Instituto Universitário de Lisboa

Departamento de Ciências e Tecnologias da Informação

Plataforma de registo e gestão de resultados de testes de intrusão e auditorias automáticas

Pedro Jorge Duarte Carrilho Pereira Cananão

Uma dissertação apresentada de acordo com os requerimentos para o grau de

Mestre em Engenharia Informática

Orientador

Prof. Dr. Carlos Serrão, Professor Auxiliar

ISCTE-IUL

Co-Orientador

Eng. Nuno Teodoro, Information Security Director

Seedrs

Setembro 2016

Resumo

Com o crescimento a nível dos sistemas de informação e do número de empresas que cada vez mais utiliza os mesmos, vai existir uma necessidade de se realizar auditorias à segurança dos sistemas.

Normalmente numa auditoria, o auditor que realiza os testes vai usar vários métodos e ferramentas para realizar os testes, e cada uma dessas ferramentas geram um relatório diferente em relação à outra, e no final disto tudo vai ser necessário realizar um relatório para se entregar à organização com os resultados de todos os testes, mas atualmente ainda não existe nenhum *template* para estes relatórios.

Como tal existe a necessidade da criação de uma plataforma para as empresas que realizam as auditorias na qual seja possível agregar os resultados destas auditorias, fazendo com que esta informação seja guardada para mais tarde, se for feita uma nova auditoria ao mesmo sistema, seja possível verificar se as ameaças e as vulnerabilidades foram mitigadas para assim a organização responsável fazer uma melhor gestão da segurança do sistema.

Para tal foi efetuada esta investigação que resultou numa plataforma que tem como propósito tentar ajudar os auditores nesta tarefa, bem como possibilitar aos mesmos que possam utilizar ficheiros que são criados a partir de programas que fazem a pesquisa automaticamente por vulnerabilidades, para registarem as suas vulnerabilidades para além de ser só manualmente. Esta plataforma oferece também a possibilidade de classificar as vulnerabilidades utilizando um sistema de classificação bastante conhecido por parte dos auditores de segurança para assim permitir que as mesmas estejam com o maior detalhe possível.

Palavras-chave: Auditoria, Vulnerabilidade, Segurança, Classificação de Vulnerabilidades.

Abstract

With the growth in regard of information systems and the number of companies that increasingly uses them, there will be a need to perform security audits to the systems.

Normally in an audit, the auditor that performs the tests will use several methods and tools to perform the tests, and each of these tools generate a different report in relation to the other, and at the end of it all the different reports will be necessary to make a final report to deliver to the company with the results of all tests, but currently there isn't still a template for these reports.

As such there is a need to create a platform for companies that are responsible for performing audits, with the objective of aggregating the results of these audits, so that this information is saved for later use, for when it's made a new audit of the same system, it is possible check the threats and vulnerabilities that were mitigated so as the responsible organization can do a better management of their system security.

Thanks to this a need for a platform of this type had to be researched, and this investigation is a result of this. The platform aims to help the people responsible for the audits by providing the means to insert vulnerabilities manually or by using a file from a vulnerability scanner. This platform also offers to it's users a vulnerability scoring system that is very well know by most auditors, and thanks to it, the vulnerabilities that are registered have more details that can help everyone understand them.

Keywords: Audit, Vulnerability, Security, Vulnerability Rating.

Agradecimentos

Começo por agradecer aos meus pais, pois sem o apoio, paciência e dedicação deles nunca teria chegado a este ponto.

À minha irmã por todo o apoio que me deu, bem como tudo o que atura de mim todos os dias.

Ao orientador Carlos Serrão e coorientador Nuno Teodoro, pois sem a ajuda deles, conselhos e guia este trabalho não tinha sido realizado.

A toda a minha família e amigos, um muito obrigado por todo apoio e ajuda que me proporcionaram não só ao longo deste mestrado, mas ao longo de todo meu percurso e vida.

Ao pessoal do ISCTE, Vitor, Nuno, Duarte, André, Pedro, Manuel e muitos outros um grande obrigado pela ajuda nos trabalhos, a estudar, bem como todo o tempo que passamos juntos.

Conteúdo

Lista de Figuras

Abreviaturas

VRSS	**V**ulnerability **R**ating and **S**coring **S**ystem (ver página 8)
CVSS	**C**ommon **V**ulnerability **S**coring **S**ystem (ver página 9)
CVE	**C**ommon **V**ulnerabilities and **E**xposures (ver página 14)
OWL	**O**ntology **W**eb **L**anguage (ver página 8)
ICAT	**I**nternet **C**atalog of **A**ttacks **T**oolkit (ver página 8)
NVD	**N**ational **V**ulnerability of **D**atabase ver página 8)
IBM	**I**nternational **B**usiness of **M**achines (ver página 8)
AV	**A**ttack **V**ector(ver página 11)
AC	**A**ttack **C**omplexity (ver página 11)
Au	**Au**thentication (ver página 11)
POC	**P**roof-**O**f-**C**oncept (ver página 12)
ND	**N**ot **D**efined (ver página 13)
RL	**R**emediation **L**evel (ver página 12)
TF	**T**emporal **F**ix (ver página 12)
OF	**O**fficial **F**ix (ver página 12)
RC	**R**eport **C**onfidence (ver página 12)
CDP	**C**ollateral **D**amage **P**otential (ver página 13)
TD	**T**arget **D**istribution (ver página 13)
CR	**C**onfidentiality **R**equirement (ver página 13)
IR	**I**ntegrity **R**equirement (ver página 13)
AR	**A**vailability **R**equirement (ver página 13)
CVE	**C**ommon **V**ulnerability **E**xposure (ver página 14)
CVE-ID	**C**ommon **V**ulnerability **E**xposure-**ID**entifier (ver página 15)
CWE	**C**ommon **W**eakness **E**numeration (ver página 15)

BID **B**ugtraq **ID**entifier (ver página 16)

CERT **C**omputer **E**mergency **R**esponse **T**eam (ver página 17)

EDB **E**xploit **D**ata **B**ase (ver página 17)

IAVA **I**nformation **A**ssurance **V**ulnerability **A**lerts (ver página 20)

OWASP **O**pen **W**eb **A**pplication **S**ecurity **P**roject (ver página 30)

HTTP **H**yper**T**ext **T**ransfer **P**rotocol (ver página 33)

SOAP **S**imple **O**bject **A**ccess **P**rotocol (ver página 33)

HTML **H**yper**T**ext **M**arkup **L**anguage (ver página 54)

CSS **C**ascading **S**tyles **S**heets (ver página 54)

SQL **S**tructured **Q**uery **L**anguage (ver página 54)

IDE **I**ntegrated **D**evelopment **E**nvironment (ver página 55)

API **A**pplication **I**ntegrated **I**nterface (ver página 55)

SMTP **S**imple **M**ail **T**ransfer **P**rotocol (ver página 56)

SSL **S**ecure **S**ockets **L**ayer (ver página 56)

TLS **T**ransport **L**ayer **S**ecurity (ver página 56)

PDF **P**ortable **D**ocument **F**ormat (ver página 72)

CSRF **C**ross-**S**ite **R**equest **F**orgery (ver página 59)

LDAP **L**ight-**D**irectory **A**ccess **P**rotocol (ver página 39)

Capítulo 1

Introdução

Neste capítulo será feita a introdução à investigação desta tese de mestrado, de modo a que se enquadre o tema que vai ser abordado, bem como as perguntas que vão dar origem a esta investigação e os objetivos que se espera atingir no final da mesma.

1.1 Motivação

A grande maioria das Organizações recorrem a metodologias como testes de intrusão e análise de vulnerabilidades aos seus sistemas de uma forma regular no decorrer das atividades de auditoria de segurança da informação, procurando garantir a segurança continuada dos mesmos.

No entanto, a realização destas atividades nem sempre é gerida da melhor maneira, fazendo com que ocorram alguns problemas:

- A realização de testes de intrusão por várias entidades ou equipas não gera resultados comparáveis para a Organização em relação à classificação das vulnerabilidades;

- Como é complexo mitigar as vulnerabilidades faz com que o tempo que é decorrido entre a auditoria e a mitigação complique o acompanhamento das ações;
- Muitas vezes o acompanhamento entre as vulnerabilidades e a mitigação das mesmas, são feitas usando ferramentas de *ticketing*, sendo que estas não estão preparadas para os resultados específicos gerados pelos testes de intrusão;
- Visto que não existe uma plataforma central que agrega as atividades, isto é, os resultados e a tendência de evolução dos mesmos, faz com que a comunicação à gestão de topo seja muitas vezes complexa e por vezes até mesmo impossível;
- Inexistência de uma plataforma que agregue todos resultados dos diversos testes de intrusão que permita o acesso a partir de perfis tendo em conta as permissões associadas, facilitando assim a separação da informação que vai fornecida às equipas responsáveis por realizar a mitigação.

Assim, a principal motivação para a realização deste trabalho, consiste na necessidade do desenvolvimento de uma plataforma que consiga resolver os problemas identificados de modo a ajudar tanto os auditores como as organizações, ao nível da perceção e da comunicação dos resultados das auditorias, bem como analisar a evolução ao nível da mitigação das vulnerabilidades já descobertas.

1.2 Enquadramento

Nos dias que correm cada vez os sistemas de informação são as ferramentas base para as empresas, fatores de competitividade e, como tal, contêm e processam informação de negócio vital que necessita de ser protegida. As empresas têm de garantir que a segurança dos sistemas está assegurada e para isso são realizadas auditorias para verificar a existência de vulnerabilidades que possam ser exploradas.

O auditor vai realizar vários testes, nos quais vai utilizar vários métodos e ferramentas para realizar os mesmos, e cada uma dessas ferramentas vai gerar um relatório com resultados. No entanto, cada uma das ferramentas, produz relatórios em formatos distintos, sendo necessário integrar todos estes relatórios e informação dispersa num relatório integrado para comunicar à organização os resultados de todos os testes realizados. Atualmente ainda não existe nenhum "relatório-tipo" integrado que permita este tipo de comunicação.

Surge, por isso mesmo, a necessidade de criação de uma plataforma destinada para as empresas que realizam estas auditorias de segurança na qual seja possível agregar e gerir os resultados desses mesmos testes. Esta plataforma deve permitir que a informação dos testes seja preservada, garantindo assim que, quando mais tarde for realizada uma nova auditoria ao mesmo sistema, seja possível verificar se as ameaças e as vulnerabilidades foram mitigadas. Isto permitirá à entidade auditada efetuar uma melhor gestão da segurança do sistema.

1.3 Questões de Investigação

Ao longo do presente trabalho de dissertação ir-se-á procurar responder às seguintes questões de investigação:

- É possível fornecer aos auditores uma solução que lhes permita registar os resultados que vão obtendo dos testes de intrusão que são efetuados ?
- É possível uma plataforma suportar receber resultados de ferramentas de deteção automáticas de vulnerabilidade ?
- É possível fornecer ao auditor responsável um *dashboard*, onde o mesmo pode observar os vários riscos que as vulnerabilidades apresentam à empresa em causa ?

1.4 Objetivos

Os objetivos para esta dissertação são:

- Desenvolvimento de uma plataforma que consiga agregar os resultados de auditorias;
- A partir desses resultados obtidos ser capaz de gerar *dashboards* e gráficos evolutivos que facilitem a perceção tanto dos auditores como os gestores de topo;
- Usando a informação que é submetida para a plataforma, fazer a geração de relatórios que podem ser personalizáveis.

1.5 Metodologia de Investigação

O método de investigação que vai ser usado para o desenvolvimento desta dissertação é o Design Science Research [1]. Este método foi escolhido pois é o que se enquadra melhor para a resolução do problema que é proposto, devido à utilização de artefactos de tecnologias de informação (TI) para a solução dos problemas do mundo de negócios. Para tal vai ser necessário seguir às seguintes questões:

- Questão 1- Qual é o foco da investigação?

A investigação vai focar-se no desenvolvimento de uma plataforma que seja capaz de centralizar os resultados das auditorias de segurança, tornando assim possível a todos os que estão envolvidos no processo de auditoria possam aceder aos resultados que são produzidos das auditorias.

- Questão 2- Qual é o artefacto produzido e como é que o mesmo é representado?

O artefacto que vai ser produzido é uma plataforma onde os auditores poderão colocar os resultados dos testes de vulnerabilidades que efetuaram para uma determinada Organização, de maneira a que todos os utilizadores que tiverem as permissões necessárias possam ver os resultados que são obtidos e a evolução dos mesmos.

- Questão 3- Que processos de desenho vão ser utilizados para a criação do artefacto?

Os processos de desenho que vão ser usados para o desenvolvimento serão técnicas de desenvolvimento seguras que têm como base o OWASP Top 10, bem como um *hardening* às configurações de segurança e infraestrutura que irão servir para suportar o artefacto.

- Questão 4- Como é que os processos de desenho e o artefacto são fundamentados com base no conhecimento?

A base de conhecimento vão ser os testes de vulnerabilidade que os auditores realizaram e o artefacto vai pegar nessa informação e gerar os *dashboards* e gráficos evolutivos correspondentes à informação que é passada, tendo depois como opção a geração de relatórios personalizáveis.

- Questão 5- Que avaliações são efetuadas durante o ciclo interno de desenho? Que melhorias a nível de desenho são identificadas durante cada ciclo?

Para a avaliação do ciclo interno de desenho vão ser usados um conjunto de testes que vão servir para verificar se a informação gerada pelos *dashboards* está correta e se os gráficos evolutivos que também vão ser gerados estão de acordo com a informação fornecida pelos testes.

- Questão 6- Como é que o artefacto produzido é introduzido no ambiente aplicacional? Que métricas são usadas para demonstrar a utilidade e o melhoramento dos artefactos anteriores?

O artefacto vai ser inicialmente testado usando um número pequeno de *inputs* que são passados pelos utilizadores e ficheiros que são gerados por ferramentas de testes automáticas (por exemplo, Nessus), para verificar se os *dashboards* produzidos e gráficos estão de acordo com a informação que é fornecida para a plataforma, bem como os relatórios personalizáveis que vão ser gerados com base na informação que é fornecida pelos *dashboards* e gráficos, se tudo funcionar corretamente podemos aumentar a escala de maneira a suportar várias auditorias. As métricas que vão ser utilizadas para demonstrar a utilidade e o melhoramento do artefacto estão mencionadas na secção dos Objetivos.

- Questão 7- Que conhecimento é adicionado à base de conhecimento e de que forma?

Os gráficos evolutivos tendo em conta a informação que é fornecida pelos auditores, bem como as ferramentas automáticas, avaliando assim as políticas de segurança em causa, fazendo assim a geração de conhecimento.

- Questão 8- A questão que levou à investigação foi satisfatoriamente respondida?

De acordo com o feedback positivo que foi recebido por muitos dos utilizadores, sim pode se responder que foi possível responder satisfatoriamente à questão que levou a esta investigação.

Capítulo 2

Análise do Estado da Arte

Este capítulo vai apresentar os vários tópicos que tiveram de ser investigados com o propósito de se ter uma base de conhecimento e de comparação em relação às plataformas que existem atualmente similares à que está a ser criada, bem como os recursos necessários para a implementação da mesma.

2.1 Sistemas de Classificação de Vulnerabilidades

Para haver uma melhor perceção acerca do risco que as vulnerabilidades podem trazer para um sistema é necessário que haja uma classificação das mesmas. Nesta secção vão ser abordados alguns sistemas de classificação para serem comparados entre si para mostrar as suas vantagens e desvantagens, de maneira a escolhermos um entre os que estão aqui apresentados.

2.1.1 Uso de ontologias para classificação de vulnerabilidades em sistemas computacionais

O uso de ontologias para a classificação de vulnerabilidades [2] tem como objetivo o uso de ontologias para classificar as vulnerabilidades devido ao facto de não existir um sistema próprio para fazer a classificação, pois grande parte das ferramentas

o que fazem é apenas reportar a existência de vulnerabilidades e depois deixar ao critério do responsável que está a avaliar o sistema classifica-as. Este tipo de avaliação utiliza a informação que existe no *Common Vulnerabilities and Exposures* (CVE), sendo este um "repositório" para as vulnerabilidades existentes com o intuito de facilitar o acesso à informação acerca das vulnerabilidades [3], recorrendo também à lingua*Ontology Web Language* (OWL). Este sistema de classificação vai buscar ao CVE a informação relativa às vulnerabilidades, e depois usa o OWL para organizar hierarquicamente as vulnerabilidades e utiliza ainda o *Internet Catalog of Attacks Toolkit* (ICAT) (que atualmente é conhecida como *National Vulnerability Database*) para verificar a informação que é recebida do CVE e assim poder comparar e classificar as vulnerabilidades entre alta, média ou baixa.

