INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Avaliação de segurança dos websites das universidades Angolanas

Emanuel Agostinho Sebastião Mateus

Mestrado em Software de Código Aberto

Orientador:
Doutor Carlos José Corredoura Serrão, Professor Associado
ISCTE-IUL

Setembro, 2020

iscte

TECNOLOGIAS
E ARQUITETURA

Avaliação de segurança dos websites das universidades Angolanas

Emanuel Agostinho Sebastião Mateus

Mestrado em Software de Código Aberto

Orientador:
Doutor Carlos José Corredoura Serrão, Professor Associado
ISCTE-IUL

Setembro, 2020

Agradecimentos

A Deus por me amparar nos momentos difíceis, me dar força interior para superar as dificuldades, mostrar o caminho nas horas incertas e me suprir em todas as minhas necessidades.

Ao meu orientador, Professor Doutor Carlos José Corredoura Serrão, agradeço toda a sua orientação, as críticas construtivas e disponibilidade que permitiram a realização desta pesquisa de modos a adquirir novos conhecimentos e, muito particularmente, pela sua ínfima paciência.

Às Instituições envolvidas neste estudo pela disponibilidade na recolha dos dados.
A todos os Professores do mestrado, que contribuíram para o meu aperfeiçoamento académico.
A minha família, a qual amo muito, pelo carinho, paciência e incentivo.

A todos os que enumerei o meu sincero "Obrigado".

Resumo

A avaliação das vulnerabilidades constitui um dos procedimentos técnicos que podem ajudar a prevenir falhas graves de segurança, que quando explorados podem colocar em causa a credibilidade da marca e ou a continuidade de um determinado negócio. As universidades angolanas utilizam os seus *websites* como montra digital e portal de acesso aos serviços e aplicações que estão disponíveis na Internet 24/7, pelo que estando expostas a todo o tipo de riscos que dela podem advir.

O presente trabalho tem por objetivo avaliar o estado da segurança dos *websites* das instituições de ensino superior angolanas, e mais especificamente, identificar as vulnerabilidades mais frequentes existentes nos mesmos e quantificar o grau de criticidade das mesmas. Para alcançar estes objetivos, este trabalho apoia-se em metodologias que identificam as falhas mais frequentes em aplicações web e utiliza ferramentas automáticas para efetivamente descobrir e validar tais vulnerabilidades.

Existem várias metodologias, modelos e *frameworks* de avaliação de segurança para aplicações web, no entanto muitos destes instrumentos convergem na forma e até mesmo nas mais variadas caraterísticas de segurança que dado o seu grau de criticidade devem ser obrigatoriamente avaliadas. Logo, este estudo teve como referência instrumentos como o OWASP Top 10 e o CWE Top 25 para identificar e validar os achados descobertos durante o processo automático de avaliação de vulnerabilidades.

O presente trabalho identificou as vulnerabilidades de segurança mais frequentes nas aplicações web das universidades de angolanas e constatou que muitas destas falhas poderiam ser eliminadas se as instituições avaliadas introduzissem nos seus procedimentos técnicos e administrativos a prática regular de avaliação de vulnerabilidades e de testes de intrusão.

Palavras – Chave: Avaliação de segurança; Universidades Angolanas; OWASP; CWE.

Abstract

Vulnerability assessment is one of the technical procedures that can help prevent serious security breaches, which when exploited can undermine brand credibility and or the continuity of a business. Angolan universities use their websites as a digital showcase and gateway to their web applications that are available on the Internet 24/7 and exposed to all kinds of risks that may arise from them.

This work aims to assess the security status of the websites of Angolan higher education institutions, and more specifically, to identify the most frequent vulnerabilities existing in them and to quantify their degree of criticality. To achieve these objectives, this work is based on methodologies that identify the most frequent flaws in web applications and uses automatic tools to effectively discover and validate such vulnerabilities.

There are several methodologies, models and frameworks for evaluating security for web applications, however many of these instruments converge in the form and even in the most varied security characteristics that, given their degree of criticality, must be mandatorily evaluated. Therefore, this study had as reference instruments such as OWASP Top 10 and CWE Top 25 to identify and validate the findings discovered during the automatic vulnerability assessment process.

The present work identified the most frequent security vulnerabilities in the web applications of Angolan universities and found that many of these flaws could be eliminated if the evaluated institutions introduced the regular practice of vulnerability assessment and intrusion tests in their technical and administrative procedures.

Keywords: Vulnerability assessment; Angolan universities; OWASP; CWE.

Índice

Índice de Quadros

Índice de Figuras

Lista de Abreviaturas e Siglas

API - *Application Programming Interface*

CAPEC - *Common Attack Pattern Enumeration and Classification*

CAPTCHA - *Completely Automated Public Turing test to tell Computers and Humans Apart*

CGI - *Common Gateway Interface*

CMS - *Content Management System*

CSS - *Cascading Style Sheets*

CVE - *Common Vulnerabilities and Exposures*

CVSS - *Common Vulnerability Scoring System*

CWE - *Common Weakness Enumeration*

DSR – *Design Science Research*

DDoS - *Distributed Denial-Of-Service*

ERP - *Enterprise Resource Planning*

HKSAR - *The Government of the Hong Kong Special Administrative Region*

HTML - *Hypertext Markup Language*

HTTP - *Hypertext Transfer Protocol*

IDS - *Intrusion Detection System*

IoT - *Internet of Things*

IoV - *Internet of Vehicles*

IP - *Internet Protocol Address*

ISACA - *Information Systems Audit and Control Association*

ISSAF - *Information Systems Security Assessment Framework*

NVD - *National Vulnerability Database*

NCSC - *National Cyber Security Centre (United Kingdom)*

NIST - *National Institute of Standards and Technology*

OISSG - *Open Information Security Services Group*

OSSTMM - *Open-Source Security Testing Methodology Manual*

POC - *Proof of concept*

PTES - *Penetration Testing Execution Standard*

RGPD *- Regulamento Geral de Proteção de Dados*

SSL - *Secure Sockets Layer*

URI - *Uniform Resource Identifier*

URL - *Uniform Resource locator*

WASC - Web Application Security Consortium

XML - *Extensible Markup Language*

XSRF - *Cross-site request forgery*

XSS - *Cross-Site Scripting*

XXE - *Entidades Externas de XML*

Capítulo 1 Introdução

1.1 Enquadramento do tema

No contexto atual a Internet deixou de ser uma "simples" rede mundial de computadores e tornou-se numa verdadeira plataforma de conteúdos e de negócios onde várias empresas desenvolvem as suas atividades com o objetivo de efetuarem negócios e obterem vantagens competitivas sobre os seus concorrentes. Esta evolução levou a que fosse redefinido estrategicamente o uso que era dado aos *websites*.

Durante este processo de evolução da Internet os *websites* passaram de um conjunto de páginas HTML com conteúdos estáticos que geralmente mostravam a história das instituições, contactos, descrição dos produtos entre outros, para se transformarem em autênticas ferramentas de marketing e de distribuição de conteúdos.

Em Angola, à semelhança do que aconteceu noutras partes do mundo, os *websites* das universidades passaram por este processo de transformação e atualmente a maior parte dos mesmos são baseados em soluções como Joomla, WordPress, Umbraco e outros sistemas para a gestão de conteúdos (CMS – *Content Management Systems*) e construção de portais que servem de porta de acesso aos sistemas integrados de gestão académica e financeiro (ERP – *Enterprise Resource Planning*) das respetivas universidades. Por se tratar de aplicações web estas aplicações podem ser acedidas a partir de qualquer parte do mundo, usando um navegador web (*web browser*).

Este tipo de aplicação é critico e estratégico para a gestão das universidades, no entanto dada a sua natureza web estão expostos aos riscos que a Internet apresenta, pelo que, garantir a segurança das mesmas constituem um dos fatores críticos do sucesso, pois a falta dela pode inclusive pôr em causa a seriedade e até a continuidade das referidas instituições.

A complexidade, modularidade e a inclusão de bibliotecas de terceiros no desenvolvimento e distribuição das aplicações web tornam difícil a tarefa de garantir que a mesma esteja completamente isenta de erros (*bugs* no *software* ou má configuração dos sistemas informáticos) ou de vulnerabilidades que quando devidamente exploradas colocam em causa a segurança das aplicações.

É impossível garantir que uma aplicação web ao longo do seu tempo de vida útil não possua ou não esteja exposta a algum tipo de vulnerabilidade. No entanto existem mecanismos associados a testes de segurança informática que ao serem executados regularmente permitem detetar e reduzir o número de vulnerabilidades e evitar que as mesmas sejam exploradas por terceiros e desta forma garantido de forma iterativa a segurança das mesmas.

Em Angola, sistemas de gestão de informação como, o sistema integrado de gestão universitária da Técnica de Angola (SIGU), sistema de informação integrado da Universidade Metodista de Angola (SIIUMA), o sistema informático utilizado na gestão académica da Universidade Mandume ya Ndemufayo (SIGU-Alunos), o sistema integrado de gestão informático da Universidade Lueji A Nkonde (SIGA) ou o Oracle PeopleSoft adotado pela Universidade Católica de Angola são apenas alguns exemplos dos sistemas integrados de gestão que as universidades angolanas usam e que estão disponíveis a partir dos seus *websites*.

Nesta perspetiva, importa avaliar a segurança das aplicações web das universidades Angolanas para perceber o grau de vulnerabilidade que as mesmas apresentam e consequentemente conhecer o grau de risco a que as mesmas estão expostas.

O presente trabalho avalia as vulnerabilidades mais frequentes nas aplicações web e o impacto das mesmas na segurança das aplicações analisadas. Este trabalho recorre a um conjunto de métodos automáticos (*web scanners* e *scripts* automatizados) que permitem coletar, identificar, validar e classificar o grau de criticidade das vulnerabilidades encontradas. No final do estudo é feito uma relação causa efeito entre os achados e falha segurança das aplicações.

1.2 Motivação e relevância do tema

O número crescente de aplicações web atacadas por indivíduos ou entidades maliciosas têm causados prejuízos de milhões de euros a diversas instituições e estas vêm frequentemente as informações privadas dos utilizadores (e nalguns casos, clientes) sendo roubadas devido a falhas de segurança nas aplicações.

Em 2019, a British Airways, por não ter garantido de forma eficaz a segurança das suas aplicações web foi multada em 183 milhões de libras (cerca de 204 milhões de euros) ao

abrigo do regulamento geral de proteção de dados (RGPD) devido a um ciberataque no seu *website* em setembro de 2018 que afetava dados pessoais (emails, moradas, nomes e dados de pagamento) de milhares de passageiros (Publico, 2019).

Para que um ataque informático ocorra é necessário que determinados procedimentos técnicos sejam realizados previamente. Uma das etapas críticas e imprescindíveis envolve a descoberta de vulnerabilidades das aplicações e respetivos ecossistemas que circundam o alvo para posterior exploração das falhas e consequentemente a realização do ataque informático.

Logo, a avaliação de vulnerabilidades é uma das primeiras etapas para identificar e quantificar as falhas de segurança. Nesta perspetiva a maneira mais apropriada para evitar ataques informáticos é a realização frequente de testes de vulnerabilidades com o objetivo de ir mitigando as falhas que forem sendo encontradas impedindo que as mesmas se transformarem em efetivas falhas graves de segurança, e que possam ser usadas para explorar o sistema alvo.

As avaliações das vulnerabilidades podem ser realizadas através de ferramentas totalmente automatizadas ou manuais, nas diferentes fases do ciclo de vida de uma aplicação. O ideal seria avaliar as vulnerabilidades das aplicações durante o processo de desenvolvimento e de atualização de qualquer elemento ou ativo que faça parte do ecossistema da aplicação, incluindo servidor web ou aplicacional, sistema de gestão de base de dados, sistema operativo ou quando ocorrer alteração do hardware.

Espera-se que os resultados desta investigação possibilitem a identificação dos pontos fortes e fracos da segurança dos websites das instituições em causa e fornecer elementos que possam contribuir para incentivar o uso e a inclusão deste instrumento na rotina diárias das instituições de ensino superior de Angola por forma a aumentar o grau de segurança das suas aplicações informáticas.

1.3 Objetivos

No contexto atual muitas instituições de ensino superior em Angola incorporaram nos seus *websites* diversos módulos de software que são críticos para o sucesso e sustentabilidade das mesmas. Sistemas de gestão de conteúdos (CMS), webmail e

sistemas integrados de gestão académico e financeiros (ERP), são apenas alguns dos muitos exemplos de aplicações que estão associadas ou indexadas aos portais web destas instituições de ensino.

Dado que as referidas aplicações estão disponíveis em permanência e acessível a partir de qualquer parte do mundo elas estão expostas aos diversos riscos que a Internet apresenta.

Assim, o principal objetivo da presente investigação é:

- Determinar o nível de segurança das aplicações web das instituições de ensino superior em Angola

A avaliação da segurança das aplicações web das instituições de ensino superior em Angola resulta da necessidade de identificar e quantificar o volume de vulnerabilidades existentes e classificar o grau de risco que as mesmas apresentam, para posteriormente apontar e mitigar as falhas de segurança detetadas.

Assim, de forma sucinta os objetivos específicos deste trabalho passam por:

- Identificar e avaliar as vulnerabilidades mais frequentes das aplicações web das instituições de ensino superior Angolanas.
- Analisar e quantificar o grau de criticidade das vulnerabilidades das aplicações web das instituições de ensino superior Angolanas e o impacto das mesmas.
- Classificar o risco de segurança das aplicações web das instituições em estudo com base no grau de criticidade e impacto das vulnerabilidades encontradas.
- Promover e divulgar os resultados do presente estudo junto das instituições envolvidas, para que as mesmas possam melhorar a segurança da sua presença na Internet.

1.4 Questões da investigação

Para desenvolver um plano de trabalho que possibilitasse realizar esta investigação e respondê-la tendo em conta a metodologia adotada foi criado a seguinte questão primária da investigação que deriva dos objetivos gerais:

Qual é o atual estado da segurança das aplicações web das instituições de ensino superior de Angola?

1.5 Abordagem metodológica

O *Design Science Research* (DSR) foi a metodologia usada neste trabalho. A aplicação da referida metodologia em pesquisas relacionadas com sistemas de informação tem por objetivo a aquisição de conhecimento e entendimento que permitam o desenvolvimento e a implementação de soluções baseadas em tecnologia para solucionar problemas relevantes na área de negócios até agora não resolvidos (Hevner, *et al*. 2004).

Hevner (2007), apresentou uma *framework* (Figura 1) baseado no seu anterior trabalho (Hevner, *et al*. 2004) acrescentando os ciclos da relevância, desenho e do rigor com o propósito de tornar mais eficiente o entendimento, a execução, a avaliação e os procedimentos inerentes a pesquisa em sistemas de informação.

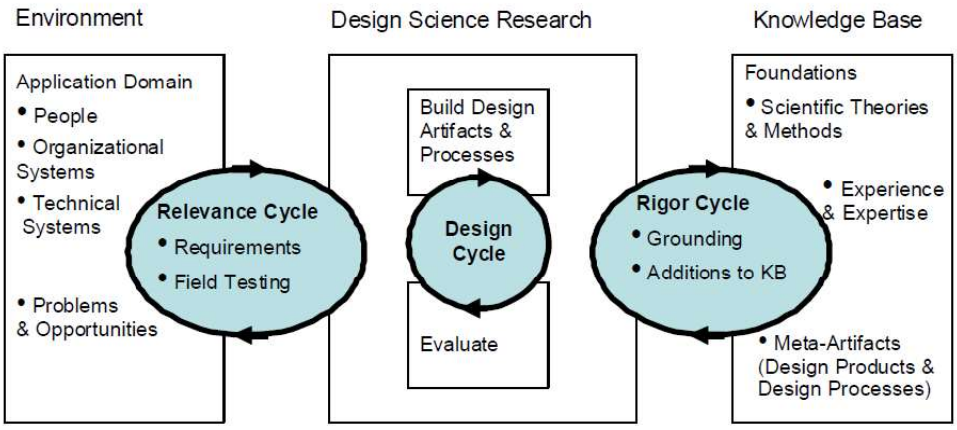

Figura 1- Ciclos de pesquisa (Hevner, 2007)

A *framework* acima referenciada (Figura 1), permite de forma cíclica realizar pesquisas que resultem na construção de artefactos, ao incorporar o rigor científico (conhecimento

existente) no *design* de soluções relevantes destinados a resolver e ou melhorar problemas organizacionais previamente identificados.

Dresch e Lacerda (2016), no seu trabalho de investigação sobre a aplicação da metodologia da Design Science Research na área da engenharia de produção identificaram a abordagem científica (Figura 2) e descreveram as etapas de pesquisa a seguir para identificar a relevância e conduzir com rigor a pesquisa científica.

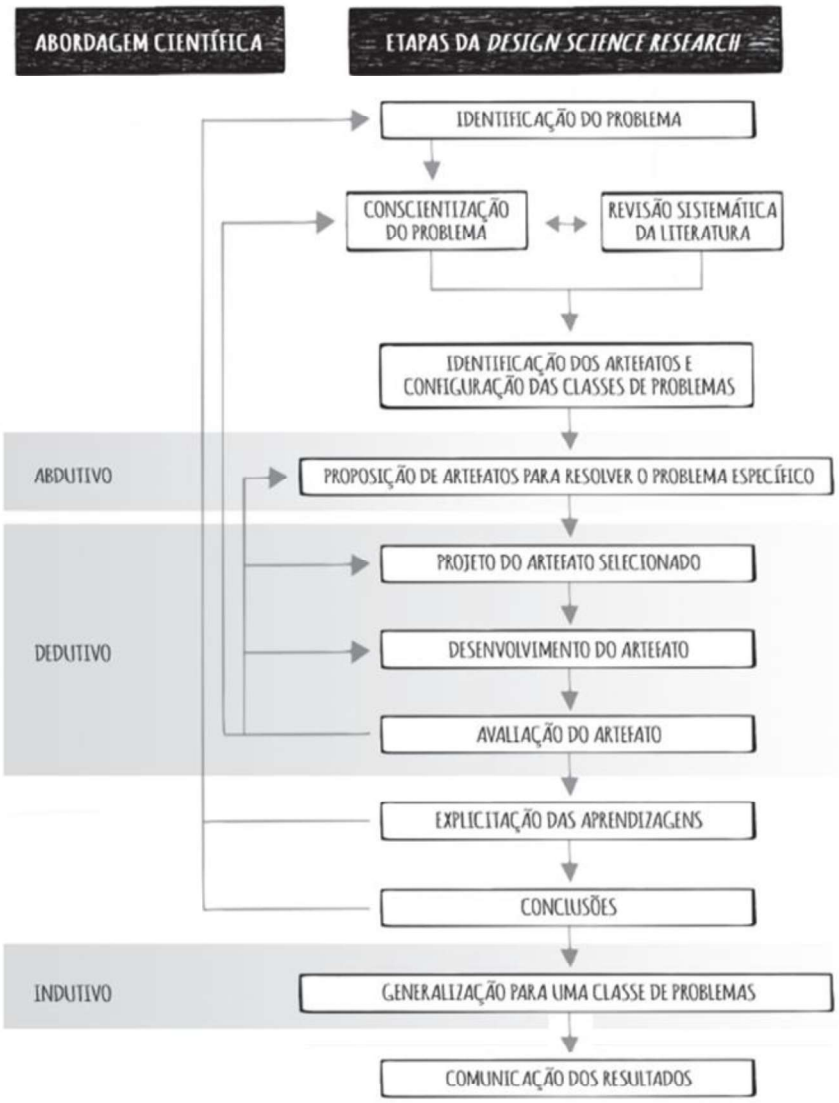

Figura 2 - Etapas da Design Science Research (Dresch & Lacerda, 2016)

A aplicação das etapas da *Design Science Research* (Figura 2), permitiu executar um conjunto de procedimentos cujo a primeira etapa teve início com a identificação do

problema e definição dos objetivos, de seguida foi efetuado a revisão da literatura que permitiu validar a relevância do tema.

De seguida efetuou-se a avaliação de vulnerabilidades das aplicações *web*, permitindo a criação e avaliação dos artefactos resultantes da utilização de um conjunto de métodos automáticos (*web scanners* e *scripts* automatizados) que serviu para identificar, validar e classificar o grau de criticidade das vulnerabilidades encontradas, culminando com a elaboração das conclusões e consequentemente a publicação dos resultados da pesquisa.

1.6 Estrutura e organização da dissertação

O presente trabalho está organizado em cinco capítulos que pretendem refletir as diferentes fases até à sua conclusão.

O primeiro capítulo introduz o tema da investigação e objetivos da mesma bem como uma breve descrição da estrutura do trabalho.

O segundo capítulo reflete o enquadramento teórico, designado por Revisão da literatura.

O terceiro capítulo é dedicado à Metodologia utilizada no processo de recolha e tratamento de dados bem como os métodos de análise utilizados.

O quarto capítulo apresenta a análise dos resultados obtidos, de acordo com a metodologia que se entendeu apropriada.

No quinto e último capítulo apresentam-se as conclusões deste estudo bem como as recomendações, limitações e trabalhos futuros.

Capítulo 2 Revisão da Literatura

2.1 Avaliação das vulnerabilidades

Uma vulnerabilidade é uma fraqueza ou uma falha numa aplicação, sistema ou rede, que quando explorado permite causar danos ao proprietário do mesmo (Zhu, 2017).

A avaliação da vulnerabilidade ou *"Análise da vulnerabilidade*" é o processo de identificação e classificação de falhas de segurança no sistema, na rede ou na infraestrutura de comunicação de uma organização (Shakeel, 2010). O referido autor afirmou que a maior vantagem de um teste de avaliação de vulnerabilidade é que ela pode prever o sucesso das contramedidas propostas e examinar a taxa de sucesso real após a sua utilização.

A análise de vulnerabilidade envolve a descoberta de um subconjunto de falhas no *input* de dados com o qual um utilizador mal-intencionado pode explorar erros de lógica numa aplicação e torná-la insegura (Sparks, Embleton, Cunningham, & Zou, 2007). Este processo pode ser feito de forma independente ou integrado como uma das etapas do teste de intrusão.

A *Open Web Application Security Project* (OWASP), instituição sem fins lucrativos que se dedica ao desenvolvimento de instrumentos que permitem aumentar a segurança de aplicações, criou e mantém atualizado um manual com uma lista consensual onde identificou dez (10) riscos de segurança mais críticos e frequentes em aplicações Web (OWASP, 2017a). A Tabela 1, identifica e descreve resumidamente estes riscos.

Tabela 1- OWASP Top 10 - Riscos de segurança

Rank	**Vulnerabilidade**
A1:2017	Injeção
A2:2017	Quebra de Autenticação
A3:2017	Exposição de Dados Sensíveis
A4:2017	Entidades Externas de XML (XXE)
A5:2017	Quebra de Controlo de Acessos
A6:2017	Configurações de Segurança Incorretas
A7:2017	*Cross-Site Scripting* (XSS)
A8:2017	Desserialização Insegura

A9:2017	Utilização de Componentes Vulneráveis
A10:2017	Registo e Monitorização Insuficiente

A MITRE Corporation, uma instituição sem fins lucrativos, mas financiada pelo governo dos Estados Unidos cujo principal objetivo é a segurança informática criou e mantém atualizado uma lista com as vulnerabilidades de segurança mais frequentes em software (MITRE, 2019).

