UM ESTUDO APLICADO A SEGURANÇA DE APLICAÇÕES WEB

A STUDY APPLIED TO WEB APPLICATION SECURITY

Felipe Delboni ORTEGA
fdelb.ortega@gmail.com
Aluno do Curso de Bacharelado em Ciência da Computação
Centro Universitário Padre Anchieta - Jundiaí, SP

Carlos Eduardo CÂMARA
dinhocamara@gmail.com
Pesquisador na Área de Ciência da Computação
Doutorado e Mestrado em Engenharia Elétrica - FEEC/UNICAMP – Campinas/SP

RESUMO

O objetivo desse trabalho, consiste em apresentar um estudo teórico orientado à segurança de aplicações Web, a partir da análise de alguns de seus principais pontos de vulnerabilidade, permitindo seu entendimento, a mitigação e uma visão compreensiva da importância de uma estratégia de desenvolvimento seguro, baseada em estudos analíticos e metodologias de mercado. De maneira específica, o projeto "*OWASP Top Ten 2021"* é a metodologia fundamental adotada neste trabalho, contemplando os pontos focais e cenários de fragilidade em comum entre a maioria das aplicações Web*,* em conjunto com demais experiências no mundo da tecnologia.

Palavras-Chave: Desenvolvimento seguro, Segurança da informação, Teste de intrusão, *Owasp Top 10 2021*.

ABSTRACT

This work consists in presenting theoretical research oriented to web application security through the analysis of some of its main points of vulnerability, allowing its understanding, mitigation and a comprehensive vision of the importance of a secure development strategy, based on analytical studies and market methodologies. In a specific way, the "OWASP Top Ten 2021" project is the fundamental methodology of this work, contemplating the focal points and common fragility scenarios among most Web Applications, together with other experiences in the technology world.

Keywords: Secure Development, Information Security, *pentest*, *Owasp Top 10 2021*.

1. INTRODUÇÃO

A tecnologia está em constante evolução e seu avanço traz consigo uma enorme dependência das capacidades computacionais, acesso à Internet e acesso à informação. E com a tendencial utilização dos dispositivos e das transações digitais, os riscos de perda, vazamento e roubo de informações, ou até mesmo a possibilidade de indisponibilidade de serviços aumentam significativamente.

As vulnerabilidades intrínsecas a aplicações e sistemas da informação, que, por vezes não são controladas, devido a incapacidade de compreensão da extensão das estruturas de comunicação complexas de hoje, podem causar enormes prejuízos financeiros as organizações. Ademais, a condição de risco passa a ser ainda maior quando as aplicações estão direcionadas para o contexto web, já que este tipo de aplicação vem numa crescente, respondendo por 67% de todos os ataques em 2021, segundo o *"Relatório de Inteligência de Ameaças Globais 2021 (GTIR)",* lançado pela empresa *"Nippon Telegraph and Telephone Corporation (NTT)*".

A fim de reduzir possíveis vulnerabilidades de segurança voltadas a aplicações Web, estratégias de desenvolvimento seguro são extremamente necessárias, já que o cenário web há algum tempo é considerado popular nas organizações, baseado em sua facilidade de acesso a serviços e gerenciamento de sistemas centralizado.

2. CONCEITOS BÁSICOS

Esta seção busca apresentar as variáveis nomenclaturas e termos que serão abordados ao longo do trabalho, e seus respectivos significados, para melhor compreensão sobre os assuntos tratados.

2.1 CWE

Segundo a MITRE, CWE (Common Weakness Enumeration – Enumerações de fraquezas comuns), é uma lista desenvolvida pela comunidade, com o intuito de informar os tipos de fraquezas de software e hardware. A empresa (MITRE), afirma que esta lista serve como uma linguagem comum e fonte de medição, para a identificação de fraquezas, mitigação e esforços de prevenção.

A CWE é patrocinada pelo Departamento de Segurança Interna dos Estados Unidos (DHS – Department of Homeland Security), Agência de Segurança Cibernética e Infraestrutura (CISA - Cybersecurity and Infrastructure Security Agency) e gerenciada pelo Instituto de Engenharia e Desenvolvimento de Sistemas de Segurança Interna (HSSEDI - Homeland Security Systems Engineering and Development Institute), operado pela MITRE Corporation (MITRE).

2.2 CVE

Segundo a Red Hat (2020), CVE (do inglês Common Vulnerabilities and Exposures), é uma lista de falhas de segurança de computadores, divulgadas publicamente. Dessa forma, as referências CVE que serão citadas neste trabalho, significam uma falha de segurança específica que recebeu um número de identificação, ID CVE.

Ainda em conformidade com a Red Hat (2020), o programa CVE assim como o CWE supervisionado é mantido pela corporação MITRE (Corporation).

2.3 MÉTODOS DE SOLICITAÇÃO HTTP

Segundo MOZILLA (2021), HTTP define um conjunto de métodos de solicitação para indicar a ação desejada a ser executada para um determinado recurso que se encontra no servidor. Esses métodos são: GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE e PATCH.

Embora também possam ser substantivos, esses métodos de solicitação às vezes são chamados de *verbos HTTP*.

2.4 JWT

Segundo a AUTH0 (2021), JSON web token (JWT), é um padrão aberto (RFC 7519) que define uma maneira compacta e independente para transmitir informações com segurança entre partes, como um objeto JSON. Um JWT pode ser enviado por meio de uma URL, por meio de um parâmetro POST ou dentro de um cabeçalho HTTP e é transmitido rapidamente.

Pode ser utilizado como forma de autenticação, quando um usuário efetua login com sucesso usando suas credenciais, Autorização, uma vez que um usuário é autenticado com sucesso ou também como troca de informações, em uma transmissão segura de dados entre as partes – por exemplo, aplicação cliente e servidor.

2.5 CORS

Segundo MOZILLA (2021), Cross-Origin Resource Sharing (CORS) é um mecanismo baseado em cabeçalho HTTP que permite a um servidor indicar qualquer origem (domínio, esquema ou porta) diferente da sua própria, a partir da qual um navegador deve permitir o carregamento de recursos.

Na prática, este conceito funciona de maneira que quando uma aplicação Front-end, JavaScript por exemplo, registrado com domínio https://dominio-frontend.com, tenta fazer uma solicitação à https://dominio-

backend.com. Por motivos de segurança, os navegadores restringiriam esta solicitação devido a uma requisição de origem cruzada. Dessa forma, a aplicação Front-end só conseguirá solicitar recursos à aplicação Back-end se os cabeçalhos CORS corretos forem configurados.

2.6 OAUTH

SOBERS (2018), diz que OAuth é um protocolo ou estrutura de autorização de padrão aberto que fornece aos aplicativos a capacidade de "acesso designado seguro", ou seja, é um protocolo de autenticação que permite com que um usuário seja aprovado em uma aplicação que está interagindo com outra, sem revelar sua senha, a partir de tokens de autorização.

2.7 NEGAÇÃO DE SERVIÇO (DoS) NEGAÇÃO DE SERVIÇO DISTRIBUÍDO (DDoS)

COSTA (2014), define DoS (do inglês, Denial of Service), como uma forma de ataque a sistemas computacionais. Consiste em uma tentativa de sobrecarga em um servidor ou computador, para que os recursos do sistema - pode-se citar como exemplo CPU, memória, banda, etc. - fiquem indisponíveis.

Já DDoS (do inglês, Distributed Denial of Service), segundo COSTA (2014), é um ataque similar ao de DoS, porém, ele atinge camadas extras, ou seja, atinge um ou mais computadores mestres, que gerenciam uma série de outros computadores, também chamados de "zumbis", causando um ataque distribuído. Dessa forma, com o acesso de vários computadores "mestres", e computadores "zumbis", o invasor os utiliza para aumentar a sobrecarga sobre determinado servidor.

2.8 PATCH

Segundo a FC BRASIL (2020), um patch - que traduzido ao português significa correção ou remendo - pode ser definido como uma solução rápida para atualizar ou corrigir um software. Patches são criados, quando há uma vulnerabilidade existente em um software, computador, dispositivos móveis ou outras máquinas em uma rede. Eles garantem que os hackers não usem a fragilidade para invadir redes corporativas.

Este termo será abordado neste trabalho em algumas menções de ataques cibernéticos ou em algumas formas de mitigação de riscos de segurança.

2.9 PHISING

PHISHING (2017), define o termo Phishing em si, como um crime cibernético em que um alvo é contatado por e-mail, telefone ou mensagem de texto por alguém que se passa por uma instituição legítima ou amigável, com o intuito de atrair indivíduos a fornecer dados confidenciais, como informações de identificação pessoal, dados bancários e senhas, resultando em roubo de identidade e perda financeira.

2.10 AUTENTICAÇÃO MULTIFATORES

HOSTMIDIA (2021), define Autenticação multifatores - ou também conhecida como autenticação de dois fatores – como o uso de dois ou mais fatores ou agentes de verificação de autenticidade. Isto é, a utilização de dois ou mais métodos para atestar a identidade de alguém para concessão de acesso a um sistema, documento ou informação.

Algumas formas de autenticação de multifatores podem ser o envio de um código numérico de verificação via Email ou SMS para uma conta/número já cadastrado previamente no sistema em questão, através de verificação biométrica como leituras de digitais ou reconhecimento facial, aplicativos de geração de códigos (como Google Authenticator), etc.

2.11 PIPELINES DE CI / CD

Por definição, o termo CI, define-se como C*ontinuous Integration*, e CD C*ontinuous Delivery*. Dessa forma, uma pipeline de CI / CD, consiste numa série de etapas que devem ser executadas para fornecer uma nova versão de software. (REDHAT, 2019)

Focada em melhorar a entrega de software, a partir de uma abordagem de DevOps, uma Pipeline CI / CD busca apresentar monitoramento e automação para melhorar o processo de desenvolvimento de aplicativos, especialmente nas fases de integração, teste, entrega e implantação. (REDHAT, 2019)

2.12 SUPPLY CHAIN ATTACK

GCSEC (2021) define um ataque de cadeia de suprimentos, como uma estratégia que visa afrontar empresas de software ou provedoras de serviços terceirizados, por meio de vulnerabilidades em sua cadeia de fornecimento, causando de certa forma, um efeito dominó.

MICROSOFT (2021) define Supply Chain Attack - ou Ataque a cadeia de suprimentos – como um tipo emergente de ameaça que tem como alvo os desenvolvedores e fornecedores de software, cujo objetivo é acessar códigos-fonte, processos de construção ou mecanismos de atualização, infectando aplicativos legítimos para distribuir malware em sua cadeia de fornecimento, ou seja, espalhá-lo para os clientes da empresa fornecedora do software atingido.

2.13 ATAQUE MITM

Malenkovich (2013) diz que um ataque MITM, ou Ataque Man-in-the-Middle, consiste em um ataque cujo invasor se posiciona entre duas partes que tentam comunicar-se, intercepta mensagens enviadas e depois se passa por uma das partes envolvidas. Dessa forma, este agente malicioso pode colocar suas armadilhas entre a vítima e sites relevantes, como sites de bancos e contas de e-mail.

A variante do ataque MITM mais utilizada é a partir de uma situação cujo agressor configura seu dispositivo wireless para atuar como ponto de WiFi, nomeando-o com um título comum em redes públicas, na qual se bem sucedido, o invasor poderá roubar todas as credenciais transacionadas. Estes ataques são

extremamente eficientes e difíceis de detectar, especialmente por usuários inexperientes ou desavisados. (Malenkovich**,** 2013)

2.14 SERIALIZAÇÃO E DESSERIALIZAÇÃO

Segundo DEVOPEDIA (2020), “desserialização” define-se no processo de transformar algum objeto serializado - por exemplo no formato JSON, ou XML - em um formato estruturado de dados - por exemplo baseado em uma classe no conceito de orientação a objetos. Podendo assim ser armazenado em banco de dados, em arquivos ou ser utilizado como parte das comunicações entre as funções do sistema.

DEVOPEDIA (2020) menciona serialização no processo inverso, isto é, coletar este objeto serializado e estruturado para aquela aplicação específica - por exemplo um objeto tipado em uma aplicação JAVA ou C# - e reconstruí-lo em um objeto compreensível – JSON, ou XML - para que outra aplicação possa realizar sua leitura.

2.15 LOG

Segundo MACHADO (2012), o registro de logs define-se no registro de dados e informações cruciais em arquivos de texto, para verificação posterior, que tem como objetivo o monitoramento e a detecção de ações impróprias nos sistemas de informação.

2.16 RANSOMWARE

KASPERSKY (2021) define Ransomware, a partir da divisão da palavra, "ransom", que significa resgate. Dessa forma, Ransomware é um software de extorsão que tem o intuito de bloquear um computador, como quaisquer arquivos presentes nele, e depois exigir um resgate para desbloqueá-lo, onde normalmente este resgate é solicitado em dinheiro. Este bloqueio pode ser realizado a partir da criptografia do sistema operacional ou apenas para arquivos individuais.