2.1.2 Vulnerability Rating and Scoring System (VRSS)

O *Vulnerability Rating and Scoring System* (VRSS) tendo em conta [4] e [5], é um sistema de classificação que foi proposto com o objetivo de tentar unificar os sistemas existentes, para a recolha de informação relativamente às vulnerabilidades este sistema usa a base de dados do X-Force da *International Business of Machines* (IBM) , a *Vupen Security* e a *National Vulnerability Database* (NVD). Combinando o uso de métodos quantitativos com métodos qualitativos, inicialmente o sistema divide as vulnerabilidades em 3 partes *High*, *Medium* e *Low*, usando as propriedades de disponibilidade, confidencialidade e integridade para as avaliar. No passo seguinte usa o método quantitativo para avaliar as vulnerabilidades dando lhe um valor entre 0 e 10, como se pode verificar na figura 2.1.

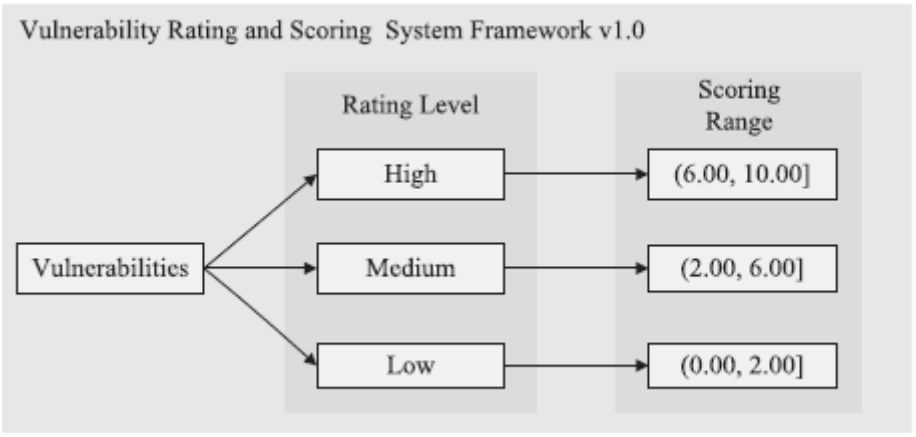


FIGURA 2.1: Classificação de Vulnerabilidades usando VRSS, retirado de [4]

2.1.3 Common Vulnerability Scoring System (CVSS)

O *Common Vulnerability Scoring System* (CVSS) com base em [6] é um sistema de classificação de vulnerabilidades criado pelo Conselho Nacional de Garantia da Infraestrutura dos Estados Unidos (NIAC), que consiste num grupo de pessoas que pertenciam ao departamento de segurança interna dos Estados Unidos, que em 2005 publicaram a primeira versão do CVSS. Atualmente o CVSS pertence à *Forum of Incident Response and Security Teams* (FIRST) [7], este sistema é composto por 3 grupos de métricas, como está representado na figura 2.2:

- **Métricas de Base** – estas métricas demonstram as características que são intrínsecas e fundamentais de uma vulnerabilidade que não se altera ao longo do tempo nem tendo em conta o ambiente do utilizador.
- **Métricas Temporárias** – métricas que representam as características de uma vulnerabilidade que se alteram ao logo do tempo sem ter em conta o ambiente do utilizador.
- **Métricas do Ambiente** – representam as características de uma vulnerabilidade que apenas interessam e são únicas tendo em conta o ambiente que o utilizador usa.

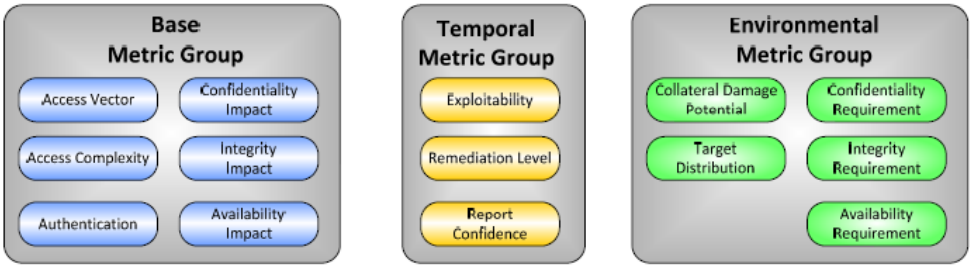


FIGURA 2.2: Grupos de métricas do CVSS, retirado de [6]

Estes grupos têm a função de definir e comunicar as características que são fundamentais para uma vulnerabilidade. O uso desta aproximação para caracterizar as vulnerabilidades dá aos utilizadores uma representação clara e intuitiva das mesmas.

Após terem sido definidos os valores para as métricas de base, usa-se a equação de base para calcular um valor, que vai estar entre 0 e 10, como está demonstrado na figura 2.3. O resultado de base pode depois ser modificado se for incluído os resultados das métricas temporais e as de ambiente, para assim o resultado refletir mais corretamente os risco que a vulnerabilidade representa para o ambiente do utilizador, mas não é obrigatório incluir o resultado dessas métricas.

As métricas usadas para calcular o resultado vão produzir um vetor. Esse mesmo vetor consiste em uma *string* em ASCII que contem os valores que foram atribuídos a cada métrica e é usado para descrever como é que o resultado é atribuído para cada vulnerabilidade, como tal os vetores têm de estar sempre representados com o valor que é atribuído à vulnerabilidade.

Métricas de Base

Como referido acima, este é o conjunto de métricas que se mantêm constantes independentemente do tempo e do ambiente do utilizador, as métricas do Vetor de Ataque, Complexidade do Ataque e de Autenticação servem para demonstrar como as vulnerabilidade é explorada e se existem outras condições que são precisas para as explorarem. Existem mais 3 métricas que medem o impacto, se as vulnerabilidades forem exploradas, podem ter nos sistemas de informação suscetíveis às

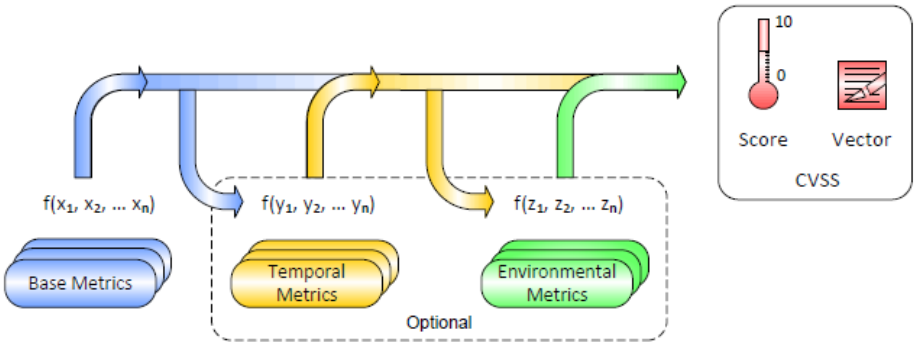


FIGURA 2.3: Funcionamento do CVSS2.0, retirado de [6]

mesmas, tendo como métricas para medir a Confidencialidade, a Integridade e a Disponibilidade.

- Vetor de Ataque (AV) - reflete o contexto da possibilidade de exploração da vulnerabilidade, o valor desta métrica vai ser maior quanto maior for a possibilidade de exploração remota do componente vulnerável. O valor que esta métrica pode tomar é Rede (N), Adjacente (A) e Local (L).

- Complexidade do Ataque (AC) – descreve as condições para lá do controlo do atacante que tem de existir para explorar a vulnerabilidade, como por exemplo a existência de certas configurações de sistema ou exceções computacionais. Esta métrica irá ter valores maiores quanto menor for a complexidade do ataque, sendo que os valores que pode tomar são Alto (H), Médio (M) e Baixo (L).

- Autenticação (Au) - esta métrica tem como propósito de medir o número de vezes que o atacante tem de autenticar-se de modo a que seja possível explorar a vulnerabilidade, quanto mais baixo for maior o valor da métrica. A mesma pode tomar os valores de Múltiplo (M), Singular (S) e de Nenhum (N).

- Impacto à Confidencialidade (C) – mede o impacto a nível da confidencialidade da informação dos recursos que são geridos pelo componente, caso haja uma vulnerabilidade seja explorada com sucesso. O valor da métrica

aumenta tendo em conta a perda de informação que exista devido ao componente que tenha sofrido o ataque, podendo tomar os seguintes valores Completo (C), Parcial (P) e Nenhum (N).

- Impacto à Integridade (I) – mede o impacto que teve na integridade a exploração da vulnerabilidade, esta é maior quanto maior for as consequências para o componente afetado. Tendo os valores de Completo (C), Parcial (P) e Nenhum (N).

- Impacto à disponibilidade (A) – mede o impacto que teve na disponibilidade do componente que foi afetado após ter sido explorada uma vulnerabilidade com sucesso, esta aumenta quanto maior for o tempo que o componente não está disponível. Podendo tomar os valores de Completo (C), Parcial (P) e Nenhum (N).

Métricas Temporais

Como foi explícito anteriormente, estas métricas medem a ameaça de uma vulnerabilidade puder sofrer modificações com o tempo para tal vão ter-se em conta os 3 fatores seguintes:

- Exploração (E) - esta métrica mede o estado das técnicas de exploração da vulnerabilidade ou a publicação do código relativo à mesma, quanto mais fácil seja a exploração da vulnerabilidade maior será o valor que a métrica irá assumir. Sendo os valores que a mesma pode ter os seguintes, Não Provado (U), Prova de conceito (POC), Funcional (F), Alto (H) e Não Definido (ND).

- Nível de Remediação (RL) – mede a prioridade que existe na tentativa de criar um *patch* ou uma solução para mitigar uma vulnerabilidade, quanto menos oficial e permanente for maior será o valor que vai tomar. Tendo como valores Não Disponível (U), Solução Alternativa (W), Reparo Temporário (TF), Reparo Oficial (O) e Não Definido (ND).

- Relato de Confiança (RC) – mede o nível de confiança que se tem na existência da vulnerabilidade e nos detalhes que são conhecidos da mesma, quanto

mais fontes de confiança maior a validarem maior vai ser o valor que esta métrica vai assumir, tendo como valores Não Definido (ND), Confirmado (C), Não Corroborada (UR), e Não Confirmado (UC) .

Métricas do Ambiente

Como indicado anteriormente, estas métricas servem para ter em conta o ambiente do utilizador onde foi encontrada a vulnerabilidade de modo a que seja possível ter uma classificação mais realista relativa à mesma, estas métricas não são obrigatórias para o cálculo do resultado final, tendo as seguintes métricas:

- Potencial Dano Colateral (CDP) - esta métrica mede a perda dos ativos via dano ou roubo de equipamentos, mede também as perdas que resultam a nível económico, esta métrica pode ser Nenhum (N), Baixo (L), Baixo-Médio (LM), Médio-Alto (MH), Alto (H) e Não Definido (ND).
- Distribuição dos Alvos (TD) - mede a extensão de sistemas que estão vulneráveis, a métrica pode assumir os valores de Nenhum (N), Baixo (L), Médio (M), Alto (H) e Não Definido (ND).
- Requerimentos de Segurança (CR, IR, AR) - esta métrica tem como propósito analisar as métricas base relativamente à confidencialidade, integridade e o impacto da disponibilidade, podendo assim haver algumas alterações ao valor que foi inicialmente medido nas métricas de base. Tendo como valores Baixo (L), Médio (M), Alto (H) e Não Definido (ND).

Equações

As equações que são implementadas pelo CVSS estão apresentadas no anexo A (ver página 93).

2.1.4 Comparação entre os diversos sistemas de classificação de vulnerabilidades

Tendo em conta que os sistemas VRSS e o uso de ontologias para a classificação de vulnerabilidades são apenas propostas de sistemas que tentam apresentar alternativas para fazer a classificação melhor mas nenhum deles chega a ser um dos mais utilizados em comparação ao CVSS. Este último atualmente é o mais utilizado para classificação de vulnerabilidades, sendo utilizado, por exemplo, pela NVD e muitas outras bases de dados de vulnerabilidades bem como em alguns sistemas das agências federais dos Estados Unidos da América [8], outro facto que faz este também ser tão usado é devido mesmo que dois analistas utilizem o CVSS para classificar uma vulnerabilidade igual o resultado de ambos vai ser sempre o mesmo [9]. Como tal vai ser usado o CVSS como sistema de classificação na plataforma tornando assim a avaliação das vulnerabilidades, bem como as perceção das mesmas mais fácil, devido a este sistema ser tão conhecido e utilizado.

2.2 Identificadores de vulnerabilidades

De maneira a que a informação das vulnerabilidades fique mais completa vai ser possível ao utilizador inserir os identificadores que correspondem a vulnerabilidades que foram catalogadas em várias plataformas. Nesta secção vão ser abordados os vários identificadores que o utilizador pode inserir de modo a complementar a informação acerca da vulnerabilidade.

2.2.1 Common Vulnerabilities and Exposures (CVE)

Como já foi mencionado, o projeto *Common Vulnerabilities and Exposures* (CVE) [10] [3], é uma lista que possui informação relativa a vulnerabilidades de segurança bem como exposições que tem como propósito atribuir nomes a problemas de segurança que são conhecidos, o grande objetivo desta listagem é a de facilitar a

partilha de informação relativamente às várias vulnerabilidades usando um processo de atribuição de identificadores. Para cada vulnerabilidade é atribuído um identificador CVE (CVE-ID), este é único para cada uma delas, como podemos observar na figura 2.4.

CVE-ID

CVE-2013-1937 Learn more at National Vulnerability Database (NVD)
• Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings

Description

Multiple cross-site scripting (XSS) vulnerabilities in tbl_gis_visualization.php in phpMyAdmin 3.5.x before 3.5.8 might allow remote attackers to inject arbitrary web script or HTML via the (1) visualizationSettings[width] or (2) visualizationSettings[height] parameter.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- FULLDISC:20130409 [waraxe-2013-SA#102] - Reflected XSS in phpMyAdmin 3.5.7
- URL:http://archives.neohapsis.com/archives/fulldisclosure/2013-04/0101.html
- MLIST:[oss-security] 20130409 Re: CVE Request: Self-XSS in phpmyadmin fixed in 3.5.8
- URL:http://openwall.com/lists/oss-security/2013/04/09/13
- MISC:http://packetstormsecurity.com/files/121205/phpMyAdmin-3.5.7-Cross-Site-Scripting.html
- MISC:http://www.waraxe.us/advisory-102.html
- CONFIRM:https://github.com/phpmyadmin/phpmyadmin/commit/79089c9bc02c82c15419fd9d6496b8781ae08a5a
- CONFIRM:http://www.phpmyadmin.net/home_page/security/PMASA-2013-1.php
- FEDORA:FEDORA-2013-5604
- URL:http://lists.fedoraproject.org/pipermail/package-announce/2013-April/103195.html
- FEDORA:FEDORA-2013-5620
- URL:http://lists.fedoraproject.org/pipermail/package-announce/2013-April/103184.html
- FEDORA:FEDORA-2013-5623
- URL:http://lists.fedoraproject.org/pipermail/package-announce/2013-April/103188.html
- MANDRIVA:MDVSA-2013:144
- URL:http://www.mandriva.com/security/advisories?name=MDVSA-2013:144
- SUSE:openSUSE-SU-2013:1065
- URL:http://lists.opensuse.org/opensuse-updates/2013-06/msg00181.html

FIGURA 2.4: Exemplo de um CVE-ID, retirado de [10]

2.2.2 Common Weakness Enumeration (CWE)

O *Common Weakness Enumeration* (CWE) [11] tem como propósito servir de uma lista de guia para defeitos comuns que existem em software que podem levar a que existam falhas de segurança que podem ser exploradas. Esta lista tem como objetivo de servir como um "dicionário comum"para todas as falhas de segurança a nível de software, com o intuito de ser utilizado pelas ferramentas de segurança para procurar por esses defeitos e ainda servir como uma base de conhecimento para os defeitos e falhas que existem de forma a que seja possível prevenir os mesmos. Para as falhas que são reportadas vão lhe ser atribuídas um CWE-ID, a informação que estes vão conter em relação às falhas são a descrição com os pormenores relativos à falha, a denominação alternativa que a mesma pode ter, uma representação de como ela funciona, como pode ser explorada e a consequência

dessa eventual exploração, soluções para a falha, o respetivo CVE-ID e outras referências que possam existir.

2.2.3 Bugtraq

O Bugtraq [12] [13] consiste numa *mailing list* que é gerida pela Symantec no site da SecurityFocus, no qual é um dos fóruns mais populares de segurança na Internet. Nesta *mailing list* vão estar referênciados os *Bugtraq Identifier* (BID), que consistem em números que são atribuídos a vulnerabilidades que são submetidas para esta lista, no qual vai estar informação relativa à classe da vulnerabilidade, a data em que esta foi reportada, a data da última atualização à mesma, que a reportou, os sistemas que estão vulneráveis contra ela, a forma como a explorar, soluções, referências externas e discussões acerca da vulnerabilidade, como pode ser observado na figura 2.5.