A Common Weakness Enumeration (CWE), é um instrumento útil e que se deve ter em conta aquando da avaliação de vulnerabilidades de uma aplicação. Ela usa uma linguagem comum, para identificar e descrever as vulnerabilidades e apresentar formas de mitigar e prevenir a ocorrência das mesmas (CWE, 2019).

Na prática, a CWE tem o seguinte propósito:

- Servir como uma linguagem comum para descrever os pontos fracos de segurança de software em arquitetura, desenho ou código.
- Servir como um medidor padrão para ferramentas de segurança de software que atendam a esses pontos fracos.
- Forneça um padrão comum para os esforços de identificação, mitigação e prevenção de fraquezas.

A Tabela 2, lista, quantifica, classifica e possui um identificador que permite obter informações detalhadas sobre as vinte e cinco (25) vulnerabilidades de segurança frequentemente encontradas em software.

Tabela 2- CWE -Top 25

Rank	***ID***	**Designação**
1	CWE-119	*Improper Restriction of Operations within the Bounds of a Memory Buffer*
2	CWE-79	*Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')*
3	CWE-20	*Improper Input Validation*
4	CWE-200	*Information Exposure*
5	CWE-125	*Out-of-bounds Read*
6	CWE-89	*Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')*
7	CWE-416	*Use After Free*
8	CWE-190	*Integer Overflow or Wraparound*
9	CWE-352	*Cross-Site Request Forgery (CSRF)*

Rank	***ID***	**Designação**
10	CWE-22	*Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')*
11	CWE-78	*Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')*
12	CWE-787	*Out-of-bounds Write*
13	CWE-287	*Improper Authentication*
14	CWE-476	*NULL Pointer Dereference*
15	CWE-732	*Incorrect Permission Assignment for Critical Resource*
16	CWE-434	*Unrestricted Upload of File with Dangerous Type*
17	CWE-611	*Improper Restriction of XML External Entity Reference*
18	CWE-94	*Improper Control of Generation of Code ('Code Injection')*
19	CWE-798	*Use of Hard-coded Credentials*
20	CWE-400	*Uncontrolled Resource Consumption*
21	CWE-772	*Missing Release of Resource after Effective Lifetime*
22	CWE-426	*Untrusted Search Path*
23	CWE-502	*Deserialization of Untrusted Data*
24	CWE-269	*Improper Privilege Management*
25	CWE-295	*Improper Certificate Validation*

A coluna *ID,* da Tabela 2 é um identificador que permite identificar univocamente uma falha de segurança e com ela obter no website da MITRE a descrição detalhada e exaustiva sobre a mesma, consultar formas de mitigar o problema bem como obter a descrição de possíveis relacionamentos com outras vulnerabilidades.

2.2 Processo de avaliação de vulnerabilidade

O processo de avaliação de vulnerabilidades implica efetuar uma análise (*scanning)* do sistema e das suas aplicações. Essencialmente, existem três (3) tipos de categorias de *scanners*: *scanners* de portas, de aplicações e de vulnerabilidades (Bairwa, Mewara, & Gajrani, 2014).

- ***Scanners* de portas**: São usados para verificar o estado as portas com o propósito para determinar se as portas do sistema operativo e dos serviços oferecidos estão abertas ou fechadas.

- ***Scanners* de aplicações**: São usados para avaliar uma aplicação específica na rede, a fim de rastrear se seus pontos fracos podem ser explorados e colocar em causa a segurança do sistema.

- ***Scanners* de vulnerabilidade**: São usados para descobrirem as vulnerabilidades num determinado sistema, e avaliar se as falhas encontradas quando exploradas por um atacante, podem colocar todo o sistema em risco.

Para a ISACA (2018), existem quatro (4) tipos de avaliação de vulnerabilidades: avaliação da rede, avaliação local (*host*), avaliação da rede sem fios (*wireless)* e avaliação dos aplicativos.

- **Avaliação ou *scans* da rede**: Este tipo de avaliação, efetua uma varredura na rede para descobrir e listar todos os ativos de redes (hardware, software) e serviços com vulnerabilidades. Para cada ativo encontrado é realizada uma ou mais varreduras para que determinados possíveis pontos fracos que possam ceder perante um ataque. Este tipo de avaliação pode ser feito remotamente.

- **Avaliação local (*host*):** Este tipo de avaliação permite realizar uma varredura local no dispositivo alvo ou serviço que possa estar inacessível ou escapado a um tipo de varredura de rede. O processo inclui a listagem de todos os componentes (hardware, software), serviços e configurações do dispositivo a fim de identificar e enumerar os pontos fracos que possam comprometer a segurança local e ou da rede.

- **Avaliação da rede sem fios:** Este tipo de avaliação visa essencialmente, validar o estado das configurações de segurança, a legitimidade da rede e a fronteira ou isolamento da rede pública e da rede interna. Ela permite evitar que utilizadores convidados possam ter acesso à rede interna, permitindo-lhes escalarem os seus privilégios e contornarem os controlos de segurança.

- **Avaliação das aplicações:** Neste tipo de avaliação dá-se especial atenção ao software existente. Geralmente o processo envolve avaliar aplicações Web com o intuito de descobrir e enumerar vulnerabilidades e ou erros de configuração.

O processo de avaliação das vulnerabilidades pode ser realizado de forma isolada ou ser visto como sendo uma etapa de outro processo mais complexo como é o teste de intrusão

(Figura 3), que permite explorar e validar na prática os defeitos encontrados e excluir os resultados falsos positivos.

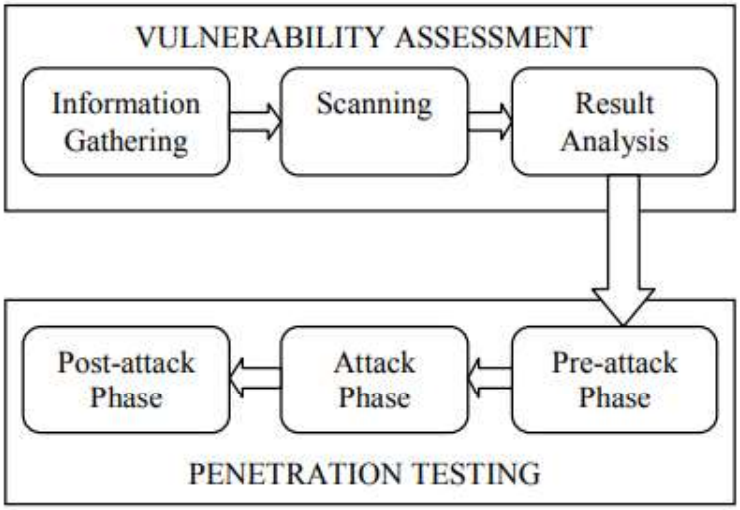

Figura 3 - Processo de avaliação de vulnerabilidades (Shah & Mehtre, 2013)

O referido processo envolve passos como obtenção da informação, análise (*scanning)* e avaliação dos resultados conforme ilustrado na Figura 3. No entanto, vários autores apresentaram um modelo formal de quatro (4) etapas, onde acrescentam a elaboração do relatório como sendo a última etapa do teste de avaliação de vulnerabilidade (Shah & Mehtre, 2014), (Vibhandik & Bose, 2015).

A KPMG (2016), elaborou um modelo (ou guião) e um cenário com duas opções que permitem estabelecer alguns parâmetros mínimos durante o processo de avaliação de vulnerabilidades. Na Tabela 3, estão identificados quatro (4) pilares que sustentam o processo de avaliação de vulnerabilidades, a saber, os utilizadores, a aplicação, o *web browser* e a área de *backend*.

Tabela 3- Adaptação do guia de avaliação de segurança

Atores	**Parâmetros**
The User	• *Password management* • *Social engineering* • *Phishing* • *update management* • *Waterhole attacks* • *Data governance* • *Administrative access*
The Application	• *Cross-site scripting*

		• *Weak input validation* • *Brute force attacks* • *Zero-day exploits* • *Weak session management* • *Vulnerable libraries* • *Privileges escalation*
The Browser		• *Phishing* • *Framing* • *Click jacking* • *Man-in-the-Browser* • *Data Caching*
The Backend	*Webserver*	• *Platform vulnerabilities* • *Server misconfiguration* • *Cross-site request forgery* • *Weak input validation* • *Brute force attacks*
	Database	• *SQL Injection* • *Privileges escalation* • *Data dumping* • *OS command execution*

Fonte: *(KPMG, 2016)*

O guia de avaliação de segurança elaborado pela KPMG (Tabela 3) permite identificar de forma simples e clara as fronteiras, as áreas, os parâmetros que devem ser usados durante o processo de avaliação das vulnerabilidades. Neste guia, o fator humano e as ferramentas utilizadas para acederem as aplicações (*Web Browser*) são escrutinadas e incluídos no processo de avaliação de segurança.

2.2.1 Abrangência do processo avaliação de vulnerabilidade

A abrangência dos testes de avaliação de vulnerabilidades dependerá sempre do escopo, do tipo de teste e técnicas a serem adotados aquando da realização da mesma. Neste contexto, as três (3) técnicas mais importantes usadas para encontrar falhas são: Testes de caixa branca, testes de caixa cinza e testes de caixa negra (Hafele, 2004), (Yüce, 2018), (PCI, 2015), (Al Shebli & Beheshti, 2018).

- **Testes de caixa branca**

  Teste realizado com conhecimento da estrutura interna, design e/ou implementação (inclusive com acesso ao código-fonte) do objeto que está a ser

testado. Neste tipo de teste o auditor recebe todas as informações com antecedência, como documentação de design, dados de login entre outros (NCSC, 2020).

Para o autor, é importante que um auditor especializado esteja envolvido nos testes e que a programação relevante para a segurança seja avaliada de maneira estruturada. Na prática, os testes de intrusão e revisões de código são geralmente realizados por diferentes especialistas.

- **Testes de caixa cinza**

  Teste realizado com conhecimento parcial da estrutura interna, desenho e/ou implementação do objeto que está a ser testado. NCSC (2020), exemplifica o caso em que um auditor recebe informações parciais com antecedência, como como os dados de login de um utilizador com uma determinada função dentro do sistema, para que seja possível testar o que um atacante com certos direitos em um sistema poderia alcançar ou escalar os privilégios atribuídos.

  Os testes de caixa cinza combinam o melhor dos testes de caixa branca e negra. Eles visam a análise privilegiada do ambiente de tempo de execução e da interface da camada de apresentação, bem como do código-fonte (Sreenivasa, 2012).

  O autor acrescenta, que o referido teste permite uma análise mais abrangente de um aplicativo e que a eficiência é obtida abordando os problemas de ambos os lados, visto que a disponibilidade de informações permite uma maior precisão de deteção e recomendações mais granulares que permitem uma compreensão mais completa da prova de impacto.

- **Testes de caixa negra**

  Neste tipo de testes nenhuma informação prévia é fornecida; o auditor não sabe nada exceto, por exemplo, um endereço IP, nome de domínio ou empresa nome. Isso geralmente é mais realista em termos de informações que um atacante real teria acesso, exceto que, em prática, um invasor tem muito mais tempo e pode, portanto, obter acesso a mais informações (NCSC, 2020).

A grande diferença entre estas abordagens, reside na quantidade de informação sobre os ativos de redes e sistemas que o auditor possui no momento da realização dos testes. A Tabela 4, apresenta as vantagens e desvantagens existentes entre os três tipos de testes:

Tabela 4- Vantagens e desvantagens dos tipos de teste de intrusão

Tipo	Vantagens	Desvantagens
Caixa Negra	É eficiente para grandes segmentos de código. A perceção daquilo que tem de ser feito pelo auditor é muito simples. A perspetiva dos utilizadores é claramente diferente as dos programadores (programador e testador são independentes do outro). A criação dos casos de teste faz-se de forma rápida	Apenas um número selecionado de cenários de teste é realmente realizada. Como resultado, há apenas uma cobertura limitada. Pouca clareza na especificação dos requisitos. Os casos de teste são difíceis de serem projetados. Teste difíceis de serem implementados
Caixa Branca	Permite detetar erros escondidos no código-fonte. É possível cobrir todos os cenários que vão ser testados.	Requer um auditor qualificado para executá-lo. Muitos aspetos não vão ser testados, pois é muito difícil cobrir todos os cenários e descobrir todos os erros.
Caixa Cinza	O auditor depende da interface e especificação funcional em vez de código fonte. O teste é realizado do ponto de vista do utilizador e não do que o ponto de vista do programador. Permite a criação de testes inteligentes. Este tipo de teste é imparcial.	A cobertura do teste é limitada, pois o acesso ao código-fonte não está disponível. É difícil associar a identificação de defeitos em aplicações distribuídas. Muitos cenários de casos de testes podem não ser testados. Se o arquiteto de software já tiver executado um caso de teste, os testes podem ser redundantes.

Fonte: *(Mohd. Ehmer & Farmeena, 2012)*.

2.3 Como efetuar os testes de avaliação de vulnerabilidades

O processo de avaliação de vulnerabilidades envolve um conjunto de passos que devem dados a priori e a posteriori para que o mesmo produza resultados que permitam obter informações confiáveis e conclusivas que permitem conhecer o grau de vulnerabilidade dos seus sistemas de informação. E em caso de risco o relatório final deverá incluir um plano que permita remediar ou eliminar as falhas de seguranças encontradas.

A realização do teste avaliação vulnerabilidades envolve no mínimo quatro (4) fases, começa com o planeamento, passa pela identificação do alvo para depois realizar o

processo do *scanning* ou identificação das vulnerabilidades, no final analisa-se os resultados e elaboração de um relatório com os achados e formas de mitigar as vulnerabilidades encontradas (Shah & Mehtre, 2014), (Shinde & Ardhapurkar, 2016), (Goel & Mehtre, 2015).

O modelo apresentado na Figura 4, ilustra os passos necessários para a realização dos testes de avaliação de vulnerabilidades bem como a sua relação com os testes de intrusão.

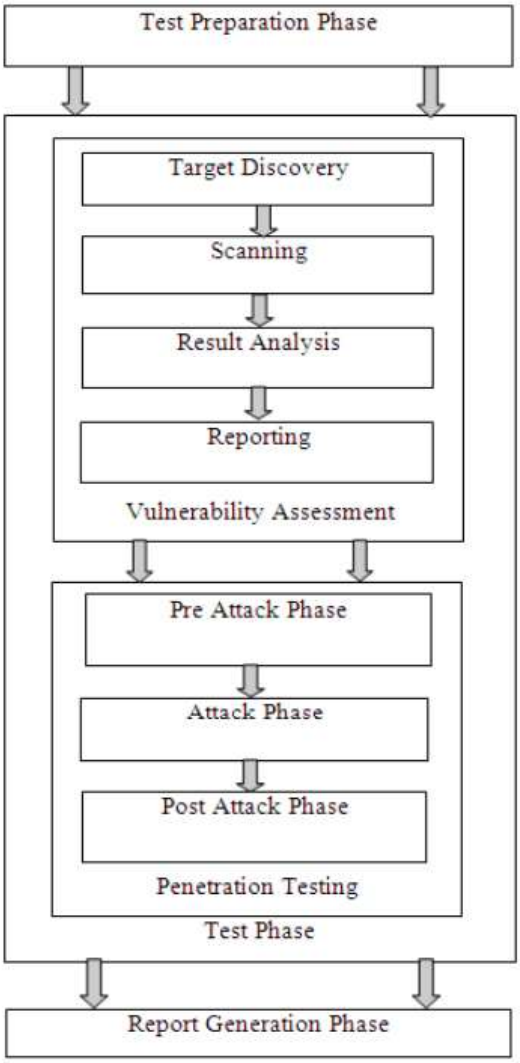

Figura 4- Fases dos testes de segurança (Shah & Mehtre, 2014)

A partir dela é possível identificar pelo menos cinco (5) etapas que devem ser percorridas para a realização da mesma. Dos quais destaca-se as seguintes fases:

Planeamento: Nesta fase, o auditor e a organização se reúnem para decidir o escopo, os objetivos, o tempo e a duração dos testes. Segundo os autores do referido modelo, deve-

se também serem validados ou assinados toda a documentação legal para os devidos efeitos.

Obtenção das informações sobre o alvo: O auditor recolhe as informações cruciais relativas ao alvo. Estas informações incluem dados importantes relativamente aos sistemas de informações e infraestrutura tecnológica que são obtidos mediante *scanning* de redes ou de capturas de cabeçalho *HTTP*.

***Scanning* das vulnerabilidades:** É o processo através do qual o auditor executa uma varredura de todo o sistema com o objetivo de identificar e listar as vulnerabilidades encontradas. Durante o processo os auditores utilizam várias ferramentas e técnicas que permitem analisar em detalhe cada um dos componentes do sistema para encontrar qualquer falha ou brecha no funcionamento do sistema, como serviços indesejados, portas abertas, conexões remotas, cifras ou senhas fracas, etc. A varredura termina com a lista de potenciais vulnerabilidades do alvo que podem levar à divulgação de informação ou perda de dados.

Análise dos resultados: Esta fase está dependente da saída (*output*) da fase de varredura e analisa o conjunto de vulnerabilidades e outras ameaças identificadas após a varredura. O papel do auditor nesta fase, visa priorizar as vulnerabilidades identificadas com base em sua gravidade e impacto.

Em termos práticos, os resultados obtidos na fase de varredura podem ser afetados ou influenciados pela quantidade de resultados falsos positivos, esses falsos positivos são detetados e removidos nesta fase, otimizando assim a lista inicial tornando-a mais eficiente e precisa. Posteriormente, as vulnerabilidades identificadas são classificadas e ordenadas pelo seu grau de gravidade para posterior exploração ou correção.

Relatório: Após a realização com sucesso das fases iniciais, o auditor documenta as diversas operações realizadas e os resultados obtidos em todo o processo de avaliação das vulnerabilidades. Nesta fase, os autores esclarecem, que o auditor gera um relatório do trabalho feito, listando o conjunto de vulnerabilidades, identificado o seu nível de gravidade e outros detalhes. Esta lista pode ainda ser usada pela organização para solicitar remediações para o mesmo.

2.4 Tipos de scanners de avaliação de vulnerabilidades

Os *scanners* de vulnerabilidade podem ser amplamente divididos em dois grupos: scanners de rede (*network-based*) e scanners local (*host-based*) (HKSAR, 2008), (Shakeel, 2010).

Scanners de rede (*network-based*)

Shakeel (2010), ao comparar os dois tipos de *scanner*s de vulnerabilidade, afirmou que a baseada em rede vem em primeiro lugar devido à sua capacidade de identificar sistemas vulneráveis numa rede. Segundo o autor, um administrador/auditor de rede deve adotar este processo primeiro, ao conduzir os testes de avaliação de vulnerabilidade visto que a mesma fornece os resultados imediatos de vulnerabilidades altamente graves que precisam de uma solução rápida.

Um *scanner* baseado em rede é geralmente instalado em uma única máquina que faz a varredura de vários outros *hosts* na rede. Ele ajuda a detetar vulnerabilidades críticas, como *firewalls* mal configurados, servidores da *Web* vulneráveis, riscos associados a *software* fornecido pelo fornecedor e riscos associados à administração de rede e sistemas (HKSAR, 2008).

O referido autor, descreve que este tipo de scanners podem ser utilizado para fazer varredura que determinam a lista de portas de rede abertas em sistemas remotos; Scanners de servidor da web que avaliam as possíveis vulnerabilidades (por exemplo, arquivos potencialmente perigosos ou CGIs) em servidores da web remotos; Scanners de aplicativos da web que avaliam os aspetos de segurança de aplicativos da web (como *script* entre sites e injeção SQL) em execução num determinado servidor.

Scanners local (*host-based*)

Um scanner local é instalado na máquina a ser verificado e tem acesso direto a dados de baixo nível, como serviços específicos e detalhes de configuração do sistema operativo do referido equipamento (HKSAR, 2008). Segundo o autor, este tipo de scanner pode, no entanto, fornecer informações sobre atividades arriscadas do utilizador, como o uso de senhas de fraca segurança ou até mesmo nenhuma senha.

Shakeel (2010), afirmou que a avaliação da vulnerabilidade que utiliza um scanner local permite que um administrador/auditor de rede elimine os riscos de segurança de dentro de suas organizações. Segundo o autor, esta ferramenta de avaliação de vulnerabilidade executa os testes da perspetiva de um utilizador com uma conta local e com permissões de acesso aos sistemas existentes.

O autor conclui dizendo que este tipo de avaliação com recurso a este tipo de scanner permite validar o tipo de privilégio/acesso que um utilizador terá ao se conectar à rede local, mesmo a partir de uma conta de convidado, visto o mesmo pode explorar as falhas de segurança nos servidores locais e pode acabar assumindo o controle dos sistemas locais da organização.

O Intruder (2015), descrever que existem tecnicamente três (3) tipos de scanners de avaliação de vulnerabilidades: *Scanners* de rede (*Network-based*), *scanners* baseados em agente (*Agent-based*) e *scanners* de aplicações web *(Web-application).*

Para o referido autor, os scanners de rede são assim chamados porque eles varrem os sistemas através da rede, sondando o sistema em busca de portas e serviços abertos e, em seguida, investigando cada serviço para obter mais informações, fraquezas de configuração ou vulnerabilidades conhecidas.

O *scanning* baseado em agente é realizado mediante instalação prévia de um software cliente (agente) em cada dispositivo a ser auditado, e que possuem capacidade para podem executar varreduras de vulnerabilidade local e relatar os resultados a um servidor central (Intruder, 2015).

Quanto ao *scanner* de aplicações web, o autor descreve que estes são um tipo especializado de *scanner* de vulnerabilidade que se concentra em encontrar pontos fracos em aplicações web, visto que tradicionalmente, este tipo de aplicações funcionam "rastreando" através de um website ou aplicativo de maneira semelhante a um mecanismo de pesquisa, enviando uma série de sondagens para cada página ou formulário que encontra para descobrir pontos fracos.

O autor conclui dizendo que os *scanners* de aplicações *Web* geralmente são bons na identificação de fraquezas diretas como: Injeções de SQL simples e falhas de script entre sites, pontos fracos de controle de acesso, exposição de informações confidenciais, pontos

fracos em fluxos de trabalho de várias etapas (*workflows*), pontos fracos envolvendo o armazenamento de uma carga (*payload*) e pontos fracos baseados na sessão de um utilizador.

2.5 Vantagens e desvantagens da avaliação de vulnerabilidades

Nesta secção iremos apresentar as principais vantagens e desvantagens do processo de avaliação de vulnerabilidades.

Vantagens

O processo de avaliação de vulnerabilidade analisa uma ampla gama de problemas de rede e, em seguida, identifica a fraqueza da rede que precisa ser corrigida. Esse processo também identifica vulnerabilidades como configuração incorreta e não conformidade com políticas. Outra vantagem de uma avaliação de vulnerabilidade é que ela sempre o manterá um passo à frente dos invasores. É o processo proativo mais poderoso de proteção da segurança de uma organização (Shakeel, 2010).

Um *scanner* de vulnerabilidade ajuda a verificar o inventário de todos os dispositivos na rede. O inventário inclui o tipo de dispositivo, versão do sistema operacional e nível de *patch*, configurações de hardware e outras informações relevantes do sistema. Essas informações são úteis no gerenciamento e rastreamento de segurança (HKSAR, 2008).