2.17 APT

APT (do inglês, Advanced Persistent Threats), como o nome "avançado" sugere, um ataque persistente avançado, utiliza técnicas de invasão contínuas, clandestinas e sofisticadas para obter acesso a um sistema e permanecer dentro dele por um período prolongado, com consequências potencialmente destrutivas.

2.18 DEVOPS E DEVSECOPS

MXM SISTEMAS (2020) define o conceito de DEVOPS como:

> “*O DevSecOps — desenvolvimento, segurança e operação — pode ser conhecido como a evolução do DevOps — desenvolvimento e operação. Embora o DevOps também envolva*

> *segurança, o foco do conceito está em agregar as práticas mais eficazes de desenvolvimento de software.*"

De acordo com a definição de DEVOPS apresentada, é possível concluir que embora equipes de DevOps estejam por vezes encarregadas de atividades de segurança, seu verdadeiro foco é no auxílio e suporte das equipes de desenvolvimento de software.

Com base nisso, MXM SISTEMAS (2020) define DEVSECOPS conforme abaixo:

> *"O DevOps faz parte de uma cultura de engenharia de software que tem como objetivo principal unificar a operação e o desenvolvimento e, por isso, tem sido cada dia mais adotada. No entanto, a segurança ainda é uma medida secundária nesse modelo. Logo, o DevSecOps é a opção que visa resolver essa questão."*

2.19 TESTE DE INTRUSÃO

Teste de intrusão ou muito conhecido pelo termo *pentest*, é o método de simulação de ataques cibernéticos em uma aplicação, com o objetivo de diagnosticar suas fragilidades de segurança, possibilitando a solidificação de melhores estratégias de defesa (BLOG DA CCM, 2019). Para LIMA (2020, p. 20), sua execução, em contextos específicos, pode ser benéfica na análise dos requisitos não-funcionais de um sistema, como desempenho, usabilidade e segurança.

SALGADO (2014) enaltece que este método de diagnóstico fortalece o ambiente tecnológico, pois é executado a partir do ponto de vista do atacante e não se compara às auditorias de segurança baseadas na norma ISO/IEC 27001, uma vez que estas são passivas, poucos intrusivas e aplicadas por meio de entrevistas aos funcionários. (apud CINTO, 2015, p. 21).

O posicionamento de SALGADO traz uma visão crítica, sobre às auditorias de segurança baseadas nas normas e diretrizes ISO/IEC 27001:2006, em comparação aos testes de penetração. Em vista disso, partindo da perspectiva da relação dos *pentests* com o padrão de segurança ISO/IEC 27001:2013, de acordo com SEGOVIA (2016), apenas a realização de análise de vulnerabilidade já mantém a conformidade com a própria ISO 27001:2013, porém não garantirá discernir o quão vulnerável a aplicação está. Este resultado só poderá ser obtido através da exploração detalhada pelo teste aplicado.

Para a sua execução, SEGOVIA (2016) recomenda direcionar os testes de intrusão em algumas fases (Planejamento, Coleta de Informações, Modelagem de Ameaças, Análise de vulnerabilidade, Exploração, Pós-exploração e Relatórios), na qual pode-se relacioná-las com o que diz a metodologia (do inglês Penetration Testing Execution Standard) PTES.

O padrão de execução do teste de penetração (PTES, do inglês Penetration Testing Execution Standard) (PTES, 2017), é uma metodologia de testes de intrusão. Ela é baseada em uma divisão de 7 seções principais, com objetivo de realizar a cobertura total necessária de um pentest.

O fluxo da Figura 1 abaixo simboliza as 7 etapas da metodologia PTES, com suas devidas descrições em seguida (LIMA, 2020, p. 22-23):

Figura 1. Fluxo das etapas da metodologia PTES

Fonte: LIMA, 2020, p. 22.

1. **Pré-engajamento** (pre-engagement interactions): etapa para definir o escopo e objetivo do teste de intrusão;

2. **Levantamento de informações** (intelligence gathering): etapa para coletar informações da AST, como usuário e senha, por exemplo;

3. **Modelagem de ameaças** (threat modeling): etapa para desenvolver estratégias para minimizar riscos associados à presença de vulnerabilidades na AST;

4. **Análise de vulnerabilidades** (vulnerability analysis): etapa para identificar vulnerabilidades que podem ser exploradas pelo testador na AST;

5. **Exploração** (exploitation): etapa para o testador estabelecer acesso à AST a partir de vulnerabilidades existentes;

6. **Pós-exploração** (post exploitation): etapa para manter acesso à AST e esconder possíveis evidências da invasão;

7. **Relatar resultados** (reporting): etapa para produzir um relatório com os resultados obtidos nos testes.

A sigla AST conforme LIMA (2020) é definida como Aplicação Sob Teste.

Segundo Sanches (2018), em um *pentest* pode-se utilizar diversas metodologias de mercado, que já foram aplicadas e testadas por outras empresas, consolidadas em uma espécie de script, de forma que o pentester não fique preso somente em uma metodologia, mas que a utilize como referência para investigação dos riscos.

Ainda de acordo com Sanches, atualmente, as principais metodologias de mercado para a realização de testes de intrusão em aplicações Web são: PTES (Penetration Testing Execution Standard), OSSTMM (Open Source Security Testing Methodology Manual), NIST 800-115 (National Institute of Standards and Technology), e OWASP Top 10 (Open Web Application Security Project), que será a metodologia referência a ser utilizada neste trabalho. (Sanches, 2018)

3. OWASP

Partindo ao foco deste trabalho, a apresentação da metodologia base será enfatizada permitindo que o leitor tenha maior compreensão e interpretação metodológica, acerca das definições técnicas a serem expostas nas próximas seções.

3.1 SOBRE A FUNDAÇÃO

Fundada em 1º de dezembro de 2001, a OWASP (Open Web Application Security Project - Projeto Aberto de Segurança em Aplicações Web) é uma fundação sem fins lucrativos que trabalha em prol da segurança da informação. Uma metodologia de mercado, formada por desenvolvedores, pesquisadores e especialistas em segurança da informação, que disponibilizam conteúdos educacionais através de conferências, treinamentos, artigos técnicos, e projetos de software de código aberto, liderados pela comunidade. (OWASP, 2021)

A entidade produz uma variedade de materiais de forma colaborativa, transparente e aberta, para empresas e profissionais da área manterem e implementarem aplicações confiáveis, além de fornecer um dos principais projetos de pesquisa utilizados para testes de intrusão e desenvolvimento seguro, o já citado OWASP TOP 10. (OWASP, 2017)

3.2 O PROJETO TOP 10

Segundo a própria OWASP (OWASP, 2021), o projeto Top 10 é um documento padrão de conscientização de segurança de aplicativos da web para desenvolvedores. Ele representa um amplo consenso sobre os riscos de segurança mais críticos para aplicativos da web.

Em seu último lançamento de 2021 (OWASP, 2021), a fundação busca estimular as empresas para que adotem este documento, a fim de garantir a minimização dos riscos abordados, e complementa que a utilização do OWASP Top 10 seja talvez o primeiro passo mais eficaz para mudar a cultura de desenvolvimento de software em uma organização, para que códigos mais seguros sejam produzidos.

Esta metodologia, na maioria dos contextos, é referência útil para orientar os desenvolvedores, a partir de problemas comuns que tornam um código inseguro. Conforme as aplicações são moldadas, baseadas nas detecções e abordagens dos riscos, é certo que haverá o aumento da resistência às ameaças cibernéticas.

A construção do projeto é realizada de maneira híbrida. De acordo com o OWASP, esta abordagem mista é adotada porque os dados estatísticos refletem apenas fatos conhecidos do passado, mas podem não cobrir tendências recentes, uma área que complementa os resultados da pesquisa.

A definição dos 10 riscos da lista, é compilada a partir de dois métodos, conforme será citado abaixo:

1. Coleta de dados

O processo de coleta de dados é o que define oito dos dez riscos da lista, que consiste no período de coleta de dados estatísticos sobre vulnerabilidades de aplicações web, encontradas em vários processos. Os dados são fornecidos por organizações parceiras, que realizam serviços de testes em aplicações de outras instituições ou contribuem com dados de teste internos e, doravante, os enviam para a OWASP a partir de uma convocatória de dados.

Segundo a OWASP (OWASP, 2021), a fim de tornar este o maior e mais abrangente conjunto de dados de segurança de aplicativos, foram fornecidos dados de mais de 500 mil aplicativos. Ainda em conformidade com a OWASP (OWASP, 2021), as empresas doadoras dos dados para a edição atual são:

AppSec Labs; Cobalt.io; GitLab; *HackerOne; HCL Technologies; Micro Focus; PenTest-Tools; Probely; Sqreen; Veracode; WhiteHat (NTT);*

2. Pesquisa da comunidade Top 10

A pesquisa da comunidade têm intuito de questionar à especialistas em segurança e desenvolvimento na linha de frente, suas visões e previsões a respeito de fraquezas essenciais que podem não ser refletidas apenas em dados estatísticos. Seus resultados implicam na determinação de duas das dez categorias do projeto.

Esta metodologia de pesquisa foi implementada pela fundação na última edição lançada em 2017, visto que segundo a OWASP (OWASP, 2021), esta é uma forma valiosa de permitir que os indivíduos da comunidade identifiquem riscos importantes que podem ter passado despercebidos na coleta dos dados pelas organizações.

A pesquisa segue o seguinte fluxo no qual inicialmente são mapeados CWEs iminentes de entrar na lista do Top 10, ou apresentados em eventos significativos. Estes CWEs são retornados para a comunidade para avaliação, para que em seguida seja realizada a tabulação dos resultados. Esta tabulação consiste em comparar os resultados de pesquisa com a análise dos dados, onde a partir disso, são selecionados os dois riscos que receberem a maioria dos votos dos especialistas, e ainda não estão presentes nos dados analisados.

4. ANÁLISE DE RISCOS

Desde 2017, quando a última lista com as dez maiores vulnerabilidades foi lançada, analisando a categorização de riscos realizada pela OWASP há 4 anos atrás, comparada com o recente lançamento de 2021, é possível mensurar algumas das novas tendências de fragilidades de segurança, no ciclo de desenvolvimento de aplicações Web.

A figura 2, apresenta um comparativo das vulnerabilidades da última edição com o lançamento de 2021.

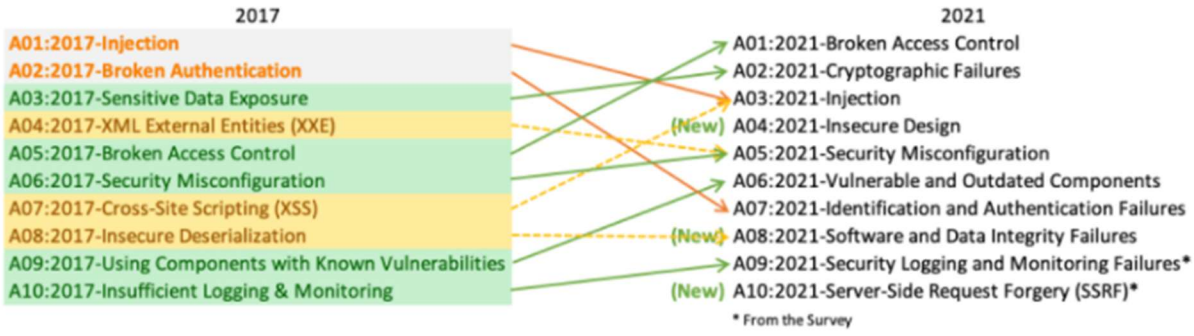

Figura 2. Comparativo OWASP TOP 10 do ano 2017 a 2021

Fonte: OWASP, 2021.

Em uma análise comparativa a partir da Figura 2, é possível perceber que todas as categorias foram mantidas de 2017 para 2021, porém com transformações em suas classificações. De maneira geral, em alguns

casos houve a consolidação de mais de uma categoria da última edição, sendo agrupada em apenas uma. De acordo com a OWASP (2021) estas alterações ocorreram devido a necessidade de concentrar na causa raiz em vez do sintoma.

Houve também a inclusão de três novas categorias, que são: A04:2021-Insecure Design, focada nos riscos relacionados a falhas de design e arquitetura de aplicações; *A08:202-Software and Data Integrity Failures* com o intuito de realizar suposições relacionadas a atualizações de software e A10:2021-Server-Side Request Forgery (SSRF), que foi a primeira das duas categorias adicionadas pelo processo de pesquisa da comunidade da OWASP (OWASP, 2021).