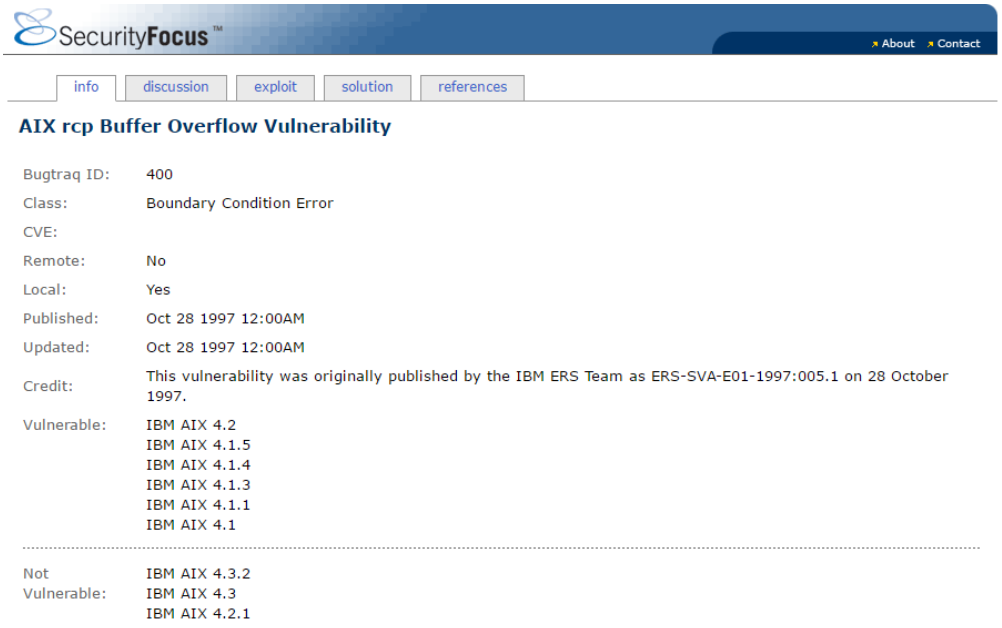
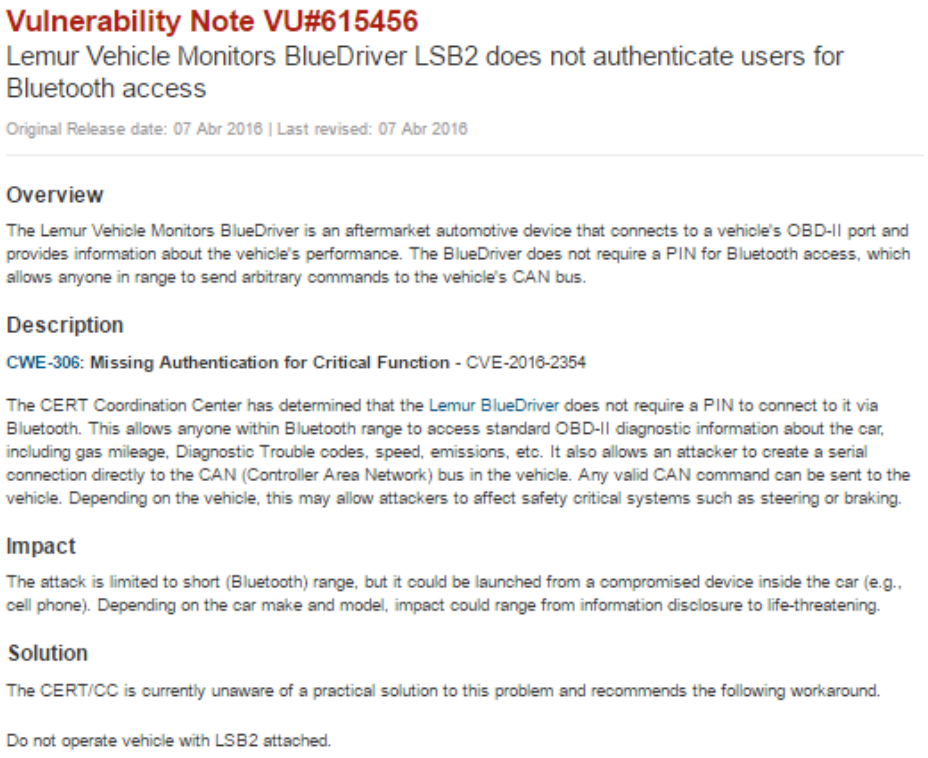


FIGURA 2.5: Exemplo de um BID, retirado de [13]

2.2.4 Computer Emergency Response Team (CERT)

O *Computer Emergency Response Team* (CERT) [14] [15] consiste em um grupo de experts na área de segurança de computadores que estão responsáveis por incidentes. Este grupo criou uma base de dados denominada de *Vulnerability Notes Database*, onde disponibilizam informação acerca das vulnerabilidades que encontram com um resumo sobre as mesmas, informação técnica mais detalhada, maneiras como as mitigar e quem é afetado pelas mesmas, como pode ser observado na seguinte figura 2.6.



Vulnerability Note VU#615456

Lemur Vehicle Monitors BlueDriver LSB2 does not authenticate users for Bluetooth access

Original Release date: 07 Abr 2016 | Last revised: 07 Abr 2016

Overview

The Lemur Vehicle Monitors BlueDriver is an aftermarket automotive device that connects to a vehicle's OBD-II port and provides information about the vehicle's performance. The BlueDriver does not require a PIN for Bluetooth access, which allows anyone in range to send arbitrary commands to the vehicle's CAN bus.

Description

CWE-306: Missing Authentication for Critical Function - CVE-2016-2354

The CERT Coordination Center has determined that the Lemur BlueDriver does not require a PIN to connect to it via Bluetooth. This allows anyone within Bluetooth range to access standard OBD-II diagnostic information about the car, including gas mileage, Diagnostic Trouble codes, speed, emissions, etc. It also allows an attacker to create a serial connection directly to the CAN (Controller Area Network) bus in the vehicle. Any valid CAN command can be sent to the vehicle. Depending on the vehicle, this may allow attackers to affect safety critical systems such as steering or braking.

Impact

The attack is limited to short (Bluetooth) range, but it could be launched from a compromised device inside the car (e.g., cell phone). Depending on the car make and model, impact could range from information disclosure to life-threatening.

Solution

The CERT/CC is currently unaware of a practical solution to this problem and recommends the following workaround.

Do not operate vehicle with LSB2 attached.

FIGURA 2.6: Exemplo de uma vulnerabilidade reportada na *Vulnerability Notes Database*, retirado de [14]

2.2.5 Exploit Database (EDB)

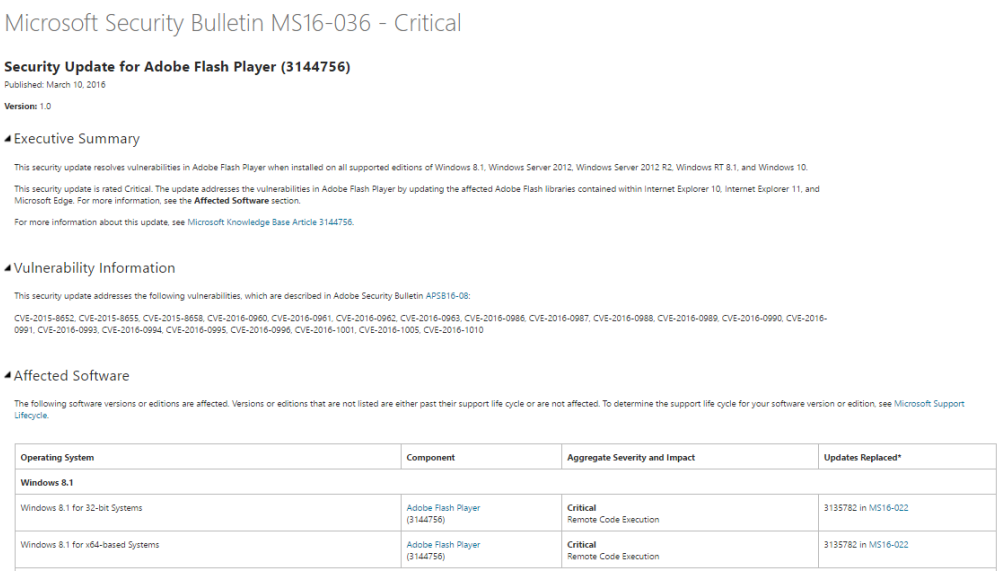
Como o próprio nome indica a *Exploit Database* [16] consiste numa base de dados que contém informação relativa a formas de explorar as vulnerabilidades que são tornadas públicas, esta foi desenvolvida para quem efetua testes de segurança e

faz pesquisas relativamente a vulnerabilidades. A informação que cada entrada desta base de dados vai ter é o identificador que lhe é atribuído (EDB-ID), o CVE relativo à vulnerabilidade, a data de quando foi descoberta, o autor, o link relativo ao software vulnerável, a versão do mesmo e o sistema operativo em que foi testado o *exploit*. Na figura 2.7 temos um exemplo de um *exploit* que está reportado e verificado no EDB.

FIGURA 2.7: Demonstração de um exploit reportado na *Exploit Database*, retirado de [16]

2.2.6 Microsoft Security Bulletins

Os Microsoft Security Bulletins [17] são boletins de segurança que são feitos pela equipa da Microsoft que é responsável pela segurança e que emite com o intuito de reportar vulnerabilidades que são descobertas, sendo estes criados mensalmente. Muitas destas vulnerabilidades após terem sido reportadas nestes boletins são lançados *updates* por parte da Microsoft para fazer com que os sistemas deixem de estar expostos às mesmas. Cada vulnerabilidade reportada vai ter um indentificador próprio para a mesma ficando depois identificada como por exemplo, MS16-036, em que o 16 é identifica o ano em que a vulnerabilidade é reportada e o 036 o número da vulnerabilidade, na figura 2.8 podemos observar como é reportado uma vulnerabilidade no Security Bulletin.



Microsoft Security Bulletin MS16-036 - Critical

Security Update for Adobe Flash Player (3144756)

Published: March 10, 2016

Version: 1.0

Executive Summary

This security update resolves vulnerabilities in Adobe Flash Player when installed on all supported editions of Windows 8.1, Windows Server 2012, Windows Server 2012 R2, Windows RT 8.1, and Windows 10.

This security update is rated Critical. The update addresses the vulnerabilities in Adobe Flash Player by updating the affected Adobe Flash libraries contained within Internet Explorer 10, Internet Explorer 11, and Microsoft Edge. For more information, see the **Affected Software** section.

For more information about this update, see Microsoft Knowledge Base Article 3144756.

Vulnerability Information

This security update addresses the following vulnerabilities, which are described in Adobe Security Bulletin APSB16-08:

CVE-2015-8652, CVE-2015-8655, CVE-2015-8658, CVE-2016-0960, CVE-2016-0961, CVE-2016-0962, CVE-2016-0963, CVE-2016-0986, CVE-2016-0987, CVE-2016-0988, CVE-2016-0989, CVE-2016-0990, CVE-2016-0991, CVE-2016-0993, CVE-2016-0994, CVE-2016-0995, CVE-2016-0996, CVE-2016-1001, CVE-2016-1005, CVE-2016-1010

Affected Software

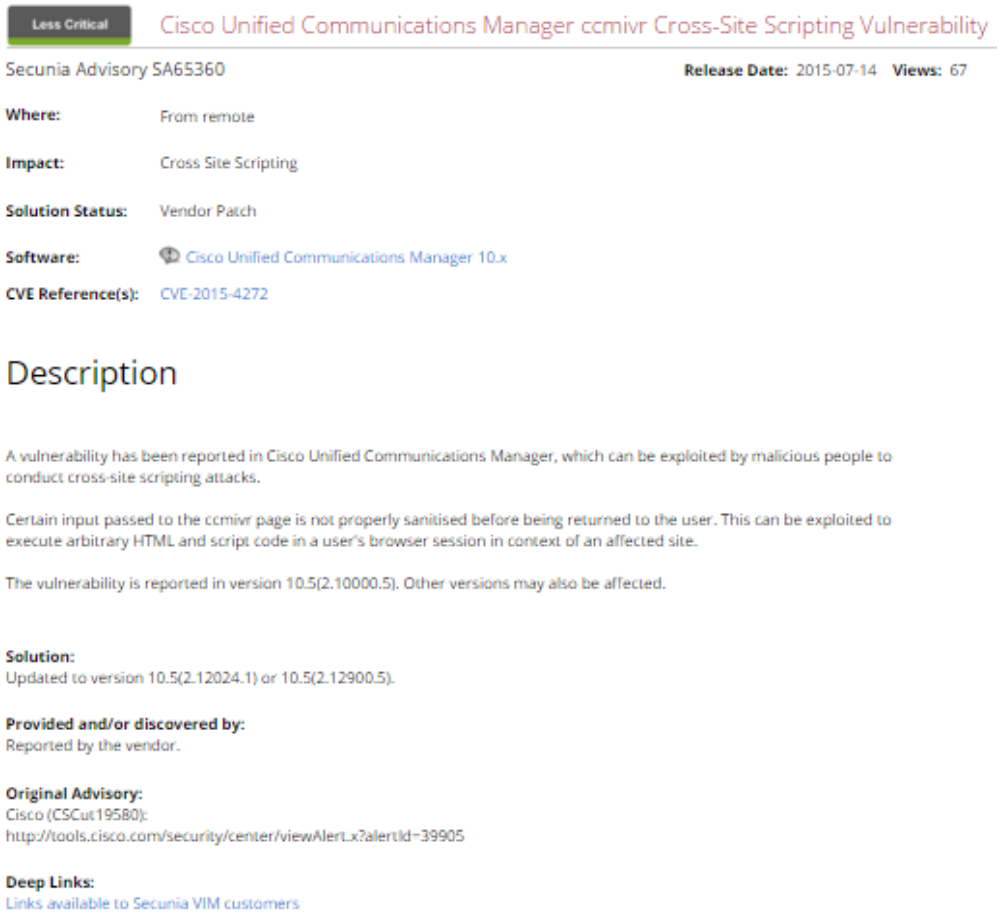
The following software versions or editions are affected. Versions or editions that are not listed are either past their support life cycle or are not affected. To determine the support life cycle for your software version or edition, see Microsoft Support Lifecycle.

Operating System	Component	Aggregate Severity and Impact	Updates Replaced*
Windows 8.1			
Windows 8.1 for 32-bit Systems	Adobe Flash Player (3144756)	**Critical** Remote Code Execution	3135782 in MS16-022
Windows 8.1 for x64-based Systems	Adobe Flash Player (3144756)	**Critical** Remote Code Execution	3135782 in MS16-022

FIGURA 2.8: Exemplo de uma vulnerabilidade reportada em um Microsoft Security Bulletin, retirado de [17]

2.2.7 Secunia

A Secunia [18] é uma empresa que desenvolve software relacionado com a segurança, que possui uma base de dados de vulnerabilidades que é atualizada pelos seus profissionais e também permite qualquer utilizador que se inscreva no seu site e deseje reportar alguma vulnerabilidade que tenha descoberto. Os dados que vão ser guardados relativamente às vulnerabilidades vão ser o nome da mesma, uma descrição detalhada, a solução, referências do CVE, o que é afetado por ela, o impacto que tem, quem a reportou e ligações com mais informação relativa à mesma, como é possível observar-se na figura 2.9.



Less Critical

Cisco Unified Communications Manager ccmivr Cross-Site Scripting Vulnerability

Secunia Advisory SA65360 **Release Date:** 2015-07-14 **Views:** 67

Where: From remote

Impact: Cross Site Scripting

Solution Status: Vendor Patch

Software: Cisco Unified Communications Manager 10.x

CVE Reference(s): CVE-2015-4272

Description

A vulnerability has been reported in Cisco Unified Communications Manager, which can be exploited by malicious people to conduct cross-site scripting attacks.

Certain input passed to the ccmivr page is not properly sanitised before being returned to the user. This can be exploited to execute arbitrary HTML and script code in a user's browser session in context of an affected site.

The vulnerability is reported in version 10.5(2.10000.5). Other versions may also be affected.

Solution:
Updated to version 10.5(2.12024.1) or 10.5(2.12900.5).

Provided and/or discovered by:
Reported by the vendor.

