STEFANINI, R**. Infra News Telecom**. Disponível em: https://infranewstelecom.com.br/principais-preocupacoes-de-seguranca-da-informacao/>. Acesso em: 25 ago. 2021.

SUMMERS, R. C. **Secure computing**: threats and safeguards. New York: McGraw-Hill, 1997.

VIDAL, H. H. **Por que se preocupar com a segurança da informação?** NET SABER - ARTIGOS. Disponível em: <http://artigos.netsaber.com.br/resumo_artigo_24860/artigo_sobre_por-que-se-preocupar-com-a-seguranca-da-informacao-> Acesso em: 13 jun. 2022.