Segundo uma análise realizada pela HKCERT (2021), foram resumidos alguns pontos a respeito das alterações:

- A Exposição de Dados Sensíveis (Sensitive Data Exposure) passou a ser reclassificada como Falhas Criptográficas, devido a este tipo de falha ser, na verdade, a causa raiz do sintoma de exposição de dados confidenciais.
- Cross-Site Scripting (XSS) foi incluso na categoria de Injeção (Injection), o que tecnicamente, é uma injeção de script direcionada a outros usuários que acessaram o site afetado. Este é um ataque diferente dos métodos de injeções que acontecem do lado do servidor, como SQL, comandos do sistema operacional e injeções LDAP.
- A vulnerabilidade XML External Entities (XXE) passou a ser recategorizada à "Configuração Incorreta de Segurança" (Security Misconfiguration), já que muitas vezes é um erro cometido pelo desenvolvedor ao configurar incorretamente os analisadores XML em seu código.
- A desserialização insegura (Insecure Deserialization) é colocada em "Falhas de Integridade de software e de dados". Este, por ser um problema de proteção insuficiente no tratamento de objetos serializados, leva à adulteração de dados e à transmissão desses à aplicação, a partir de uma fonte não confiável. Um número significativo de desserializações inseguras é atribuído à vulnerabilidade de código aberto, que deve ser gerenciada ativamente e não deve ser negligenciada.

4.1 BROKEN ACCESS CONTROL

O controle de acesso impõe a política de modo que os usuários não possam agir fora de suas permissões pretendidas. Dessa forma, a quebra do controle de acesso, normalmente leva à divulgação, modificação ou destruição de dados, informações ou ao desempenho de uma funcionalidade comercial do sistema, que se encontra fora dos limites do usuário. (OWASP, 2021)

OWASP (2021), diz que, partindo da quinta posição, para líder do ranking, 94% dos aplicativos foram testados para alguma forma de controle de acesso quebrado, com uma taxa de incidência média de 3,81% e tem o maior número de ocorrências no conjunto de dados, contribuído com mais de 318k.

SIEMBA (2021) diz que muitos gigantes do negócio já foram vítimas deste ataque, devido à falta de implementação de um mecanismo de controle de acesso adequado, levando a uma autenticação fraca, falta de controle de acesso de nível funcional e gerenciamento de sessão.

Segundo Goldman (2016) dados fiscais como impostos e salários de mais de 431.000 funcionários da rede varejista foram vazados, tornando-se acessíveis através do site W-2 eXpress da empresa Equifax, uma agência de crédito que segundo Krebs (2016) torna os formulários W-2 eletrônicos acessíveis para download para muitas empresas, incluindo a Kroger.

> *"According to a letter Kroger sent to employees dated May 5, thieves were able to access W-2 data merely by entering at Equifax's portal the employee's default PIN code, which was nothing more than the last four digits of the employee's Social Security number and their four-digit birth year. "(*Krebs, 2016)

IDX (2016) resume a relação entre ambas as empresas (Kroger e Equifax Inc.) e o ataque ocorrido, de forma que, a Kroger contratou a Equifax para fornecer seu conveniente sistema eletrônico W-2 aos próprios funcionários da Kroger. Porém, em seu e-mail para funcionários, a gigante do varejo (Kroger) reconheceu que o site W-2 eXpress da Equifax, usa informações de login padrão com base em SSNs (Social Security Number - número de identificação pessoal nos Estados Unidos) e datas de nascimento. Conforme já mencionado por Krebs (2016), já que apenas o ano de nascimento de quatro dígitos era necessário, este foi um ataque relativamente fácil para os criminosos.

Formas de exploração

Abaixo, consta-se algumas vulnerabilidades comuns de controle de acesso OWASP (2021):

- Violação do princípio de privilégio mínimo ou negação por padrão, onde o acesso deve ser concedido apenas para determinados recursos, funções ou usuários, mas por falta de tratamento, se encontra disponível para qualquer pessoa.
- Omissão de verificações de controle de acesso, a partir de modificações diretas na URL (como adulteração de parâmetros), no estado interno da aplicação, na página HTML ou utilizando uma ferramenta de ataque que modifica as solicitações que vão para a API.
- Permissão da visualização ou edição da conta de outro usuário, fornecendo seu identificador exclusivo.
- Acesso à API sem controles de acesso de métodos de solicitação HTTP POST, PUT e DELETE.

- Elevação de privilégio. Agir como um usuário sem estar logado ou agir como um administrador quando logado como um usuário. Exemplo: Forçar a navegação para determinadas páginas, autenticado como um usuário não autenticado, ou navegar para páginas privilegiadas como um usuário padrão.
- Manipulação de metadados, como reproduzir ou adulterar um token de controle de acesso JSON Web Token (JWT), um cookie ou um campo oculto manipulado para elevar privilégios ou abusar da invalidação de JWT.
- A configuração incorreta do CORS permite o acesso à API de origens não autorizadas / não confiáveis.

Mitigação

Segundo a HACKSPLANNING, não existe uma solução única para implementar corretamente o controle de acesso, basta que sua estratégia de controle de acesso abranja três aspectos:

- **Autenticação**: Identificar corretamente um usuário quando ele retorna ao aplicativo.
- **Autorização**: Decidir quais ações um usuário deve e não deve ser capaz de executar depois de ter sido autenticado.
- **Verificação de permissão**: Avaliar a autorização no momento em que um usuário tenta executar uma ação.

Como forma de prevenção, a OWASP (2021) sugere que os tratamentos de controles de acessos devem ser aplicados do lado do servidor confiável, de forma que o invasor não consiga realizar modificações na verificação de controle de acesso. A fundação, lista os seguintes pontos como forma de mitigação:

- Exceto para recursos públicos, negar por padrão.
- Implementação e reutilização de mecanismos de controle de acesso em todo o aplicativo, incluindo a minimização do uso de Cross-Origin' Resource Sharing (CORS).
- Os controles de acesso ao modelo devem impor a propriedade do registro, em vez de aceitar que o usuário possa criar, ler, atualizar ou excluir qualquer registro.
- Os requisitos de limite de negócios de aplicativos exclusivos devem ser impostos por modelos de domínio.

- Desativar a lista de diretórios do servidor da web e certificar de que os metadados do arquivo - por exemplo o arquivo *.git* – arquivo do sistema GIT de controle de versão de arquivos do projeto - e os arquivos de backup não estejam presentes nas raízes da web.
- Registro de falhas de controle de acesso e notificação aos administradores da aplicação, quando apropriado (por exemplo, falhas repetidas).
- Limite de taxa ao acesso da API e do controlador, afim de minimizar os danos de um possível conjunto de ferramentas de ataque automatizado.
- Os identificadores de sessão com estado devem ser invalidados no servidor após o logout. Os tokens JWT sem estado devem ter vida curta para que a janela de oportunidade para um invasor seja minimizada. Para JWTs de longa duração, é altamente recomendável seguir os padrões OAUTH para revogar o acesso.

4.2 CRYPTOGRAPHIC FAILURES:

Anteriormente conhecida como Exposição de dados confidenciais (*Sensitive Data Exposure*) como mostra na Figura 2 da seção 3, segundo a OWASP (2021) este é mais um sintoma amplo do que uma causa raiz, onde esse ajuste de categoria está focado nas falhas relacionadas à criptografia (ou falta dela). O que muitas vezes leva à exposição de dados confidenciais.

Segundo dados disponibilizados pela OWASP (2021), 79,33% das aplicações testadas detectaram falhas criptográficas, apontando uma taxa de incidência máxima de 46,44%, e uma taxa de incidência média de 4,49%, com um total de 233.788 ocorrências.

Um grande caso a ser citado, conforme a IMMUNIWEB (2021), é a violação massiva contra a Equifax, em 29 de julho de 2017, que causou o roubo de dados altamente sensíveis de 143 milhões de usuários nos Estados Unidos – cerca de 44% da população. De acordo com Ivanova (2017), o ex-CEO da empresa repassou a um comitê do congresso que a empresa utiliza muitas técnicas de proteção de dados, porém esses dados específicos não foram criptografados em repouso.

IMMUNIWEB (2021) apresenta um raciocínio em que, tecnicamente, uma exposição de dados confidenciais pode ser causada por meio de vários outros riscos, como Injeções, quebra de controle de acesso, etc. Porém, apesar das vulnerabilidades que precedem o roubo de dados e custos subsequentes após o roubo de dados, existe fundamentalmente apenas um risco: a exposição real desses dados. Dessa forma, IMMUNIWEB (2021) conclui que se os dados forem realmente mantidos em segurança, e devidamente criptografados, as demais vulnerabilidades passariam a contar menos, e as consequências seriam muito menores.

IMMUNIWEB (2021) diz que a criptografia pode ser dividida em algumas vertentes, onde primeiramente, a criptografia dispõe de dois estados: em repouso (ou seja, em armazenamento); e em trânsito (ou seja, sendo transportados de um local para outro). Pode-se completar conforme diz a OWASP (2021), em que, determinar as necessidades de proteção dos dados em trânsito e em repouso deve ser o primeiro passo da implementação de criptografia de dados, principalmente se esses dados se enquadrarem nas leis de privacidade, por exemplo, Regulamento geral de proteção de dados (GDPR). Pode-se citar também a própria Lei geral de proteção de dados (LGPD) no Brasil.

Formas de exploração

De acordo com uma análise realizada por HOLMWOOD (2021), sobre as várias formas de ataque descritas pela OWASP (2021), foram divididas três categorias, conforme abaixo:

1. **Operacional**:

    - Chaves criptográficas fracas, falta de gerenciamento ou falta de rotação das mesmas, levando a um acesso não autorizado, como legítimo.
    - Downgrades de protocolos. Segundo Educalingo na computação, o termo Downgrade refere-se ao retorno de um software para uma versão anterior, ou seja, é o oposto da atualização. Estes podem ser rebaixados para remover recursos não utilizados ou com erros, e aumentar a velocidade e/ou a facilidade de uso. Partindo para uma definição focada em Downgrades de protocolos, segundo Naziridis (2021), downgrades podem ser considerados também como ataques de rede, que forçam os computadores a abandonar um tipo seguro de conexão, como uma conexão criptografada, recorrendo a versões mais antigas e vulneráveis.
    - Mensagens de erro que podem levar a inferências sobre os dados sendo transportados.
    - Transmissão de dados em texto não criptografado.

2. **Uso incorreto de criptografia:**

    - Aleatoriedade insuficiente, o que pode tornar o texto cifrado previsível.
    - Uso de algoritmo ou protocolo criptográfico antigo ou fraco.

3. **Má criptografia:**

    - Uso de funções hash obsoletas, como MD5 ou SHA1, ou funções hash não criptográficas usadas quando funções hash criptográficas são necessárias.

Mitigação

A mitigação de falhas criptográficas pode ser realizada de diversas maneiras, dependendo de seu escopo e da análise realizada sobre os dados do sistema. Segundo a CIPHERSTASH (2021), os destaques são:

- Classificação dos dados processados, armazenados ou transmitidos por um aplicativo, com o objetivo de entender quais dados serão necessários, e para defender e identificar os controles apropriados.
- Criptografia de todos os dados classificados como "confidenciais".
- Utilização de uma criptografia que forneça sigilo de encaminhamento, para garantir que os dados criptografados no passado não possam ser descriptografados caso as chaves de sessão sejam expostas.
- Verificação da eficácia das configurações de criptografia de forma independente.

4.3 INJECTION

WALLARM (2017) diz que um ataque de injeção ou *Injection*, é quando usuários com intenções maliciosas injetam sua própria entrada maliciosa através de quaisquer entradas de dados possíveis, que sejam controláveis pelo próprio usuário, incluindo variáveis de ambiente, parâmetros, serviços da web externos e internos, funcionalidade de importação ou qualquer outra possibilidade para os usuários inserirem seus próprios vetores de ataque. WALLARM (2017), complementa que estas entradas maliciosas vindas do usuário levam a execução remota do código, que pode ocasionar a perda de dados, ações não autorizadas, corrupção de dados e muito mais.

Este risco pode ser explorado de diversas formas, a partir de quaisquer processos que recebam a entrada do usuário. O ataque de injeção SQL ou *SQL Injection*, é uma das mais comuns maneiras. (WALLARM, 2017)

PINTO e col. (2019) dizem que a injeção SQL é um tipo ataque que ocorre quando usuários com intenções maliciosas, injetam um código malicioso a partir de formulários de entrada de dados, sem validação. Conforme PINTO e col. (2019), pode-se citar alguns exemplos de entradas de dados de usuários, como, formulários de cadastro como nome, data de nascimento, endereço, também formulários de login ou campos de pesquisa presentes em websites.