Original Advisory:
Cisco (CSCut19580):
http://tools.cisco.com/security/center/viewAlert.x?alertId=39905

Deep Links:
Links available to Secunia VIM customers

FIGURA 2.9: Exemplo de uma vulnerabilidade reportada na base de dados de vulnerabilidades da Secunia, retirado de [18]

2.2.8 Information Assurance Vulnerability Alerts

O *Information Assurance Vulnerability Alerts* (IAVA) [19] [20] é um sistema de alertas utilizado por parte de organizações que trabalham com o departamento da defesa dos Estados Unidos da América para uniformizar o anúncio relativo a vulnerabilidades bem como a mitigação das mesmas. Estes alertas são lançados quando é encontrada uma vulnerabilidade que coloca em risco os sistemas do departamento da defesa.

2.3 Plataformas de registo de auditorias existentes

Esta secção tem como objetivo mostrar soluções já existentes similares à plataforma que foi o alvo de estudo e de desenvolvimento neste trabalho, bem como mostrar o que torna a plataforma proposta única em relação às soluções que aqui são apresentadas.

2.3.1 Bsafe/Cross-Platform Audit

O Bsafe/Cross-Platform Audit [21] é uma plataforma de gestão de registos empresariais e uma ferramenta de monitorização de atividades na base de dados, com o foco em organizações que estejam a correr vários sistemas e em plataformas diferentes. O mesmo serve para consolidar eventos de auditoria específicos e faz com que os mesmos fiquem disponíveis para a gestão de topo e auditores com uma interface de uso fácil.

As plataformas que são suportadas por esta solução são:

- IBM i (AS400)
- IBM z (Mainframe)
- Windows
- SQL Server
- Unix/AIX
- Linux
- Oracle
- Sun Solaris

Esta solução permite monitorizar a atividade de um utilizador através de computadores diferentes em diversas plataformas e apresentar a atividade no ecrã num registo (log) de eventos usando também o formato gráfico para complementar.

Como Funciona

A plataforma monitoriza e recolhe informação sobre os eventos de segurança de auditoria que ocorrem em cada computador, estes depois podem ser vistos e classificados diretamente e ficam logo prontos para serem transferidos e guardados no repositório central quando se quiser. A importação da informação sobre as auditórias pode ser feita a qualquer altura ou então pode ser programado para ser feito periodicamente em intervalos de tempo definidos pelo utilizador. Também oferece a escolha de especificar o grupo de eventos de auditoria que queremos importar. Os mesmos eventos que são importados de diferentes plataformas para o Cross-Platform Audit são guardados em formato uniforme para serem filtrados e analisados como se tivessem sido originados no computador onde está a correr a solução.

Os eventos de segurança que são guardados em cada computador são definidos pela política de auditoria, sendo que o Cross-Platform Audit deixa que seja o auditor a definir como é que esta vai funcionar para cada computador e que tipos de eventos vão ser guardados.

Principais características

- Uma interface gráfica única baseada na consola de gestão para todas as plataformas – Uma consola única de gestão para todas as plataformas da qual é possível gerir os registos consolidados e aceder aos diferentes nodos monitorizados;
- Múltiplos tipos de eventos – Inclui eventos a nível de sistema, TCP/IP, de instruções de SQL, entre outros;

- SOC (Centro de segurança operacional) – Ferramenta gráfica para análise dos eventos de segurança, hábitos e incidentes, no qual também é possível gerar ficheiros com todas as tabelas de resumo e gráficos que são apresentados;
- Políticas de gestão de auditorias – Define o tipo de registos de eventos que vão ser guardados pelos computadores;
- Ferramentas de conformidade – Cria um *template* de base tendo em conta as politicas de conformidade com verificações de desvios e opções de reparação.

Benefícios

- Coleções de vários formatos de dados em uma base uniforme;
- Uma monotorização compreensiva em um ambiente de multiplataformas;
- Filtro capaz de encontrar o evento usando características especificas;
- Análise gráfica dos dados estatísticos de segurança;
- Correlação de eventos que são aparentemente diferentes para uma auditoria uniforme;
- Informação completa e compreensiva de auditoria para cada evento.

2.3.2 FireEye Série CM

A FireEye série CM [22] consiste numa plataforma distribuída em rede entre outras plataformas fornecidas pela FireEye, fazendo com que a informação seja partilhada em tempo real entre as várias plataformas, tendo ainda a funcionalidade de gerar relatórios de auditoria com a informação que recolhe das outras plataformas.

Principais Funcionalidades

- Oferta de controlos integrados para a distribuição com várias plataformas;

- Permitir a prevenção de ameaças mistas usando correlação "multivetorial";
- Plataforma dedicada que em pouco tempo pode ser distribuída;
- *Dashboard* de segurança no qual fornece a informação relativa ao status da proteção contra ataques avançados e direcionados;
- Torna mais rápida a criação de relatórios e auditorias devido à existência de um repositório de eventos de segurança;
- Simplificação na gestão das várias plataformas FireEye, reduzindo também o tempo que é gasto em gestão de configurações, atualizações de ameaças e atualizações de software.

2.3.3 Analyzer

O Analyzer [23] é uma ferramenta de análise de tráfego *web-based* e de geração de relatórios de fácil uso, fornecendo informação em tempo real bem como uma visão evolutiva ao nível de histórico do estado da rede, da segurança e do seu rendimento.

Benefícios

- Relatórios gráficos detalhados que permitem a análise das ameaças e atividades;
- Syslog de nova geração que simplifica o resumo de dados;
- Fornece uma ideia forte sob o estado e comportamento do acesso remoto seguro e da proteção contínua de dados;
- Relatórios de "relance" facilitam a análise rápida;
- Compilação de informação facilita a geração de relatórios;
- O relatar de várias ameaças fornece informação instantânea relativamente a ameaças e ataques;

- Conhecimento de novos ataques fornece informação mais detalhada sobre ataques específicos.

2.3.4 Titania Nipper Studio

O Titania Nipper Studio [24] é uma solução que faz auditoria à rede, fazendo uma verificação às configurações nativas dos dispositivos, permitindo ao auditor criar relatórios de auditoria personalizáveis.

Principais características

- Auditoria de segurança
  - Produz uma auditoria de segurança detalhada em que foca as vulnerabilidades das configurações;
  - Resume a segurança dos dispositivos de rede (firewall, routers e switches);
  - Classifica as vulnerabilidades tendo em conta a gravidade e a facilidade de correção das mesmas;
  - Fornece um plano de ação fácil para mitigar as vulnerabilidades tendo em conta as configurações;
  - Tem uma função de comparação de relatórios de auditoria de modo a ver o progresso que foi feito;
- Relatórios personalizáveis
  - Exporta secções específicas do relatório da auditoria de segurança;
  - Escolha a partir de uma grande variedade de opções de configuração dentro do relatório ou ao classificar informação no documento;
  - Modifica o *template* de base do relatório para se adaptar às nossas necessidades;
  - Constrói um ou vários perfis para guardar as configurações;

- Usa o rating para vulnerabilidades CVSS v2;
- Inserção das nossas próprias métricas.

2.3.5 Faraday

O Faraday [25] consiste num IDE para quem realiza testes de intrusão, esta oferece ao utilizador um ambiente de testes similar ao que já está familiarizado tendo uma consola para realizar todos os comandos que o utilizador usa para fazer os testes que normalmente executa via a linha de comandos do seu terminal. Após ter feito a recolha de informação a aplicação gera um *dashboard* para possibilitar a visualização dos dados que são recolhidos nos testes, aqui vai ser possível ver as vulnerabilidades que são descobertas, informação relativa aos hosts, os serviços em que fora reportados mais vulnerabilidades e um histórico de comandos que foram utilizados.

Esta aplicação oferece alguns plugins para várias ferramentas que fazem a deteção de vulnerabilidades automaticamente como Nessus, Metasploit, Nmap, OpenVas, Burp, entre outras. O Faraday apenas é suportado em MacOSX ou em plataformas Linux, não sendo ainda suportado em máquinas que corram Windows, tendo uma versão gratuita para os utilizadores experimentarem sem ter todas as funcionalidades disponíveis sendo que por exemplo, a funcionalidade de gerar relatórios apenas está incluída para os utilizadores privados e a possibilidade de personalizar os mesmo apenas está disponível para a versão para empresas.

2.3.6 Penteston

O Penteston [26] é uma plataforma que está disponível online que permite aos utilizadores que se registem a possibilidade de executar as ferramentas de scan automático que estão integradas na plataforma para fazerem os testes de intrusão, que após terem sido executados os testes é possível a geração de um relatório que é personalizável pelos utilizadores. As ferramentas disponíveis para fazer os testes de intrusão são as seguintes.

- Nmap
- DNSenum
- DNSRecon
- WPScan
- Joomscan
- Nikto Web Scanner
- OWASP ZAP

O Penteston também permite importação de scans que tenham sido realizados pelo Acunetix e o Nessus, a mesma também oferece uma funcionalidade de possibilitar a comparação entre dois relatórios que são gerados pela plataforma.

2.3.7 Dradis

Dradis [27] consiste numa plataforma *open-source* capaz de ser executada em qualquer ambiente, necessitando apenas de um browser. A mesma tem como propósito de guardar as vulnerabilidades que são reportados por auditores e assim permitir que os utilizadores possam colaborar complementando com mais informação, também permite aos utilizadores gerar relatórios acerca das vulnerabilidades personalizáveis e fornece ao utilizador a possibilidade de fazer *upload* de vulnerabilidades a partir de ficheiros que são gerados de scanners automáticos de vulnerabilidades, como o Nessus, Nmap, Metasploit entre outros.

2.3.8 Comparação entre as plataformas

Na tabela 2.1 estão presentes as principais características que vão servir de base para a plataforma a desenvolver e vão servir de base de comparação entre estas plataformas e a plataforma.

	C1	C2	C3	C4	C5	C6
Bsafe/Cross-Platform Audit	N	N	N	S	N	S
FireEye série CM	N	N	N	S	N	N
Analyzer	N	S	N	S	N	N
Titania Nipper Studio	N	S	S	S	N	S
Faraday	S	S	N	S	S	S*
Penteston	N	N	N	S	S	S
Dradis	N	S	N	N	S	S

Tabela 2.1: Comparação entre plataformas

Legenda:

- S- Sim / N - Não
- C1 - Disponibilização de duas interfaces, uma para a organização cliente e outra para a organização fornecedora.
- C2 - Ter capacidade de introduzir novas vulnerabilidades, em qualquer uma das duas interfaces.
- C3 - Geração de gráficos usando o CVSS 2.0.
- C4 - Geração de *dashboards* globais que demonstrem todas as vulnerabilidades com os KPIs específicos.
- C5 - Implementar interfaces que são capazes de importar resultados de scanners de vulnerabilidades automáticos.
- C6 - Geração de relatórios personalizáveis.

* Apenas disponível para a versão da empresa.

As características que foram apresentadas acima são consideradas como a base de comparação, devido a estes serem consideradas as principais funcionalidades de base para o desenvolvimento da plataforma servindo assim como uma base para

podermos comparar todas as plataformas que foram investigadas para assim se mostrar as vantagens do que está a ser desenvolvido.

Como se pode reparar na tabela 2.1 nenhuma destas plataformas oferece mais do que uma interface, seja uma para o cliente e outra para organização, bem como não possibilitam a importação de resultados que são passados através de scans de ferramentas de deteção automáticas de vulnerabilidades, pois estas plataformas normalmente oferecem o seu próprio scanner de vulnerabilidades e a partir deste faz a geração de *dashboards* e informação relevante em relação às vulnerabilidades encontradas. Uma das plataformas que é semelhante ao que é proposto para esta dissertação é o Titania Nipper Studio, devido à capacidade de deixar o utilizador inserir as vulnerabilidades bem como a disponibilização de personalização de relatórios e o uso do CVSS para classificar as vulnerabilidades. O Faraday e o Penteston também se aproximam bastante da plataforma proposta, sendo que estas oferecem versões pagas para assim se ter acesso a todas as funcionalidades mas ainda nas versões gratuitas já estão disponíveis quase todas as caraterísticas, no caso do Penteston oferecem a possibilidade de geração de relatórios personalizáveis, enquanto no caso do Faraday essa funcionalidade só está disponível para a versão empresarial. Outra plataforma que se assemelha muito com a que está a ser desenvolvida é a Dradis, esta para além de oferecer tudo gratuitamente ainda dá aos seus utilizadores a possibilidade de fazer o *upload* de ficheiros que são gerados a partir de vários scanners automáticos e integrar os seus resultados na plataforma, bem como a geração de relatórios personalizáveis com a informação que queremos apresentar acerca das vulnerabilidades, apenas não oferece interface para o cliente, nem *dashboards* para os utilizadores devido a esta ser uma plataforma mais virada para os *experts* de segurança.

Como podemos observar nenhuma das plataformas examinadas conseguem satisfazer as caraterísticas que foram propostas como base para o desenvolvimento da plataforma, sendo que muitas destas são plataformas que são pagas ou estão dependentes de outros serviços e não oferecem a liberdade ao utilizador de inserir

a informação que deseja ou então utilizar um scanner de vulnerabilidades automático para fazer os seus testes de intrusão, bem como fornecer ao utilizador opção de geração de relatórios que possam ser personalizados.

2.4 Classificação de Vulnerabilidades

De forma a que seja possível agregar as vulnerabilidades é necessário que as mesmas sejam classificadas para assim ser possível aos utilizadores conseguirem obter mais informações relativas à vulnerabilidade, podendo até encontrar outras que sejam similares que possuam a mesma classificação e para as quais já possa existir uma solução que seja aplicável à sua vulnerabilidade.

O Open Web Application Security Project (OWASP) é uma organização sem fins lucrativos que trabalha no ramo da segurança informática, em que qualquer pessoa que esteja disposta a participar e a ajudar na mitigação dos riscos de segurança é bem-vindo, pois todo o material que disponibilização é *open-source*.

O OWASP possui dois projetos, um para as aplicações móveis e outro para as aplicações web, os quais têm como grande propósito identificar as 10 falhas de segurança mais críticas para as aplicações, como se pode observar nos seguintes parágrafos.

2.4.1 OWASP Top 10

Este foi o top 10 mais recente lançado pela OWASP tendo em conta [28]:

- A1: Injeção de dados - As falhas por injeção ocorrem quando dados que não são de natureza segura são enviados para um interpretador, estes dados podem levar ao mesmo a executar comandos não desejáveis ou permitir que o atacante tenha acesso a informação. Um exemplo deste tipo de ataques é a injeção via SQL.

- A2: Quebra de autenticação e Gerenciamento de Sessão - As funções da aplicação que gerem a autenticação e a sessão não estão bem implementadas, tendo assim como consequência possibilitar que um atacante consiga comprometer as palavras passe e os *tokens* da sessão.

- A3: Cross-Site Scripting (XSS) - Esta falha acontece quando uma aplicação recebe dados que não os filtra nem os valida, permitindo assim a execução de scripts que são enviados pelos atacantes que podem ter como consequência, por exemplo, o redireccionamento dos utilizadores para sites maliciosos.

- A4: Referência Insegura e Direta a Objetos - Esta falha ocorre quando um programador deixa exposto referências a um objeto interno, arquivos, diretorias ou registo da base de dados, isto permite ao atacante manipular estas referências caso não exista uma verificação ao acesso ou outro género de segurança imposto para impedir estes eventuais acessos.

- A5: Configuração Incorreta de Segurança - Para a aplicação ter uma boa segurança é necessário que a mesma esteja implementada e configurada corretamente ao nível da framework, do servidor web, o servidor da aplicação e a base de dados. Estas configurações têm de ser mantidas, tendo também de se atualizar o software para garantir que não existam falhas a esse nível.

- A6: Exposição de Dados Sensíveis - Esta falha acontece quando existe dados sensíveis (dados de cartões de crédito, palavras passe, ids de autenticação) que não estão encriptados, como tal estes dados devem ter uma proteção extra e mais algumas medidas de segurança adicionais para além das que o navegador fornece.

- A7: Falta de Função para Controlo do Nível de Acesso - Existem certas funcionalidades em que é necessário verificar os direitos de acesso que os utilizadores possuem para tal é necessário executar verificações para o controlo do acesso a nível do servidor para cada uma das funcionalidades.

- A8: Cross-Site Request Forgery (CSRF) - Esta falhar permite ao atacante obrigar o browser da vítima a criar pedidos à aplicação que está vulnerável

e esta por sua vez vai aceitar os pedidos pois aceita a aplicação pensa que é a vítima que os está a pedir.

- A9: Utilização de Componentes Vulneráveis Conhecidos - Aplicações que usem componentes que são conhecidos por estarem vulneráveis, faz com que as mesmas fiquem expostas permitindo assim que possam existir ataques que podem trazer graves consequências às aplicações.

- A10: Redirecionamentos e Encaminhamentos Inválidos - Se não existir uma validação adequada por parte da aplicação, os atacantes podem fazer com que os utilizadores sejam redirecionados para onde quiserem, como por exemplo sites de *phishing* ou de *malware*.