Permite o encadeamento de vulnerabilidades para entender o impacto total de todos os problemas descobertos e tomar medidas que permitem mitigar as falhas de segurança encontradas.

Desvantagens

O HKSAR (2008), descreve que uma das desvantagens dos *scanners* de vulnerabilidade é o facto de só poderem avaliar um instantâneo do tempo *(snapshot)* em termos do status de segurança de um sistema ou rede. Portanto, a varredura deve ser realizada regularmente, pois novas vulnerabilidades podem surgir ou as alterações na configuração do sistema podem introduzir novas brechas de segurança.

Uma outra desvantagem apontada pelo autor reside no facto dos *scanners* de vulnerabilidade serem projetados para descobrir apenas vulnerabilidades conhecidas. Ele

não pode identificar outras ameaças de segurança, como aquelas relacionadas a questões físicas, operacionais ou processuais.

Gera um incoerente e opressor quantidade de dados junto com algum falso positivo resultados, falha ao identificar vetores de ataque lógico, como falhas de lógica de aplicativo e reutilização de senha e produz recomendações de remediação que são genéricos e baseados na saída da ferramenta (Shinde & Ardhapurkar, 2016).

As vantagens dos testes de avaliação de vulnerabilidades superam as desvantagens apresentadas, pelo facto das mesmas por um lado contribuírem para uma maior perceção do grau de vulnerabilidades das aplicações e seus ecossistemas. Por outro lado, o resultado produzido pela avaliação das vulnerabilidades pode ser usado como sendo a primeira de duas fases de um processo mais abrangente de validação da segurança quando incluído como parte de um teste de intrusão.

2.6 Diferença entre testes de intrusão e avaliação de vulnerabilidades

O teste de intrusão ou *pentesting* é uma prática de testes de segurança que consiste em usar técnicas e ferramentas de atacantes (*hackers)* para avaliar o grau de segurança de um sistema informático, redes de computadores ou aplicações.

Para Engebretson (2011), o teste de intrusão pode ser definido como uma tentativa legal e autorizada de localizar e explorar com êxito sistemas informáticos com o objetivo de tornar esses sistemas mais seguros. O referido autor descreve que o processo inclui investigar vulnerabilidades e realizar ataques de prova de conceito (POC) para demonstrar que as vulnerabilidades são reais (Engebretson, 2011).

Enquanto o teste de intrusão é atividade normal de segurança, o ciberataque independentemente das suas motivações é um ato criminoso que causam sérios danos à instituição lesada. No final de um ciberataque o atacante elimina os vestígios da sua atividade para que a vítima não se aperceba do ataque sofrido (Hostway, 2017).

O teste de intrusão tem alguns elementos em comum com o processo de avaliação de vulnerabilidades, aliás não é possível fazer teste de intrusão sem passar pela descoberta das vulnerabilidades. No entanto, alguns profissionais confundem estes dois (2)

procedimentos. A Tabela 5, compara os dois instrumentos de segurança em termos de objetivo, como e quando os mesmos devem ser realizados (PCI, 2015).

Tabela 5- Avaliação de vulnerabilidades e o teste de intrusão

	Avaliação de vulnerabilidades	**Teste de intrusão**
Objetivo	Identificar, classificar e relatar as vulnerabilidades que, se exploradas, podem resultar em comprometimento intencional ou não intencional de um sistema.	Identificar formas de explorar as vulnerabilidades para contornar ou anular os controlos de segurança dos componentes do sistema.
Quando	Deve ser realizado pelo menos trimestralmente ou após alterações significativas.	Pelo menos uma vez por ano e ou sempre que ocorrem mudanças estruturais nos sistemas ou aplicações.
Como	Normalmente, são utilizados uma variedade de ferramentas automatizadas combinadas com a verificação manual de problemas identificados.	Um processo manual que pode incluir o uso da verificação de vulnerabilidades ou outras ferramentas automatizadas, resultando em um relatório abrangente.
Relatório	No relatório deve constar todos os riscos e potenciais danos que podem ser causados pelas vulnerabilidades descobertas, classificando-as de acordo com o sistema de pontuações básico da *Common Vulnerability Scoring System* (NVD) e da *Common Vulnerability Scoring System* (CVSS). Deve-se incluir no relatório se a avaliação foi feita de dentro para fora ou vice-versa.	Deve incluir a descrição de cada vulnerabilidade verificada e ou possível problema descoberto. Especificar os riscos que uma determinada a vulnerabilidade pode representar, incluindo métodos específicos de como e até que ponto ele pode ser explorado. Além das vulnerabilidades conhecidas, o relatório deve indicar também o estado de obsolescência do alvo avaliado.
Duração	Tempo relativamente curto, geralmente de alguns segundos a vários minutos por *host* verificado.	O procedimento pode durar dias ou semanas, dependendo do âmbito do teste e do tamanho do ambiente a ser testado. O tempo de execução dos testes podem aumentar atendendo a complexidade e a abrangências e a profundidade dos mesmos.

Fonte: *(PCI, 2015)*

2.7 Web Scanners

Web *scanners* são programas ou script automatizados que percorrem ou rastreiam uma determinada aplicação Web em busca de problemas comuns de segurança. Estes problemas estão normalmente relacionados com as principais falhas de segurança

encontradas em aplicações Web e que estão catalogados pela OWASP, como sendo os dez (10) tipos de vulnerabilidades frequentemente encontrado em aplicações Web.

O rastreamento das aplicações *Web* é feito com recurso a um tipo específico de *software* designado como web *crawler* (também conhecido com *robot* ou *spider*). Estes, além de terem a capacidade de automaticamente efetuarem *download* de páginas web, são capazes de criarem e manterem atualizado um índice com todos os links existentes nas páginas e efetuar um relacionamento entre elas, podendo para o efeito permitir que os utilizadores possam efetuar consultas sobre o índice e consequentemente localizarem páginas e ou conteúdos relacionados (Najork, 2010).

O processo de rastreamento recursivo e automático de páginas web é simples e rápido. No entanto existem algumas limitações técnica e de segurança que podem impedir que o mesmo possa preencher um formulário e submetê-lo devido a existência de regras de segurança que exigem o preenchimento do CAPTCHA ou da conclusão de uma regra de negócio.

CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*) são medidas de segurança generalizadas na *World Wide Web* que impedem que programas automatizados abusem dos serviços online (Ahn, Maurer, McMillen, Abraham, & Blum, 2008). Por exemplo, humanos podem ler texto distorcido ou selecionar apenas um subconjunto de imagens relacionadas num determinado contexto, mas os programas de computador atuais ainda não podem fazê-lo com sucesso.

A Figura 5 encontra-se retratada as características de CAPTCHA.

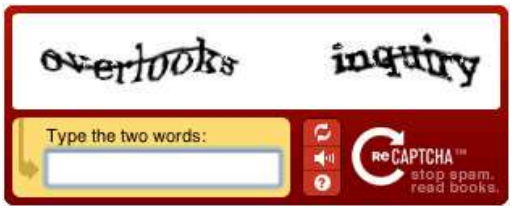

Figura 5 – Representação visual de um CAPTCHA (CAPTCHA, 2019)

A Google na tentativa de criar um CAPTCHA mais simples e seguro introduziu a versão 3 do seu CAPTCHA (reCAPTCHA V3), conforme indicado na Figura 6.

Figura 6- Google reCAPCHA

A existência de regras de negócios, também podem levar a que certas páginas ficam por ser validadas, se as mesmas não forem acedidas na sequência correta. Por exemplo, a consulta do detalhe de um determinado produto, as validações de um carrinho de compras exigem que passos prévios sejam dados para poder visualizar e ou ter acesso as mesmas.

Este tipo de *script* automatizado desempenha um papel importante na obtenção de informações que posteriormente devem ser usados nas fases subsequentes dos testes de avaliação de segurança. É importante referenciar que no decorrer do processo de web *scanning*, são também listados a existência de formulários sem validação do CAPTCHA, bem como a presença de XML, CSS, JavaScript, WebAssembly, e outros tipos de tecnologias que correm do lado do cliente e que caso de falha podem comprometer a segurança das aplicações.

Capítulo 3 Metodologia

3.1 Desenho de investigação

A pesquisa é um processo sistemático de coleta, análise e interpretação de informações (dados) para aumentar nossa compreensão de um fenômeno sobre o qual estamos interessados ou preocupados (Leedy & Ormrod, 2015).

Os referidos autores desenvolveram uma *framework* (Figura 7), que permite identificar os componentes que devem fazer parte do desenho de investigação com o propósito de obter elementos que permitam satisfazer os critérios científicos e responder as questões da investigação.

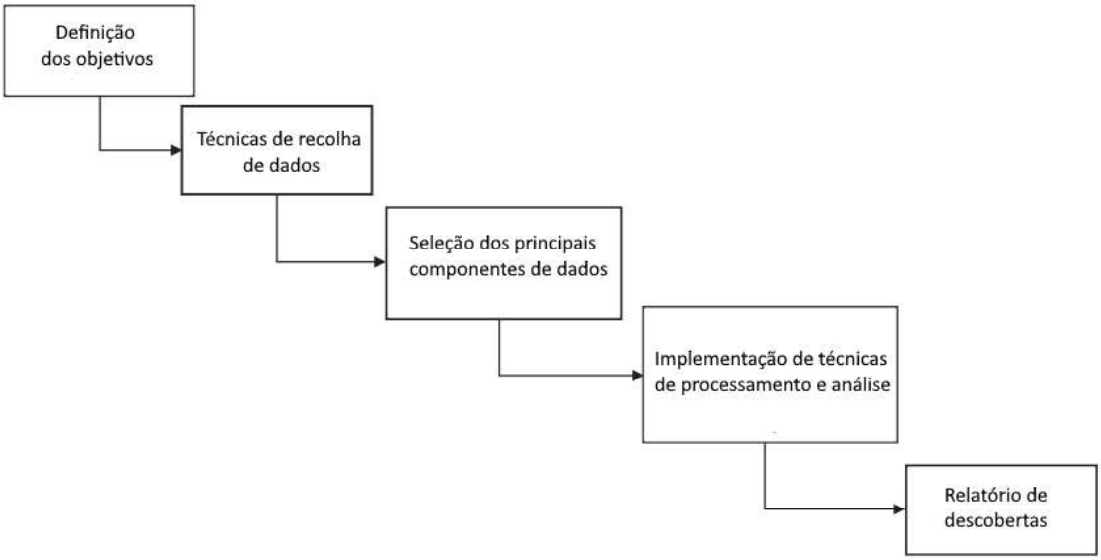

Figura 7 - Research Design Framework (Leedy & Ormrod, 2015)

3.1.1 Metodologia dos testes de auditoria

No contexto atual existem várias metodologias e padrões relacionados com testes de intrusão e avaliação de vulnerabilidades que podem ser usados para avaliar a segurança de aplicações web. Destas metodologias destacam-se o NIST SP 800-115, ISSAF (Information Systems Security Assessment Framework), o OISSG (Open Information Security Services Group), o OSSTMM (Open-Source Security Testing Methodology Manual) e o PTES (Penetration Testing Execution Standard) (Haubris & Pauli, 2013).

Os referidos autores, deixam claro que o propósito dos padrões para teste de intrusão é fornecer um esboço básico sobre o que é um teste de intrusão e dar um esboço das etapas que devem ser seguidas. No entanto, Haubris &Pauli (2013), esclarecem que os diferentes padrões que existem têm vários prós e contras e que a escolha de um deve ser baseada no propósito do teste a ser realizado.

O presente trabalho, faz uso da metodologia e padrões definidos pela NIST SP 800-115 (NIST, 2008) pelo facto de ser mais simples, objetivo, flexível, com atividades recursivas e acima de tudo, por ser um guia com recomendações amplamente aceite pelas agências reguladoras dos Estados Unidos da América.

O referido padrão apresenta recomendações práticas e procedimentos técnicos e administrativos para a realização de testes de avaliação de vulnerabilidades e ou auditorias em aplicações e sistemas informáticos. Na prática, esta metodologia de teste de intrusão identifica quatro fases ou etapas que devem ser seguidas para que se realize com sucesso testes de intrusão conforme indicado na Figura 8.

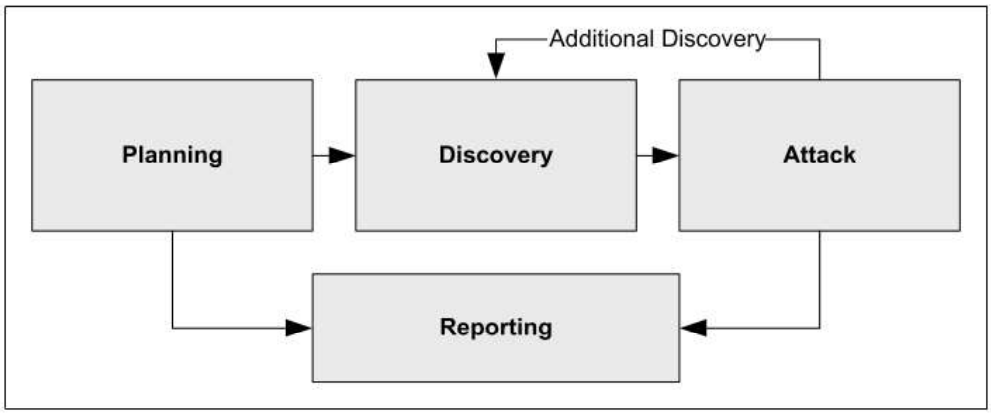

Figura 8 - Metodologia de teste de intrusão (NIST,2008)

As quatros (4) etapas que fazem parte metodologia NIST SP 800-115 são: Planeamento, a descoberta, o ataque e a elaboração de um relatório. No entanto, apesar de existir uma sequência de ações desde o planeamento ao relatório, nota-se que após a fases do planeamento existe a opção para começar a elaborar o relatório. Outro aspeto importante reside no facto desta metodologia destacar um momento de recursividade entre a fase de ataque e a da descoberta.

Os autores desta metodologia descrevem as quatro (4) fases dos testes de auditoria (Planeamento, Descoberta, Ataque e Relatório) do seguinte modo:

- **Planeamento**

  Na fase de planeamento, as regras são identificadas, a aprovação da auditoria pelas entidades é finalizada e documentada e as metas dos testes são definidas. Segundo os autores, esta fase estabelece as bases para um teste de intrusão bem-sucedido. No entanto nenhum teste real ocorre nesta fase.

- **Descoberta**

  A fase de descoberta do teste de intrusão inclui duas partes. A primeira parte é o início dos testes reais e cobre a coleta e verificação de informações. A identificação dos serviços e das portas da rede, é realizada para identificar os potenciais alvos. Além da identificação da porta e do serviço, outras técnicas são usadas para coletar informações na rede alvo.

  A segunda parte da fase de descoberta é a análise de vulnerabilidade, que envolve comparar os serviços, aplicativos e sistemas operativos avaliados com base de dados de vulnerabilidade. Segundo os autores desta metodologia o processo pode ser realizado de forma automática ou manual por auditores experientes.

- **Ataque**

  Executar um ataque é o cerne de qualquer teste de intrusão, nesta fase dá-se o processo de verificação de vulnerabilidades identificadas na fase anterior, tentando explorá-las. Os autores, esclarecem que, se um ataque for bem-sucedido, a vulnerabilidade será confirmada e serão tomadas medidas preventivas para mitigar a falha de segurança.

- **Relatório**

  A fase de relatório ocorre simultaneamente com as outras três fases do teste de intrusão. Segundo os autores, na fase de planeamento são documentados procedimentos técnicos e administrativos no relatório ao passo que nas fases de descoberta e ataque, os achados são documentados e mantidos em relatórios preliminares.

> Os autores desta metodologia, descrevem que na conclusão do teste, geralmente é desenvolvido um relatório para descrever as vulnerabilidades identificadas, apresentar uma classificação de risco e fornecer orientação sobre como mitigar as fraquezas descobertas.

3.1.2 Planificação da auditoria

O *Planeamento* é o primeiro de quatro (1/4) etapas descritas no guia NIST SP 800-115 que define a metodologia usada para a realização deste trabalho. Durante esta fase foram realizadas reuniões com todas as entidades cujo as aplicações web foram previamente validadas como estando aptas para serem auditadas.

Na prática, a pré-auditoria consistiu em realizar duas atividades uma pesquisa pelo nome das universidades nos motores de buscas da Google e Bing para depois obter o endereço dos *websites*. De seguida fez a validação dos respetivos endereços onde foram excluídas as universidades com *websites* em desenvolvimento ou quebrados. Deste modo foi possível classificar dez (10) das dezanove (19) como sendo aptas para o estudo.

Para cada instituição com *website* válido foi-lhes apresentado uma proposta para participarem num estudo acadêmico na qual as mesmas iriam se beneficiar de uma auditoria gratuita que iria validar o grau de segurança das suas aplicações web e alertá-las sobre possíveis falhas de segurança em que as mesas estão expostas.

Durante a aceitação da proposta foram discutidos e definidos os procedimentos técnicos e administrativos a serem observados antes, durante e no final da auditoria. Nos procedimentos administrativos ficou acordado que os resultados da auditoria deveriam se sigilosos, e que as verdadeiras identidades das universidades seriam salvaguardadas mediante a substituição do nome das mesmas por um código que não permitisse a sua identificação.

Tecnicamente, ficou acordado que os testes deveriam ser realizados exclusivamente pela Internet, fora das instalações das instituições, dispensando as mesmas de fornecerem quaisquer informações sobre utilizadores, sistema de informação, infraestrutura tecnológica ou do código fonte das aplicações (testes de caixa negra).

Outro aspeto acordado, foi a definição de que os testes deveriam ser feitos a noite nos fins de semana e somente durante o período de pausas ou férias acadêmicas. Também ficou acordados a entrega do relatório da auditoria as respetivas universidades.

3.1.3 Implementação da metodologia dos testes de auditoria

A aplicação prática desta metodologia pode ser resumida no fluxograma da Figura 9, que descreve o processo de realização dos testes.

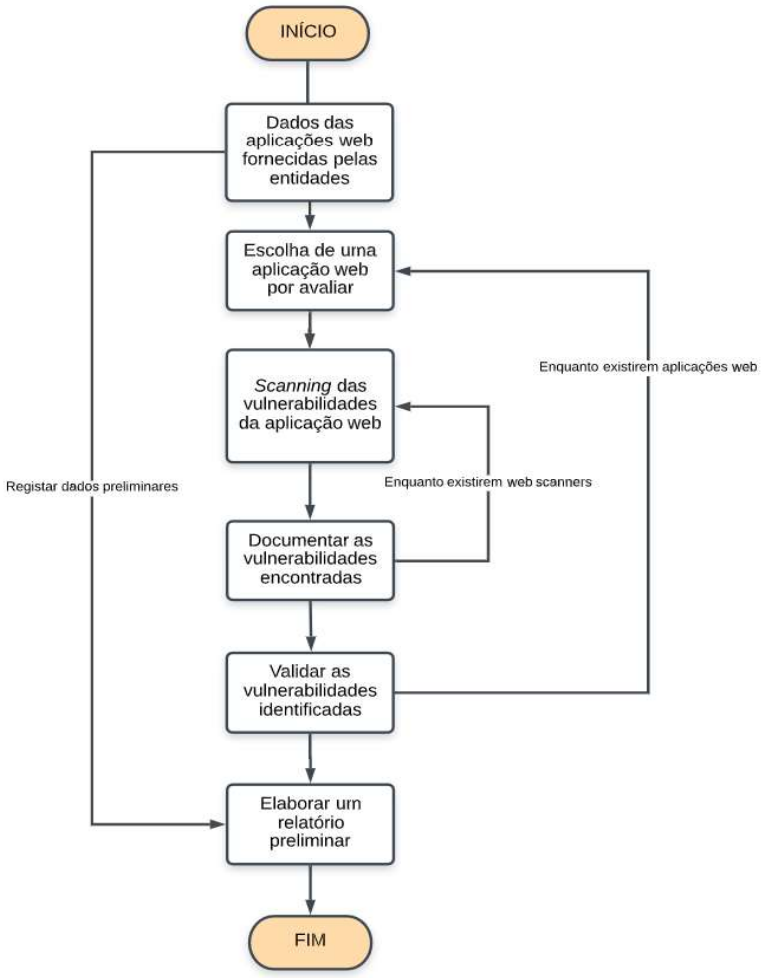

Figura 9- Visão geral do processo de auditoria[1]

A implementação a metodologia de auditoria descrita no guia NIST SP 800-115, sofreu algumas alterações e adaptações visto que o presente trabalho apenas faz uma avaliação das vulnerabilidades, logo a fase de ataque não foi realizada por fazer parte do teste intrusão.

[1] Adaptação da imagem original de (Teodoro, 2011)

Como se pode observar na Figura 9, a parte técnica da auditoria começa após a definição de um conjunto de procedimentos e técnicos e administrativos como a definição dos objetivos, escopo e regras da auditoria. Depois obtém-se uma lista das aplicações web a serem avaliadas, a seguir é realizado o *scanning* de todas as aplicações web e o resultado dos achados são documentados em relatórios preliminares.

No final deste processo é realizado a validação manual ou automática das vulnerabilidades onde são confirmadas ou não as falhas de segurança encontradas, seguindo a elaboração de um relatório final a ser entregue a entidade responsável pelas aplicações avaliadas.

A Figura 10, ilustra o procedimento técnico para a efetiva validação das vulnerabilidades encontradas.

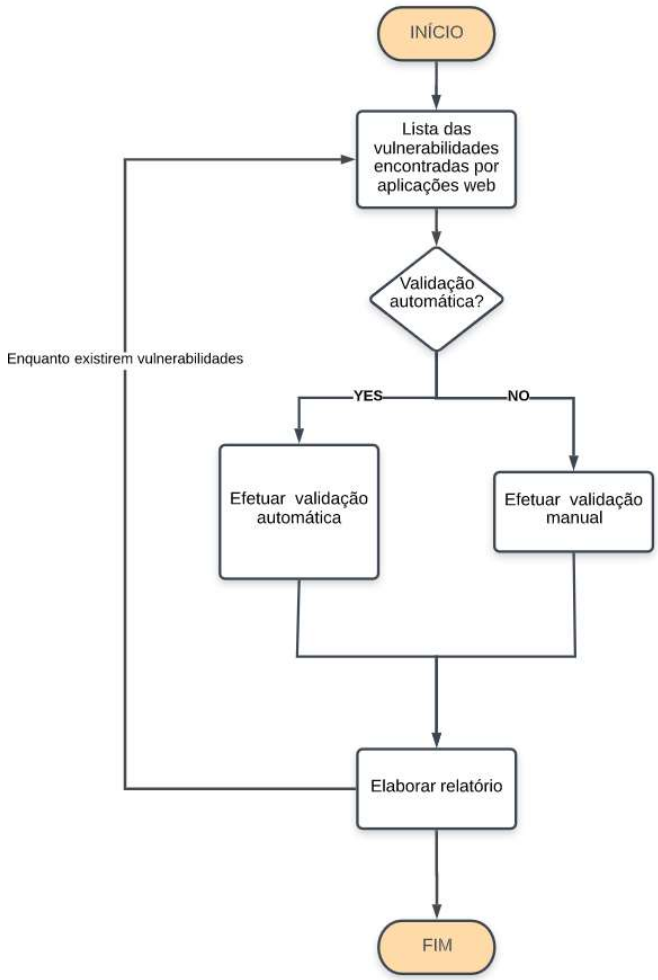

Figura 10- Metodologia para validar as vulnerabilidades encontradas

A partir da Figura 10, podemos verificar que existem duas opções para validar as vulnerabilidades encontradas: A automática e manual. A opção para validar as vulnerabilidades de forma automática faz uso dos resultados obtidos pelos outros web *scanners.* Ou seja, este procedimento consiste em selecionar uma determinada vulnerabilidade de segurança encontrada numa aplicação e verificar se para a mesma falha algum das outras três (3) ferramentas tenham reportado falha similar.