De acordo com uma pesquisa de Akamai, provedora de serviços em nuvem com sede nos Estados Unidos:

> (...) *"SQL injection and Local Filer Inclusion attacks comprised over 85 percent of the attack vectors recorded. SQL injection attacks accounted for 65 percent of web-based attack vectors from November 2017 to March 2019. "*(TECHMONITOR, 2019)

Em uma análise comparativa, a partir da Figura 2, seção 3, além de sua queda no posicionamento de 2017 a 2021, observa-se que foram fundidas duas categorias principais em uma só, unificando INJECTION e CROSS-SITE SCRIPTING. O que é coerente, já que o CROSS-SITE SCRIPTING embora seja diferente de outros métodos de injeção do lado do servidor, é afinal, um tipo de ataque de injeção. (HKCERT, 2021)

A respeito do CROSS-SITE SCRIPTING, KirstenS (2021) diz que:

> *"Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user."* (KirstenS, 2021)

Em conformidade com KirstenS (2021), após uma brecha de execução de script maliciosa ser devidamente explorada por um atacante, o navegador do usuário, por achar que o script veio de uma fonte confiável, irá permitir que o script possa acessar qualquer cookie, token de sessão ou outras informações confidenciais retidas pelo navegador e usadas naquele site.

Considerado estatisticamente o maior e mais frequente risco, a injeção se manteve em primeiro lugar no ranking OWASP Top 10 em 2017. Em uma análise comparativa, no ano de 2021, a ex-líder dos rankings de 2017 deslizou para a terceira posição no ranking, na qual, segundo a OWASP, 94% das aplicações testadas apontaram alguma forma de injeção com uma taxa de incidência máxima de 19%, uma taxa de incidência média de 3% e 274k ocorrências. (OWASP, 2021)

IMMUNIWEB (2021) aponta que o impacto desse tipo de ataque pode variar a partir das funcionalidades da aplicação e do tipo da injeção que está sendo explorada, porém um ataque bem sucedido pode ser o causador de perda de dados, ações não autorizadas, corrupção de dados e ganho de acesso a dados sensíveis de usuários, como ocorreu com a violação de Heartland em 2008.

MESSMER (2009) diz que o ataque a Heartland Payment Systems comprometeu os dados dos cartões que cruzaram a rede varejista, juntamente com detalhes pessoais de muitos clientes. Segundo a IMMUNIWEB (2021), apesar dos dados exatos não terem sido divulgados, este foi por muitos anos apontado como a maior violação de dados de todos os tempos.

Formas de exploração

Segundo a OWASP (2021), uma aplicação é vulnerável a ataques de injeção quando:

- Os dados fornecidos pelo usuário não são validados, filtrados ou higienizados pelo aplicativo.

- Consultas dinâmicas ou chamadas não parametrizadas sem escape ciente do contexto são usadas diretamente no interpretador.
- Dados hostis são usados nos parâmetros de pesquisa de mapeamento relacional de objeto (ORM - Object Relational Mapping Tools) para extrair registros confidenciais adicionais.
- Dados hostis são usados diretamente ou concatenados. O SQL ou comando contém a estrutura e os dados maliciosos em consultas dinâmicas, comandos ou procedimentos armazenados.

Mitigação

OWASP (2021) aponta que para prevenir a injeção, deve-se manter os dados separados dos comandos e consultas, onde logo abaixo elencamos algumas maneiras de mitigar este problema, de acordo com a própria OWASP:

- A utilização de uma API segura, evitando o uso do interpretador SQL inteiramente, fornecendo uma interface parametrizada com Object Relational Mapping Tools (ORMs). ORMs, segundo Cadu (2011), é uma técnica de mapeamento de objeto relacional que permite fazer uma relação dos objetos de classes mapeadas com os dados que os mesmos representam em seu devido banco de dados.
- Para quaisquer consultas dinâmicas, realizar tratativa de escape para caracteres especiais, usando a sintaxe de escape específica para esse interpretador. No entanto, esta forma de prevenção pode não cobrir totalmente um ataque de Injection, já que estruturas SQL, como nomes de tabelas, nomes de colunas e assim por diante, não podem ter escape. Portanto, nomes de estruturas fornecidos pelo usuário são perigosos, e mesmo um tratamento como este, não seria capaz de cobrir esta falha. Este é um problema comum em softwares de elaboração de relatórios.
- Uso do LIMIT e outros controles SQL em consultas para evitar a divulgação em massa de registros no caso de injeção de SQL. GALLAGHER (2020) diz que o LIMIT é uma clausula SQL, que restringe quantos registros serão retornadas a partir de uma consulta SQL.

4.4 INSECURE DESIGN

Segundo a OWASP (2021), o Design Inseguro (Insecure Design) é uma nova categoria para 2021, que se concentra nos riscos relacionados a falhas de design e arquitetura, onde busca alertar os desenvolvedores para que façam mais uso de modelagem de ameaças, padrões de design seguros e arquiteturas de referência, a fim de antecipar ataques e aplicar medidas preventivas para proteger adequadamente as aplicações Web.

Com uma cobertura de 40 CWEs mapeados e um total de 262.407 ocorrências, a partir dos dados de testes disponibilizados pela OWASP (2021), segundo a própria fundação, o design inseguro não deve ser considerado como a fonte de todas as outras 10 categorias de riscos principais, pois há uma diferença entre design inseguro e implementação insegura.

HKCERT (2021) menciona que a lógica por traz de ferramentas automatizadas, se não devidamente controladas podem causar riscos de segurança para as empresas que as utilizam. O incidente de vazamento de dados no Facebook no início de 2021, é um exemplo disso. Segundo RIGUES (2021), o vazamento de dados de 533 milhões de usuários foi causada por atores maliciosos que abusaram de uma ferramenta de importação de contatos para obter uma quantidade limitada de dados sobre um perfil. RIGUES (2021), complementa:

> *"Segundo a empresa, os dados não são resultado de uma invasão aos seus sistemas, mas sim do uso de uma técnica conhecida como scraping (raspagem ou coleta de dados), onde ferramentas automatizadas são usadas para coletar dados em páginas e perfis publicamente disponíveis."*

Com isso pode-se concluir que o mau uso de uma ferramenta também pode ser um risco, e não somente ataques a partir de vulnerabilidades diretas. A principal causa dessa violação foi um uma arquitetura insegura aplicada no uso da ferramenta. Um design de aplicação mal elaborado pode levar a graves consequências para um negócio.

Design Seguro x Design Inseguro

Segundo descrito pela OWASP (2021), um design seguro pode ter defeitos de implementação que levam a vulnerabilidades que podem ser exploradas. Já um design inseguro não pode ser corrigido nem mesmo por uma implementação perfeita, pois, por definição, na arquitetação base do projeto, os controles de segurança necessários não foram criados para a defesa contra ataques específicos.

Mitigação

IMMUNIWEB (2021) lista algumas etapas que podem ser úteis na projeção de uma aplicação:

- Implementação SDLC para ajudar a avaliar e projetar controles relacionados à segurança e à privacidade.
- Implementação de uma biblioteca de padrões de design seguros para utilização na aplicação.
- Avaliação do processo de autenticação de aplicativos, controle de acesso, lógica de negócios e a utilização da modelagem de ameaças para identificação de possíveis vetores de ataque.
- Criação de testes de integração e verificações para validar se todos os fluxos críticos são resistentes ao modelo de ameaça projetado.
- Atenção aos recursos computacionais e ao limite do consumo pelo usuário do serviço

4.5 SECURITY MISCONFIGURATION

Segundo MANAGEENGINE, Configurações incorretas de segurança ou *Security Misconfiguration* são controles de segurança que são configurados incorretamente ou de forma insegura, colocando sistemas e dados em risco, como alterações de configuração mal documentadas ou a utilização de configurações padrão.

De acordo com dados disponibilizados pela OWASP (2021) 90% dos aplicativos foram testados para alguma forma de configuração incorreta, com uma taxa de incidência média de 4%, e mais de 208 mil ocorrências de Enumerações de Fraquezas Comuns (CWEs), nesta categoria de risco, dentre elas. A Fundação complementa que com muitas mudanças em softwares altamente configuráveis, não é surpreendente ver essa categoria subir.

Em uma análise comparativa a partir da Figura 2, na seção 3, é possível observar que a categoria XML External Entities (XXE), presente no 4º lugar da edição de 2017, foi contida em Security Misconfiguration. Segundo Vicente (2019), a essência dos riscos de injeção de XXE é uma configuração incorreta nos servidores que aceitam XML como entrada de aplicativos *clients* não confiáveis, na qual, invasores abusam desse aspecto para injetar cargas úteis de XMLs personalizados, para extrair dados, falsificar solicitações do lado do servidor (SSRF), ou conduzir tentativas de negação de serviço (DoS).

Segundo IMMUNIWEB (2021), um relatório de abril de 2018 da IBM apontou algumas mudanças interessantes nas tendências de segurança ao longo de 2017.

> *"IBM reports that breaches related to bad configuration jumped by 424% in 2018, accounting for nearly 70% of compromised records over the year."* (IMMUNIWEB, 2021)

Formas de exploração

OWASP (2021) informa que um aplicativo pode ser vulnerável nessa categoria se contemplar alguns critérios, onde pode-se citar os destaques como:

- Falta de proteção de segurança apropriada ou permissões configuradas incorretamente em serviços em nuvem.
- Recursos desnecessários são ativados ou instalados.
- Contas e senhas padrão ainda ativas e inalteradas.
- O tratamento e mensagens de erros revelam erros excessivamente informativos aos usuários.
- Para sistemas atualizados, recursos mais recentes de segurança desabilitados ou não configurados com segurança.
- As configurações de segurança de Frameworks (Spring, ASP.NET) e de bibliotecas não definidas para proteger os valores da aplicação.

- Software desatualizado ou vulnerável (Consulte a seção 3.6 – VULNERABLE AND OUTDATED COMPONENTS).

Mitigação

IMMUNIWEB (2021) busca acalmar esta situação da seguinte maneira:

> *"Fortunately, most security misconfiguration risks are known and documented. This means automated testing resources are especially useful for detecting this kind of problem, especially if the testing tools are well-integrated into".*

Porém, mesmo com as mais atualizadas documentações de ferramentas e tecnologias, este tipo de falha sempre pode vir a acontecer. Dessa maneira, abaixo a OWASP (2021) busca apresentar algumas formas de prevenção:

- Um processo de proteção repetível torna mais rápido e fácil implantar outro ambiente que esteja devidamente bloqueado. Os ambientes de desenvolvimento, controle de qualidade e produção devem ser todos configurados de forma idêntica, com credenciais diferentes usadas em cada ambiente. Este processo deve ser automatizado para minimizar o esforço necessário para configurar um novo ambiente seguro.
- Uma plataforma mínima sem recursos, componentes, documentação e amostras desnecessários torna-se inviável e de risco.
- Remoção de recursos e estruturas não utilizados.
- Uma tarefa para revisar e atualizar as configurações apropriadas para todas as notas de segurança, atualizações e patches como parte do processo de gerenciamento de patches (Consulte a seção 3.6 – VULNERABLE AND OUTDATED COMPONENTS).
- Revisão de permissões de armazenamento em nuvem (por exemplo, uma aplicação que utiliza a ferramenta AWS S3, verificar as permissões dos buckes S3).
- Uma arquitetura de aplicativo segmentada fornece separação eficaz e segura entre componentes ou locatários, com segmentação, conteinerização ou grupos de segurança em nuvem (ACLs). Segundo a AWS (2021) Network ACL, é uma lista de controle de acesso à rede, que consiste numa camada de segurança opcional para uma VPC (Amazon Virtual Private Cloud), que funciona como um firewall para controlar o tráfego de entrada e saída de uma ou mais sub-redes.

- Envio de diretivas de segurança para aplicações clients, por exemplo, cabeçalhos de segurança. Por exemplo, como será citado no tópico de mitigação da seção 4.7 IDENTIFICATION AND AUTHENTICATION FAILURES, onde com uma diretiva de segurança, é possível definir que se um usuário errar a senha três vezes, dentro de um período de meia hora, a sua conta deve ficar bloqueada até que um Administrador desbloqueie.
- Um processo automatizado para verificar a eficácia das configurações em todos os ambientes.

4.6 VULNERABLE AND OUTDATED COMPONENTS

Na última edição OWASP TOP 10 2017 (OWASP, 2017), nomeado como Uso de Componentes com vulnerabilidades conhecidas *(Using Components with Known Vulnerabilities)*, esta categoria passou da 9ª posição em 2017 para a 6ª posição em 2021, no qual consiste em um problema conhecido que de acordo com a OWASP (2021), houve dificuldades em testar e avaliar os riscos para esta categoria. Além de ser a única categoria que não tem nenhuma Vulnerabilidade e Exposições Comuns (CVEs Common Vulnerabilities and Exposures) mapeada para os CWEs (Common Weakness Enumeration) incluídos.

Segundo a OWASP (2021), esta categoria foi a segunda colocada na pesquisa da comunidade Top 10, porém apesar de sua alta relevância a partir da pesquisa designada pelos especialistas que colaboraram para com a sua análise, a mesma já havia dados suficientes para se enquadrar no índice Top 10.

De acordo com Nayak (2021), essa mudança de status significativa reflete a importância crescente dessa vulnerabilidade no desenvolvimento de aplicações modernas e a preocupação crescente em como a comunidade de segurança passará a ver esse risco.