2.4.2 OWASP Mobile Top 10

Este é o Top 10 mais recente relativo às vulnerabilidades de aplicações móveis relatado em [29]:

- M1: Fraco Controlo do Lado do Servidor - Esta falha acontece devido a falhas na programação, neste caso devido a não existir uma verificação da informação passada pela componente móvel faz com que a aplicação fique insegura e permitindo assim que o atacante possa enviar código malicioso para o servidor de maneira a explorá-lo podendo criar assim uma das vulnerabilidades que estão reportadas no OWASP Top 10.

- M2: Armazenamento de Dados Inseguro - Acontece quando os programadores assumem que os utilizadores ou o *malware* não iram aceder ao sistema de ficheiros dos dispositivos móveis, deixando assim exposta informação que pode ser considerada sensível. Para se evitar que seja difícil ou até impossível aceder à informação é necessário que os dados que ficam guardados no sistema de ficheiros relativos à aplicação estejam encriptados, ou então que não se guarde nenhuma informação.

- M3: Proteção na Camada de Transporte Insuficiente - Esta falha ocorre devido à falta de proteção a nível da camada de transporte, e isto acontece frequentemente pois muitas aplicações móveis não se protegem a nível do tráfego de rede e apenas usam SSL/TLS para proteger a autenticação, deixando assim os dados que são enviados e recebidos expostos.

- M4: Fuga de Dados não Intencional - Ocorre quando os *developers* deixam informação sensível em localizações do dispositivo móvel de fácil acesso para outras aplicações que também estejam presentes no mesmo dispositivo.

- M5: Fraca Autenticação e Autorização - Esta falha aparece devido a fracos métodos de controlo da autenticação, ou até a falta dos mesmos, que vai assim permitir ao atacante que execute funções a nível da aplicação ou do servidor de *backend* anonimamente.

- M6: Criptografia Quebrada - Esta falha só ocorre quando se usa algoritmos de encriptação fracos ou com falhas para se encriptar as informações sensíveis, deixando-as assim vulneráveis caso um atacante as consiga capturar.

- M7: Injeção do Lado do Cliente - Para esta falha ocorrer é necessário que seja executado código malicioso no dispositivo móvel do cliente através da aplicação móvel, este por sua vez vai tentar comprometer a framework da aplicação, podendo levar a mesma a realizar ações que podem ser comprometer informação que está no dispositivo.

- M8: Decisões de Segurança Via Inputs não Confiáveis - Tal acontece quando existem campos que estão escondidos ou valores que são usados para distinguir utilizadores com mais permissões que outros e caso um atacante consiga intercetar as chamadas e mexer com os parâmetros, podendo levar à aplicação a ter comportamentos indevidos ou ainda ceder permissões de alto nível ao atacante.

- M9: Má Gestão da Sessão - Para manter o estado da ligação entre a aplicação e os servidores de *backend*, a aplicação usa tokens de sessão para manter o estado e estes são enviados via HTTP ou SOAP. Esta falha ocorre quando

o cookie que é gerado para a sessão é partilhado inadvertidamente com um atacante durante uma das transações entre os servidores e a aplicação.

- M10: Falta de Proteção Binária - A falta de proteção binária facilita o trabalho que o atacante tem de analisar, de recriar e de modificar uma aplicação. Esta falha é muito comum, devido a muitas aplicações serem lançadas sem uma proteção deste género.

2.4.3 CWE Sans Top25

O CWE Sans Top 25 [30] consiste numa lista com os erros mais críticos e mais comuns que podem levar à existência de vulnerabilidades sérias a nível do software. A lista é composta pelos seguintes 25 erros mais comuns:

- Injeção de SQL - Existe um risco ao utilizar o SQL de um atacante conseguir influenciar as *queries* de SQL de modo a poder roubar informação ou modificar a informação que está guardada numa base de dados SQL.
- Injeção de comandos no sistema operativo - acontece quando se permite invocar outro programa no sistema operativo, mas deixa-se que seja *inputs* não seguros de utilizadores a gerar o comando que vai ser corrido na execução desse programa.
- Buffer Overflow - Ocorre quando se tenta meter um *input* muito grande num espaço de memória que é mais pequeno que o mesmo, criando assim um erro de buffer overflow.
- Cross-site Scripting (XSS) - acontece quando não se tem cuidado com os *inputs* que são passados para as plataformas, os atacantes aproveitam estas falhas para conseguir injetar código malicioso de modo a que a plataforma comece a executar código que não é suposto.
- Falha de autênticação em funções críticas - é uma falha que ocorre quando se deixa exposto nas aplicações funcionalidades que são críticas para o funcionamento da mesma sem que haja um processo de autenticação.

- Falha de autorização - é uma falha que ocorre quando se permite não limita as permissões aos utilizadores, permitindo aos mesmos que possam fazer tudo na aplicação, caso um atacante se aproveite desta falhar é possível que o mesmo consiga aceder a toda a informação que desejar.

- Uso de credências *Hard-coded* - Ocorre quando se coloca uma password secreta ou uma chave criptográfica no programa, se a mesma for igual em todo o software esta pode tornar-se uma vulnerabilidade se for descoberta.

- Falta de encriptação em informação sensível - ocorre quando se envia informação que não está encriptada para locais que estão fora de controlo da aplicação, e caso essa informação não esteja encriptada se houver um atacante a escutar pode assim recolher a informação.

- Não restrição de upload de ficheiros de tipos perigosos - esta falha ocorre quando não existe controlo por parte dos ficheiros que são enviados, podendo assim expor a plataforma a sérios riscos.

- Dependência de *inputs* de fontes não fiáveis em decisões de segurança - é uma falha que acontece quando a plataforma não verifica a informação que o utilizador passa para a mesma, senão existir este controlo é possível que possam ocorrer problemas graves.

- Execução com privilégios que não são necessários - quando se dá privilégios mais durante mais tempo do que é requerido pode representar um risco elevado. Quando se utiliza privilégios extra, uma aplicação tem acesso a recursos que normalmente o utilizador da aplicação não consegue utilizar.

- Cross-Site Request Forgery (CSRF) - esta falha tem como grande alvo o utilizador da plataforma, na qual o atacante tenta enganar o utilizador utilizando pedidos da nossa aplicação.

- Cruzamento de caminhos - esta falha ocorre quando se usa *inputs* dos utilizadores para fazer do nome dos ficheiros, isto pode resultar em o caminho mostrar uma diretoria que não é suposta, relevando assim os ficheiros dentro da mesma.

- Download de código sem verificação da integridade - uma falha que ocorre quando se transfere código que não está verificado e se corre o mesmo podendo resultar em modificações à plataforma, deixando a mesma exposta.

- Autorização incorreta - ocorre quando o atacante consegue ultrapassar os controlos de permissões que são implementados pelos programadores, permitindo a que o mesmo consiga alterar certos aspetos da plataforma.

- Inclusão de funcionalidades de uma esfera de controlo não fiável - é uma falha que ocorre quando se importa algo para a plataforma correr, mas o que está a ser importado está comprometido, podendo assim o atacante modificar a aplicação.

- Designação de permissões incorretas para um recurso crítico - ocorre quando o atacante consegue chegar aos programas críticos ou ficheiros com permissões existentes que possam permitir tornar os recursos disponíveis ou capazes de serem modificados por todos.

- Uso de funções potencialmente perigosas - é uma falha que aparece quando um programador utiliza funcionalidades a qual eles não garantem que são seguras, deixando assim a plataforma exposta.

- Uso de um algoritmo quebrado ou arriscado - esta falha ocorre quando se utiliza algoritmos criptográficos que não são seguros na proteção de informação sensível.

- Cálculo incorreto do tamanho do buffer - este problema ocorre quando o programador tem de gerir a memória dos buffers, mas esta gestão não é feita de maneira correta, gerando assim uma falha de *buffer overflow*.

- Restrição imprópria para as quantidades excessivas de tentativas de autenticação - quando não é restringido o número de vezes que um utilizador pode tentar repetir, o atacante pode criar um programa que teste várias combinações até conseguir aceder, criando assim esta falha.

- Redireccionamento aberto - ocorre quando se permite ao utilizadores especificarem-se links que sejam para fora da aplicação se estes não forem verificados, pode levar a que um atacante se aproveite desta falha para poder realizar um ataque de phishing por exemplo.

- Formato das strings não controlado - esta falha ocorre quando não se controla as *strings* que são passadas pelos utilizadores, oferecendo assim a possibilidade aos atacantes de poder controlar o input e até por vezes correr o seu próprio código.

- Overflow de Integer - esta falha vai ocorrer quando a operação de aritmética que é executada excede o tamanho do *integer* que a vai guardar.

- Uso de hash de uma via sem *salt* - é uma falha que consiste em mesmo que seja feita uma hash para salvaguardar algo, mas se este hash for constante o atacante se o conseguir capturar apenas uma vez o recurso ao hash perde todo o seu valor.

2.4.4 Web Application Security Consortium (WASC) Threat Classification

A classificação de ameaças da WASC [31] foi criada graças a uma cooperação entre os membros do WASC com o propósito de clarificar e organizar as ameaças à segurança de um site. Este projeto foi criado para desenvolver e promover terminologias para servirem como base para descrever os problemas que são encontrados.

A classificação está dividida em duas, em ataques e em fraquezas.

Ataques

- Abuso de funcionalidades - Consiste em uma técnica de ataque que usa as funcionalidades e capacidades do próprio site para se atacar a sim mesmo e a outros.

- Força bruta - é um método utilizado para tentar determinar um valor desconhecido, recorrendo a um processo automático para tentar várias possibilidades.

- Buffer Overfow - uma falha que ocorre quando é escrita demasiada informação para um bloco de memória, ou buffer, que este consegue suportar.

- Spoofing de conteúdo - permite ao atacante injetar um *payload* malicioso que mais tarde vai ser interpretado como conteúdo legitimo da aplicação.

- Previsão de credênciais/sessão - consiste num método de se fazer passar ou de roubar um web site de um utilizador.

- Cross-site scripting (XSS)- é uma técnica de ataque que consiste em fazer "ecoar"o código que pertence ao atacante para a instância no browser do utilizador.

- Cross-site request forgery - é um ataque que força a vitima a enviar um pedido HTTP para o alvo sem os mesmos terem conhecimento ou a intenção de o fazer, de modo a executarem ações no lugar da vitima do ataque.

- Negação de serviço - consiste num ataque com o intuito de impedir que um web site consiga funcionar corretamente.

- Fingerprinting - um dos métodos mais comuns que consiste em o atacante verificar a presença que existe online do utilizador e examinar o máximo de informação que é possível acerca do mesmo.

- Formato das strings - este ataque é capaz de alterar o fluxo de uma aplicação recorrendo às funcionalidades da biblioteca de formatação de *strings* para aceder a outros espaços da memória.

- Contrabando da resposta Hypertext Transfer Protocol (HTTP) - é uma técnica que consiste em "contrabandear"2 respostas HTTP do servidor para o cliente, através de um dispositivo de HTTP intermédio que espera uma reação do servidor.

- Separação da resposta HTTP - este ataque tem como base a habilidade do atacante enviar um pedido de HTTP que força o servidor a ter de formar um *output stream*, que é depois interpretado pelo alvo do ataque como duas respostas em vez de ser apenas uma resposta, que é caso normal.

- Contrabando do pedido HTTP - é um ataque que abusa das discrepâncias no *parsing* de um não *Request for Comments* (RFC) compatível com os pedidos de HTTP entre dois dispositivos para "contrabandear"o pedido para o segundo dispositivo "através"do primeiro dispositivo.

- Separação do pedido HTTP - este ataque permite que o se force o browser a enviar pedidos de HTTP, fazendo ataques de XSS e "envenenando"a cache do browser.

- Overflow de Integer - consiste numa condição que ocorre quando o resultado de uma operação de aritmética, excede o máximo de tamanho de tipo de *integer* que é utilizado para o guardar.

- Injeção Light Directory Access Protocol (LDAP) - é uma técnica que ataque que explora web sites que criam declarações LDAP através de inputs de utilizadores.

- Injeção via comando de mail - consiste numa técnica utilizada para explorar os servidores de mail e aplicações de webmail que constroem declarações *Simple Mail Transfer Protocol* (SMTP)/*Internet Message Access Protocol* (IMAP) para inputs do utilizador que não são "sanetizados"corretamente.

- Injeção via byte a null - é uma técnica de exploração ativa usada para ignorar os filtros de verificação de sanitização da infraestrutura web, adicionando-lhe um URL codificado com caracteres de bytes a null, para a informação fornecida pelo utilizador.

- Comandar o sistema operativo (OS) - é um ataque que consiste em executar comandos do sistema operativo que não estão autorizados.

- Cruzamento de caminhos - é um ataque que permite aos atacantes aceder a ficheiros, diretorias e comandos que residem potencialmente fora da diretoria de root do documento web.

- Local de recurso previsível - é uma técnica que consiste em descobrir conteúdo e funcionalidades escondidas de um web site.

- Inclusão de ficheiros remotos (RFI) - é usado para explorar os mecanismos de inclusão dinâmica de ficheiros nas aplicações web.

- Routing Detour - é um tipo de ataque de *man in the middle* onde os intermediários podem ser injetados ou "sequestrados"para desviar mensagens com conteúdo sensível para outra localização.

- Fixação da sessão - é um ataque que força o identificador de sessão do utilizador para um valor explícito.

- Abuso do array de Simple Object Access Protocol (SOAP) - este ataque é direcionado aos arrays XML SOAP, caso o mesmo seja modificado pode forçar a que um servidor SOAP crie um array grande na memória da máquina, podendo originar assim um ataque de negação de serviço usando estes arrays.

- Injeção *Server Side Includes* (SSI) - consiste numa técnica de *exploit* do lado do servidor que permite ao atacante enviar código para a aplicação que irá mais tarde ser executado a nível local por parte do servidor web.

- Injeção SQL - consiste numa técnica utilizada para explorar as aplicações que criam declarações SQL através dos *inputs* de utilizadores.

- Abuso de redireccionador de URL - é uma técnica que consiste em abusar da funcionalidade de redireccionamento que é muito utilizada por websites, este não representa diretamente uma falha de segurança mas pode ser utilizado por atacantes de modo a fazer a vítima pensar que está a navegar para a página correta.

- Injeção de XPath (XML Path) - é uma técnica usada para explorar aplicações que construam queries XPath através de *inputs* dos utilizadores para navegar documentos de eXtensible Markup Language (XML).
- Explosão de atributo XML - consiste num ataque de negação de serviço contra analisadores de XML.
- XML de entidades externas - esta técnica aproveita-se da funcionalidade do XML de construir documentos dinamicamente ao mesmo tempo que os processa.
- Expansão de entidades XML - este ataque explora a capacidade do XML de definir o tipo de documentos, que permite a criação de macros customizáveis, de chamadas a entidades que podem ser usadas através do documento.
- Injeção de XML - consiste num ataque que visa a manipular ou comprometer a lógica de uma aplicação ou serviço XML.
- Injeção de XQuery - semelhante á injeção SQL mas com vista a atacar as *queries* de XML, chamadas de XQueries.

Fraquezas

- Má configuração da aplicação - consiste em explorar falhas que existam na configuração das aplicações web.
- Indexação de diretorias - este problema acontece quando não é configurado um ficheiro para servir de root para a diretoria do website, o que por vezes pode levar a acontecer quando se tenta abrir a página, revelar todo o que está presente nessa diretoria.
- Permissões impróprias para a diretoria - este é um problema que ocorre quando são dadas permissões erradas para os ficheiros, pasta e ligações simbólicas. Sendo este um risco para a confidencialidade, integridade e disponibilidade da aplicação web.

- Inputs lidados impropriamente - este é uma das fraquezas mais comuns nas aplicações atualmente, a mesma ocorre quando não existe nenhum mecanismo de validação, filtração, sanitização, ao codificar ou de descodificar parte da informação que é passada pelos utilizadores. A má gestão dos *inputs* podem levar a vulnerabilidades críticas no sistema e aplicações.

- Outputs lidados impropriamente - esta falha refere-se a como uma aplicação gera a informação de *output*, se a mesma não for bem gerida isto pode levar a vulnerabilidades e execução de ações que não estavam planeadas por parte da aplicação.

- Vazamento de informação - esta falha refere à existência de uma falha na plataforma que revela informação sensível, podendo ser informação da plataforma ou informação relativa ao utilizador.

- Indexação insegura - esta falha consiste num risco à confidencialidade da informação do website, isto acontece quando se indexa a processos ficheiros aos quais não és suposto existir acesso, podendo levar a que informação seja exposta.

- Anti automação insuficiente - esta falha ocorre quando a aplicação permite a um atacante automatizar um processo que é suposto ser realizado manualmente.

- Insuficiente autenticação - é uma falha que ocorre quando o website permite ao atacante aceder a contéudo sensível ou funcionalidades sem se autenticar corretamente.