Por outro lado, a validação manual consiste em selecionar uma determinada falha de segurança encontrada e que não foi validada por uma outra ferramenta (na validação automática) e manualmente ir à aplicação web e inspecionar a veracidade do achado.

No presente trabalho a escolha pela validação manual ocorreu nos casos em falhas de segurança estavam relacionadas com a descoberta de diretórios que permitiam a listagem ficheiros suscetíveis de conterem informações sensíveis e para confirmar a existência de componentes (biblioteca e *frameworks* JavaScript) com vulnerabilidades conhecidas.

3.1.4 Ferramentas de auditoria utilizadas

Durante o processo de auditoria foram usadas as seguintes ferramentas: WhatWeb, OWASP ZAP, Skipfish, OpenVAS e Penetration-Tools (PT). A WhatWeb é uma ferramenta de reconhecimento de tecnologias da web enquanto as demais são ferramentas web *scanning.*

A WhatWeb foi a primeira ferramenta a ser utilizada, visto que ela foi usada para reconhecer ou colher dados como endereço IP, localização geográfica dos servidores, e informações sobre os sistemas de gestão dos conteúdos de conteúdo (CMS), plataformas de blog, identificação das versões das bibliotecas JavaScript e CSS, descrição das linguagem de programação utilizadas no desenvolvimento de certas aplicações, tipos de servidores da web e respetivas versões (Kali, 2020).

Ela é uma ferramenta *open-source* e tem mais de 1700 *plug-ins*, que lhe conferem uma grande capacidade para diferentes tipos de tecnologias web.

As ferramentas OWASP ZAP, Skipfish, OpenVAS e Penetration-Tools são web *scanners,* isto é, são ferramentas automatizadas que fazem a varredura de aplicações web, normalmente executados de fora para dentro da instituição a ser auditada, e serve para procurar vulnerabilidades de segurança, que podem comprometer a curto, médio ou longo

prazo o normal funcionamento das aplicações. Estas aplicações também têm a capacidade para detetar a configuração incorreta de diferentes aplicações, ativos de rede e aplicações em fim de vida.

Algumas destas ferramentas possuem funcionalidades que permitem efetuar a gestão das vulnerabilidades encontradas ao longo de um período e possuem um sistema de verificação de falhas de segurança que as permitem efetuar recomendações sobre os procedimentos técnicos que devem ser executados para mitigar ou eliminar as falhas encontradas.

Dos web *scanners* utilizados, três são *open-source* (OWASP ZAP, Skipfish, OpenVAS) enquanto a Pentest-Tools é software proprietário baseado numa solução que funciona a partir de uma *cloud*.

3.1.5 Recursos computacionais de apoio a auditorias

Para execução técnica da auditoria, foi necessário utilizar e configurar alguns recursos relacionados com hardware, software e conexão acesso a Internet de banda larga 197.5 Mbps download e 4.16 Mbps upload). O uso de máquinas virtuais permitiu que os testes fossem executados na mesma máquina física apesar de terem sido usados dois tipos de sistemas operativos diferentes (Windows e Linux), com diferentes configurações de memória.

O *scanner* OAWSP ZAP (versão 2.8.0) foi instalado diretamente numa máquina Lenovo ThinkPad com Windows 10 Enterprise, 24 Gb de RAM, Intel(R) Core (TM) i7 -8650 CPU @1.90GHz, ao passo que os *scanners* Skipfish e OpenVAS foram executados numa máquina virtual (Oracle VM VirtualBox V 6.1), sendo que para a Skipfish (versão 2.10b) foi executada dentro do Kali Linux (versão 2020.2) ao passo que o OpenVAS (versão 9) foi utilizado dentro da versão customizada fornecida pelo projeto Greenbone, conforme apresentado na Tabela 6.

A avaliação das vulnerabilidades de segurança com recurso ao *scanner* Pentest-Tools ou PT como também é conhecida, foi efetuado numa solução hospedada numa *cloud* (Pentest-Tools.com), pelo que a empresa proprietária do *software* não divulga informações sobre o *hardware* usado para hospedar a aplicação e tão pouco divulga outros recursos computacionais alocado aos testes.

Os *scanners* utilizados nos testes de vulnerabilidades foram configurados para executar os testes num período máximo de 23 horas, a exceção à regra foram os testes realizados com a ferramenta PT, que no momento da realização dos testes não possuía este tipo de parametrização. Os testes realizados consumiram cerca de 69 horas. O *scanner* PT gastou cerca de 28 horas e foi a ferramenta que mais tempo levou para executar os testes, ao passo que Skipfish levou cerca de 10 horas para concluir faseadamente todos os testes.

Tabela 6 - Recursos computacionais

Web *Scanner*	***Equipamento***	**Duração**
ZAP	Windows 10 Enterprise, 24 Gb de RAM, Intel(R) Core (TM) i7 -8650 CPU @1.90GHz	12 horas
Skipfish	Kali Linux, Oracle MV VirtualBox, 5 Gb de RAM, Intel(R) Core (TM) i7 -8650 CPU @1.90GHz	10 horas
OpenVAS	Kali Linux, Oracle MV VirtualBox, 5 Gb de RAM, Intel(R) Core (TM) i7 -8650 CPU @1.90GHz	18 horas
PT	*Cloud Computing Services*	28 horas

3.1.6 Fontes de dados

As fontes de dados primárias foram obtidas na sequência da realização dos testes de intrusão das instituições de ensino superior de Angola, reconhecida pela instituição de tutela, o Ministério do Ensino Superior e Ciência e Tecnologia de Angola, publicada no ano de 2020 (MESCTI, 2020).

O procedimento começava com validação do *URL* principal do website da universidade, e a partir deste listar todos os links existentes no domínio para de seguida avaliar o mesmo. Durante o período de avaliação, para cada link encontrado foi automaticamente verificado a existência de formulários, que por sua vez foram sujeitos a diferentes tipos de testes para validar o grau de vulnerabilidades que os mesmos apresentavam.

As fontes de dados secundárias foram obtidas com recurso a ferramenta WhatWeb, aplicação *open source* distribuída como *package* da distribuição Linux derivada do Debian projetada para análise forense digital e testes de intrusão (Kali Linux).

Os dados, foram obtidos com recurso a quatro ferramentas, sendo uma proprietária (Pentest-Tools.com) e as demais *opens source* que permitem a realização de testes automatizados (Skipfish, OWASP ZAP, OpenVAS) que exploram as falhas mais frequentemente encontradas nas aplicações web. Nos casos em que os resultados eram falsos positivos foi feita a verificação e validação manual dos mesmos.

Os testes autorizados pelas instituições, seguiram uma abordagem ética e foram exclusivamente realizados a partir da Internet, pelo que alguns tipos de dados que poderiam ser obtidos mediante a realização de testes mais exaustivos e intrusivos que poderiam causar quebras parciais ou paragem dos serviços ou dos sistemas informáticos foram ignorados para salvaguardar a continuidade do negócio das instituições em causa.

3.1.7 Análise dos dados

A primeira parte do trabalho incluiu a pesquisa e revisão bibliográfica, na qual foi dada preferência às dissertações de mestrado e doutoramento, artigos científicos, livros, e-*books* e publicações *online* das revistas de especialidade (*journals*), que continham artigos sobre avaliação de *website*s, design, redes ou médias sociais, ou seja, sobre a presença das instituições na Internet.

Na segunda parte, foram selecionadas as vinte (20) características que além de permitem avaliar riscos de segurança também serviram para uniformizar os resultados deste trabalho. Dez das vinte (10/20) características escolhidas fazem parte de um amplo consenso sobre os riscos de segurança mais críticos para aplicações da web e que podem ser encontrados no projeto da OWAP (OWASP, 2017b).

As restantes 10 (dez) características fazem parte da CWE/SANS Top 25 que listas as 25 (vinte e cinco) elementos que são classificadas como sendo as mais perigosas falhas de software (Martin, Paller, Kirby, & Christey, 2011). É importante referenciar que as vinte (20) características abaixo enumeradas possuem várias sub-características que não foram aqui listadas por uma questão de conveniência. Para o presente estudo foram escolhidas as seguintes características:

- Injeção de SQL.
- Divulgação interna de IP.
- Interface de administração insegura.
- Fixação de sessão (*Session fixation*).
- *Cross-Site Scripting* (XSS).
- Inclusão remota de arquivos.
- Injeção de comando do SO.
- Vulnerabilidade de XML entidade externa (XXE).
- Redirecionamento aberto e ou não validados.
- Tipo de sessão que não expira.
- Uso de credenciais padrão.
- Funcionalidade de depuração presente (modo *debug*).
- Inclusão de arquivo local.
- Tipo de autenticação.
- Tipo de validação de cookies (encriptação).
- Uso de comunicação segura (SSL).
- Uso de software com versões desatualizado.
- Servidor web desatualizados.
- Uso de CMS com vulnerabilidades conhecidas.
- Má configuração das aplicações e servidores web.

O processo de avaliação de vulnerabilidades foi realizado com base em testes passivos e não intrusivos que permitiram coletar, avaliar os diferentes tipos de vulnerabilidades que podem ser encontradas nas aplicações web. A ferramenta OWASP ZAP ou simplesmente ZAP foi utilizada como instrumento principal para a validar as caraterísticas obrigatórias enumeradas. Esta ferramenta foi a única das três utilizadas no presente estudo que conseguiu realizar com sucesso os testes propostos em todos websites.

A ferramenta OpenVAS, apenas conseguiu realizar testes completos de avaliação de segurança em 8 (oito) das 10 (dez) universidades. Esta situação ocorre nos 2 (dois) websites que estão sobre alçada da empresa CloudFlare, uma rede de distribuição de conteúdo, cujo seus recursos incluem proteção contra ataque de negação de serviço (*DDoS*) e funcionalidade de firewall de aplicações web (Cloudflare, 2020).

A primeira parte deste estudo no período compreendido entre realizado no período compreendido entre 01/05/2018 à 25/06/2018 e a segunda entre 20/05/2020 à 25/08/2020 e envolveu dez (10) websites das universidades Angolanas que estavam *online* durante a realização do estudo.

3.1.8 Classificação

Existem muitas abordagens para a análise de risco. Algumas abordagens envolvem a ameaça, a vulnerabilidade e o impacto (Lubis & Tarigan, 2017), (Liu, Tan, Fang, & Lok, 2012). Sendo que teríamos genericamente a seguinte fórmula:

Risco = Ameaça * Vulnerabilidade * Impacto

A OWASP (2017), apresenta o um modelo adaptável de acordo com a instituição, simplificado e aplicável as aplicações web, onde o destaque vai para a probabilidade e impacto. Desta forma teríamos a seguinte fórmula:

Risco = Probabilidade * Impacto

Neste modelo, faz-se uma estimativa de probabilidade de ocorrência exploração da vulnerabilidade e estima-se o impacto da mesma pode causar, estes dois elementos servem para calcular o grau de severidade geral risco que calculado qualitativamente pode ser baixo, médio ou alto e qualitativamente os valores vão de 0 a 9 conforme ilustrado na Figura 11.

Níveis de probabilidade e impacto	
0 to <3	BAIXO
3 to <6	MÉDIO
6 to 9	ALTO

*Figura 11- Níveis de probabilidade e impacto (*OWASP, 2019*)*

No modelo da OWASP (2019), a probabilidade de ocorrência de ataque e o seu impacto é quantificado (Figura 11), sendo que numa escala de 0 a 9, é atribuído a classificação de baixo quando os valores atribuídos forem inferiores a 3, médio quando os valores forem superiores ou iguais a 3 e inferior a 6 ao passo que os valores entre 6 e 9 são considerados como altos.

Os graus de classificação obtidos com a criação dos níveis de probabilidade e impacto (Figura 11), permitiram a OWASP a criar uma matriz de avaliação de risco (Figura 12) que permite identificar e calcular rapidamente o grau de criticidade das vulnerabilidades de segurança.

Gravidade de risco geral				
Impacto	ALTO	Médio	Alto	Crítico
	MÉDIO	Baixo	Médio	Alto
	BAIXO	Notas	Baixo	Médio
		BAIXO	MÉDIO	ALTO
	Probabilidade			

Figura 12 - Matriz de avaliação de risco (OWASP, 2019)

A matriz para o cálculo do risco apresentado pela OWASP (Figura 12) pode ser facilmente calculado, bastando para o efeito encontrar o ponto de interseção do impacto em relação a probabilidade. Por exemplo, se uma determinada vulnerabilidade tem um impacto alto e a probabilidade da ocorrência da mesma for baixa então o seu é médio.

No presente estudo, foi tido em consideração a matriz de avaliação de risco da OWASP, para classificar o risco das vulnerabilidades encontradas. Os graus de criticidade foram classificados com base na sua perigosidade, ou seja, todas as vulnerabilidades existentes que quando exploradas comprometem imediatamente a segurança da aplicação e que podem colocar em causa a continuidade do negócio foram classificadas como sendo do tipo alto, por exemplo: “Injeção de SQL”.

No sentido oposto, todas a vulnerabilidades de segurança que apenas se limitavam a fornecer informações sobre as aplicações e sobre os ecossistemas tais como versões dos sistemas operativos, versões dos servidores web e de outras aplicações foram classificadas como sendo de pouca gravidade (nível baixo).

As vulnerabilidades de segurança que não foram considerados como sendo muito grave ou de pouca gravidade passaram a ser consideradas como sendo se nível médio. Exemplo: *Cookies* HTTP inseguros.

O processo de aplicação da fórmula de classificação do risco, do nível de gravidade e classificação do impacto baseado na matriz da OWASP neste estudo foi calculado, implementado e validado automaticamente pela ferramenta *open source* OWASP ZAP que faz parte de um dos projetos desenvolvido pela própria OWASP. Esta ferramenta permite gerar um relatório com os resultados da avaliação de segurança, nela é descrito a quantidade de vulnerabilidades encontradas bem coma a respetiva classificação de risco (baixo, médio e alto).

A ferramenta Skipfish baseia-se no impacto que a vulnerabilidade pode causar no sistema e nos dados para classificar automaticamente o risco, que pode ser resumido em baixo, médio, alto. A Tabela 7, descreve a relação entre o impacto das vulnerabilidades e o risco de segurança.

Tabela 7- Classificação de vulnerabilidades com Skipfish

Impacto	**Risco**
Vulnerabilidades cujo impacto não comprometem nem o sistema ou os dados. Exemplo: uso de formulários HTML sem proteção XSRF (*Cross-site Request Forgery*)	BAIXO
Vulnerabilidades que potencialmente podem levar ao comprometimento dos dados. Exemplo: Submissão de passwords em formulários sem SSL.	MÉDIO
Vulnerabilidades que potencialmente podem levar ao comprometimento do sistema. Exemplo: Injeção de SQL	ALTO

Fonte: (Google, 2012)

A OpenVAS (*Open Vulnerability Assessment System)* é uma ferramenta que além de permitir a avaliação de segurança, monitoriza e faz a gestão continua das vulnerabilidades. Ela gera automaticamente a relatórios com vários tipos de gráficos com os diferentes tipos de vulnerabilidades encontradas. Esta aplicação baseia-se na National Vulnerability Database (NVD) uma base de dados do governo dos Estados Unidos da América que regista e inclui listas de verificação de segurança, falhas de software relacionadas à segurança, configurações incorretas, nomes de produtos e métricas que permitem identificar, calcular o impacto das vulnerabilidades sobre a segurança.

Para definir o grau de severidade, o OpenVAS recorre ao Common Vulnerability Scoring System (CVSS) um instrumento padrão, gratuito e aberto para avaliar a gravidade das vulnerabilidades de segurança do sistema de informáticos, que na sua versão 2, classifica-os como sendo baixo, médio e alto (NVD, 2020).

Todas a ferramentas utilizadas neste estudo, seguem o mesmo padrão comportamental ao encontrarem uma falha de segurança, ou seja, para cada vulnerabilidade encontrada, as ferramentas identificam univocamente as falhas mediante um código de identificação padronizado (CWE), a partir do qual é possível obter informações detalhadas sobre a respetiva falha de segurança, usando a NVD ou outra base de dados.

De igual modo, todas a ferramentas utilizadas apresentam possíveis soluções, ou sequência de instruções de devem ser dadas para mitigar ou eliminar a respetiva vulnerabilidade.

A Pentest-Tools (PT) é uma ferramenta proprietária de teste de intrusão online que permite descobrir e relatar vulnerabilidades em sites e infraestruturas de rede. Esta ferramenta possui um conjunto de funcionalidades que vão desde o *scanning* passando pela geração automática de relatórios, com os achados (dados estatísticos e elementos gráficos e inclusão de sugestões sobre como mitigar ou eliminar as falhas de seguranças encontradas), classificação dos diferentes graus de vulnerabilidades e para cada falha de segurança a ferramenta descreve um sumário a relacionar as vulnerabilidades encontradas com as falhas mais frequentes que podem ser encontradas em aplicações *W*eb.

Além disto, a ferramenta também permite a gestão centralizada das vulnerabilidades encontradas ao longo de um período, mostrando automaticamente o grau de melhoria e ou degradação das falhas de segurança encontradas ao longo do tempo.

Capítulo 4 Resultados

4.1 Resultados individuais por *scanners*

Em Angola existem cerca de dezanove (19) universidades, no entanto durante a realização deste estudo nove (9) delas não possuíam um website em pleno funcionamento por problemas técnicos ou pela ausência de um *website* oficial, condição que levou a exclusão das mesmas. No entanto, por motivos de confidencialidade optou-se omitir o nome das dez (10) instituições e as respetivas aplicações que foram alvos de auditoria neste trabalho, substituindo-as por um nome codificado apenas por questão estatística.

Ao longo deste trabalho foram utilizadas quatro (4) ferramentas, dos quais três (3) *open-source* (OWASP ZAP, Skipfish, OpenVAS) e uma proprietária (Pentest-Tools). Estas ferramentas foram selecionadas sobretudo pela capacidade que as mesmas possuem e gerar artefatos. Ou seja, capacidade para fazer *scanning* automático e recomeçar o procedimento em caso de falhas na execução dos mesmos, capacidade em gerar gráficos, relatórios com a descrição dos achados, bem como a classificação automática das vulnerabilidades encontradas tendo em o padrão CWE.

A existência de várias comunidades a volta das ferramentas usadas, e o facto das mesmas terem sido usadas com sucesso em estudos relacionados com testes de intrusão (Ottosson, 2018); (OWASP, 2014); (Teodoro, 2011); (Amankwah et al., 2020), influenciaram a escolha dos respetivos web *scanners*.

A escolha do *scanner* proprietário deveu-se ao facto do mesmo efetuar o *scanning* automático e produzir relatórios que incluem elementos gráficos, que facilitam a rápida leitura dos achados bem como a inclusão de possíveis soluções para resolver e mitigar as falhas de segurança encontradas. A ferramenta PT também identifica as falhas de segurança encontradas com base no padrão CWE, caraterística relevante para o presente trabalho.

O presente trabalho usa a matriz de avaliação de risco da OWASP (2019), que designa como alta, média e baixa as vulnerabilidades de segurança tendo em conta a gravidade, o impacto que as mesmas podem causar e a probabilidade de as mesmas ocorrerem, conforme ilustrado anteriormente na Figura 11.

4.1.1 OWASP ZAP

O uso da ferramenta ZAP permitiu apurar que 30% das universidades avaliadas apresentavam tipos de vulnerabilidades de alto risco de segurança como a mais alta classificação, (70%) apresentaram vulnerabilidades de segurança do nível médio ao passo que nenhuma delas apresentou apenas vulnerabilidades de segurança de baixa gravidade como a classificação mais alta, conforme ilustrado na Figura 13.

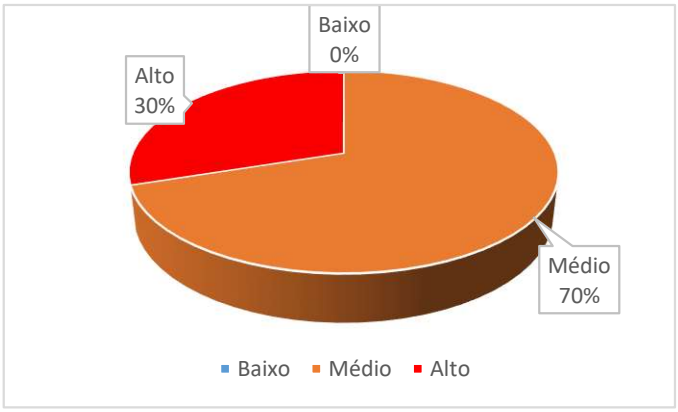

Figura 13- Grau de severidade usando ZAP

A Figura 14, apresenta os resultados da avaliação geral das vulnerabilidades das universidades angolanas. Os resultados apresentados foram agrupados por categorias tendo em conta o grau de criticidade que as mesmas vulnerabilidades representam.

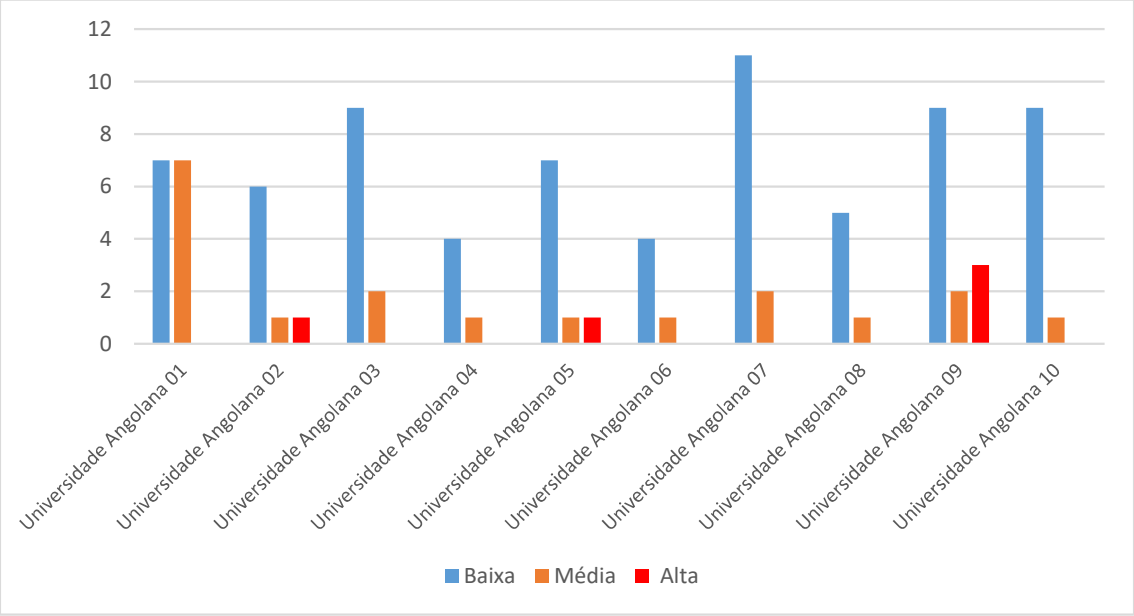

Figura 14- Avaliação das vulnerabilidades usando ZAP

A partir da Figura 14, podemos constatar que todas as universidades avaliadas apresentaram pelo menos alguma vulnerabilidade de segurança de média e baixa gravidade. No entanto três (3) das dez (10) universidades apresentam falhas graves.