IMMUNIWEB (2021) diz que atualmente o número de aplicativos que fazem o uso de componentes pré-existentes em vez de serem totalmente codificados do zero é cada vez maior, onde este problema é cada vez mais grave, devido a tendência de desenvolvedores utilizarem cada vez mais componentes de código aberto expostos à comunidade. Sob a pressão da entrega, estes componentes, por vezes, não são suficientemente verificados.

Uma publicação realizada por UCHILL (2021) para a revisa SC Magazine, menciona que Chris Wysopal - fundador e diretor de tecnologia do testador de segurança de aplicativos automatizado Veracode - estima que 90% das aplicações modernas utilizam componentes de código aberto. UCHILL (2021) complementa que mesmo perante a estes riscos, o código fonte aberto é onipresente nas aplicações por vários motivos, como a economia de tempo e dinheiro durante o desenvolvimento de uma funcionalidade ou execução de testes.

Formas de exploração

De acordo com a OWASP (2021), uma aplicação é vulnerável à esta categoria a partir de alguns fatores, conforme será citado abaixo:

- Desconhecimento das versões dos componentes e dependências que são utilizadas na aplicação (tanto do lado do cliente quanto do lado do servidor).
- Software vulnerável, sem suporte ou desatualizado, podendo incluir também toda sua estrutura externa, como Sistema Operacional, Servidor Web, Sistema de Gerenciamento de Banco de Dados (DBMS), APIs e bibliotecas.
- Ausência da execução de varredura de vulnerabilidades regularmente.
- Ausência de correção da plataforma, estruturas e dependências subjacentes de maneira oportuna e baseada em riscos. Isso geralmente acontece em ambientes em que a correção é uma tarefa mensal ou trimestral sob controle de alterações, deixando as organizações abertas a dias ou meses de exposição desnecessária, a vulnerabilidades corrigidas.
- Ausência de testes de compatibilidade de bibliotecas atualizadas ou patches.
- Ausência de proteção das configurações dos componentes, dependências e ferramentas auxiliares (Categoria citada na seção 3.5 SECURITY MISCONFIGURATION).

Mitigação

IMMUNIWEB (2021) diz que a atenção a respeito de qualquer vulnerabilidade em componentes de código aberto continua sendo um grande problema e, complementa, que a conscientização é a melhor defesa de uma empresa contra riscos de vulnerabilidades conhecidas.

Segundo IMMUNIWEB (2021):

> *"A web application firewall (WAF) can also be employed since defence in depth is always a good principle. But WAF is no silver bullet, and researchers have repeatedly demonstrated they can frequently be bypassed by competent attackers"*

OWASP (2021), cita algumas formas de mitigação para este risco, como:

- Remoção de dependências não utilizadas, recursos, componentes, arquivos e documentações desnecessárias.
- Levantamento contínuo das versões dos componentes do lado do cliente e do lado do servidor (por exemplo: estruturas, bibliotecas) e suas dependências, usando ferramentas como OWASP Dependency-Check, retire.js, etc.

- Monitoramento de bibliotecas e componentes sem manutenção ou que não possuem lançamentos de patches de segurança para versões anteriores. Se não for possível obter patches pelo responsável pela biblioteca, por exemplo, considerar implantar um patch virtual para monitorar, detectar ou proteger contra o problema descoberto.
- Coletar componentes e dependências apenas de fontes oficiais, por meio de links seguros.

4.7 IDENTIFICATION AND AUTHENTICATION FAILURES

Segundo a IBM (2021), *Identificação* se resume à capacidade de constatar exclusivamente um usuário de um sistema ou um aplicativo em execução no sistema. *A autenticação* consiste na capacidade de provar que um usuário ou aplicativo é genuinamente quem essa pessoa ou aplicativo alega ser. Esta categoria trata-se das vulnerabilidades intrínsecas a estes dois componentes.

De acordo com a Figura 2, na seção 3, esta categoria decaiu da 2ª posição (anteriormente nomeada como *Broken Authentication* – Autenticação Quebrada no ranking 2017) para a 7ª posição no ranking 2021, ficando atrás apenas dos ataques de injeção. Sua alteração na nomenclatura diz respeito ao mapeamento de novas CWES relacionadas à falha de identificação, conforme mencionado pela OWASP (2021).

Formas de exploração

IMMUNIWEB (2021) realiza uma análise baseada na divisão dos três padrões de ataque que exploram a vulnerabilidade de Autenticação Fraca, segundo a OWASP (2021), que são: *Credential Stuffing*, *Brute Force Access* e *Session hijacking*.

1. **Credential Stuffing**: Consiste no uso de ferramentas automatizadas para testar uma lista de nomes de usuários e senhas válidos, roubados de uma aplicação, contra outra aplicação. Normalmente estas listas são obtidas também a partir da "Dark Web", que são nada menos que frutos de vazamentos de credenciais de usuários de outras empresas no passado, que consequentemente são utilizados para realização de novos ataques em novos alvos. IMMUNIWEB (2021) menciona que:

   > *"In 2017, researchers discovered a file on the dark web containing 1.4 billion compromised username and password combinations, in plain text format. These were compiled from numerous earlier breaches and made available for anyone to use."*

2. **Brute Force Access**: Ataques de força bruta podem ser definidos técnicamente como o procedimento de tentativa de todas as possibilidades de senhas e usuários diferentes até que ambos sejam encontrados. Em alguns casos, este método pode ser desnecessário, já que os invasores utilizam primeiramente listas de senhas mais comuns, para tentativa de acesso.

3. **Session hijacking**: Aplica-se no cenário onde há a exploração de uma sessão de um usuário legítimo autenticado. Após o usuário se autenticar, é gerado um ID ou token de sessão, onde este é armazenado em algum local na máquina do usuário (como cookie de sessão, Local Storage – armazenamento local de dados de um navegador Web - ou até mesmo anexado a URL do navegador). Posteriormente, quando este usuário efetuar o logout da sessão, o token ou ID de sessão deve ser removido também. Caso o mesmo não seja removido, este pode ser aproveitado por um invasor, permitindo-o "sequestrar" a sessão do usuário legítimo e executar qualquer ação permitida a este usuário.

Mitigação

Com base nas formas de prevenção sugeridas pela OWASP, é possível destacar os seguintes pontos:

- Implementação de autenticação multifatores, para evitar o autopreenchimento de credenciais, ataques de força bruta e ataques de reutilização de credenciais roubadas.
- Não utilização de credenciais padrão, especialmente para usuários administradores.
- Implementação de verificações de senha fracas, como também validar senhas baseadas em históricos do usuário, ou em relação a lista de senhas inseguras, como cita a GATEFY (2019) no ranking das 100 piores senhas de 2019.
- Mapeamento de complexidade e políticas de rotação de senha com as diretrizes do Instituto Nacional de Padrões e Tecnologia (NIST) 800-63b, cuja a seção 5.1 das Diretrizes de Identidade Digital publicada pela NIST (Atualizada em 2020) apresenta políticas de senha modernas, baseadas em evidências e regras de aceite. NIST (2019)
- Garantir que os PATHs das páginas de registro, recuperação de credencial e API sejam protegidos contra ataques de enumeração de contas usando as mesmas mensagens para todos os resultados. Segundo MICROSOFT (2021), um ataque de enumeração de contas consiste em um cenário cujo um invasor usa um dicionário com milhares de nomes de usuário, ou ferramentas para tentar adivinhar nomes de usuário no domínio. Este tipo de ataque pode ser ainda mais preciso no cenário conforme diz KASPERSKY (2021), onde, caso o hacker consiga verificar se determinado usuário está cadastrado na aplicação, bastará configurar um ataque de força bruta para encontrar a senha correspondente, economizando tempo e esforço (apud TIINSIDE, 2021).
- Limitar e atrasar cada vez mais as tentativas de login malsucedidas, porém é necessário cuidado para não criar um cenário de negação de serviço.
- Registro de todas as falhas e alerta aos administradores quando o enchimento de credenciais, força bruta ou outros ataques forem detectados.
- Utilização de um gerenciador de sessão integrado e seguro do lado do servidor que gera um novo token de sessão aleatório com alta entropia após o login. O identificador de sessão não deve estar no URL, e ser armazenado com segurança e invalidado após o logout.

4.8 SOFTWARE AND DATA INTEGRITY FAILURES

Com base na Figura 2, da seção 3, a categoria de desserialização insegura, apresentada na 8ª posição da última edição TOP 10, foi re-enquadrada nesta nova categoria e, segundo a OWASP (2021), esta trata-se de uma nova categoria, focada em realizar suposições relacionadas à falta de verificação de integridade durante atualizações de software, dados críticos e pipelines de CI / CD.

Segundo HKCERT (2021), este risco relaciona-se ao código ou infraestrutura não protegida contra violações de integridade, e pode ser considerado um dos mais impactantes. HKCERT (2021) complementa que um exemplo disso, é a violação da SolarWinds Orion, ocorrida em dezembro de 2020.

SolarWinds é uma empresa que produz softwares de soluções de gerenciamento de rede, sistemas, banco de dados e segurança de TI (SOLARWINDS, 2021), na qual segundo Turner (2020), um de seus produtos de gerenciamento de rede, chamado Orion, sofreu um ataque, cujo os agentes maliciosos incorporaram código malicioso ao software, adulterando-o antes que uma atualização fosse lançada para todos os seus usuários, o que resultou na implantação de malware para os clientes da SolarWinds.

Segundo AIQON (2020), os responsáveis por este ataque foram Hackers russos, pertencentes a um grupo de ameaça APT29 conhecido como Cozy Bear, no qual utilizaram o tipo de ataque conhecido como "supply chain attack", ou Ataque de cadeia de suprimentos.

Pode-se citar por exemplo o produto Orion da empresa SolarWinds. Com isso, após um malware instalado, os atacantes por vezes conseguem obter controle total sobre as redes do cliente do fornecedor.

Com base nisso, conclui-se que este ataque veio a acontecer devido à falta de um processo eficaz de verificação de integridade do código. HCKERT (2021) diz que este código malicioso foi entregue a mais de 18.000 organizações da SolarWinds, causando um dos ataques cibernéticos mais sérios até hoje.

Formas de exploração

Conforme citado, uma das principais formas de ataque dessa categoria pode ser nas atualizações de software automáticas. Segundo IMMUNIWEB (2021), atualmente muitas empresas aderem para suas aplicações, essa funcionalidade de atualização automática de software, o que pode levantar certas preocupações sobre a integridade dos dados durante o processo de atualização. IMMUNIWEB (2021) cita que um ataque MitM também pode ser uma forma de ataque, no qual se bem sucedido, o invasor pode enviar um código malicioso para o aplicativo durante o processo de atualização do software.

CWE-494 (2021), cita uma vertente desta forma de ataque descrita no último parágrafo, conforme abaixo:

> *"The product downloads source code or an executable from a remote location and executes the code without sufficiently verifying the origin and integrity of the code."*

Além do vetor de ataque citado acima, pode-se citar também o problema na desserialização de dados não confiáveis. CWE-502 (2021) cita esta vulnerabilidade como:

> *"The application deserializes untrusted data without sufficiently verifying that the resulting data will be valid. "*

Com base na citação acima, para fonte de informação, a desserialização e serialização, são mencionadas na seção 2.14 SERIALIZAÇÃO E DESSERIALIZAÇÃO, em conceitos básicos.

Dessa forma, a partir dos ataques de desserialização de dados não confiáveis, CWE-502 (2021), define algumas consequências para a integridade da aplicação, onde os invasores podem modificar objetos ou dados inesperados que foram considerados protegidos contra modificação, e também complementa sobre consequências para a disponibilidade da aplicação, como um cenário de uma função da aplicação que faz uma suposição sobre quando terminar, baseada no conteúdo de um atributo do tipo String. Esta função poderia facilmente nunca terminar, podendo causar graves problemas no desempenho da aplicação.

Mitigação

OWASP (2021) sugere algumas formas de mitigação para este risco:

- Utilização de assinaturas digitais ou mecanismos semelhantes para verificar se o software ou os dados são da fonte esperada e não foram alterados.
- Certificar-se das bibliotecas e dependências, como npm ou Maven. O Maven é uma ferramenta de construção e resolução de dependências popular para a linguagem Java, assim como o NPM é para a linguagem Javascript.
- Utilização de uma ferramenta de segurança da cadeia de suprimentos de software, como OWASP Dependency Check ou OWASP CycloneDX, seja usada para verificar se os componentes não contêm vulnerabilidades conhecidas.
- Revisão do código-fonte da aplicação, para evitar alterações de configuração indesejadas, ou introdução de código malicioso no software.
- Certificar-se que a pipeline de CI/CD tenha segregação, configuração e controle de acesso adequados, a fim de garantir a integridade do código que flui pelos processos de construção e implantação.
- Certificar-se que nenhum dado serializado ou não criptografado não seja enviado a clientes não confiáveis sem alguma forma de verificação de integridade ou assinatura digital para detectar adulteração ou repetição dos dados serializados.