- Autorização insuficiente - este problema vai ocorrer quando a aplicação não realiza verificações adequadas a nível das autorizações para garantir que o utilizador não pode executar funções ou aceder a informações à qual não tem permissões.

- Recuperação de password insuficiente - é uma falha que ocorre quando o website permite a um atacante obter, modificar ou recuperar a password de outro utilizador.

- Processo de validação insuficiente - ocorre quando uma aplicação web não é capaz de prevenir um atacante de interferir com o fluxo da aplicação ou com a lógica de negócio da aplicação.

- Funcionalidade de expirar a sessão é insuficiente - ocorre quando a aplicação web permite ao atacante reutilizar um credenciais de uma sessão antiga ou um identificador antigo de sessão para conseguir ter permissões.

- Insuficiente proteção na camada de transporte - esta falha permite que as informações estejam expostas, criando assim uma oportunidade para comprometer a segurança da aplicação e/ou roubar informação sensível dos utilizadores.

- Má configuração do servidor - esta fraqueza ocorre muitas vezes devido a manter-se, no servidores web, os ficheiros de origem e as configurações muitas vezes desnecessários, e isto abre a porta a potenciais ataques.

Capítulo 3

Análise, Desenho e Implementação

Neste capítulo vai ser abordado toda a parte que corresponde ao desenvolvimento da plataforma, desde o levantamento de requisitos, o desenho da mesma, até à apresentação do estado final da mesma.

3.1 Requisitos da Plataforma

Esta secção vão estar apresentados os requisitos que foram identificados para o desenvolvimento da plataforma, os mesmos são os seguintes:

- Permitir o registo de novos utilizadores.
- Permitir que os utilizadores possam fazer a gestão do seu perfil, podendo alterar a sua palavra-passe, bem como apagar a sua conta.
- Permitir ao utilizador registar auditorias de segurança.
- Permitir que o utilizador consiga registar por cada auditoria as vulnerabilidades que correspondem à mesma.
- Permitir que o utilizador a possibilidade de fazer o upload de ficheiros de uma aplicação de *scanner* de vulnerabilidades automáticas.

- Fornecer ao utilizador *dashboards* que são gerados tendo em conta as vulnerabilidades reportadas, para assim fornecer mais informação aos utilizadores.
- Permitir a administração da plataforma por parte de utilizadores que serão os seus administradores.
- Permitir a geração de relatórios personalizáveis que contenham toda a informação relativa à auditoria em que foram pedidos.

3.2 Arquitetura da plataforma

A plataforma vai possuir uma base de dados relacional onde irá ser guardada toda a informação sobre os utilizadores, as vulnerabilidades, as auditorias e as empresas para as quais são realizadas as auditorias. Esta base de dados vai estar em *back-end*, a plataforma para comunicar com a mesma vai ser via *querys* em SQL que o PHP executa para assim se conseguir gerir a informação, bem como apresentar a informação que está guardada.

Para tal vão existir ficheiros de PHP que gerem os pedidos que os utilizadores fazem ao servidor para assim este ir buscar a informação à base de dados ou mexer com a informação tendo em conta os pedidos. Sendo que o *front-end* que é apresentado ao utilizador está todo em HTML, CSS e JavaScript, sendo que assim só seja necessário o utilizador usar um browser para utilizar a plataforma sem que seja necessário instalar mais alguma aplicação, como se pode verificar na figura 3.1.

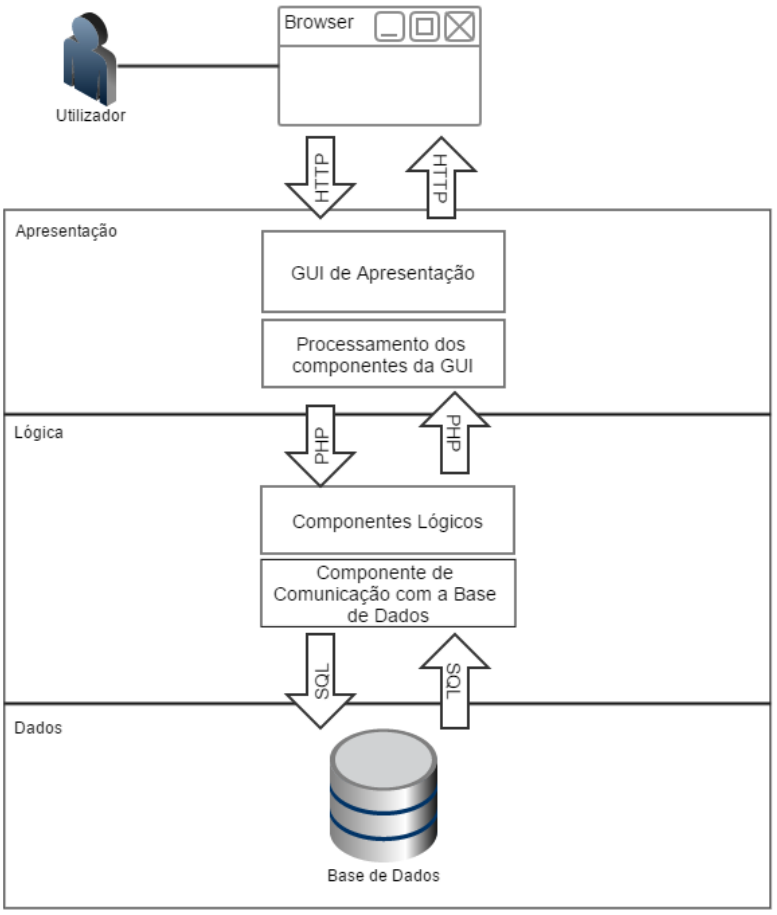


FIGURA 3.1: Arquitetura desenhada para a pltaforma

Como observamos na figura 3.1, o utilizador interage com a GUI que lhe é apresentada no seu browser, e cada ação efetuada por ele vai ter de passar pelas várias partes que que foram apresentadas anteriormente na figura 3.1, sendo que a parte de apresentação são as páginas, como por exemplo o login.php, e vamos ter um gestor para esta página que para o caso do login.php vai ser o handleLogin.php. O gestor vai tratar de todos os pedidos que venham da página de modo a que a plataforma se comporte da maneira correta e permita fazer os pedidos à base de dados, que são realizados chamando um dos ficheiros que possuem essas funcionalidades, que neste caso será o functionsLogin.php. Podemos observar este tipo de interação na figura 3.2, onde observamos como a plataforma gere o processo de login.

Tendo em conta a arquitetura que foi implementada, as páginas em que o utilizador interage vão ter um gestor, em PHP, que está à espera dos pedidos que são enviados pela página para assim ele poder realizar a ação desejada. Isto pode ser observado na figura 3.2 que mostra a interação da plataforma ao efetuar o processo de login.

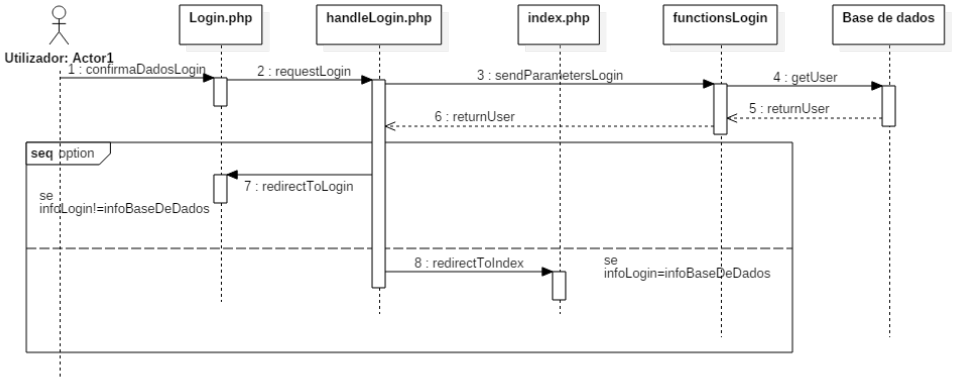


FIGURA 3.2: Diagrama de sequência que ilustra a ação de login

3.3 Base de Dados

A plataforma foi desenvolvida para ser instalada num servidor permitindo assim aos utilizadores usarem apenas o seu browser para lhe aceder. Para ser possível guardar os dados que os utilizadores passam vai ser necessário a existência de uma base de dados, a mesma vai ter a forma da seguinte figura 3.3.

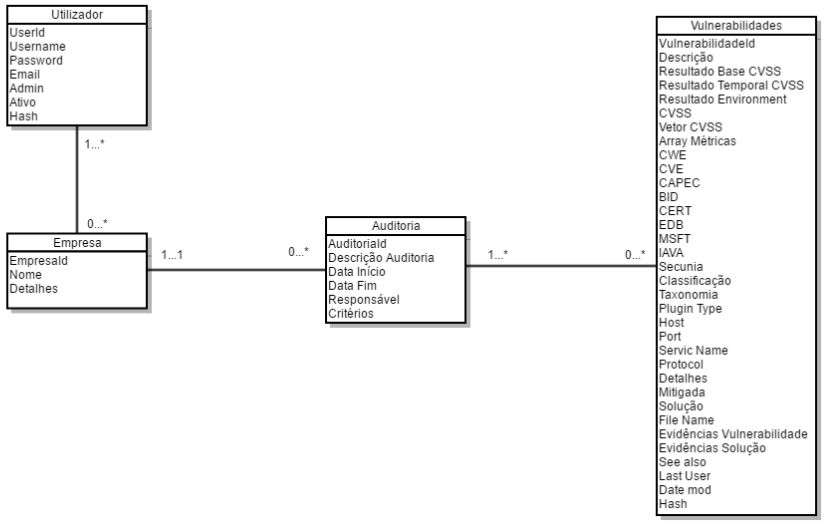


FIGURA 3.3: Desenho da base de dados relativo à plataforma

Classe Users

Nesta classe vai ser guardada toda a informação correspondente aos utilizadores, incluindo as suas credenciais de acesso à plataforma.

Detalhes sobre os atributos da classe Users:

- UserId - identificador que é atribuído ao utilizador logo após o seu registo gerado automaticamente pela base de dados depois de este ser inserido na tabela.
- Username - é o nome que o utilizador escolhe para si quando se está a registar, o mesmo é único para cada utilizador que se vai registar.
- Password - corresponde a uma *string* que o utilizador ao registar-se vai definir para efetuar o seu login em conjunto com o Username.
- Email - vai ser o endereço de email associado ao mesmo que também vai ser único na plataforma e para assim quando existir alterações na password ou mesmo na altura do registo para efetuar a ativação da conta.

- Admin - consiste numa *flag* para identificar se o utilizador tem permissões de administrador ou não.
- Ativo - é também uma *flag* que diz respeito a dizer se o utilizador tem a conta ativa ou não.
- Hash - é um campo que serve como forma de determinar se o link que o utilizador recebeu corresponde ao mesmo,

Classe Empresa

Nesta classe vais estar presente a informação que serve para definir a empresa bem como alguns detalhes que podem ser úteis para os utilizadores.

Detalhes sobre os atributos da classe Empresa:

- EmpresaId - identificador que vai ser atribuído pela base de dados à empresa quando é registada pela primeira vez, este é único.
- Nome - corresponde ao nome que vai ser atribuído à empresa quando a mesma é registada.
- Detalhes - este campo serve para indicar informação extra acerca da empresa.

Classe Auditoria

Nesta classe vamos ter os atributos que são necessários para se ter a informação que é necessária para ser criada uma auditoria, desde das datas em que se inicia, bem como a data de fim, os critérios da auditoria e o responsável pela mesma.

Detalhes sobre os atributos da classe Auditoria:

- AuditoriaId - corresponde ao identificador que é gerado pela base de dados e é atribuído à auditoria, quando é efetuado o registo da auditoria.
- Descrição Auditoria - este atributo tem como propósito indicar qual é a descrição que é dada à auditoria.

- Data Início - indica a data em que foi iniciada a auditoria.
- Data Fim - indica a data em que a auditoria foi terminada.
- Critérios - este campo tem como objetivo descrever os critérios usados para a execução da auditoria.

Classe Vulnerabilidades

Esta é a tabela que vai ter mais atributos, pois para se ter o máximo de informação possível acerca das vulnerabilidades, de modo a que seja possível qualquer utilizador conseguir retirar informação que possa ser pertinente para esta ou futuras vulnerabilidades.

Detalhes sobre os atributos da classe Vulnerabilidade:

- VulnerabilidadeId - corresponde ao identificador que a base de dados gera quando é registada uma vulnerabilidade na mesma.
- Descrição - neste campo serve para guardar a descrição que o utilizador
- Resultado Base do CVSS - corresponde ao resultado do *base score* do CVSS.
- Resultado Temporal do CVSS - corresponde ao resultado do *temporal score* do CVSS.
- Resultado Environment - corresponde ao resultado do *environment score* do CVSS.
- Vetor CVSS - neste atributo é guardado o vetor que resulta do cálculo do CVSS
- Array Métricas - neste campo vão ser guardadas as métricas do CVSS que são inseridas pelo o utilizador após o utilizador ter confirmado o resultado final do CVSS.
- CWE - atributo que vai guardar os identificadores do CWE relativos à vulnerabilidade.

- CVE - atributo que vai guardar os identificadores do CWE relativos à vulnerabilidade.
- CAPEC - atributo que vai guardar os identificadores do CVE relativos à vulnerabilidade.
- BID - atributo que vai guardar os identificadores do BID relativos à vulnerabilidade.
- CERT - atributo que vai guardar os identificadores do CERT relativos à vulnerabilidade.
- EDB - atributo que vai guardar os identificadores do EDB relativos à vulnerabilidade.
- MSFT - atributo que vai guardar os identificadores do MSFT relativos à vulnerabilidade.
- IAVA - atributo que vai guardar os identificadores do IAVA relativos à vulnerabilidade.
- Secunia - atributo que vai guardar os identificadores do Secunia relativos à vulnerabilidade.
- Classificação - este campo vai guardar a classificação que o utilizador vai atribuir à vulnerabilidade, ela pode ser OWASP Top Ten, OWASP Top Ten Mobile ou então não definida.
- Taxonomia - dependedo da classificação que foi atríbuida pelo o utilizador este campo pode ter os valores relevantes à taxonomia do OWASP Top Ten, do OWASP Top Ten Mobile ou então não ter nenhum valor caso a classificação seja não definida.
- Plugin Type - neste campo é guardado informação relativa ao tipo de *plugin* que está associado à vulnerabilidade.
- Host - corresponde ao *host* em que a vulnerabilidade foi descoberta.

- Port - indica o porto onde existe a vulnerabilidade.
- Service Name - campo que guarda informação acerca do serviço correspondente à vulnerabilidade.
- Protocol - campo que vai ter a informação relativamente ao protocolo que está associado à vulnerabilidade.
- Detalhes - neste campo vai estar todos os detalhes que o utilizador inserir à cerca da vulnerabilidade.
- Mitigada - este campo serve para guardar a flag que indica se a vulnerabilidade está ou não mitigada.
- Solução - campo no qual vai ser inserida a solução, caso essa existas, da vulnerabilidade.
- File Name - corresponde ao nome do ficheiro onde a vulnerabilidade existe.
- Evidências Vulnerabilidade - corresponde ao nome de um ou mais ficheiros de imagem que demonstram a existência da vulnerabilidade.
- Evidências Solução - corresponde ao nome de um ou mais ficheiros de imagem que demonstram que foi encontrada uma solução para a vulnerabilidade.
- See Also - campo em que vai ficar guardada mais informação que o utilizador ache pertinente em relação à vulnerabilidade.
- Last User - indica o último utilizador que modificou a vulnerabilidade.
- Date Mod - indica a data em que foi feita a última modificação à vulnerabilidade.
- Hash - corresponde a uma combinação única que é gerada inicialmente para ficar associada à vulnerabilidade tornando a mesma única.

3.4 Implementação

Nesta secção vai ser apresentado tudo o que foi utilizado para a criação da plataforma, das ferramentas que foram utilizadas, bem como uma explicação de como a mesma funciona.

3.4.1 Tecnologias Utilizadas

Nesta secção vão ser apresentadas todas as tecnologias que foram utilizadas para o desenvolvimento da plataforma proposta.

3.4.1.1 Hypertext Preprocessor (PHP)

O PHP [32] [33] é uma das linguagens na web mais utilizadas, esta é uma linguagem de programação *server-side* e *open-source*, em que consiste em utilizar scripts do lado do servidor que vão produzir uma resposta específica tendo em conta cada pedido que é feito pelo utilizador. Uma das grandes vantagens do PHP é a capacidade de ter o HTML embutido no código, isto faz com que seja possível para os utilizadores misturar ambas as linguagens de maneira a fazer que o HTML apresente os resultados das ações que são efetuados pelo PHP. Outra vantagem que possui como foi referido anteriormente é desta se tratar de uma linguagem *server-side*, isto significa que o código de PHP não aparece nunca no *front-end* devido ao código ser executado no servidor. A razão que levou ao PHP ser escolhido para o desenvolvimento da plataforma, para além das vantagens referidas anteriormente, são também a portabilidade que possui, bem como a sua estabilidade e integração com quase todas as base de dados que existem.