O uso da referida ferramenta, conforme ilustrados na Figura 15, permitiu aferir que quantitativamente foram encontradas setenta (70) falhas ou vulnerabilidades de baixa gravidade, dezanove (19) de nível médio e cinco (5) de alta gravidade.

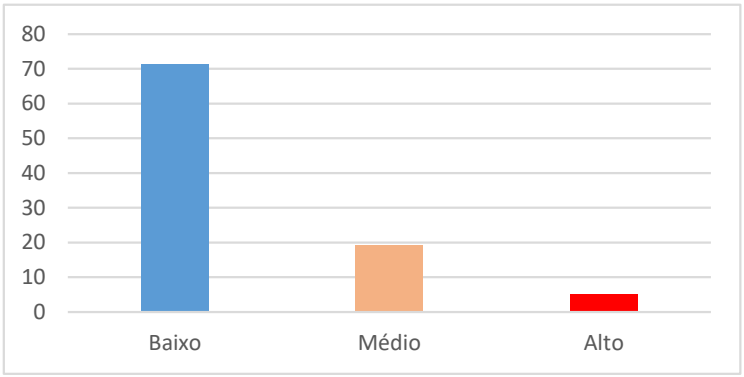

Figura 15 - Resultado quantitativo da avaliação dos riscos usando ZAP

Adicionalmente, na Figura 16, podemos verificar que pelo menos cinco (5) das dez (10) falhas mais frequentes encontradas nas aplicações web foram descobertas pelo *scanner* ZAP. Não obstante a ferramenta ter descoberto um tipo de falha muito grave (Injeção), a categoria com o maior número de falhas está relacionada com a exposição de dados sensíveis, seguindo-se a má proteção de com *scripts* entre *sites* (*XXS*), má configuração de segurança e a utilização de componentes com vulnerabilidades conhecidas.

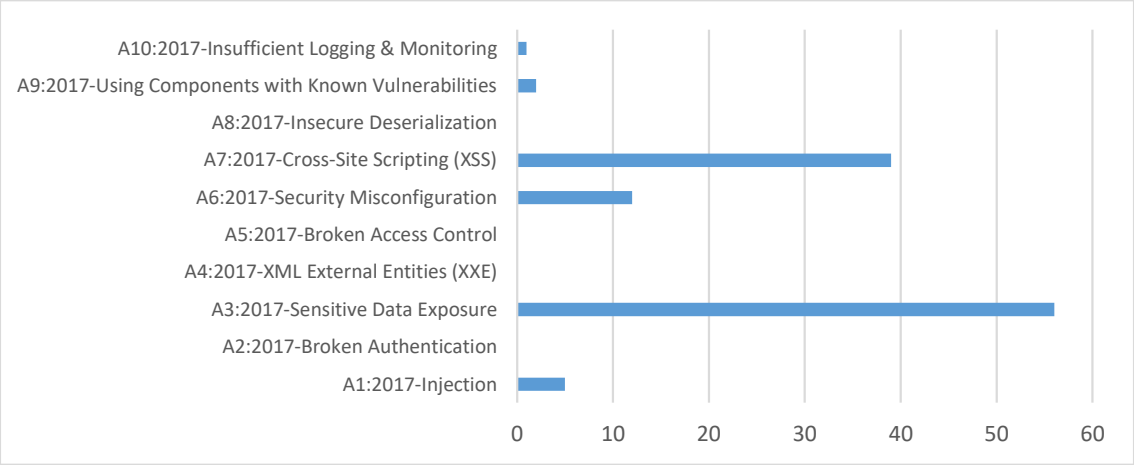

Figura 16- Distribuição dos resultados no OWASP Top 10 usando o ZAP

4.1.2 Skipfish

A utilização da ferramenta Skipfish permitiu apurar que 40% das universidades avaliadas apresentavam tipos de vulnerabilidades de alto risco como a mais alta classificação, (50%) apresentam vulnerabilidades de nível médio ao passo que (10%) delas apresentou um nível gravidade baixo como classificação mais alta, conforme ilustrado na Figura 17.

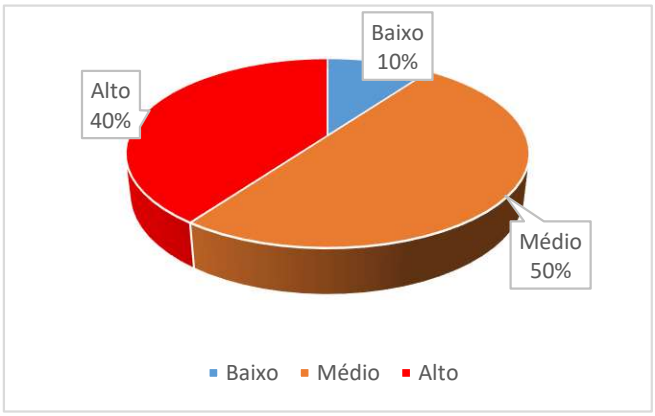

Figura 17- Grau de severidade usando Skipfish

A utilização da ferramenta Skipfish conforme ilustrado na Figura 18, permitiu apurar que todas as universidades avaliadas apresentam pelo menos alguma vulnerabilidade e baixa gravidade. No entanto quatro (4) das dez (10) universidades apresentam falhas graves.

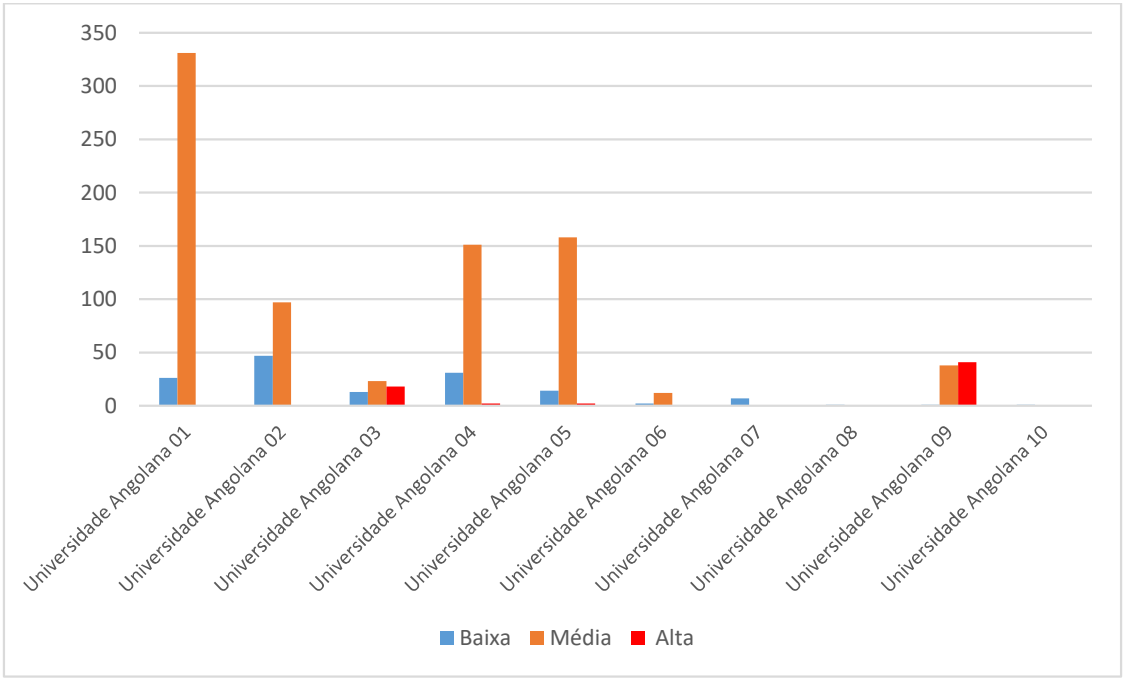

Figura 18 - Avaliação das vulnerabilidades com Skipfish

O uso da referida ferramenta, conforme ilustrados na Figura 19, permitiu aferir que quantitativamente foram encontradas cento e quarenta e três (143) falhas ou vulnerabilidades de baixa gravidade, oitocentos e dez (810) de nível médio e sessenta e três (63) falhas de alta gravidade.

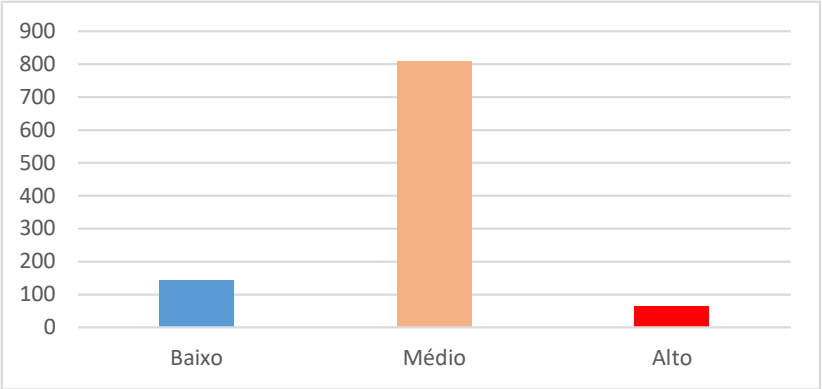

Figura 19- Resultado quantitativo da avaliação dos riscos usando Skipfish

Adicionalmente, na Figura 20, podemos verificar que pelo menos quatro (4) das dez (10) falhas mais frequentes encontradas nas aplicações web foram descobertas pelo scanner Skipfish. Não obstante a ferramenta ter descoberto um tipo de falha muito grave (Injeção), a categoria com o maior número de falhas está relacionada com a exposição de dados sensíveis, seguindo-se a má proteção de com *scripts* entre *sites* (*XXS*) e a má configuração de segurança.

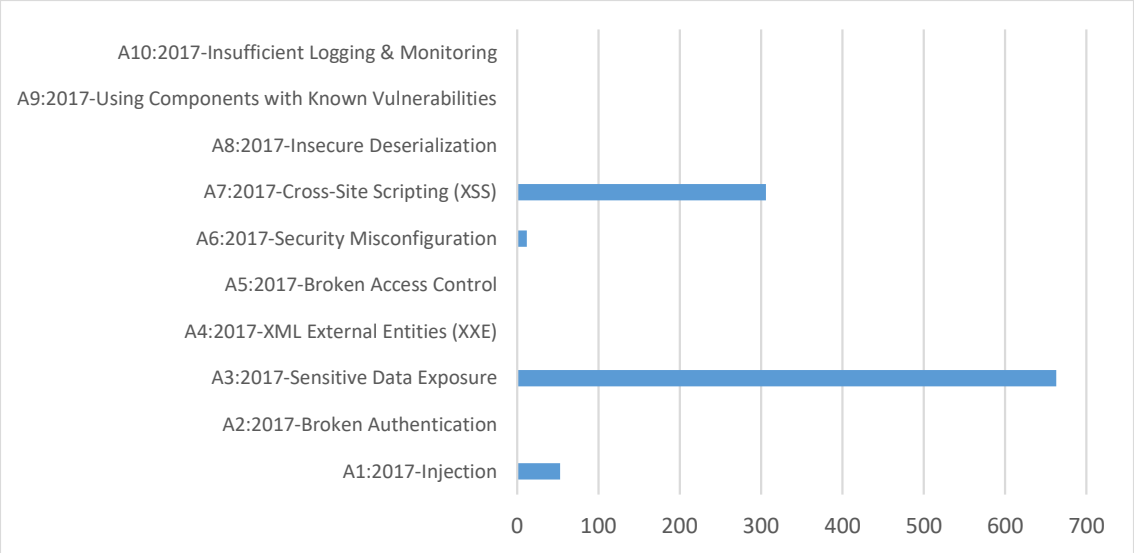

Figura 20- Distribuição dos resultados no OWASP Top 10 usando o Skipfish

4.1.3 OpenVAS

A terceira ferramenta utilizada neste estudo foi a OpenVAS, este *web scanners* não conseguiu transpor certas limitações técnica (*firewall*) de duas aplicações web. No entanto, oito (8) das dez (10) universidades foram avaliadas com sucesso, permitindo aferir que 25% delas apresentavam falhas de alta gravidade como classificação mais alta, 25% apresentam vulnerabilidades de nível médio ao passo que as restantes (25%) apresentavam um nível gravidade baixo como classificação mais alta, conforme ilustrado na Figura 21.

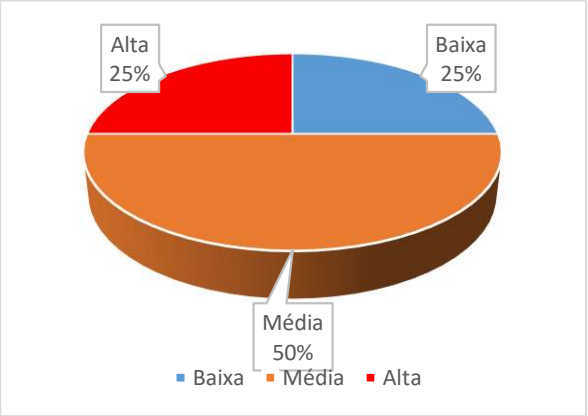

Figura 21- Grau de severidade usando OpenVAS

A utilização da ferramenta OpenVAS conforme ilustrado na Figura 22, permitiu apurar que os resultados da avaliação de duas (2) universidades foram inconclusivos e outras duas (2) não apresentaram qualquer falha de segurança. No entanto duas (2) das dez (10) universidades apresentam falhas graves.

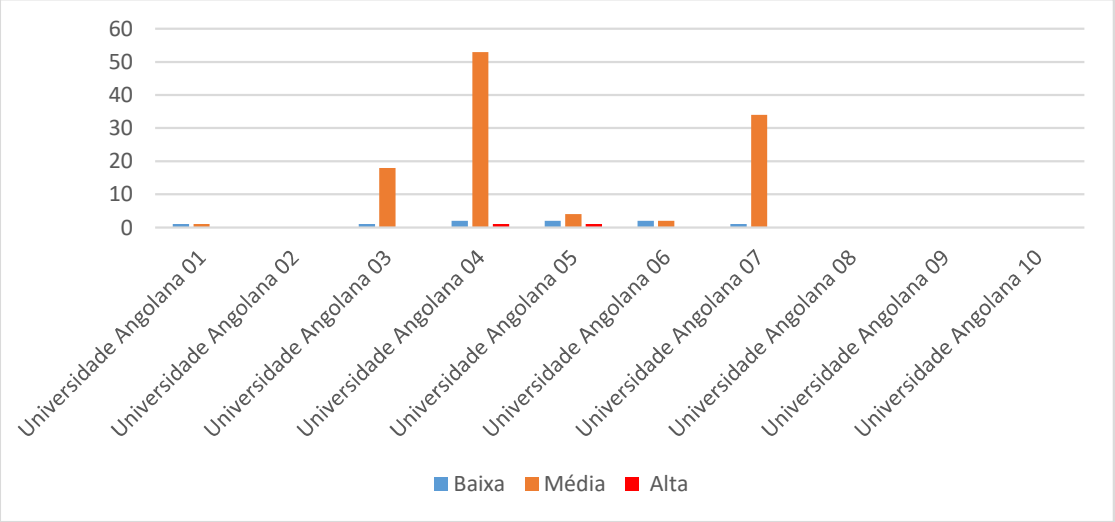

Figura 22- Avaliação das vulnerabilidades com OpenVAS

Apesar da existência de limitações técnicas acima enumeradas, os resultados obtidos com a utilização da ferramenta OpenVAS (Figura 23) permitiu aferir que quantitativamente foram encontradas nove (9) vulnerabilidades de baixa gravidade, cento e doze (112) de nível médio e duas (2) falhas de segurança de alta gravidade.

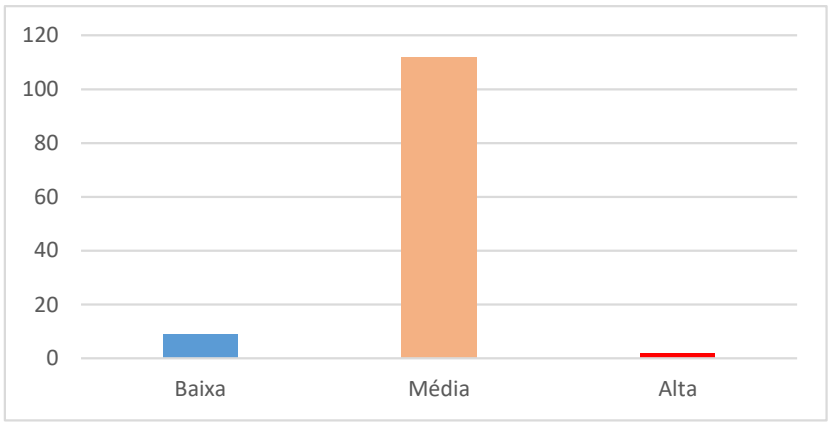

Figura 23 - Resultado quantitativo da avaliação dos riscos usando OpenVAS

Adicionalmente, na Figura 24, podemos verificar que pelo menos quatro (4) das dez (10) falhas mais frequentes encontradas nas aplicações web foram descobertas pelo *scanner* OpenVAS. Não obstante a ferramenta ter descoberto um tipo de falha muito grave (Injeção), a categoria com o maior número de falhas está relacionada com a má configuração de segurança, seguindo-se a má proteção de com *scripts* entre *sites* (*XXS*) e a exposição de dados sensíveis.

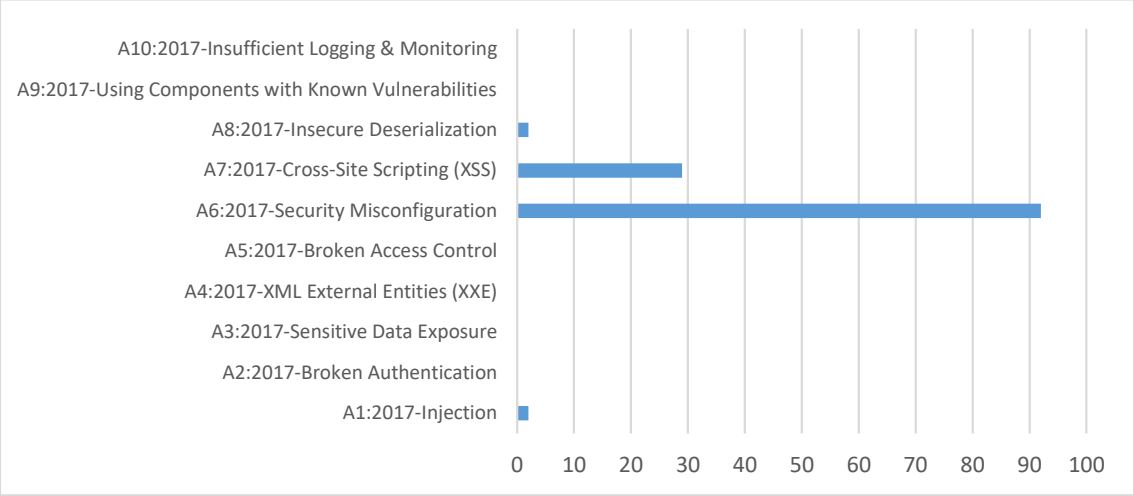

Figura 24- Distribuição dos resultados no OWASP Top 10 usando o OpenVAS

4.1.4 Pentest-Tools (PT)

O uso da ferramenta PT, permitiu apurar que 38% das universidades avaliadas apresentavam tipos de vulnerabilidades de alto risco como a mais alta classificação, (54%) apresentaram vulnerabilidades do nível médio e as restantes (8%) apresentaram vulnerabilidades de baixa gravidade, conforme ilustrado na Figura 25.

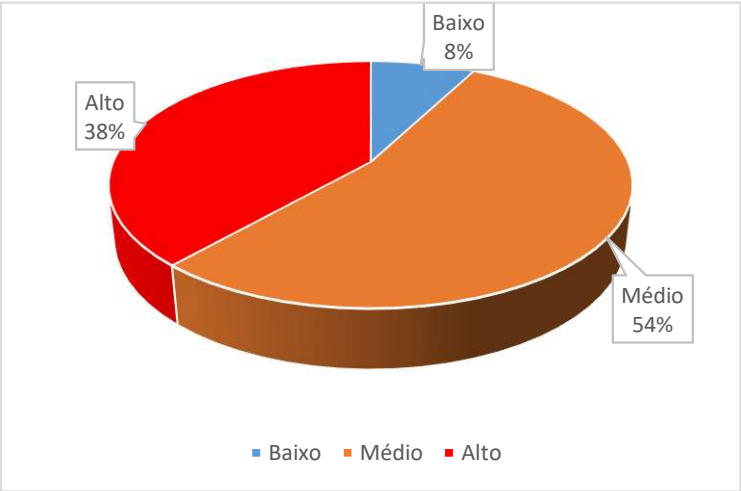

Figura 25- Grau de severidade usando PT

A Figura 26, apresenta os resultados da avaliação geral das vulnerabilidades das universidades angolanas. Os resultados apresentados foram agrupados por categorias tendo em conta o grau de criticidade que as mesmas vulnerabilidades representam.

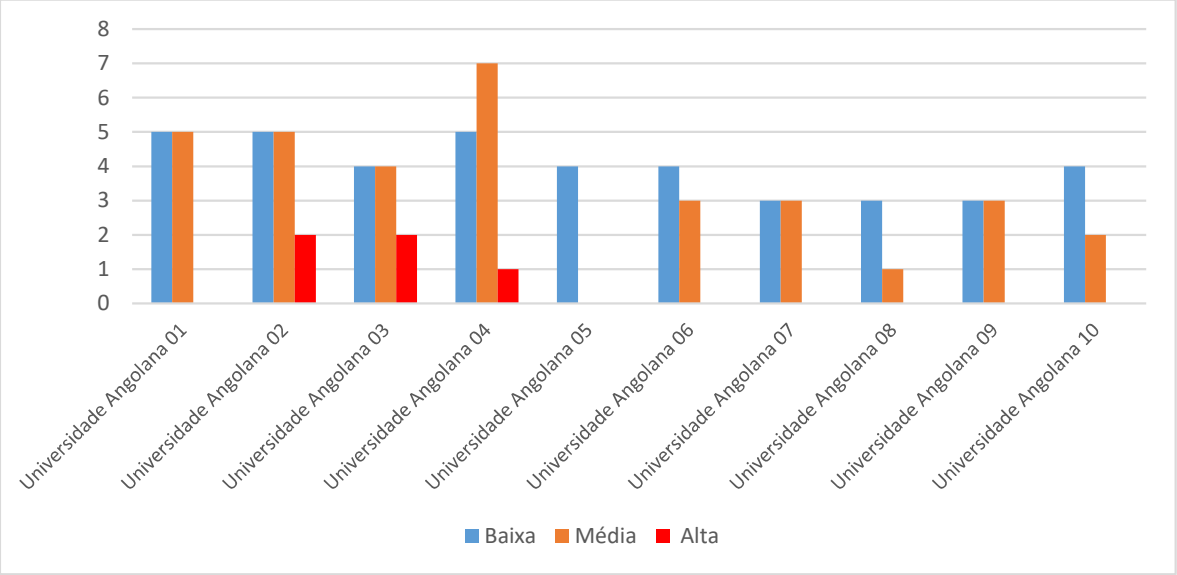

Figura 26- Avaliação das vulnerabilidades usando PT

A partir da Figura 26, podemos constatar que todas as universidades avaliadas apresentaram pelo menos alguma vulnerabilidade baixa gravidade. No entanto cinco (5) das dez (10) universidades apresentam falhas graves.