4.9 SECURITY LOGGING AND MONITORING FAILURES

A 9ª posição do ranking OWASP TOP 10 (2021), a partir da Figura 2, seção 3, foi considerada a 5ª categoria da pesquisa da comunidade da última edição de 2017, anteriormente classificada como Insuficient Logging & Monitoring e agora denominada Security Logging and Monitoring Failures.

Conforme a OWASP (2021), a categoria foi expandida para incluir mais tipos de falhas, incluindo falhas que podem afetar diretamente a visibilidade, o alerta de incidentes e a perícia.

Segundo os dados disponibilizados pela OWASP (2021), esta categoria apresentou um total de 53.615 ocorrências para 53,67% das aplicações testadas, com 4 CWEs mapeadas, onde, com base na amplitude dos dados e fontes mapeadas, esta categoria visa ajudar a detectar, escalar e responder às violações ativas. Sem registro e monitoramento, as violações não podem ser detectadas. OWASP (2021)

O relatório sobre custos de violação de dados realizado pelo Instituto Ponemon e pela IBM (2021), aponta que o tempo médio para identificar e conter uma violação de dados é de aproximadamente 287 dias e complementa que, quanto mais tempo demora a identificação, mais cara será as consequências para a organização.

Este relatório citado no parágrafo acima, de acordo com a IBM, oferece percepções de 537 violações reais entre 17 países e regiões, e 17 empresas que providenciam taxas e médias globais, afim de ajudar a compreender os riscos cibernéticos em um mundo de mudança.

IMMUNIWEB (2021) aponta a extensão deste problema, de forma que se uma aplicação web e os incidentes do servidor forem monitorados incorretamente, atividades suspeitas podem facilmente passar despercebidas. E complementa que, um método de registro bem implementado, com alertas sempre que surgirem anomalias e um monitoramento diligente, permite que ações sejam mais rapidamente tomadas contra a exploração de vulnerabilidades.

Formas de exploração

OWASP (2021) descreve alguns dos principais pontos atrelados a esta categoria:

- Ausência de registros em eventos auditáveis, logins com falhas e transações de alto valor, ou se existentes, armazenados apenas localmente.
- Avisos e erros geram mensagens de log inexistentes, inadequadas ou pouco claras.
- Logs de aplicativos e APIs não são monitorados para atividades suspeitas.
- Analisadores de vulnerabilidades e varreduras por ferramentas de teste de segurança de aplicativo dinâmico não acionam alertas.

- Não detecção ou alerta para ataques ativos em tempo real ou quase em tempo real.

Mitigação

OWASP (2021) orienta que os desenvolvedores implementem alguns ou todos os controles a seguir, a depender do risco do aplicativo:

- Todas as falhas de login, controle de acesso e validações de entrada do lado do servidor, devem ser registradas para identificação de contas suspeitas ou maliciosas e retidas por tempo suficiente para permitir análise posterior.
- Geração de logs em um formato flexível, para que as soluções de gerenciamento dos logs possam ser acessadas e consumidas facilmente. IMMUNIWEB (2021) diz que o melhor uso dos logs é a revisão pós-evento e descoberta das áreas da aplicação que foram comprometidas.
- Os dados de registro devem ser codificados corretamente para evitar injeções ou ataques nos sistemas de registro ou monitoramento.
- As transações de alto valor devem possuir uma trilha de auditoria com controles de integridade, para evitar adulteração ou exclusão, como o uso de tabelas de banco de dados limitadas somente à inserção de dados.
- Estabelecimento de monitoramento e alertas eficazes para que atividades suspeitas sejam detectadas e respondidas rapidamente. Normalmente estas ações são realizadas pelas equipes de operações de segurança da informação.
- Estabelecimento e adoção de planos de resposta e recuperação de incidentes, como são apontados pelo o Instituto Nacional de Padrões e Tecnologia (NIST) 800-61r2 ou similares. (NIST, 2012)

Além dos pontos mencionados pela OWASP (2021), IMMUNIWEB (2021), apresenta outras abordagens, na qual sugere que uma das boas práticas de testar riscos de registros inadequados é a partir da utilização de um Pentest, que investigará e tentará violar a aplicação, de forma simulada – como já descrito na seção 2.1.

A respeito dos registros de logs, IMMUNIWEB (2021), cita que sua utilização com criptografia seria a melhor alternativa, porém pode ser um processo caro em termos de desempenho e desenvolvimento.

4.10 SERVER-SIDE REQUEST FORGERY (SSRF)

Na última posição do ranking TOP 10, de acordo com a Figura 2 na seção 3, encontra-se a categoria "Server-Side Request Forgery (SSRF)" ou Falsificação de solicitação do lado do servidor. Conforme a

OWASP (2021), esta é uma nova categoria adicionada a partir da pesquisa da comunidade Top 10, ocupando o primeiro lugar no levantamento realizado pelos especialistas.

De acordo com os dados disponibilizados pela OWASP (2021), de 75% das aplicações testadas para este risco, foi apresentada uma taxa máxima de 2%. Dessa forma, é perceptível a sua baixa taxa de incidência, na qual, embora não esteja ilustrado nos dados, a OWASP (2021) diz que esta categoria representa um cenário indicado pela comunidade de segurança, por ser um risco importante a ser abordado.

Para PORTSWIGGER (2021) um ataque SSRF bem-sucedido pode muitas vezes resultar em ações não autorizadas, acesso a dados dentro da organização ou quaisquer ataques mal-intencionados que levariam os analistas a entender que o problema se origina da organização que hospeda o aplicativo vulnerável. Seja na própria aplicação vulnerável ou em outros sistemas Back-end com os quais a aplicação atacada pode se comunicar.

Segundo IMMUNIWEB (2021), as falsificações de solicitações a partir do servidor vieram a se tornar uma das vulnerabilidades mais discutidas em 2021, devido a grandes danos causados por agentes APT, e por ransomwares.

Pode-se citar um caso de ataque cibernético semelhante que ocorreu com a Microsoft, em março de 2021. Segundo Mackie (2021), os servidores Exchange da empresa – um produto Microsoft, que fornece servidores de e-mail e calendário a pequenas e médias empresas SPRINGPEOPLE (2018) – teriam sofrido um ataque de um ransomware, afetando um total de 400.000 servidores conectados à Internet.

Segundo a SBARAGLIA (2021), após as falhas terem sido descobertas, a Microsoft implementou rapidamente patches de segurança, mas até que as atualizações fossem instaladas, muitas empresas ainda se mantiveram vulneráveis.

Uma análise realizada pela UNIT 42 (2021) apresenta uma visão geral de forma ilustrada sobre a exploração das vulnerabilidades do Microsoft Exchange Server. Segundo informa a empresa, foram mapeadas um total de 4 vulnerabilidades encadeadas, que são:

- CVE-2021-26855
- CVE-2021-26857
- CVE-2021-26858
- CVE-2021-27065

UNIT 42 (2021), descreve o fluxo seguido pelos agentes maliciosos baseados nas CVE's citadas acima, da seguinte forma:

> *"For this attack to be successful, the adversary would first need to identify an on-premises Microsoft Exchange Server that is able to receive untrusted connections from an external*

source on port 443. If an adversary is successful in securing a connection, they can then exploit CVE-2021-26855 to authenticate themselves as a Microsoft Exchange server. This can be followed by the exploitation of CVE-2021-26857, CVE-2021-26858 and CVE-2021-27065 post-authentication, allowing the adversary to gain remote access."

A análise fornecida pela empresa, complementa que se o acesso for obtido, a execução de comandos remotos, ou o upload de um Web Shell – por exemplo Chopper China - que permitiria ao invasor roubar dados e executar ações maliciosas, como baixar arquivos remotos.

A Figura 3 a seguir, apresenta de maneira ilustrada os atores de ameaça responsáveis pela violação e suas atividades descritas:

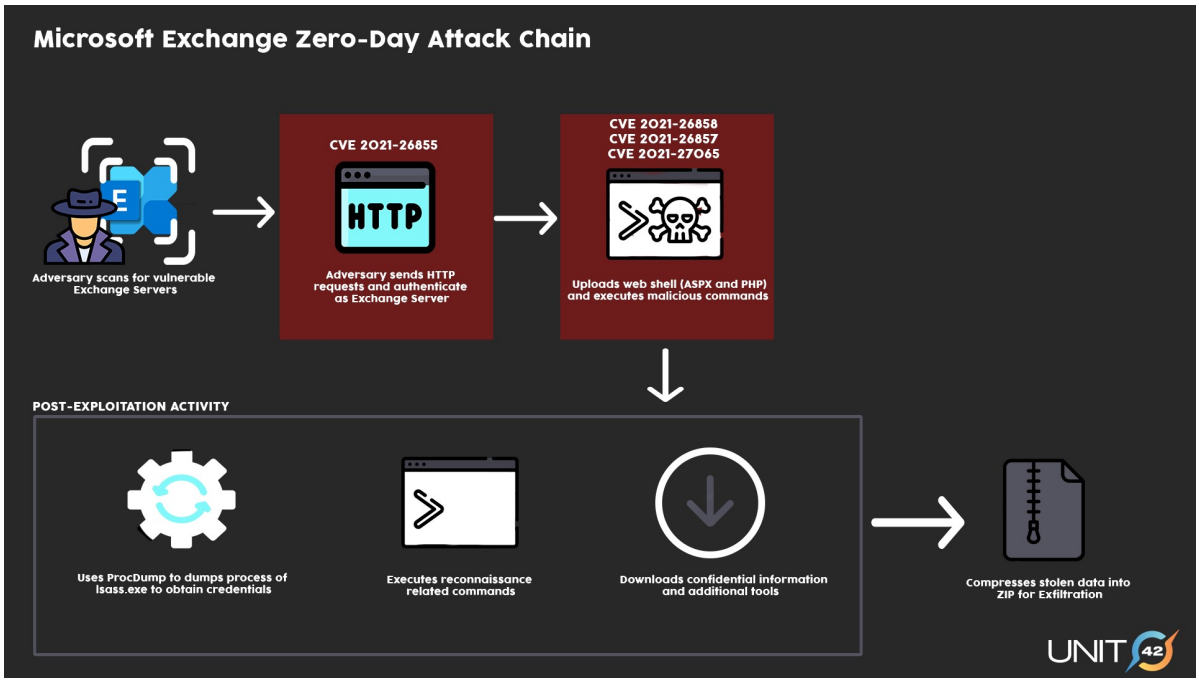

Figura 3. "*Threat actors can chain together four zero-day vulnerabilities to gain unauthorized access to Microsoft Exchange Servers."*

Fonte: UNIT 42 (2021)

Em complemento com a análise fornecida pela UNIT 42 (2021), analisando a Figura 3, o quadro que apresenta *POST-EXPLORATION ACTIVITY*, consiste em apresentar algumas das atividades pós-exploratórias a serem realizadas (também as mais comuns). Conforme apresenta abaixo do ícone de engrenagem, um *Procdump*, teria a função de despejar a memória do processo LSASS (Local Security Authority Subsystem Service - Serviço de Subsistema de Autoridade de Segurança Local), para obter credenciais. Dessa forma, após a obtenção das credenciais, segundo a UNIT 42 (2021), os invasores costumam comprimir esses arquivos de

dados sensíveis em utilitários de compactação (por exemplo: 7zip ou WinRar), para posteriormente exfiltrar estes dados roubados.

A partir disso, é possível apontar que, a primeira e talvez principal vulnerabilidade explorada na violação contra os servidores da Microsoft Exchange, mencionada nos parágrafos acima, foi a CVE-2021-26855, que segundo a GRUNZWEIG e col. (2021), a mesma pode ser descrita como uma vulnerabilidade de falsificação de solicitação do lado do servidor ou *SERVER-SIDE REQUEST FORGERY (SSRF)*, que pode ser executada remotamente, sem nenhuma tipo de autenticação.

Formas de Exploração:

- Comprometimento de serviços internos, no qual o invasor pode abusar dos serviços internos para conduzir outros ataques, como execução remota de código ou negação de serviço (DoS). OWASP (2021);
- Filtragem ausente ou insuficiente da entrada fornecida por um invasor, que posteriormente é usada pelo aplicativo para iniciar uma conexão com uma aplicação terceira. (IMMUNIWEB, 2018);
- Segundo POSTSWIGGER, algumas aplicações transmitem dados em formatos cuja especificação permite a inclusão de URLs, que podem ser solicitados pelo analisador de dados para o formato. Um exemplo disso é a partir da transmissão de dados estruturados do cliente para o servidor no formato de dados estruturados XML. Dessa forma, conforme abordada na seção 3.5 (SECURITY MISSCONFIGURATION), quando uma aplicação aceita e realiza a leitura de dados no formato XML, pode ser vulnerável à injeção XXE (XML External Entity attack) que, por sua vez, passa a ser vulnerável à SSRF via XXE;
- A partir de ataques de Phising, o SSRF pode ser usado em conjunto com redirecionamentos HTTP para redirecionar a vítima a uma página da web maliciosa. Isso pode ser usado para servir malware, afim de colher credenciais (IMMUNIWEB, 2018).