3.4.1.2 HTML e CSS

O HTML [34] é a linguagem que é usada para se desenvolver as páginas de web, usando *tags* para descrever os elementos que o utilizador quer que sejam apresentados nas páginas. O hipertexto desta linguagem é o que permite a apresentação de conteúdos, bem como a ligação entre outros conteúdos da web. O CSS [35] que é usado em conjunto com o HTML, esta é usada com o objetivo de descrever como os elementos vão estar expostos nas páginas, dando às páginas um visual mais apelativo.

3.4.1.3 SQL

A linguagem SQL [36] é usada para gerir informação que está dentro dos sistemas de base de dados relacionais, o uso desta linguagem nesta plataforma deve-se ao facto da base de dados ser relacional sendo que este vai ser o meio de comunicação entre a plataforma e a base de dados, ou seja, o PHP vai executar *queries* cada vez que seja necessário inserir ou pedir informação, apagar e atualizar.

3.4.1.4 JavaScript

O JavaScript [37] [38] é uma linguagem orientada por objetos que foi desenvolvida com o intuito de estender as páginas web com código que fosse executado do lado do cliente, e que atualmente tem vindo a aumentar os meios onde é utilizada, como por exemplo em aplicações para os escritórios e ambientes de desenvolvimento. O uso do JavaScript serve como complemento ao HTML, pois oferece aos programadores a oportunidade de criar funcionalidades para nas suas páginas de web, desde servir para fazer validações, alterar o CSS da página e até alterar conteúdos do HTML e os seus atributos também.

3.4.2 Ferramentas

Dentro desta secção vão ser mencionadas todas as ferramentas que foram utilizadas para o desenvolvimento da plataforma.

3.4.2.1 Netbeans

O Netbeans [39] é um IDE para desenvolver aplicações ou programar, este permite que se programe em várias linguagens, desde PHP, HTML, Javascript, Java, C++ e outros. Este IDE fornece a indentação em todas as linguagens para facilitar a compreensão do código, bem como a capacidade de sugerir funções para ajudar o *developer* a programar, devido a ser baseado em Java este oferece também a vantagem de ser portável e assim o código que aqui for desenvolvido pode ser executado em outros sistemas operativos, tendo em conta que por vezes é necessário realizar algumas alterações, como por exemplo, alterar a maneira como é escrita o caminho para diretorias, visto que difere de Linux para Windows.

3.4.2.2 Multiplataforma, Apache, MariaDB, PHP e Pearl (XAMPP)

O XAMPP [40] é uma distribuição do Apache muito simples de se utilizar que tem como propósito criar um servidor local, para que os *developers* possam testar as suas plataformas ou até websites sem a necessidade de recorrer a um servidor online. A maior vantagem de se usar esta ferramenta, é a mesma oferecer um ambiente que é similar a muitos servidores de alojamento online, que ajuda imenso fornecendo assim uma base de teste à plataforma antes que se passe para o servidor e fique disponível para as pessoas.

3.4.2.3 Highcharts

O Highcharts [41] consiste numa API desenvolvida em JavaScript *open-source* que desenha gráficos passando os parâmetros desejados. Esta API oferece vários tipos

de gráficos aos utilizadores bem como a capacidade de os personalizar, oferecendo assim um grande leque de opções ao utilizador, pois o próprio programador se quiser pode fazer alterações ao Highcharts caso seja necessário e não está limitado apenas ao que é fornecido.

3.4.2.4 CKEditor

O CKEditor [42] é uma API de texto HTML *open-source* que fornece as propriedades que um processador de texto normal oferece, dando ainda a oportunidade ao programador de escolher as funcionalidades que deseja que estejam presentes para os campos de texto que quer definir na plataforma. Esta API é muito útil pois em vez de se ter campos de texto onde apenas é possível escrever sem ter opção de aumentar o tamanho da letra bem como outras funcionalidades.

3.4.2.5 Securimage

O Securimage [43] é uma API *open-source* que fornece um script de CAPTCHA de maneira às plataformas que o usem possam certificar-se que o utilizador não é um bot. Isto serve como um meio de segurança para as plataformas pois assim evitam que exista spam e como foi dito antes serve de como medida de prevenção contra bots.

3.4.2.6 PHPMailer

O PHPMailer [44] é uma biblioteca para o PHP para fazer o envio de emails. O PHP por si só já oferece uma função mail() para tratar do envio de emails, mas esta é bastante difícil de configurar e não suporta nenhuma assistência no que corresponde ao uso de funcionalidades que o HTML oferece e o envio de anexos, bem como ainda oferecer a integração do protocolo de SMTP e autenticação via SSL e TLS.

3.4.2.7 CVSS 2.0 Calculator

O CVSS 2.0 Calculator [45] como o próprio nome indica é uma calculadora *open-source* que foi desenvolvida em JavaScript para efetuar o cálculo do CVSS 2.0 respeitando todos os parâmetros que estão enunciados em [6]. Esta calculadora oferece ainda a descrição das várias métricas, a capacidade de gerar um link com o vetor que foi calculado e a capacidade de o utilizador passar o vetor caso queira calcular o resultado tendo o mesmo.

3.4.2.8 Bootstrap

O Bootstrap [46] é uma *framework* que usa HTML, CSS e JavaScript para suportar o desenvolvimento de aplicações para a web. Esta é uma ferramenta muito útil pois dá-nos um *template* de base para o desenvolvimento das plataformas e este é suportado para todas as plataformas, sejam telemóveis, tablets ou computadores.

3.4.3 Segurança

Um dos aspetos mais importantes em qualquer plataforma é a sua segurança, e neste caso tendo em conta a sua arquitetura uma das vulnerabilidades que a afeta mais é a de SQL Injection, que consiste em um utilizador malicioso conseguir injetar código SQL de maneira a que perturbe o funcionamento da plataforma, para impedir que isto aconteça vai ser utilizado um mecanismo que se chama de Prepared Statements.

As Prepared Statements consistem em *templates* que são preparados com a *query* que desejamos fazer à base de dados na qual deixamos os campos que queremos preencher com pontos de interrogação para que depois se associe as variáveis que recebemos aos campos desejados da *query*. Um exemplo de uma destas prepeared statements está na seguinte função em baixo que consiste em ir buscar informação acerca de uma auditoria à base de dados passando o identificador da mesma.

```
function getAuditInfo($id){
     require '../database/database.php';
    $sqlstmtCheck = $con->prepare("SELECT auditoriaid,
    vulnerabilidadeid,scanid, descricaoAudit, dataInicio, dataFim,
    responsavel, criterios FROM auditoria WHERE auditoriaid=?");
    $sqlstmtCheck->bind_param("s", $id);
    $sqlstmtCheck->execute();
    $sqlstmtCheck->bind_result($auditoriaid,$vulnerabilidadeid,
    $scanid,$descricaoAudit,$dataInicio,$dataFim,$responsavel,
    $criterios);
    $sqlstmtCheck->store_result();
     while ($sqlstmtCheck->fetch()) {
         $arrayInfoAudit=array();
         $arrayInfoAudit["auditoriaid"]=$auditoriaid;
         $arrayInfoAudit["descricao"]=$descricaoAudit;
         $arrayInfoAudit["arrayVulnerabilidadeId"]= unserialize(
    $vulnerabilidadeid);
         $arrayInfoAudit["arrayScanId"]= unserialize($scanid);
         $arrayInfoAudit["dataInicio"]=$dataInicio;
         $arrayInfoAudit["dataFim"]=$dataFim;
         $arrayInfoAudit["responsavel"]=$responsavel;
         $arrayInfoAudit["criterios"]=$criterios;
     }
     $sqlstmtCheck->close();
     return $arrayInfoAudit;
}
```

Lista de 3.1: Exemplo de uma Prepared Statement

Outro ponto que se deve ter cuidado é com os *inputs* que são passados pelos utilizadores, para tal é necessário que que os inputs dos utilizadores sejam "limpos"de maneira a que não causem problemas à plataforma. Para tal vai ser usada a seguinte função.

```
function cleanVariables($variable) {
    $variable = stripslashes($variable);
    $variable = strip_tags($variable);
    return $variable;
```

```
5 }
```

Lista de 3.2: Função utilizada para limpar os inputs

Nesta função vão estar duas funcionalidades que são oferecidas pelo PHP para fazer a filtragem das varáveis que recebem, em que o striplashes tem como propósito remover as barras invertidas e o strip_tags serve para remover as *tags* de HTML que possam vir nas variáveis.

Para evitar que a plataforma sofra ataques de Cross-Site Request Forgery (CSRF), este ataque consiste em forçar o utilizador a executar ações que o atacante deseja, após o mesmo se ter autenticado com sucesso na plataforma. De modo a que isto não aconteça todas as páginas vai ser gerado um token que fica na sessão e outro que fica associado ao formulário que o utilizador vai executar, sendo que a ação do utilizador só vai ser efetuada após se verificar se ambos os *tokens* coincidirem. Para isso vai existir a classe tokenCSRF.php que vai ser usada para gerar os *tokens*, bem como os verificar apresentada em baixo.

```
1  class Token{
2      public static function generate(){
3          return $_SESSION['token']= base64_encode(
       openssl_random_pseudo_bytes(32));
4      }
5      public static function check($token){
6          if(isset($_SESSION['token']) && $token === $_SESSION['token'
       ]){
7              unset($_SESSION['token']);
8              return true;
9          }
10         return false;
11     }
12 }
```

Lista de 3.3: Classe usada para gerar e fazer a verificação dos tokens

Quando o utilizador entra numa página é logo executada a função generate() na qual vai ser criado o *token* que fica na sessão e na página onde o utilizador está,

quando o utilizador executa uma ação os dois *tokens* serão comparados usando a função check(), caso ambos os *tokens* sejam iguais a ação vai ser realizada com sucesso, caso contrário não irá acontecer nada.

3.5 Funcionamento da plataforma

Nesta seção vai ser abordado todas as páginas com que os utilizadores interagem na plataforma, de modo a que seja possível verificar as funcionalidades de cada uma individualmente e as ações que os utilizadores podem fazer nas mesmas.

3.5.1 Login

A primeira página que vai ser apresentada ao utilizador vai ser a de login como se pode ver na figura 3.4.

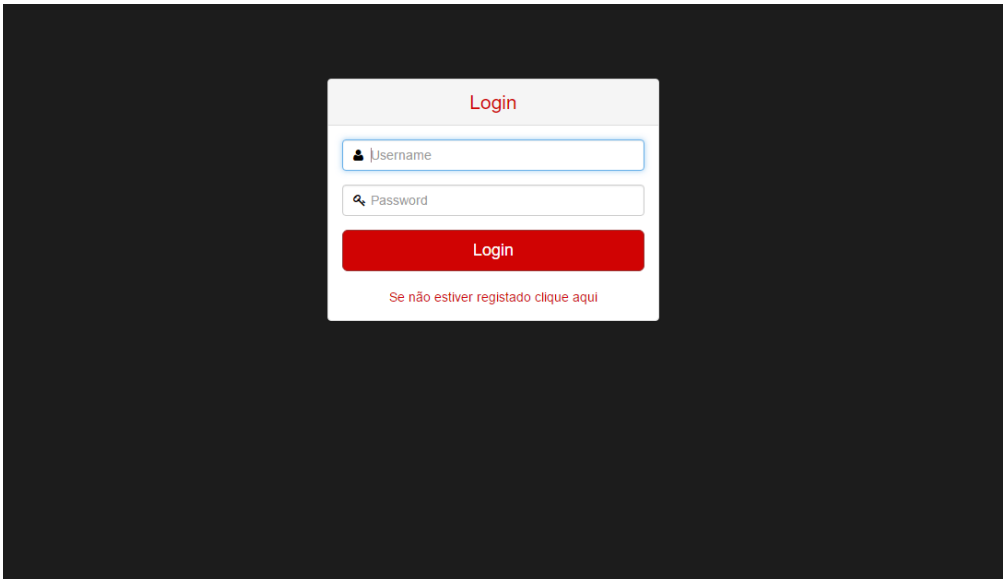


FIGURA 3.4: Página de login da plataforma

3.5.2 Registar

Antes de ser possível o utilizador ter acesso à plataforma, vai ser necessário primeiro começar por se registar na plataforma, fornecendo o seu email, escolhendo um username e password para ele e depois ainda tem inserir um código para verificar que o utilizador não é um bot, como se pode observar na figura 3.5.

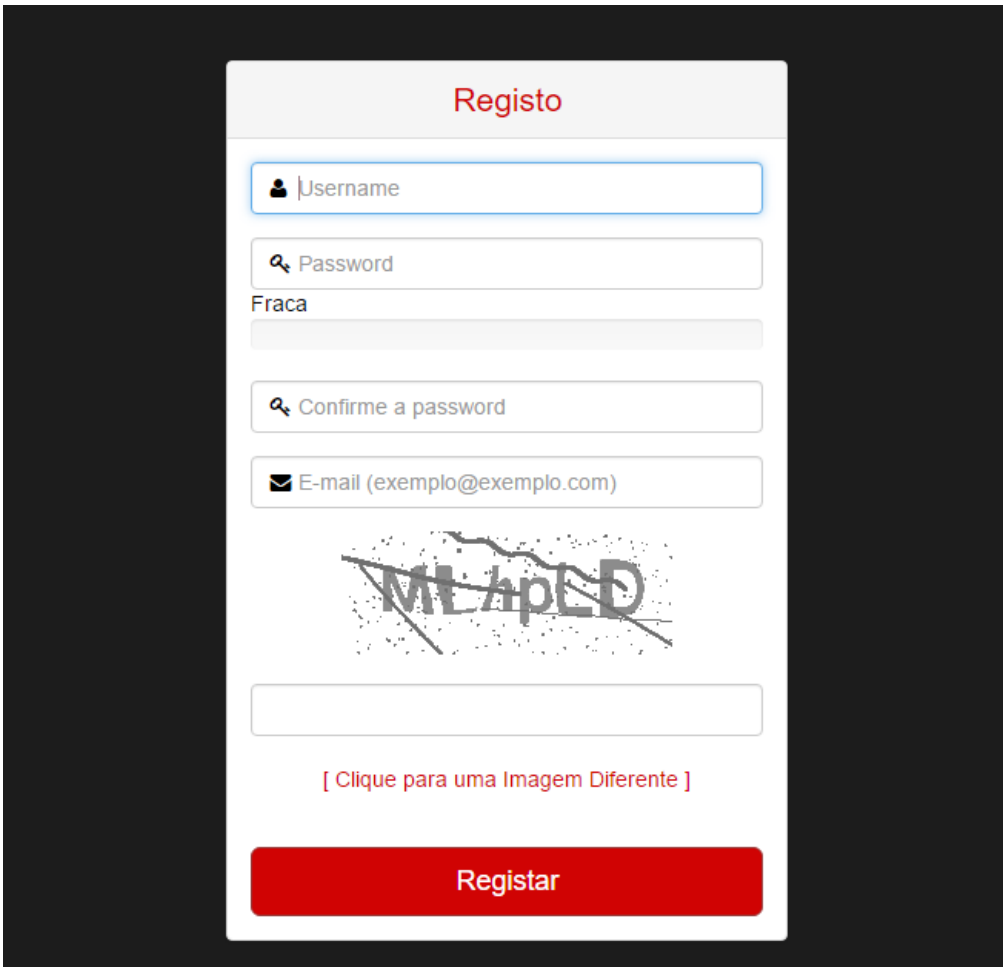


FIGURA 3.5: Página de registo da plataforma

3.5.3 Página inicial

Depois de ter sido registado com sucesso irá receber um email com um link para ativar a sua conta. Após a conta ter sido ativada com sucesso, já vai poder iniciar a sua sessão para poder assim aceder à plataforma. Após o login ter sido efetuado

com sucesso o utilizador vai passar para a página inicial da plataforma na qual irá existir um gráfico que mostra todas as vulnerabilidades que foram reportadas tendo em conta a empresa que está selecionada, como podemos observar na figura 3.6.