O uso da referida ferramenta, conforme ilustrados na Figura 27, permitiu aferir que quantitativamente foram encontradas quarenta (40) falhas ou vulnerabilidades de baixa gravidade, trinta e três (33) de nível médio e cinco (5) de alta gravidade.

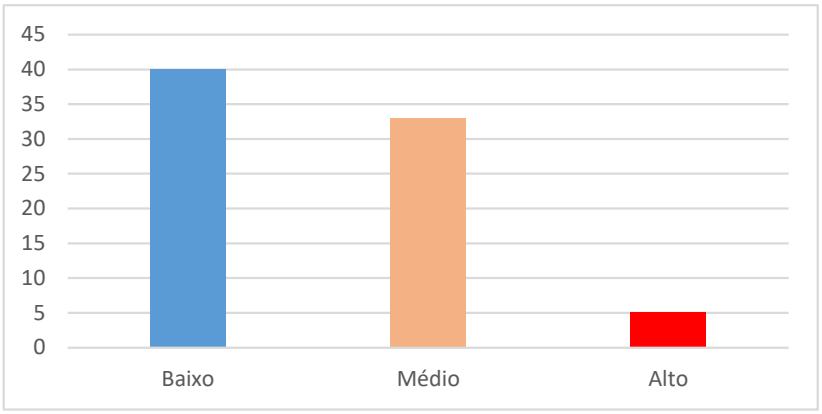

Figura 27 - Resultado quantitativo da avaliação dos riscos usando PT

Adicionalmente, na Figura 28, podemos verificar que pelo menos seis (6) das dez (10) falhas mais frequentes encontradas nas aplicações web foram descobertas pelo *scanner* PT. Não obstante a ferramenta ter descoberto um tipo de falha muito grave (Injeção), a categoria com o maior número de falhas está relacionada com a má configuração de segurança, seguindo-se a exposição de dados sensíveis, má proteção de com *scripts* entre *sites* (*XXS*), e falhas relacionadas com a autenticação e proteção de áreas reservadas.

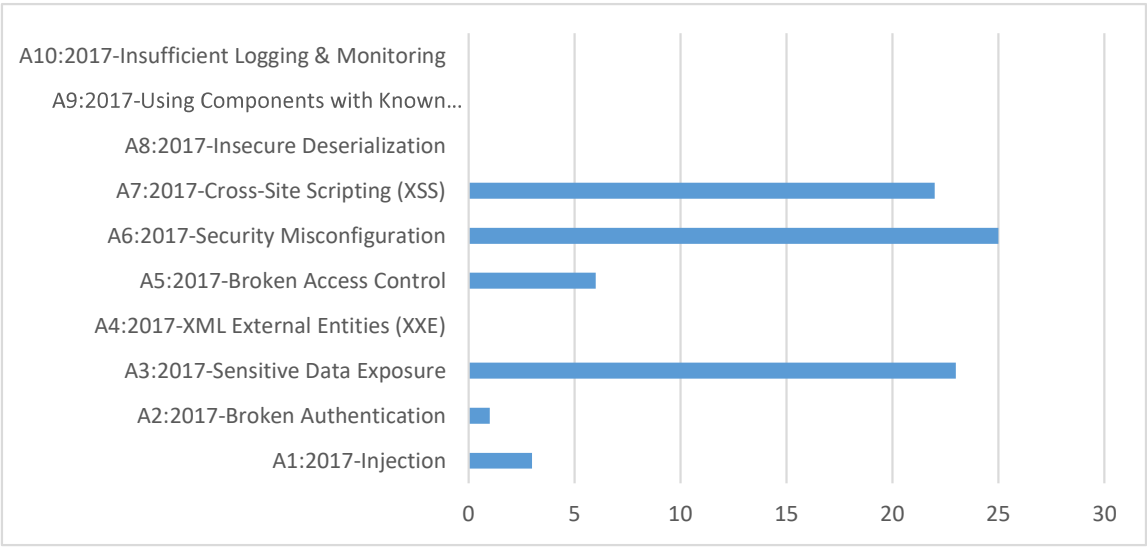

Figura 28 - Distribuição dos resultados no OWASP Top 10 usando o PT

4.2 Resultados coletivos por web scanners

Os resultados da avaliação de segurança com recurso a quatro (4) ferramentas de avaliação de vulnerabilidades (Tabela 8), permitiu por um lado verificar que em 90% dos casos pelo menos duas das três (3) ferramentas identificaram o nível de gravidade, ou seja, corroboraram com o mesmo resultado. Por outro lado, em 30% dos casos as quatro (4) ferramentas apresentaram exatamente o mesmo nível de gravidade, aumentando desta forma o grau de confiança dos resultados apresentados.

Tabela 8- Resultados qualitativos por web scanners

Universidades	**Avaliação geral qualitativa**			
	ZAP	**Skipfish**	**OpenVAS**	**PT**
Universidade Angolana 01	Média	Média	Média	Média
Universidade Angolana 02	Alta	Média	Baixa	Alta
Universidade Angolana 03	Média	Alta	Média	Alta
Universidade Angolana 04	Média	Alta	Alta	Média
Universidade Angolana 05	Alta	Alta	Alta	Baixa
Universidade Angolana 06	Média	Média	Média	Média
Universidade Angolana 07	Média	Baixa	Média	Média
Universidade Angolana 08	Média	Média	N/A	Média
Universidade Angolana 09	Alta	Alta	N/A	Média
Universidade Angolana 10	Média	Média	N/A	Média

A avaliação dos resultados gerais quantitativos (Tabela 9), permitiu aferir que a ferramenta Skipfish, com mil e dezasseis (1016) falhas identificadas é a ferramenta que mais resultados produziu.

Tabela 9- Resultados quantitativos por web scanners

Scanner	Vulnerabilidades			Total
	Baixa	Média	Alta	
ZAP	71	19	5	95
Skipfish	143	810	63	1016
OpenVAS	9	112	2	123
PT	40	33	3	76

Esta ferramenta (Skipfish) identificou sessenta e três (62) vulnerabilidades são do tipo alto, oitocentos e dez (810) e cento e quarenta e três (143) são do tipo médio e baixo

respetivamente. A OpenVAS foi a ferramenta que ficou em segundo lugar ao apresentar um total de cento e vinte e três falhas de segurança, sendo duas (2) de alta gravidade, cento e doze (112) de média gravidade e nove (9) considerada como sendo falha de baixa gravidade.

A ferramenta ZAP conseguiu identificar cerca de noventa e cinco (95) vulnerabilidades de segurança. Destas cinco (5) são falhas de alta gravidade, dezanove (19) de média gravidade e setenta e uma (71) foram considerados como sendo vulnerabilidades de baixa gravidade.

A quarta ferramenta usada neste trabalho conseguiu identificar cerca de setenta e seis (76) vulnerabilidades de segurança. Destas três (3) são falhas de alta gravidade, trinta e três (33) de média gravidade e quarenta (40) foram considerados como sendo vulnerabilidades de baixa gravidade.

Todas as falhas identificadas no presente estudo fazem parte da lista de riscos mais frequentes em aplicações web publicadas pela OWASP. Aspetos relacionados com o uso de componentes, bibliotecas JavaScript e ou sistemas com vulnerabilidades conhecidas, má configuração dos servidores, exposição de dados sensíveis, pouca ou nenhuma proteção contra *cross-site scripting,* ausência de comunicações seguras (SSL) e a não proteção adequada com injeção (*SQL Injection*) foram alguns dos achados que conduziram a classificação apresentada.

Devido a natureza dos testes realizados (caixa preta) e o facto de ter sido realizado de fora para dentro mediante o uso de web *scanner* nem todas caraterísticas ou tipos de vulnerabilidades foram passíveis de avaliação, elementos como *desserialização*, *logging* e monitorização fazem parte dessa lista.

Os resultados negativos apresentados foram fortemente influenciados pelo facto de que a maior parte dos *websites* das universidades Angolanas não protegem suficientemente os dados que trafegam entre aplicativos clientes e o servidor com total segurança (ausência de certificados de segurança *SSL*), colocando em causa a integridade dos mesmos. Outro aspeto relevante a ter em conta reside no facto dos servidores não estarem devidamente configurados, visto que os mesmos expõem informações sensíveis sobre configurações, parametrizações, sistema operativo e de algumas aplicações instaladas nos servidores.

Durante o processo de avaliação de vulnerabilidades, também foi possível verificar que todos os websites avaliados não possuíam qualquer tipo de configuração de segurança na exposição de cabeçalhos HTTP (*HTTP security headers*), a ausência deste tipo de configuração permite que um invasor possa incorporar o *website* da instituição num *iframe* de uma aplicação de terceiros.

Este tipo de falha permite ao invasor manipular certos os atributos de exibição do *iframe*, para capturar dados dos utilizadores e induzi-los a realizarem ações sem o consentimento dos mesmos, situação comum em ataques do tipo *phishing*. Um outro aspeto negativo encontrado está relacionado com o facto de algumas aplicações não validam suficientemente a entrada dos dados e expondo as aplicações vulnerável a ataques de injeção, sendo o mais comum o ataque por injeção de *SQL*.

4.3 Comparação entre grupos de ferramentas

A avaliação de vulnerabilidades com recurso a um s*canner* proprietário PT foi realizada no período compreendido entre 01/05/2018 à 25/06/2018, e dois (2) anos mais tarde foi realizado o segundo trabalho com recurso exclusivo a ferramentas *open-source.*

Atendendo que todas as ferramentas utilizadas neste trabalho têm como base ou padrão os instrumentos da OWASP e a CWE, e os resultados obtidos foram agrupados dentro das mesmas categorias (OWASP Top 10), foi possível comparar os resultados obtidos com o uso destas ferramentas que possuem diferentes tipos licenciamentos. O facto da ferramenta PT ter sido executada a partir da versão *online* (*cloud)* e as demais ferramentas terem sido feitas localmente, a mesma não influenciou os resultados.

Deste modo, ao compararmos os dois trabalhos e consequentemente o grau de vulnerabilidade obtidos pelos *scanner* PT (proprietário) e ZAP (*open-source*), notamos que quatro (4) universidades (01, 04, 06 e 10) mantêm inalterado o grau de vulnerabilidade de nível médio, a universidade Independente também conserva o grau de vulnerabilidade de alto risco, ao passo que a duas (2) universidades (07 e 08) veem o seu grau de vulnerabilidade a baixar de nível médio para baixo. De igual a universidade (Jean Piaget) diminui o seu grau de risco ao passar do nível alto para médio.

No entanto, duas (2) universidades (05 e 09) veem o seu estado de segurança a deteriorar-se ao longo dos vinte e quatro (24) meses, ao passarem do grau baixo para alto e do grau médio para alto respetivamente. Conforme ilustrado na Figura 29.

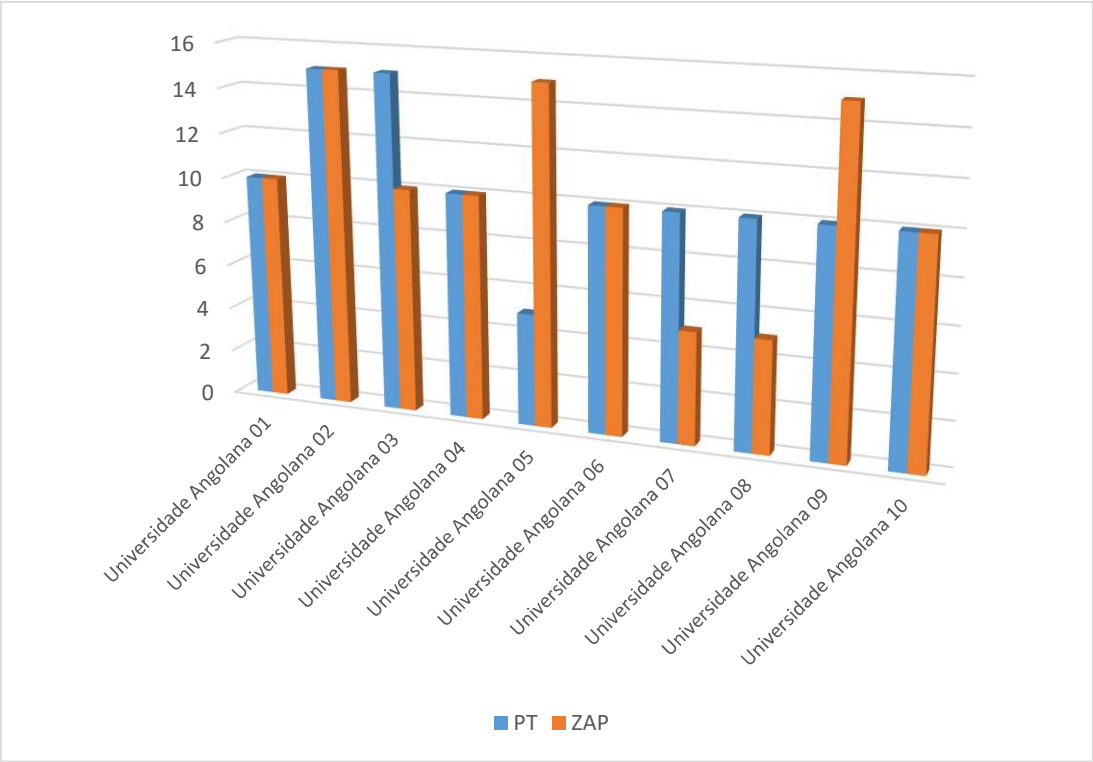

Figura 29- Comparação entre os resultados do scanner PT e ZAP

Relativamente a ferramenta ZAP, quando comparado com a PT, notamos que quatro (4) universidades (01, 06, 08 e 10) mantêm inalterado o grau de vulnerabilidade de nível médio, a universidade 03 também conserva o grau de vulnerabilidade de alto risco, ao passo que a duas (2) universidades (07 e 02) veem o seu grau de vulnerabilidade a baixar de nível médio para baixo e de alto para médio respetivamente.

No entanto, três (3) universidades (04, 05 e 09) veem o seu estado de segurança a deteriorar-se ao longo dos vinte e quatro (24) meses, ao apresentarem o valor mais elevado na escala da avaliação de vulnerabilidades, conforme ilustrado na Figura 30.

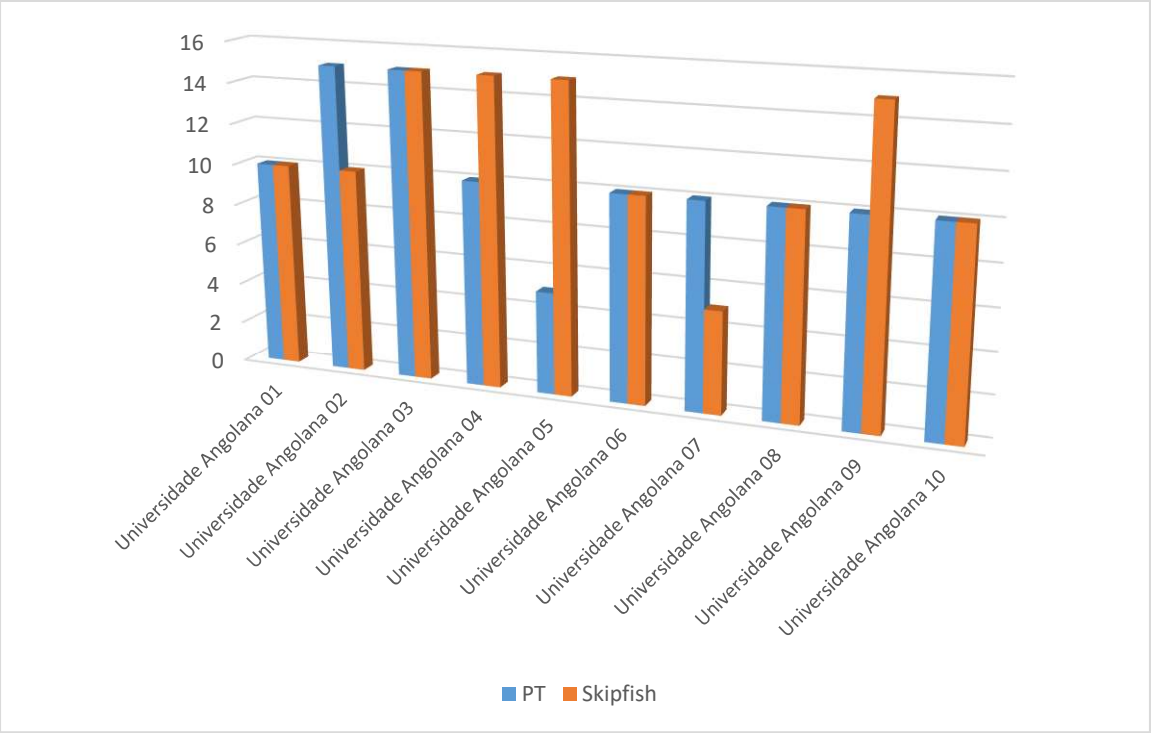

Figura 30 - Comparação entre os resultados do scanner PT e Skipfish

Quando voltamos a comparar a ferramenta proprietária (PT) com uma outra *open-source* (OpenVAS), notamos que a ferramenta OpenVAS não conseguiu transpor as barreiras de segurança das aplicações das universidades 09 e 10.

No entanto, foi possível apurar que três (4) universidades (01, 06 e 10) mantêm inalterado o grau de vulnerabilidade de nível médio, ao passo que a três (3) universidades (02, 03 08) veem o seu grau de vulnerabilidade a baixar, sendo que para as universidades 02 e 08 descem para de nível baixo de segurança e a universidade 03 saí de um nível de vulnerabilidade alta gravidade para médio.

Adicionalmente, podemos verificar que duas (2) universidades (04 e 05) veem o seu estado de segurança a deteriorar-se durante a realização das duas avaliações de vulnerabilidades, ao atingirem o grau mais alto de gravidade em termos de vulnerabilidade, conforme ilustrado na Figura 31.

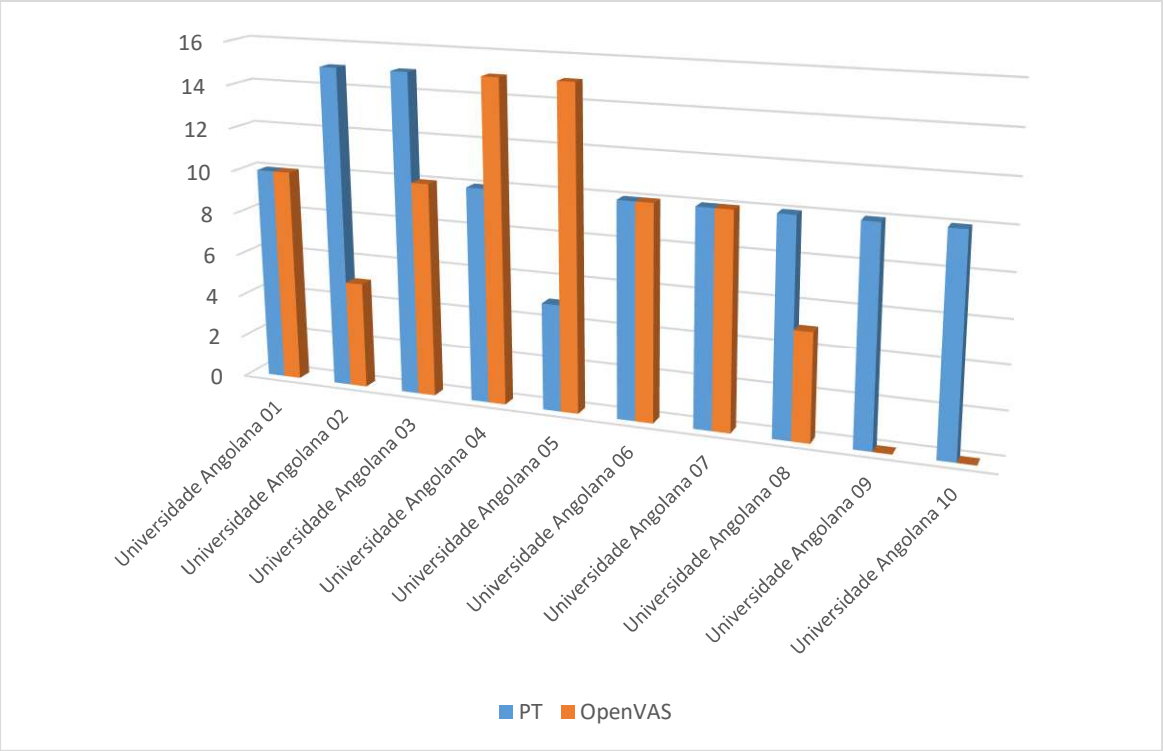

Figura 31 - Comparação entre os resultados do scanner PT e OpenVAS

Ao compararmos todos os resultados produzidos pelos scanners proprietário e *open-souce* (Figura 32) realizados no intervalo de um ano, podemos aferir o seguinte que algumas universidades mantiveram o grau de vulnerabilidade, outras melhoraram ao passo que para um número reduzido de universidades registou-se a degradação do nível de segurança.

A partir da Figura 32, podemos constatar que duas (2) universidades (01e 06) mantêm inalterado o grau de vulnerabilidade de nível médio ao longo do período avaliado, três (3) universidades (04, 09 e a 05) em dez (10) veem o seu nível de vulnerabilidade agravado, pois sobem de vulnerabilidade de segurança de baixa e média gravidade para alto respetivamente. Adicionalmente, podemos igualmente constatar que três (3) universidades (07, 08 e 10) embora apresentando algumas oscilações o seu grau de vulnerabilidade baixa ou mantém-se consoante o tipo de *scanners* utilizado.