Mitigação

OWASP (2021), sugere algumas formas de prevenção de SSRF, implementado alguns ou todos os controles de defesa que serão citados, conforme abaixo:

- **Da camada de rede:**

1. Segmentação de funcionalidades de acesso a recursos remotos em redes separadas para reduzir o impacto de SSRF.

2. Implementação de políticas de firewall, como “negar por padrão” ou regras de controle de acesso à rede para bloquear todo o tráfego da intranet, exceto o essencial. Segundo Dutran, Chrispim e Ferreira (2019), *negar por Padrão* é uma política conhecida por bloquear todo tipo de dado, seja ele chegando ou saindo da rede, que não está explicitamente permitido pelas regras. A mesma reduz drasticamente a chance de haver ataques, e é uma política considerada mais segura do que permitir que todo tráfego que não está explicitamente proibido passe pelo Firewall.
3. Estabelecimento de uma propriedade e um ciclo de vida para regras de firewall, baseadas em aplicativos.
4. Registrar todos os fluxos de rede aceitos e bloqueados em firewalls (Conforme citado na seção 3.9 SECURITY LOGGING AND MONITORING FAILURES).

- **Da camada de aplicativo:**

1. Limpeza e validação de todos os dados de entrada fornecidos pelo cliente.
2. Implementação do esquema de URL, porta e destino, com uma lista de permissões positiva.
3. Não enviar respostas brutas aos clientes, sempre realizar o refinamento de seu conteúdo.
4. Desativar redirecionamentos HTTP.

5. CONSIDERAÇÕES FINAIS

A partir da análise apresentada, é notável os cenários de vulnerabilidades interligadas entre si, ou mesmo o agrupamento de mais de uma vulnerabilidade sendo explorada simultaneamente ou não. Quanto mais brechas encontradas em uma aplicação, maiores podem ser os impactos e mais difícil será sua mitigação. Dessa forma, este trabalho buscou conduzir cada risco citado, apresentando inicialmente fatores históricos característicos de cada um, e em seguida, partindo para o ponto teórico de exploração e mitigação.

Dentre os riscos expostos, sua abrangência consiste numa abordagem preventiva, porém buscando uma condução ampla de cenários de ciberataques, para que além das tratativas diretas de fragilidades sejam realizadas, as mais viáveis maneiras de investigação e diagnóstico de problemas de segurança sejam implementadas.

O assunto exposto neste trabalho é de grande valia para qualquer profissional de TI, sem ressalvas, e, portanto, espera-se que os dados e as informações expostas contribuam para esclarecer as principais vulnerabilidades em aplicações Web, inteiradas em 2021 e baseadas em uma metodologia de reconhecimento internacional.

Para uma pesquisa futura, voltada ao tema abordado, podem ser exploradas na prática as diversas formas de ataque citadas para cada vulnerabilidade, a partir de ferramentas de testes automatizadas, a fim de relatar e relacionar as fragilidades encontradas, com as indicadas pelo projeto "*OWASP Top 10*".

6. REFERÊNCIAS BIBLIOGRÁFICAS

AIQON. **Ataque à cadeia de suprimentos da SolarWinds**. 2020. Disponível em: < https://aiqon.com.br/blog/ataque-a-cadeia-de-suprimentos-da-solarwinds/>. Acesso em: 13 de nov. 2021.

AUTH0. **JSON Web Tokens**. 2021. Disponível em: <https://auth0.com/docs/security/tokens/json-web-tokens>. Acesso em: 24 de nov. 2021.

AWS. **Network ACLs**. 2021. Disponível em: < https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>. Acesso em: 25 de nov. 2021.

BATTISTI, Julio. **Windows 7: Lição 156 - Capítulo 06 - Diretivas Locais de Segurança - Conceito**. 2020. Disponível em: <https://www.juliobattisti.com.br/artigos/windows7/capitulo06/cap06_19.asp>. Acesso em: 25 de nov. 2021.

BLOG DA CCM. **Teste de intrusão: o que é, a importância e como utilizar**. 2019. Disponível em: <https://blog.ccmtecnologia.com.br/post/teste-de-intrusao-o-que-e-a-importancia-e-como-utilizar >. Acesso em: 16 de out. 2021.

Cadu. **ORM: Object Relational Mapper**. DEVMEDIA. 2011. Disponível em: < https://www.devmedia.com.br/orm-object-relational-mapper/19056 >. Acesso em: 05 de nov. 2021.

CINTO, Nícholas Antunes. **Teste de Vulnerabilidades em Aplicações Web**. TCC - Instituto Municipal Educacional do Município de Assis – IMESA e à Fundação Educacional do Município de Assis – FEMA. Assis, p. 63. 2015.

COSTA, Matheus Bigogno. **O que é DoS e DDoS?** CANALTECH. 2014. Disponível em: <https://canaltech.com.br/produtos/o-que-e-dos-e-ddos/>. Acesso em: 25 de nov. 2021.aut

CWE-494. **CWE-494: Download of Code Without Integrity Check**. 2021. Disponível em: <https://cwe.mitre.org/data/definitions/494.html/>. Acesso em: 14 de nov. 2021.

CWE-502. **CWE-502: Deserialization of Untrusted Data**. 2021. Disponível em: < https://cwe.mitre.org/data/definitions/502.html/>. Acesso em: 14 de nov. 2021.

DEVOPEDIA. **Data Serialization**. 2020. Disponível em: <https://devopedia.org/data-serialization/>. Acesso em: 25 de nov. 2021.aut

DUTRA, Renan; NATHALIA, Chrispim; WESLLEY, Ferreira. DEL POLI UFRJ. 2021. **Firewall: Tecnologias**. Disponível em: < https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/firewall/tecnologias.html/>. Acesso em: 20 de nov. 2021

EDUCALINGO. **Downgrade**. 2021. Disponível em: < https://educalingo.com/pt/dic-en/downgrade>. Acesso em: 01 de nov. 2021.

FC BRASIL. **O que é patch e para que serve esse programa?** 2020. Disponível em: < https://www.fcbrasil.com.br/blog-fcb/133-o-que-e-gerenciamento-de-patch-e-como-ele-funciona>. Acesso em: 25 de nov. 2021.

GALLAGHER, James. **SQL Limit: A Beginner's Guide**. CARRER KARMA. 2020. Disponível em: <https://careerkarma.com/blog/sql-limit//>. Acesso em: 05 de nov. 2021.

GATEFY. **Confira a lista das piores senhas de 2019**. 2019. Disponível em: < https://gatefy.com/pt-br/blog/confira-lista-piores-senhas-2019/>. Acesso em: 09 de nov. 2021.

GCSEC. **Ataques à cadeia de fornecimento: o que são?** 2021. Disponível em: < https://gcsec.com.br/ataques-a-cadeia-de-fornecimento-o-que-sao/index.html/>. Acesso em: 14 de nov. 2021.

GOLDMAN, Jeff. Kroger. **Wendy's, Kiddicare Suffer Data Breaches**. eSecurity Planet. 2016. Disponível em: <https://www.esecurityplanet.com/networks/kroger-wendys-kiddicare-suffer-data-breaches/>. Acesso em: 27 de out. 2021.

GRUNZWEIG, Josh; MATTHEW, Meltzer; SEAN. Koessel, Steven Adair, Thomas Lancaster. VOLEXITY. 2021. **Operation Exchange Marauder: Active Exploitation of Multiple Zero-Day Microsoft Exchange Vulnerabilities**. Disponível em: < https://www.volexity.com/blog/2021/03/02/active-exploitation-of-microsoft-exchange-zero-day-vulnerabilities/>. Acesso em: 20 de nov. 2021

HACKSPLANNING. **Ensuring Proper Access Control**. 2021. Disponível em: < https://www.hacksplaining.com/prevention/broken-access-control>. Acesso em: 28 de out. 2021.

HKCERT. **OWASP Top 10-2021 is Now Released.** 2021. Disponível em: < https://www.hkcert.org/blog/owasp-top-10-2021-is-now-released>. Acesso em: 24 de out. 2021.

HOLMWOOD, Lindsay. **Cryptographic Failures is now #2 on the OWASP Top 10**. CIPHERSTASH. Disponível em < https://cipherstash.com/blog/2021-10-13-owasp-top-10-number-2-cryptographic-failures/>. Acesso em: 01 de nov. 2021.

HOSTMIDIA. **O que é MFA?** Por que é importante? 2021. Disponível em: <https://www.hostmidia.com.br/blog/o-que-e-mfa/>. Acesso em: 26 de nov. 2021.aut

IBM. **Security concepts and mechanisms: identification and authentication**. 2021. Disponível em: < https://www.ibm.com/docs/en/ibm-mq/9.0?topic=mechanisms-identification-authentication/>. Acesso em: 09 de nov. 2021.

IBM; PONEMON. **How much does a data breach cost? Cost of a Data Breach Report 2021 explores ways to help mitigate risk**. 2021. Disponível em: <https://www.ibm.com/security/data-breach/>. Acesso em: 15 de nov. 2021

IDX. **The Kroger/Equifax W-2 Breach: What Can We Learn from It?** 2016. Disponível em: < https://www.idx.us/knowledge-center/the-kroger-equifax-w-2-breach-what-can-we-learn-from-it/>. Acesso em: 28 de out. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Cryptographic Failures Practical Overview**. 2021. Disponível em: <https://www.immuniweb.com/blog/OWASP-cryptographic-failures.html/>. Acesso em: 01 de nov. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Identification and**

Authentication Failures Practical Overview. 2021. Disponível em: < https://www.immuniweb.com/blog/OWASP-identification-and-authentication-failures.html/>. Acesso em: 09 de nov. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Injection Practical Overview**. 2021. Disponível em: <https://www.immuniweb.com/blog/OWASP-injection.html/>. Acesso em: 05 de nov. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Insecure Design Practical Overview**. 2021. Disponível em: <https://www.immuniweb.com/blog/OWASP-insecure-design.html/>. Acesso em: 05 de nov. 2021.

IMMUNIWEB. **Monitoring Failures Practical Overview**. 2021. Disponível em: <https://www.immuniweb.com/blog/OWASP-security-logging-and-monitoring-failures.html/>. Acesso em: 15 de nov. 2021

IMMUNIWEB. **OWASP Top 10 in 2021: Security Logging and**

Monitoring Failures Practical Overview. 2021. Disponível em: <https://www.immuniweb.com/blog/OWASP-security-logging-and-monitoring-failures.html/>. Acesso em: 15 de nov. 2021

IMMUNIWEB. **OWASP Top 10 in 2021: Security Misconfiguration Practical Overview**. 2021. Disponível em: < https://www.immuniweb.com/blog/OWASP-security-misconfiguration.html/>. Acesso em: 06 de nov. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Server-Side Request Forgery**

(SSRF) Practical Overview. 2021. Disponível em: < https://www.immuniweb.com/blog/OWASP-server-side-request-forgery-ssrf.html/>. Acesso em: 17 de nov. 2021

IMMUNIWEB. **OWASP Top 10 in 2021: Software and Data Integrity**

Failures Practical Overview. 2021. Disponível em: < https://www.immuniweb.com/blog/OWASP-software-and-data-integrity-failures.html/>. Acesso em: 14 de nov. 2021.

IMMUNIWEB. **OWASP Top 10 in 2021: Vulnerable and Outdated**

Components Practical Overview. 2021. Disponível em: < https://www.immuniweb.com/blog/OWASP-vulnerable-and-outdated-components.html/>. Acesso em: 08 de nov. 2021.

IMMUNIWEB. **Server-Side Request Forgery [CWE-918].** 2018. Disponível em: <https://www.immuniweb.com/vulnerability/ssrf.html/>. Acesso em: 17 de nov. 2021

IVANOVA, Irina. **Equifax ex-CEO: Hacked data wasn't encrypted**. CBS NEWS. 2017. Disponível em <https://www.cbsnews.com/news/equifax-ex-ceo-hacked-data-wasnt-encrypted/>. Acesso em: 01 de nov. 2021.

KASPERSKY. **O que é uma ameaça persistente avançada (APT)?** 2021.Disponível em: <https://www.kaspersky.com.br/resource-center/definitions/advanced-persistent-threats>. Acesso em: 26 de nov. 2021.

KASPERSKY. **Ransomware: definição, prevenção e remoção**. 2021.Disponível em: <https://www.kaspersky.com.br/resource-center/threats/ransomware>. Acesso em: 26 de nov. 2021.

KirstenS. **Cross Site Scripting (XSS).** OWASP. 2021. Disponível em: < https://owasp.org/www-community/attacks/xss/>. Acesso em: 03 de nov. 2021.

KREBS, Brian. **Crooks Grab W-2s from Credit Bureau Equifax**. KrebsonSecurity. 2016. Disponível em: <https://krebsonsecurity.com/2016/05/crooks-grab-w-2s-from-credit-bureau-equifax/>. Acesso em: 27 de out. 2021.

LIMA, Luis Felipe de. **Teste de intrusão para aplicações web: um método com planejamento em inteligência artificial**. Tese (Mestrado em Informática no Programa de PósGraduação em Informática, Setor de Ciências Exatas) - Universidade Federal do Paraná. Curitiba, p. 111. 2020.

MACHADO, Marcel Jacques. **Segurança da Informação: uma Visão Geral sobre as Soluções Adotadas em Ambientes Organizacionais**. Curitiba: UFPR, 2012. Trabalho de Graduação – Bacharelado em Ciência da Computação, Universidade Federal do Paraná, Curitiba, 2012.

MACKIE, Kurt. Microsoft Releases Out-of-Band Security Patches for Exchange Server. Redmond. 2021. Disponível em: < https://redmondmag.com/articles/2021/03/02/exchange-server-zero-day-patches.aspx/>. Acesso em: 18 de nov. 2021

MALENKOVICH, Serge. **O que é um Ataque Man-in-the-Middle?** Kaspersky. 2013. Disponível em: <https://www.kaspersky.com.br/blog/what-is-a-man-in-the-middle-attack/462/>. Acesso em: 26 de nov. 2021.

MESSMER, Ellen. **Heartland: 'Largest Data Breach Ever'**. CSO Online. 2009. Disponível em: <https://www.csoonline.com/article/2123599/heartland---largest-data-breach-ever-.html/>. Acesso em: 05 de nov. 2021.

MICROSOFT. **Supply chain attacks**. 2021. Disponível em: <https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/supply-chain-malware>. Acesso em: 26 de nov. 2021.

MICROSOFT. **Tutorial: Alertas de reconhecimento**. 2021. Disponível em: < https://docs.microsoft.com/pt-br/defender-for-identity/reconnaissance-alerts/>. Acesso em: 10 de nov. 2021.

MITRE. **Common Weakness Enumeration**. 2021. Disponível em: <https://cwe.mitre.org/index.html>. Acesso em: 24 de nov. 2021.

MOZILLA. **Cross-Origin Resource Sharing (CORS)**. 2021. Disponível em: < https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>. Acesso em: 24 de nov. 2021.

MOZILLA. **HTTP request methods**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>. Acesso em: 24 de nov. 2021.

MXM SISTEMAS. **DevSecOps e DevOps: entenda as diferenças**. 2020. Disponível em: <https://www.mxm.com.br/blog/devsecops-devops-diferenca/>. Acesso em: 26 de nov. 2021.

NAYAK, Amukta. **OWASP Top 10 Deep Dive: Getting a Clear View on Vulnerable and Outdated Components**. Rapid7. 2021. Disponível em: < https://www.rapid7.com/blog/post/2021/11/08/owasp-top-10-deep-dive-getting-a-clear-view-on-vulnerable-and-outdated-components/>. Acesso em: 08 de nov. 2021.

NAZIRIDIS, Nick. **Perguntas frequentes: ataques de rede e problemas de segurança**. SSL.com. 2021. Disponível em < https://www.ssl.com/pt/faqs/ataques-de-rede-e-problemas-de-seguran%C3%A7a/>. Acesso em: 01 de nov. 2021.

NIST. **Computer Security Incident Handling Guide: Recommendations of the National Institute of Standards and Technology**. 2012. Disponível em: <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf/>. Acesso em: 16 de nov. 2021.

NIST. **Digital Identity Guidelines Authentication and Lifecycle Management: 5.1.1 Memorized Secrets**. 2019. Disponível em: < https://pages.nist.gov/800-63-3/sp800-63b.html#memsecret/>. Acesso em: 09 de nov. 2021.

NTT. **Global Threat Intelligence Report Executive Guide**. 2021. Disponível em < https://hello.global.ntt/en-us/insights/2021-global-threat-intelligence-report>. Acesso em 14 set. 2021.

OWASP. **A01:2021 – Broken Access Control**. 2021. Disponível em: < https://owasp.org/Top10/A01_2021-Broken_Access_Control/> Acesso em: 27 de out. 2021.

OWASP. **A02:2021 – Cryptographic Failures**. 2021. Disponível em: < https://owasp.org/Top10/A02_2021-Cryptographic_Failures//> Acesso em: 01 de nov. 2021.

OWASP. A03:2021 – Injeção. 2021. Disponível em: < https://owasp.org/Top10/pt_BR/A03_2021-Injection/>. Acesso em: 03 de nov. 2021.

OWASP. **A04:2021 – Insecure Design**. 2021. Disponível em: < https://owasp.org/Top10/A04_2021-Insecure_Design/>. Acesso em: 05 de nov. 2021.

OWASP. **A05:2021 – Security Misconfiguration**. 2021. Disponível em: < https://owasp.org/Top10/A05_2021-Security_Misconfiguration/>. Acesso em: 06 de nov. 2021.

OWASP. **A06:2021 – Vulnerable and Outdated Components**. 2021. Disponível em: < https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/>. Acesso em: 08 de nov. 2021.

OWASP. **A07:2021 – Identification and Authentication Failures**. 2021. Disponível em: < https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/>. Acesso em: 09 de nov. 2021.

OWASP. **A08:2021 – Software and Data Integrity Failures**. 2021. Disponível em: < https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures//>. Acesso em: 13 de nov. 2021.

OWASP. **A09:2021 – Security Logging and Monitoring Failures**. 2021. Disponível em: < https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/>. Acesso em: 15 de nov. 2021.

OWASP. **A10:2021 – Server-Side Request Forgery (SSRF)**. 2021. Disponível em: <https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/>. Acesso em: 16 de nov. 2021.

OWASP. **A9:2017-Using Components with Known Vulnerabilities**. 2017. Disponível em: <https://owasp.org/www-project-top-ten/2017/A9_2017-Using_Components_with_Known_Vulnerabilities/>. Acesso em: 08 de nov. 2021.

OWASP. **About the OWASP Foundation. 2021**. Disponível em: < https://owasp.org/about/>. Acesso em: 24 de out. 2021.

OWASP. **OWASP Top 10 – 2017: The Ten Most Critical Web Application Security Risks**. 2017. Disponível em: < https://wiki.owasp.org/images/0/06/OWASP_Top_10-2017-pt_pt.pdf>. Acesso em: 24 de out. 2021.

OWASP. **OWASP Top Ten. 2021**. Disponível em: <https://owasp.org/www-project-top-ten/>. Acesso em: 24 de out. 2021.

OWASP. **Top Ten Data Driven (partially)**. 2021. Disponível em: < https://www.owasptopten.org/thedata/>. Acesso em: 24 de out. 2021.

OWASP. **Top Ten Survey. 2021**. Disponível em: < https://www.owasptopten.org/thesurvey/>. Acesso em: 24 de out. 2021.

PHISHING. **What Is Phishing?** [2017]. Disponível em: < https://www.phishing.org/what-is-phishing/>. Acesso em: 26 de nov. 2021.aut

PINTO, Juliane Borsato Beckedorff; COSTA, Lucas da Silva; GODOY, Leonardo Buck. **Segurança em Aplicações Web: Um estudo do SQL Injection**. TCC - Faculdade de Tecnologia de Americana. Americana, p. 16. 2019.

PODJARNY, Guy. **Which of the OWASP Top 10 Caused the World's Biggest Data Breaches?** Snyk.io. 2017. Disponível em: <https://snyk.io/blog/owasp-top-10-breaches/>. Acesso em: 27 de out. 2021.

PORTSWIGGER. **Server-side request forgery (SSRF). 2021**. Disponível em: < https://portswigger.net/web-security/ssrf/>. Acesso em: 16 de nov. 2021.

PTES. **Main Page: High Level Organization of the Standard**. 2014. Disponível em: <http://www.pentest-standard.org/index.php/Main_Page>. Acesso em: 20 de out. 2021.

RED HAT. **What is a CVE?** 2020. Disponível em: < https://www.redhat.com/en/topics/security/what-is-cve>. Acesso em: 24 de nov. 2021.

REDHAT. **What is a CI/CD pipeline?** 2019. Disponível em: <https://www.redhat.com/en/topics/devops/what-cicd-pipeline>. Acesso em: 26 de nov. 2021.

RIGUES, Rafael. **Facebook atribui recente vazamento de dados de usuários a "scraping". Olhar Digital**. 2021. Disponível em: <https://olhardigital.com.br/2021/04/07/seguranca/facebook-atribui-recente-vazamento-de-dados-de-usuarios-a-scraping/>. Acesso em: 06 de nov. 2021.

SANCHES, Alan. Metodologias de Testes de Intrusao Pentest. Esecurity. 2019. Disponível em: < https://esecurity.com.br/metodologias-de-testes-de-intrusao-pentest/ >. Acesso em: 20 de out. 2021.

SBARAGLIA, Giorgio. **The main cyber attacks of the first semester of 2021 targeting companies and organizations. Flashstart**. 2021. Disponível em: < https://flashstart.com/the-main-cyber-attacks-of-the-first-semester-of-2021-targeting-companies-and-organisations/>. Acesso em: 19 de nov. 2021

SEGOVIA, Antonio Jose. **How to use penetration testing for ISO 27001 A.12.6.1**. Advisera. 2016. Disponível em: <https://advisera.com/27001academy/blog/2016/01/18/how-to-use-penetration-testing-for-iso-27001-a-12-6-1/ >. Acesso em: 17 de out. 2021.

SIEMBA. **OWASP Top 10: Broken Access Control. 2021**. Disponível em: < https://www.siemba.io/post/owasp-top-10-broken-access-control>. Acesso em: 27 de out. 2021.

SOBERS, Rob. **What is OAuth? Definition and How it Works**. VARONIS. 2018. Disponível em: <https://www.varonis.com/blog/what-is-oauth/>. Acesso em: 25 de nov. 2021.

SOLARWINDS. Solar Winds. 2021. Disponível em: https://www.solarwinds.com/pt/>. Acesso em: 13 de nov. 2021.

SPRINGPEOPLE. **Know What Is An Exchange Server And How It Works**. 2018. Disponível em: < https://www.springpeople.com/blog/what-is-an-exchange-server-and-how-it-works/>. Acesso em: 17 de nov. 2021

TECHMONITOR. **SQL Injection Attacks on the Rise, As Gaming Industry Under Attack from Credential Stuffing**. 2019. Disponível em: <https://techmonitor.ai/techonology/cybersecurity/sql-injection-attacks/>. Acesso em: 03 de nov. 2021.

TIINSIDE. **Um terço das empresas sofreram com ataques por causa de senhas e políticas fracas de segurança, diz Kaspersky**. 2021. Disponível em: < https://tiinside.com.br/16/08/2021/um-terco-das-empresas-sofreram-com-ataques-por-causa-de-senhas-e-politicas-fracas-de-seguranca-diz-kaspersky/>. Acesso em: 10 de nov. 2021.

TURNER, Jeremy. **December 2020 SolarWinds breach: What you need to know**. Coalitioninc. 2020. Disponível em: < https://www.coalitioninc.com/blog/december-2020-solarwinds-breach-what-you-need-to-know/>. Acesso em: 13 de nov. 2021

UCHILL, Joe. **Google pitches security standards for 'critical' open-source projects**. SC Media. 2021. Disponível em: < https://www.scmagazine.com/news/security-news/google-pitches-security-standards-for-critical-open-source-projects/>. Acesso em: 09 de nov. 2021

UNIT 42. **Palo Alto Networks. Threat Assessment: Active Exploitation of Four Zero-Day Vulnerabilities in Microsoft Exchange Server.** 2021. Disponível em: < https://unit42.paloaltonetworks.com/microsoft-exchange-server-vulnerabilities//>. Acesso em: 19 de nov. 2021

WALLARM. **A1: Injection 2017 OWASP**. 2021. Disponível em: <https://www.wallarm.com/what/a1-injection-2017-owasp>. Acesso em: 03 de nov. 2021.

7. AGRADECIMENTOS

Primeiramente a Deus, que assim como em todos momentos desafiadores da vida, foi refúgio e esperança. A toda minha família por todo o apoio, incentivo e base fundamental para todos os meus objetivos de vida. A todos os professores do curso de Ciência da Computação, por todo o conteúdo lecionado, juntamente com todo o conhecimento de voz experiente transmitido, em especial ao prof. Dr. Carlos Eduardo Câmara, pelas aulas e pela orientação para a execução deste trabalho. À meus amigos de sala, pela parceria de todos esses anos e ao Centro Universitário Padre Anchieta, pela estrutura fornecida.