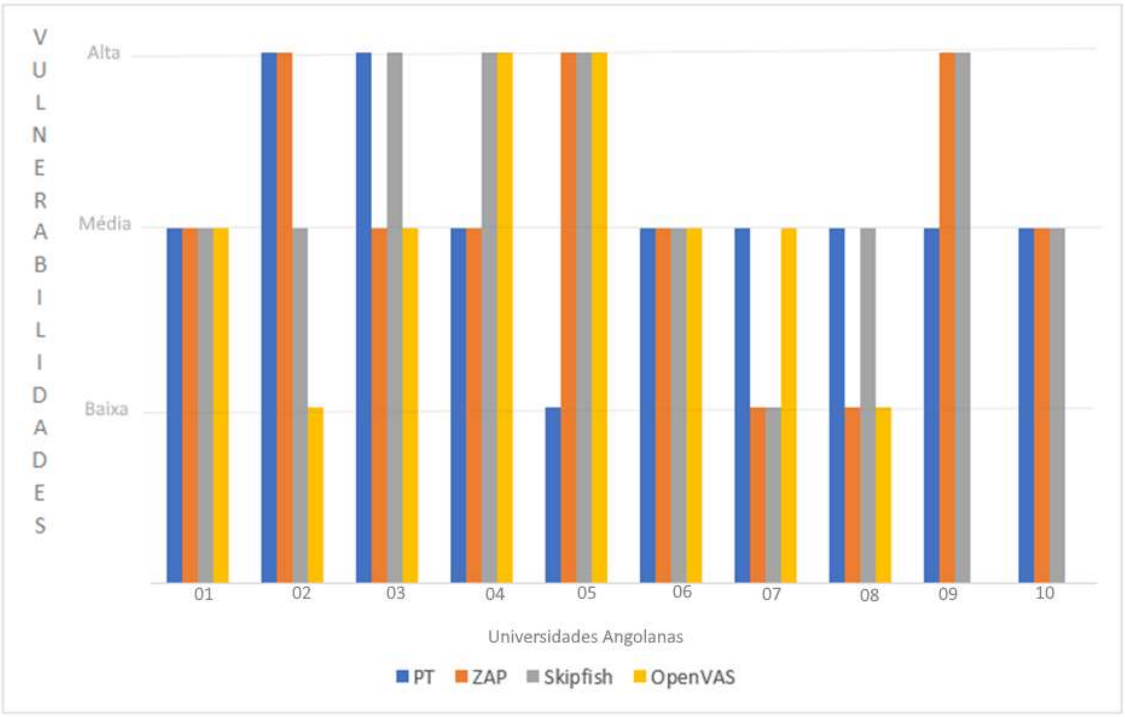

Figura 32 - Distribuição por scanners dos resultados qualitativos no OWASP Top 10

Estes resultados qualitativos dos trabalhos realizados permitem concluir que as universidades 01 e 06 são as que apresentam resultados mais estáveis, ou seja, conseguem manter o resultado (grau de vulnerabilidade média gravidade) ao longo de vinte e quatro (24) meses, não obstante o facto de terem sido usados quatro ferramentas de avaliação de vulnerabilidades.

Adicionalmente, é importante descrever que apesar dos resultados deste estudo terem sido apresentados graficamente com base no instrumento desenvolvido pela a OWASP (Top 10), as categorias apresentadas tiveram em conta a relação existente entre a OWASP Top 10 (2017) e CWE Top 25 conforme apresentado na Tabela 10.

Tabela 10 - Mapeamento entre CWE e OWASP Top 10 2017

A1 - Injection - *(1027)*
Improper Neutralization of Special Elements used in a Command ('Command Injection') - (77)
Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') - (78)
Improper Neutralization of Argument Delimiters in a Command ('Argument Injection') - (88)
Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') - (89)
Improper Neutralization of Special Elements used in an LDAP Query ('LDAP Injection') - (90)
XML Injection (aka Blind XPath Injection) - (91)
SQL Injection: Hibernate - (564)

Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') - (917)
Improper Neutralization of Special Elements in Data Query Logic - (943)
A2 - Broken Authentication - *(1028)*
Improper Authentication - (287)
Unprotected Storage of Credentials - (256)
Use of Single-factor Authentication - (308)
Session Fixation - (384)
Insufficiently Protected Credentials - (522)
Unprotected Transport of Credentials - (523)
Insufficient Session Expiration - (613)
Unverified Password Change - (620)
Weak Password Recovery Mechanism for Forgotten Password - (640)
A3 - Sensitive Data Exposure - *(1029)*
Storage of File With Sensitive Data Under FTP Root - (220)
Improper Certificate Validation - (295)
Missing Encryption of Sensitive Data - (311)
Cleartext Storage of Sensitive Information - (312)
Cleartext Transmission of Sensitive Information - (319)
Key Management Errors - (320)
Missing Cryptographic Step - (325)
Inadequate Encryption Strength - (326)
Use of a Broken or Risky Cryptographic Algorithm - (327)
Reversible One-Way Hash - (328)
Exposure of Private Personal Information to an Unauthorized Actor - (359)
A4 - XML External Entities (XXE) - *(1030)*
Improper Restriction of XML External Entity Reference - (611)
Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion') - (776)
A5 - Broken Access Control - *(1031)*
Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') - (22)
Improper Access Control - (284)
Improper Authorization - (285)
Direct Request ('Forced Browsing') - (425)
Authorization Bypass Through User-Controlled Key - (639)
A6 - Security Misconfiguration - *(1032)*
Configuration - (16)
Generation of Error Message Containing Sensitive Information - (209)
Exposure of Information Through Directory Listing - (548)
A7 - Cross-Site Scripting (XSS) - *(1033)*
Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') - (79)
A8 - Insecure Deserialization - *(1034)*
Deserialization of Untrusted Data - (502)
OWASP Top Ten 2017 Category A9 - Using Components with Known Vulnerabilities - (1035)

A10 - Insufficient Logging & Monitoring - *(1036)*
Omission of Security-relevant Information - (223)
Insufficient Logging - (778)

Fonte: (MITRE, 2018).

A Tabela 10 além de apresentar a relação entre a OWASP e a CWE, descreve categorias, subcategorias, classes, subclasses e enumera detalhadamente as falhas associadas a cada uma delas bem como o grau de perigosidade das mesmas. Por este facto foi possível enquadrar e agrupar as vulnerabilidades encontradas neste trabalho dentro da classificação OWASP Top 10 sem nunca ter de referenciar diretamente a CWE.

4.4 Avaliação técnica dos resultados

O processo de avaliação das vulnerabilidades com recurso a diferentes *scanners* permitiu a produção automática de quarenta (40) relatórios de auditoria, sendo que para cada aplicação o web, foram produzidos quatro relatórios, a razão de um (1) relatório por web *scanner*. As secções seguintes descrevem a forma como os resultados foram produzidos.

4.4.1 ZAP

Esta ferramenta gráfica, permite realizar o *scanning* de uma determinada aplicação e enumerar automaticamente as vulnerabilidades e agrupa-os tendo em conta o tipo de vulnerabilidade encontrada, e utiliza diferentes tipos de cores para identificar automaticamente os diferentes níveis de vulnerabilidades encontrados. Ou seja, a cor vermelha está associado a vulnerabilidades do tipo alto, as vulnerabilidades do tipo médio são identificadas pela cor laranja, a cor amarela serve para identificar as vulnerabilidades de baixa gravidade ao passo que os alertas sobre informações que não se enquadram em nenhuma das categorias anteriores estão identificadas pela cor azul. Conforme ilustrado na *Figura* 33.

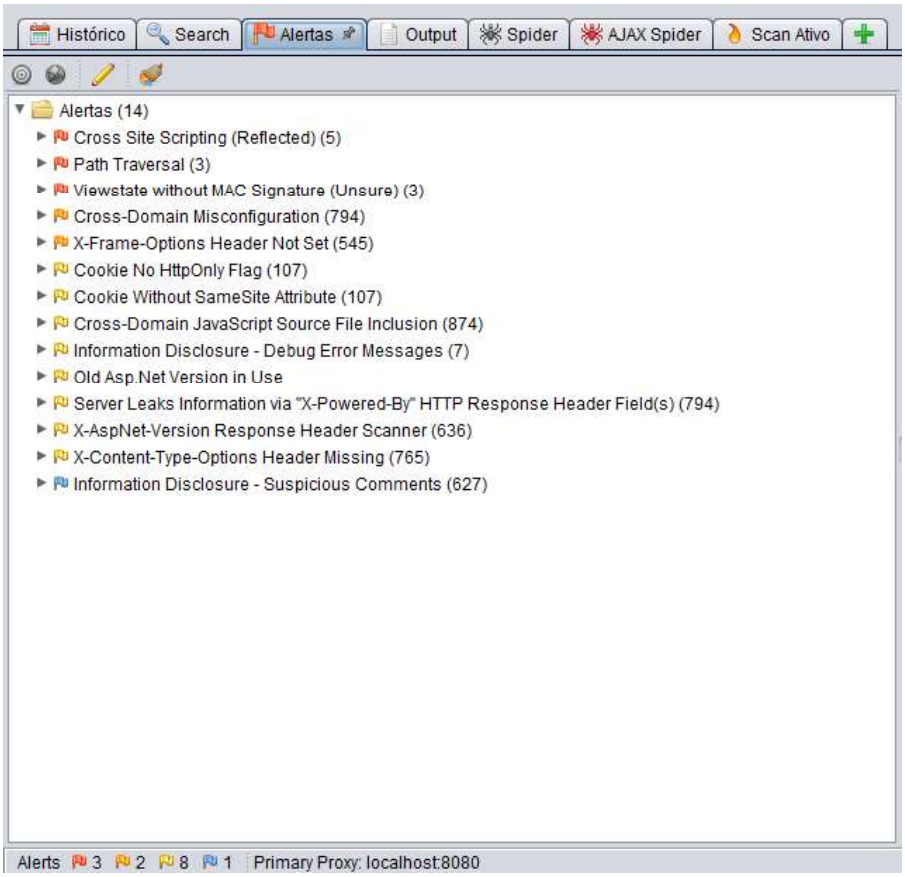

Figura 33-Resultados obtidos pela ferramenta ZAP

A referida ferramenta além de permitir pausar e recomeçar os testes numa outra altura, permite gerar relatório em diferentes formatos, sendo que para cada vulnerabilidade encontrada a ferramenta permite listar o detalhe da mesma.

No detalhe sobre uma determinada vulnerabilidade a ferramenta, enumera o risco, o grau de confiança sobre o risco, identifica os parâmetros utilizados para a cálculo do risco, o tipo de ataque e a evidência (comando SQL ou o *script)* utilizada para identificar a falha de segurança. Outro elemento relevante incluído no detalhe sobre a falha de segurança é a identificação unívoca dos códigos da CWE e ao projeto Web Application Security Consortium (WASC).

Adicionalmente, esta ferramenta faz uma a identificação unívoca das falhas em relação CWE e WASC ao identificá-las pelos seus códigos ou ID e descreve detalhadamente o problema encontrado, apresenta possíveis soluções para resolver ou mitigar a falha de segurança, e ao mesmo tempo inclui hiperligações aos *websites* da CWE e do projeto WASC. Conforme ilustrado na Figura 34.

Figura 34 - Detalhe das vulnerabilidades obtidas pela ferramenta ZAP

4.4.2 Skipfish

Esta ferramenta a semelhança do ZAP, também faz *scanning* automático de uma determinada aplicação e enumerar as vulnerabilidades de segurança. É uma aplicação baseada em linha de comando, mas com capacidade para gerar relatórios em HTML.

Ela também usa um sistema de cores para identificar o grau de vulnerabilidades das falhas encontradas. Ou seja, a cor vermelha serve para identificar as vulnerabilidades do tipo alto, as vulnerabilidades do tipo médio são identificadas pela cor laranja, a cor azul serve para identificar as vulnerabilidades de baixa gravidade ao passo que os alertas sobre informações que não se enquadram em nenhuma das categorias anteriores estão identificadas pela cor cinza. Existe uma categoria identificada pela cor verde e está associada a pequenas falhas de segurança dignas de verificação por parte do auditor.

O relatório em HTML gerado por esta ferramenta encontra-se dividido em três secções, sendo que na primeira mostra o número de vulnerabilidades encontradas separadas pelo nível de gravidade, a segunda faz uma listagem de todos os tipos de documentos encontrados (binários, *JavaScript*, GIF, PNG, JPG, XML, HTML entre outros) e agrupa-

os de acordo com o seu tipo. Na última secção estão identificadas as falhas de segurança enumeradas na primeira secção.

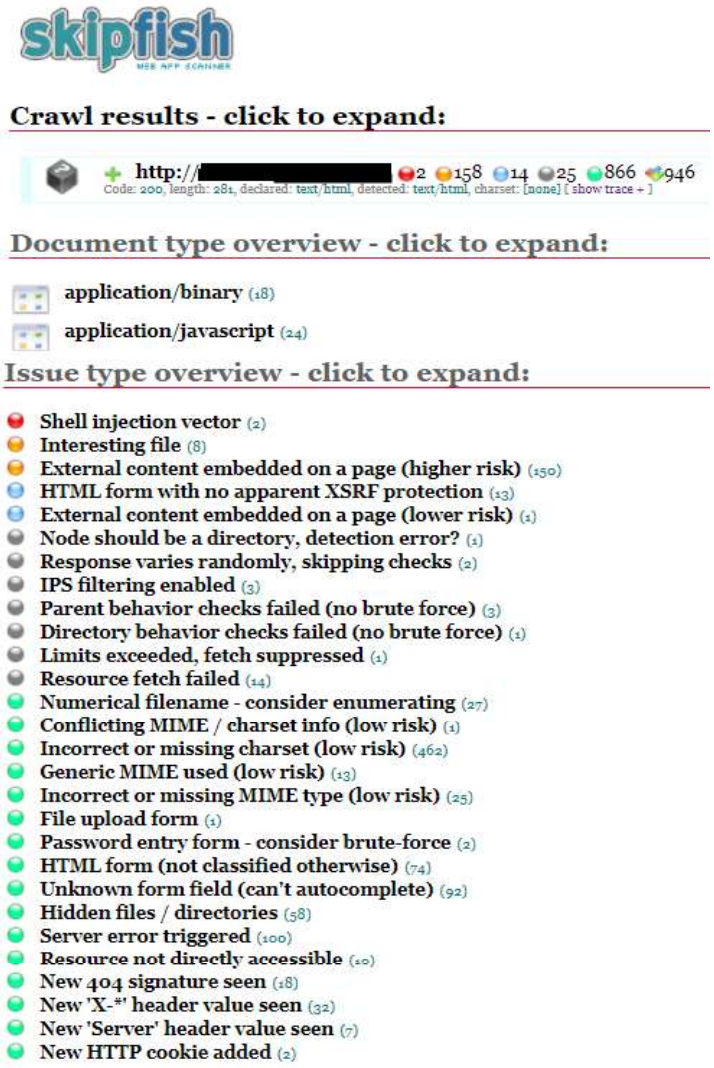

Figura 35- Resultados obtidos pela ferramenta Skipfish

O relatório produzido por esta ferramenta não faz uma associação direta entre a falha de segurança encontradas e o padrão das falhas descritas pela CWE, no entanto a forma como elas estão agrupadas e identificadas permite que o auditor possa fazê-lo com pouca margem de erro.

4.4.3 OpenVAS

Esta ferramenta possui uma interface web e permite a gestão de *scanning* de vários períodos. Esta funcionalidade permite acompanhar o estado ou a evolução das vulnerabilidades ao longo do tempo. Ela também usa o sistema de cores para identificar

automaticamente os diferentes níveis de vulnerabilidades encontrados. Conforme ilustrados na Figura 36.

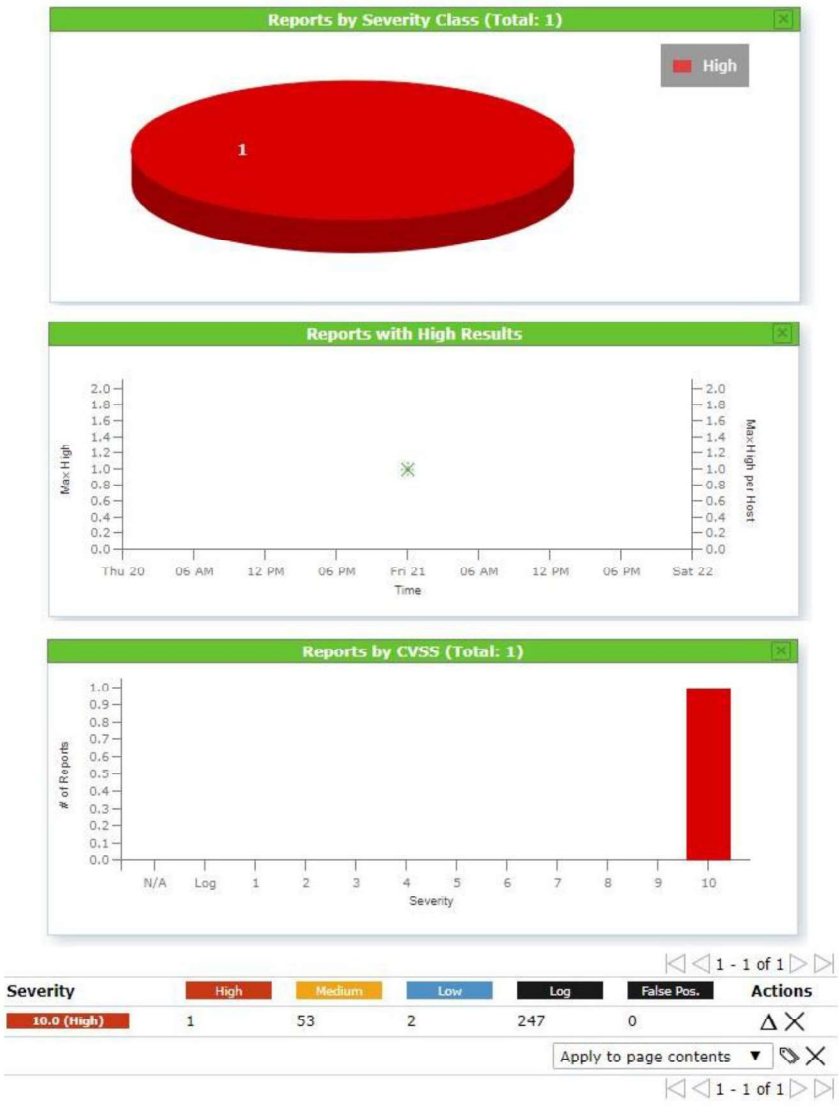

Figura 36- Resultados obtidos pela ferramenta OpenVAS

No sistema de cores acima referenciado, a cor vermelha está associado a vulnerabilidades do tipo alto, as vulnerabilidades do tipo médio são identificadas pela cor laranja, a cor azul serve para identificar as vulnerabilidades de baixa gravidade ao passo que os alertas sobre informações que não se enquadram em nenhuma das categorias anteriores estão identificadas pela cor preta.

A OpenVAS tem a capacidade de produzir resultados com elevado grau de detalhe e com uma forte componente gráfica com elevado grau de interatividade. Os dados obtidos pela ferramenta e disponíveis no relatório contêm informações que vão desde a enumeração das portas do alvo, passando pela identificação do sistema operativo, seguindo-se a lista

de vulnerabilidades encontradas que permitem ampliar e verificar o detalhe da respetiva falha de segurança.

Para cada vulnerabilidade encontrada aplicação lista o grau de severidade, a percentagem de confirmação da falha identificada, uma descrição detalhada da falha e indica possíveis soluções para eliminar e mitigar o problema. Conforme ilustrado na Figura 37.

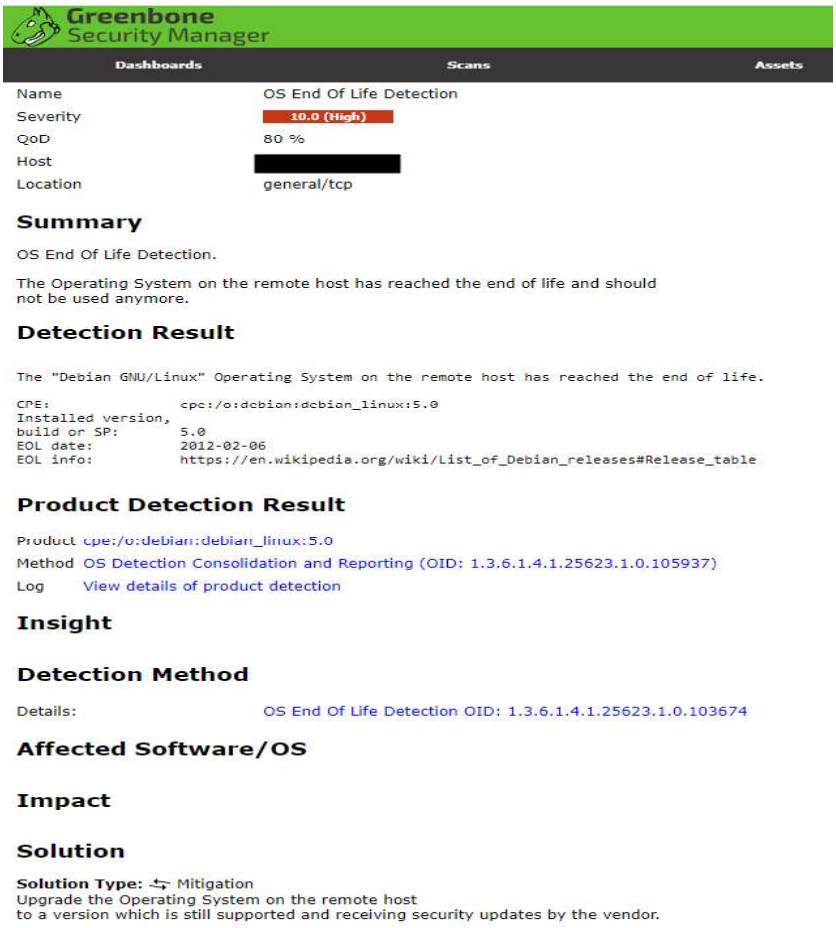

Figura 37- Vulnerabilidade identificada pela ferramenta OpenVAS

4.4.4 PT

Esta ferramenta permite efetuar testes de intrusão de forma integrada. Possui uma interface web, e a semelhança do OpenVAS também permite a gestão e o acompanhamento da evolução das vulnerabilidades encontradas. A Figura 38, ilustra a estrutura de um relatório produzido por este *scanner online*.

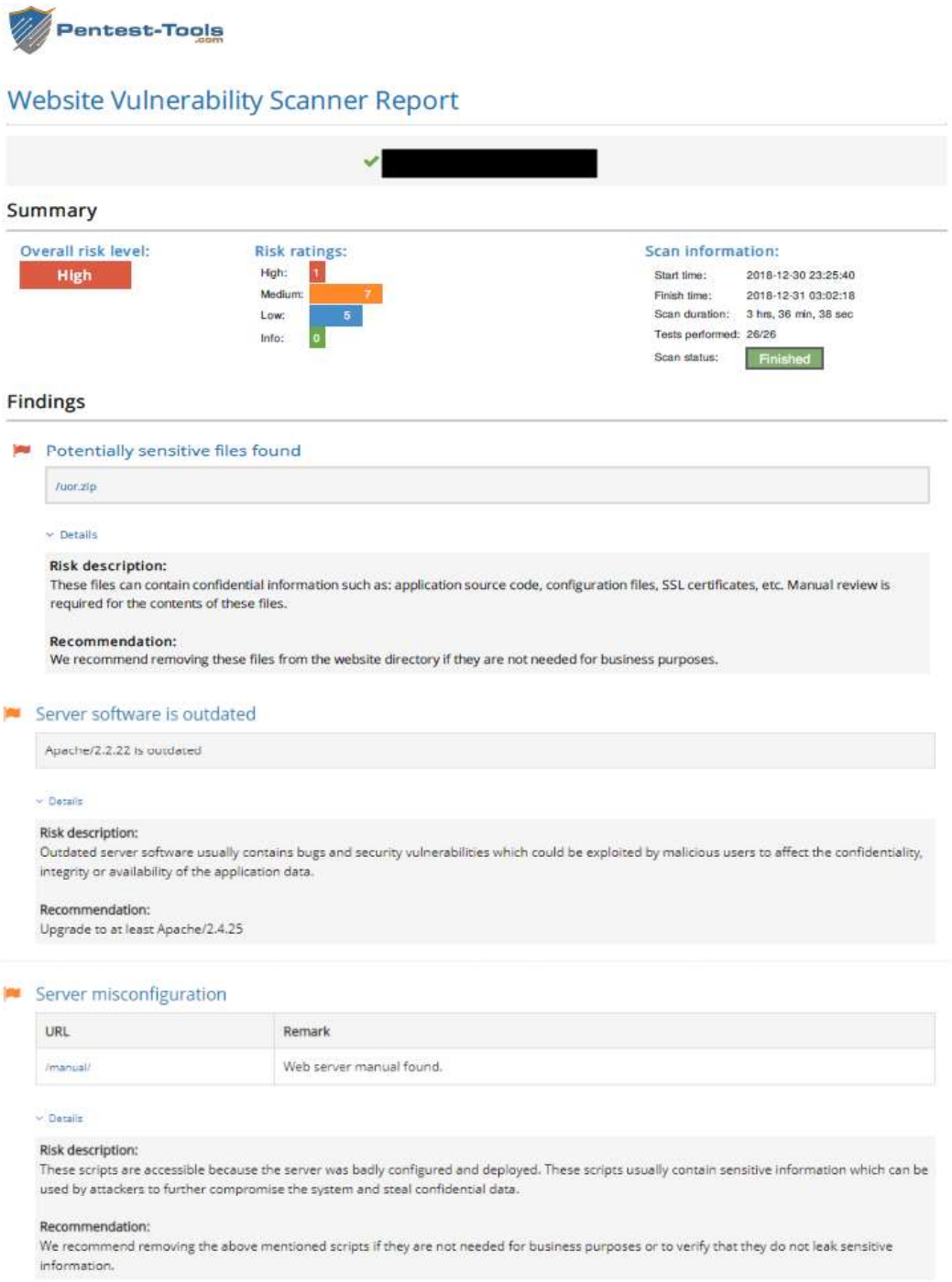

Figura 38- Vulnerabilidade identificada pela ferramenta PT

A partir da Figura 38, podemos verificar que o relatório com os achados descobertos pela ferramenta PT se encontra dividido em duas secções, sendo que a primeira destaca o resultado geral da avaliação da avaliação da aplicação *Web* e inclui a identificação sobre a quantidade dos diferentes tipos de vulnerabilidades encontradas.

A segunda secção do relatório está relacionada com as vulnerabilidades encontradas. Para cada vulnerabilidade encontrada a ferramenta cria uma descrição da falha e de pelo menos

uma recomendação sobre como resolver ou mitigar o problema. Na maior parte das vulnerabilidades as recomendações são acompanhadas de uma hiperligação que permite obter mais informações sobre a respetiva falha.

A ferramenta PT também segue o modelo de cores para automaticamente identificar os diferentes níveis de vulnerabilidades encontrados. Ou seja, a cor vermelha está associado a vulnerabilidades do tipo alto, as vulnerabilidades do tipo médio são identificadas pela cor laranja, a cor azul serve para identificar as vulnerabilidades de baixa gravidade ao passo que os alertas sobre informações que não se enquadram em nenhuma das categorias anteriores estão identificadas pela cor verde.

Independentemente, de cada web *scanner* possuir funcionalidades específicas, elas têm em comum algumas características que quando utilizado podem produzir resultados mais ou menos expressivos num determinado *scanning.* Estas características estão relacionadas com configurações capazes de tornar os testes de vulnerabilidade mais ou menos intrusivos, com maior ou menor grau de profundidade e ativar ou não a opção para excluir automaticamente dos relatórios produzidos os resultados falsos positivos.

No presente trabalho, apenas a opção para eliminar os resultados falsos positivos foi ativada para os webs *scanners* enquanto que as demais parametrizações foram usadas no modo padrão de cada ferramenta.

4.5 Validação técnica das vulnerabilidades

Os resultados obtidos pelo uso das quatro (4) ferramentas utilizadas no presente trabalho permitiu descobrir mil e duzentas e trinta e quatro (1234) vulnerabilidades, sendo que duzentas e vinte e três (223) vulnerabilidades foram consideradas com sendo de baixa gravidade, novecentos e quarenta e um (941) de média gravidade e as restantes setenta (70) vulnerabilidades foram consideradas como falhas de segurança de alta gravidade.

O processo de validação das falhas de segurança encontradas, foram efetuadas usando a confirmação automática e manual. A validação automática foi efetuada com recurso aos outros três (3) dos quatros (4) *scanners* utilizados no presente estudo. Ao passo que a validação manual por ser lenta, não padronizada, complicado e muitas vezes imprecisa foram apenas realizadas nos casos relacionadas com formulários e *frameworks* JavaScript e outros que não envolvessem criar ou adaptar *scripts* existentes para o efeito.

As tabelas seguintes (10,11 e 12) descrevem a matriz e o modelo utilizado durante a validação manual e automática. Por questões de confidencialidade foi removido e ou omitido nas tabelas apresentadas a localização real de certos ficheiros bem como a hiperligação real das instituições.

Na prática, a Tabela 11, apresenta a validação de uma vulnerabilidade de alta gravidade cujo a confirmação fora realizada manualmente. O resultado foi a descoberta de um ficheiro zipado com *software* que deveria ser removido do servidor imediatamente após a instalação da versão final do atual sistema de gestão de conteúdo (CMS), coisa que, no entanto, não aconteceu.

Tabela 11- Validação de vulnerabilidades com recurso ao scanner PT

Web *Scanner* PT	
Vulnerabilidade	Arquivos potencialmente sensíveis encontrados
Localização	*/uor.zip*
Detalhe	Esses arquivos podem conter informações confidenciais, como: código-fonte do aplicativo, arquivos de configuração, certificados SSL, etc. A revisão manual é necessária para o conteúdo desses arquivos.
Risco	ALTO
Recomendações	Recomendamos remover esses arquivos do diretório do site se eles não forem necessários para fins comerciais.
Processo de confirmação	Confirmação manual realizada pelo auditor.
Veredito	CONFIRMADO
Observação	O ficheiro em causa possui a versão do software gestão de conteúdos WordPress as respetivas credenciais de configuração.

A Tabela 12, apresenta a validação de uma vulnerabilidade de alta gravidade cujo a confirmação foi realizada automaticamente e validada positivamente por um dos outros *scanners* usados no presente estudo. O resultado foi a descoberta de *software* de extrema importância (Sistema operativo) em final de vida, confirmando deste modo o resultado da existência de vulnerabilidade de alto risco.

Tabela 12- Validação de vulnerabilidades com recurso ao scanner OpenVAS

Web *Scanner* OpenVAS	
Vulnerabilidade	Deteção de fim de vida útil do sistema operativo (*OS End Of Life*)
Localização	*general/tcp (Debian Linux 5.0)*
Detalhe	O sistema operativo no servidor remoto atingiu o fim de sua vida útil e não deve ser mais usado.
Risco	ALTO
Recomendações	Atualize o sistema operacional no servidor remoto para a versão que ainda é suportada e recebe atualizações de segurança do fornecedor.
Processo de confirmação	Vulnerabilidade confirmada pela ferramenta Skipfish.
Veredito	CONFIRMADO
Observação	Foi confirmado a utilização versão *Debian Linux 5.0*

A Tabela 13, apresenta a validação de um grupo de vulnerabilidade de média gravidade cujo a confirmação foi realizada automaticamente e validada positivamente por dois dos outros *scanners* usados no presente trabalho. O facto de as vulnerabilidades identificadas estarem geralmente relacionadas entre elas, foi possível validar a falta de proteção contra ataques do tipo *cross-Site Scripting* (XSS), *clickjacking* e *phishing* na aplicação web de uma das universidades avaliadas.

Tabela 13- Validação de vulnerabilidades com recurso ao scanner ZAP

Web *Scanner* ZAP	
Vulnerabilidade	Ataque do tipo *Cross-Site Scripting* (XSS), *clickjacking* e *phishing*
Localização	*general/tcp (Debian Linux 5.0)*
Detalhe	X-Frame-Options protege contra os ataques de tipo *clickjacking* X-XSS-Protection atenua ataques Cross-Site Scripting (XSS) X-Content-Type-Options previne possíveis ataques de *phishing* ou XSS
Risco	MÉDIO

Recomendações	Atualize o sistema operacional no servidor remoto para a versão que ainda é suportada e recebe atualizações de segurança do fornecedor.
Processo de confirmação	Vulnerabilidade confirmada pelas ferramentas PT e Skipfish
Veredito	CONFIRMADO
Observação	Foi confirmado a utilização versão *Debian Linux 5.0*

As matrizes apresentadas serviram para validar as vulnerabilidades encontradas pelos *scanners* utilizados neste trabalho. Deste modo os dados obtidos, ou seja, as vulnerabilidades validadas estão aptas para serem usadas na fase seguinte de um teste de intrusão. Neste tipo de teste o auditor usas o resultado da avaliação de vulnerabilidades para realizar um ataque simulado com o objetivo de explorar as vulnerabilidades previamente encontradas.

Capítulo 5 Conclusões

5.1 Principais conclusões

Esta dissertação de mestrado teve como objetivo avaliar a segurança dos websites das instituições de ensino superior angolanas para perceber o grau de vulnerabilidade que os mesmos apresentam.

Para tal, este estudo apoiou-se fortemente nos instrumentos e ferramentas *open-source* que identificam as falhas de segurança mais frequentes nas aplicações web para realizar a avaliação de vulnerabilidades de segurança e do ecossistema que sustentam os websites das entidades em estudo.

Desta forma, realizou-se em primeiro lugar uma revisão de literatura sobre o estado da arte da segurança em aplicações web, e dos diferentes instrumentos e ferramentas que permitem identificar, avaliar e classificar vulnerabilidades de segurança. No estudo da arte também foram apresentadas diferentes metodologias, procedimentos técnicos e administrativos sobre as etapas e o modo como se deve efetuar a avaliação das vulnerabilidades de segurança ou testes de intrusão.

Durante o estudo foi possível aferir a existência vários modelos e *frameworks* para avaliar vulnerabilidades de segurança ou para efetuar testes de intrusão. Na ausência de um *standard* alguns instrumentos como o da OWASP se destacam pelo seu nível de aceitação por parte da comunidade e pela sua abrangência, visto que a mesma tem sido usado como padrão em vários *web scanners* e incorporado em outros instrumentos de avaliação de segurança que quando usado permitem garantir a segurança das aplicações.

Do estudo que foi realizado retiraram-se algumas conclusões. Em primeiro lugar verificou-se que no período em que se realizaram os estudos (2018 e 2020) os websites das universidades angolanas não eram seguros, visto que os resultados da avaliação de segurança com recurso do web *scanner* OWASP ZAP que foi usados como a ferramenta principal deste estudo demonstrou que 30% delas apresentam um grau de vulnerabilidade de segurança do maior gravidade (alta) e as demais, ou seja, 70% apresentaram um grau de vulnerabilidade de média gravidade.

Foi possível também concluir que 70% dos websites comunicam mediante canais inseguros devido a ausência certificados de segurança (*SSL*) e fazem uso de bibliotecas JavaScript com vulnerabilidades conhecidas, ficando assim expostas a ataques do tipo *cross-site scripting*. O presente estudo constatou ainda que todos os websites fornecem demasiadas informações sobre os seus ecossistemas (Sistema operativos, versões do servidor e aplicações), situação que podem potenciar o aumento do grau de vulnerabilidades das mesmas se os sistemas não estiverem atualizados e livres de *bugs*.

Conclui-se, ainda, que as falhas de seguranças descobertas neste estudo poderiam ser evitadas se as instituições de ensino visadas adotassem a prática do *pentesting* no processo de instalação, configuração e manutenção dos ativos de redes e ou aplicações.

Este estudo enumerou a existência de instrumentos (manuais e *framework*) *open-source* que permitem a avaliação de vulnerabilidades de segurança e de testes de intrusão que quando utilizado ou se fossem utilizado por estas instituições de ensino permitiriam as mesmas conhecer o estado de segurança das suas aplicações e ou ecossistemas e consequente tomarem medidas preventivas que iriam ajudar reduzir o grau de criticidade das vulnerabilidades que as mesmas apresentam.

5.2 Trabalho futuro

A utilização de três *scanners open-source* de avaliação de vulnerabilidades para enumerar, classificar e identificar as falhas nas aplicações *web* das universidades Angolanas foram realizadas com sucesso, pelo que os objetivos do trabalho foram alcançados com sucesso.

No entanto, esta temática não se esgota por aqui, várias melhorias ainda podem ser realizadas para garantir a segurança das mesmas ao longo do tempo. Para futuros trabalhos apresentam-se as seguintes propostas:

Realização de um trabalho que explorasse a segunda parte dos testes de avaliação de segurança, visto que este trabalho apenas avaliou a primeira de duas etapas (avaliação de vulnerabilidades e teste de intrusão), ficando por fazer o ataque simulado que permitisse explorar as falhas descobertas pelo presente estudo para reclassificar as falhas, comprovar a eficiência e o grau de precisão dos *scanners* utilizados em relação às falhas encontradas.

De igual modo, a realização de um novo estudo com os objetivos similares aos atuais para perceber se foram realizadas algumas melhorias ou mitigados as vulnerabilidades que este trabalho encontrou e acima de tudo para comparar a evolução das falhas de segurança encontradas ao longo de um certo período poderia contribuir para a consciencialização da cultura de prevenção e segurança informática.

Referências bibliográficas

Abbas Saeed, F., & Abed Elgabar, E. E. (2014). Assessment of open source web application security scanners. *Journal of Theoretical and Applied Information Technology*, *61*(2), 281–287. Retrieved from www.jatit.org

Ahn, L. von, Maurer, B., McMillen, C., Abraham, D., & Blum, M. (2008). reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Sciencemag*, *321*. Retrieved from www.sciencemag.org/cgi/content/full/1160379/DC1

Al Shebli, H. M. Z., & Beheshti, B. D. (2018). A study on penetration testing process and tools. *2018 IEEE Long Island Systems, Applications and Technology Conference, LISAT 2018*, 1–7. https://doi.org/10.1109/LISAT.2018.8378035

Amankwah, R., Chen, J., Kudjo, P. K., & Towey, D. (2020). An empirical comparison of commercial and open-source web vulnerability scanners. *Software - Practice and Experience*, *50*(9), 1842–1857. https://doi.org/10.1002/spe.2870

Bairwa, S., Mewara, B., & Gajrani, J. (2014). Vulnerability scanners: a proactive approach to assess web application security. *International Journal on Computational Sciences & Applications (IJCSA)*, *4*(1). https://doi.org/10.5121/ijcsa.2014.4111

CAPTCHA. (2019). CAPTCHA: Telling Humans and Computers Apart. Retrieved from http://www.captcha.net/

Cloudflare. (2020). Comprehensive DDoS Protection. Retrieved September 1, 2020, from https://www.cloudflare.com/ddos/

CWE. (2019). *Overview - What Is CWE?* Retrieved from https://cwe.mitre.org/about/index.html

Engebretson, P. (2011). *The Basics of hacking and penetration Testing* (Elsevier, Ed.). Retrieved from www.wowebook.com

Goel, J. N., & Mehtre, B. M. (2015). Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology. *Procedia Computer Science*, *57*, 710–715. https://doi.org/10.1016/j.procs.2015.07.458

Google. (2012). skipfish - web application security scanner. Retrieved September 1,

2020, from https://code.google.com/archive/p/skipfish/wikis/SkipfishDoc.wiki

Hafele, D. (2004). *Information Security Reading Room Three Different Shades of Ethical Hacking : Black , White and Gray*. Retrieved from https://www.sans.org/reading-room/whitepapers/hackers/shades-ethical-hacking-black-white-gray-1390

Haubris, K. P., & Pauli, J. J. (2013). Improving the efficiency and effectiveness of penetration test automation. *Proceedings of the 2013 10th International Conference on Information Technology: New Generations, ITNG 2013*, 387–391. https://doi.org/10.1109/ITNG.2013.135

Hevner, A. R. (2007). A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, *19*(2), 87–92. https://doi.org/http://aisel.aisnet.org/sjis/vol19/iss2/4

Hevner, A. R., March, T. S., Jinsoo, P., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *1*(2), 75–105. https://doi.org/10.1007/BF01205282

HKSAR. (2008). An Overview Of Vulnerability Scanners. *The Government of the Hong Kong Special Administrative Region*, (February), 16. Retrieved from http://www.infosec.gov.hk/english/technical/files/vulnerability.pdf

Hostway. (2017). *The 6 Stages of a Malicious Cyber Attack Is Your Business Secure?* Retrieved from https://hostway.com/wp-content/uploads/2017/08/6-stages-of-a-malicious-cyber-attack-whitepaper.pdf

Intruder. (2015). *The Ultimate Guide to Vulnerability Scanning Vulnerability Scanning*.

Kali. (2020). WhatWeb. Retrieved from https://tools.kali.org/web-applications/whatweb

KPMG. (2016). Security Testing. In *Cyber Defense Services*. https://doi.org/10.1081/E-ESE-120044192

Leedy, P., & Ormrod, J. (2015). *Practical research: planning and design* (11th Ed.; Pearson, Ed.). Boston, MA: Pearson.

Liu, C., Tan, C.-K., Fang, Y.-S., & Lok, T.-S. (2012). The Security Risk Assessment Methodology. *Procedia Engineering*, *43*, 600–609. https://doi.org/10.1016/j.proeng.2012.08.106

Lubis, A., & Tarigan, A. (2017). Security Assessment of Web ApplicationThrough Penetration System Techniques. In *Jend. Gatot Subroto Km* (No. ISSN: 2455-1457]; Vol. 4). Retrieved from www.pancabudi.ac.id

Martin, B., Paller, A., Kirby, D., & Christey, S. (2011). *CWE - 2011 CWE/SANS Top 25 Most Dangerous Software Errors*. Retrieved from http://cwe.mitre.org/top25/

MESCTI. (2020). *Quadro legal das instituições de ensino superior privadas e respectivos cursos de graduação*. Retrieved from https://www.mescti.gov.ao/verlegislacao.aspx?id=2408

MITRE. (2019). Corporate overview. Retrieved from http://www.mitre.org/about/corporate-overview

Mohd. Ehmer, K., & Farmeena, K. (2012). A Comparative Study of White Box , Black Box and Grey Box Testing Techniques. *International Journal of Advanced Computer Science and Applications*, *3*(6), 12–15. https://doi.org/10.1017/CBO9781107415324.004

Najork, C. (2010). *Web Crawling* (Stanford university). https://doi.org/10.1561/1500000017

NCSC. (2020). *Security testing White Paper*. Retrieved from https://www.necam.com/docs/?id=2709888a-ecfd-4157-8849-1d18144a6dda

NIST. (2008). Technical Guide to Information Security Testing and Assessment Special Publication 800-115. *Nist Special Publication*, *800*, 1–80. https://doi.org/10.6028/NIST.SP.800-115

NVD. (2020). Vulnerability Metrics. Retrieved August 26, 2020, from https://nvd.nist.gov/vuln-metrics/cvss

Ottosson, H. (2018). *Penetration testing for the in- experienced ethical hacker*.

OWASP. (2014). *Testing Guide 4.0*. Retrieved from http://www.owasp.org

OWASP. (2017a). OWASP Top 10 - 2017 The Ten Most Critical Web Application Security Risks. *Owasp*, 24. Retrieved from https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf%0Ahttps://www.owasp.org/images/7/72/OWASP_Top_10-

2017_%28en%29.pdf.pdf%0Ahttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:OWASP+Top+10+-2010#1

OWASP. (2017b). Top 10-2017 Top 10 - OWASP. Retrieved June 16, 2018, from Top 10-2017 website: https://www.owasp.org/index.php/Top_10-2017_Top_10

OWASP. (2019). Risk Rating Methodology. Retrieved from https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

PCI. (2015). Penetration Testing Guidance. *Penetration Testing Guidance*, *1.0*(1.0), 40. Retrieved from https://www.pcisecuritystandards.org/documents/Penetration_Testing_Guidance_March_2015.pdf

Publico. (2019). British Airways enfrenta multa de 204 milhões de euros após ciberataque. Retrieved from https://www.publico.pt/2019/07/08/tecnologia/noticia/british-airways-enfrenta-multa-204-milhoes-euros-apos-ciberataque-1879129

Security, O. (2020). whatweb. Retrieved August 20, 2020, from https://tools.kali.org/web-applications/whatweb

Shah, S., & Mehtre, B. M. (2013). A reliable strategy for proactive self-defence in cyber space using VAPT tools and techniques. *2013 IEEE International Conference on Computational Intelligence and Computing Research, IEEE ICCIC 2013*. https://doi.org/10.1109/ICCIC.2013.6724216

Shah, S., & Mehtre, B. M. (2014). An overview of vulnerability assessment and penetration testing techniques. *Journal of Computer Virology and Hacking Techniques*, *11*(1), 27–49. https://doi.org/10.1007/s11416-014-0231-x

Shakeel, I. (2010). *The Art Of Network Vulnerability Assesment*. Retrieved from http://www.infosecinstitute.com/courses/ethical_hacking_training.html

Shinde, P. S., & Ardhapurkar, S. B. (2016). Cyber security analysis using vulnerability assessment and penetration testing. *IEEE WCTFTR 2016 - Proceedings of 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare*. https://doi.org/10.1109/STARTUP.2016.7583912

Sparks, S., Embleton, S., Cunningham, R., & Zou, C. (2007). Automated vulnerability analysis: Leveraging control flow for evolutionary input crafting. *Proceedings -*

Annual Computer Security Applications Conference, ACSAC, 477–486. https://doi.org/10.1109/ACSAC.2007.27

Sreenivasa, R. (2012). Web Application Vulnerability Assessment. *THE ISSA JOURNAL: The Global Voice of Information Security*, *2*(1).

Teodoro, N. (2011). *Auditoria de Segurança em Aplicações na World Wide Web Portuguesa* (ISCTE). Retrieved from https://repositorio.iscte-iul.pt/handle/10071/8240

Vibhandik, R., & Bose, A. K. (2015). Vulnerability assessment of web applications-a testing approach. *2015 4th International Conference on E-Technologies and Networks for Development, ICeND 2015*, 16–21. https://doi.org/10.1109/ICeND.2015.7328531

Yüce, E. (2018). *Overview of Penetration Testing Methodologies and Tools*. Retrieved from http://staff.bilkent.edu.tr/emreyuce/wp-content/uploads/sites/341/2019/02/CS479-Overview-of-pentesting-and-tools-v0.8.pdf

Zhu, C. (2017). *Experimental study of vulnerabilities in a web application* (Aalto University). Retrieved from https://pdfs.semanticscholar.org/ee97/f209f16c8911c395538383938d78cdf90242.pdf