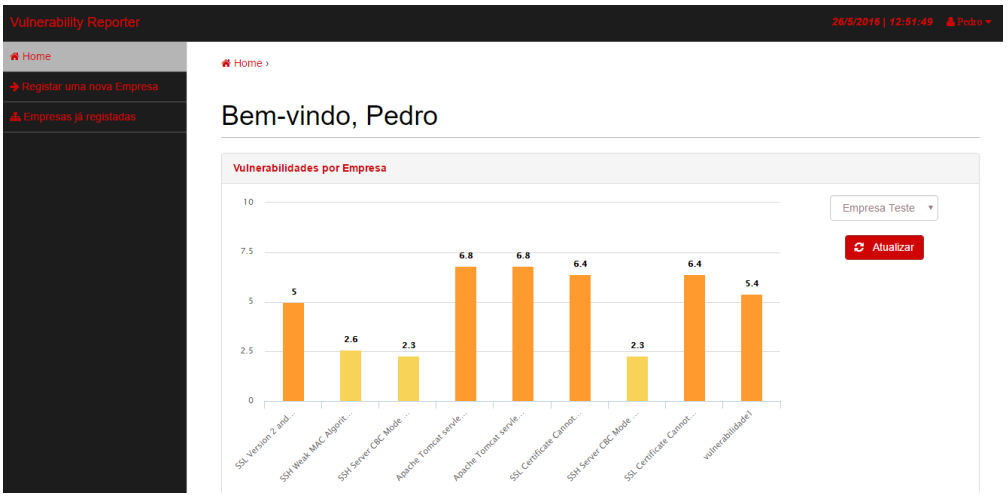


FIGURA 3.6: Página inicial da plataforma

Como se pode verificar na figura 3.6 esta página, no canto superior direito existe um menu onde está exposto o nome do utilizador. Ao clicar-se no nome vão surgir as opções de Logout, de ir para a página do perfil do utilizador e caso o utilizador possua permissões de administrador, conseguir aceder à página dos administradores da plataforma, este menu vai estar sempre disponível em todas as páginas da plataforma.

3.5.4 Perfil

No perfil que está representado pela figura 3.7, o utilizador tem as opções de conseguir alterar a sua palavra passe e de apagar a sua conta na plataforma, se escolher alterar a palavra passe o utilizador irá receber um email a confirma que foi alterada com sucesso.

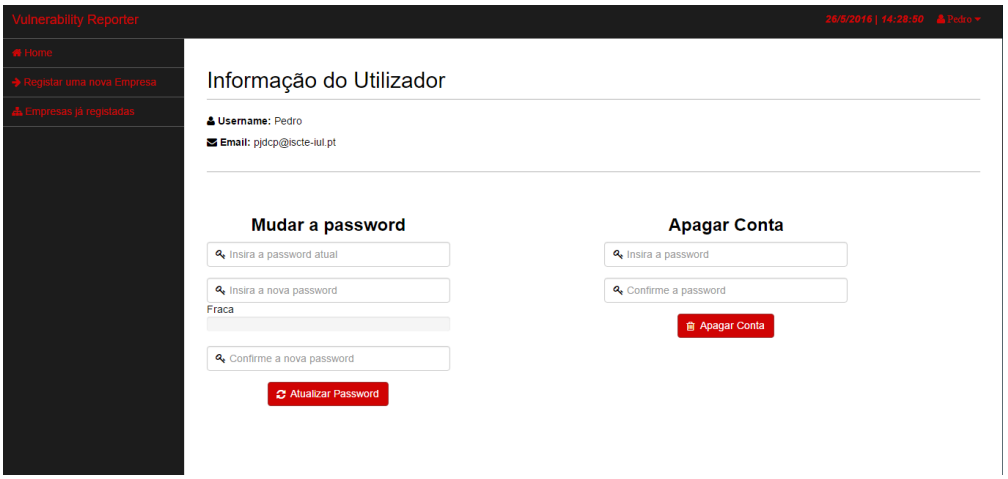


FIGURA 3.7: Página de perfil dos utilizadores da plataforma

3.5.5 Página de administrador

Na página de administrador, que está representada na figura 3.8, o utilizador vai conseguir tornar outros utilizadores administradores também, bem como poder inserir utilizadores em empresas que já existam de modo a que consigam trabalhar nas mesmas, e de poder apagar qualquer empresa que exista na plataforma.

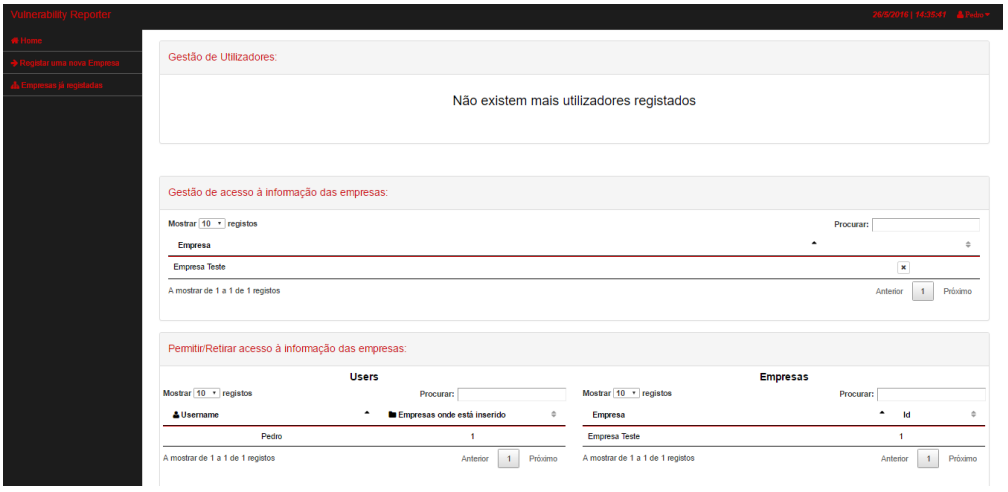


FIGURA 3.8: Página dos administradores da plataforma

3.5.6 Página de listagem das empresas

No lado esquerdo existe a opção de voltar a este ecrã, a opção de criar uma nova empresa na plataforma e a opção de ver as empresas que estão registadas e o utilizador pode aceder, como se pode observar na figura 3.9.

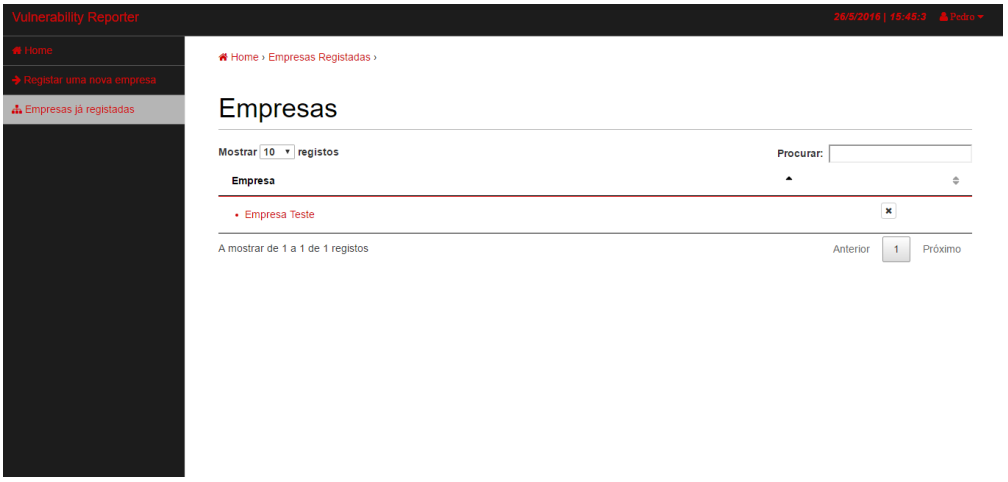


FIGURA 3.9: Página que lista as empresas com que o utilizador pode interagir

3.5.7 Página de registo das empresas

Seguindo para a página de registo de empresas que está na figura 3.10, no qual aparecem os campos com o nome da empresa e de detalhes que o utilizador ache relevante em relação à mesma, depois de o utilizador validar a empresa, a informação vai ser guardada na base de dados e a plataforma vai redirecionar o utilizador para a página da empresa.

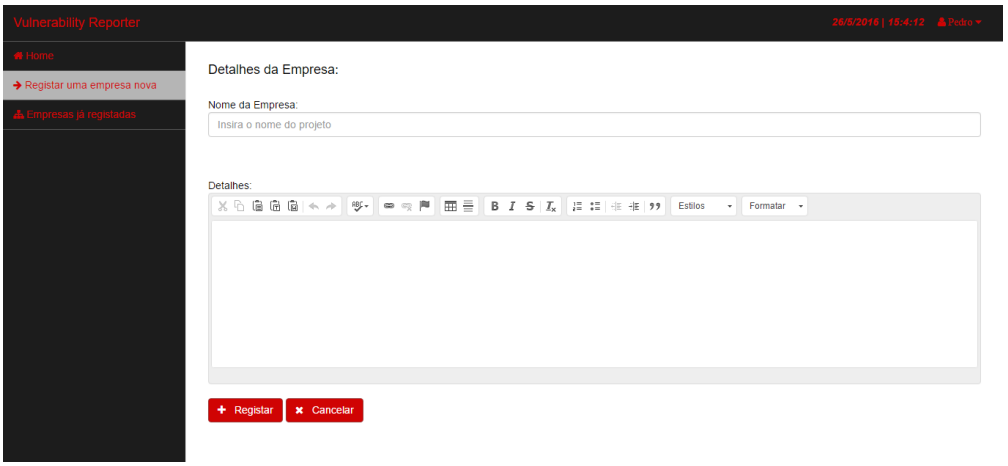


FIGURA 3.10: Página registo de empresas da plataforma

3.5.8 Página da empresa

Na página em que se apresenta a empresa, como se pode ver na figura 3.11, observamos os detalhes que inserimos um campo com as auditorias que foram ou estão a ser ainda realizadas, bem como um *dashboard* onde vão ser apresentados as vulnerabilidades que estão por mitigar e as que já foram entretanto mitigadas por auditoria. E como é possível observar na figura 3.11 existe uma opção de editar a informação da empresa caso seja necessário e isso vai abrir uma página igual à da figura 3.10, tendo apenas os campos já preenchidos com a informação que o utilizador já tinha inserido anteriormente.

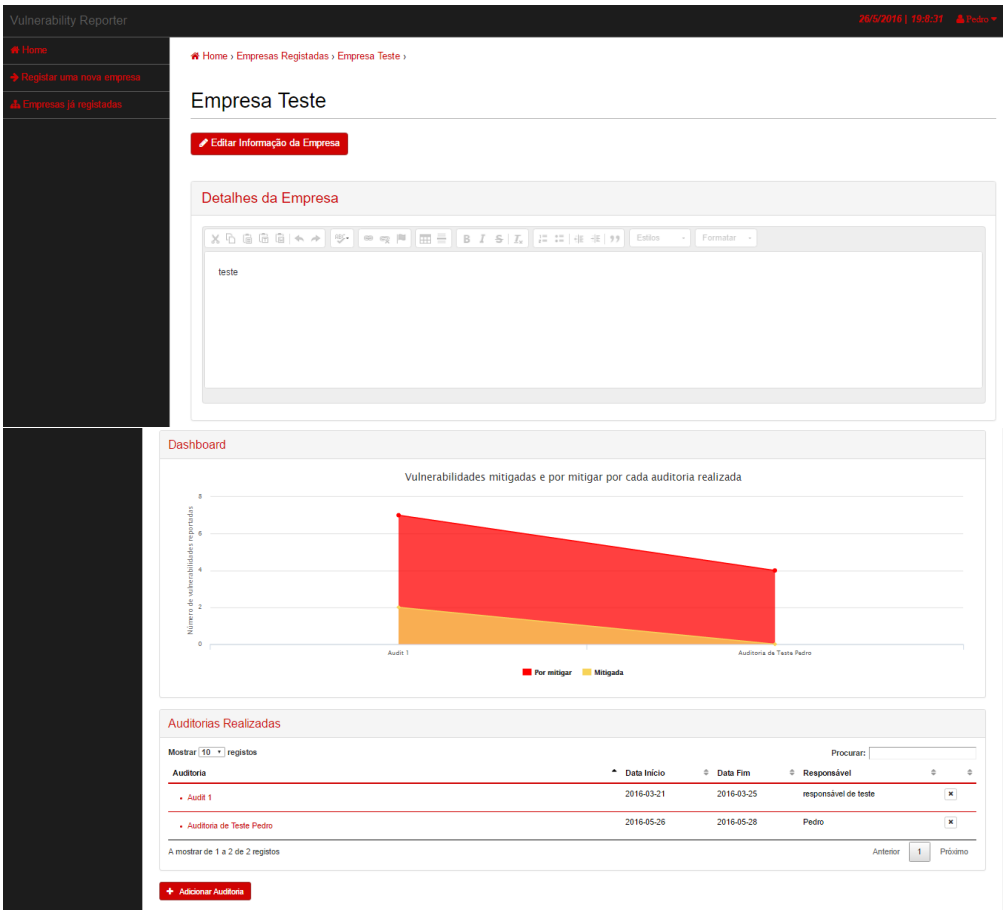


FIGURA 3.11: Página de informação acerca das empresas

3.5.9 Página de registo de auditorias

Para acrescentar de uma nova auditoria basta apenas carregar no botão que está presente na figura 3.11, e depois disso o utilizador irá passar para a página de criação da mesma. Nesta página o utilizador vai ter de inserir uma descrição para a auditora, a data de início e de fim da mesma, o responsável pela auditoria, os critérios usados e inserir as vulnerabilidades manualmente ou via um ficheiro de um scan do nessus, como se pode observar na figura 3.12.

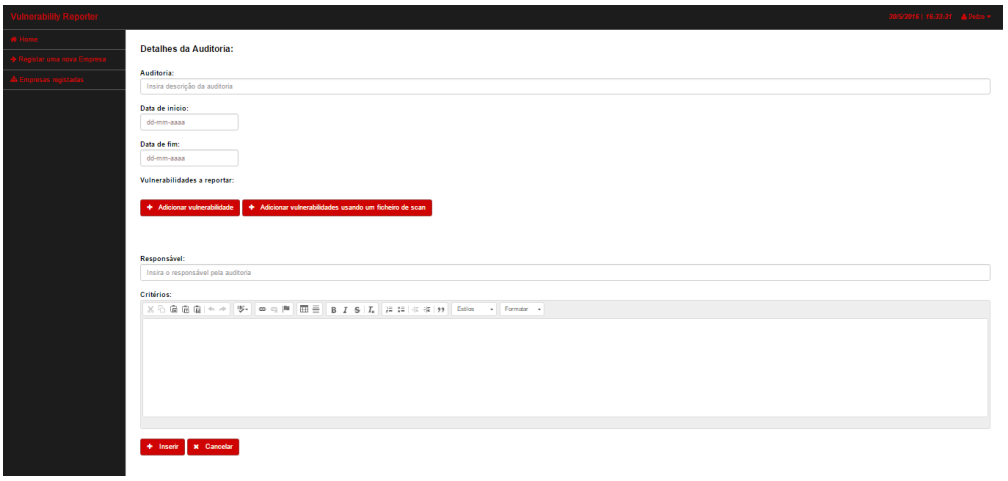


FIGURA 3.12: Página de registo de auditorias

No momento em que estamos a registar uma nova auditoria existe a possibilidade de também registar já as vulnerabilidades que possam ter surgido ou então usar um scan que possa ter realizado usando o Nessus. Depois do utilizador ter registado tudo o que deseja em relação à auditoria ele confirma os dados e carrega no botão para confirmar. Após o registo, o utilizador será redirecionado para a página da empresa, depois irá aparecer na tabela abaixo a auditoria que registo, ao clicar no nome da auditoria o utilizador vai ser enviado para a página da Auditoria.

3.5.10 Página de registo de vulnerabilidades usando um ficheiro de um scanner

Se o utilizador decidir usar a opção de fazer *upload* do ficheiro então vai ser redirecionado para a página que está representada na figura 3.13, onde o utilizador apenas precisa de selecionar a opção submeter ficheiro para escolher o scan a enviar e depois carregar em *upload* para a plataforma processar o ficheiro e extrair as vulnerabilidades do mesmo. Após as vulnerabilidades terem sido retiradas do relatório o utilizador, o mesmo irá ser redirecionado para a página onde estava anteriormente.

FIGURA 3.13: Página de upload dos ficheiros de scan

3.5.11 Página de registo de vulnerabilidades manualmente

Caso o utilizador decida inserir manualmente as suas vulnerabilidades ele vai ser direcionado para a página seguinte, apresentada na figura 3.14. Nesta página o utilizador vai poder inserir as informações necessárias, sendo que obrigatoriamente têm de estar presentes o nome da vulnerabilidade, o seu CVSS, os seus detalhes e indicar se ela já foi ou não mitigada, sendo que depois o utilizador se quiser dar mais detalhes acerca da vulnerabilidade ou modificar alguma coisa apenas precisa de carregar na opção de edição que irá existir na página em que aparecem as vulnerabilidades. Depois de o utilizador ter registado os dados relativos à vulnerabilidade o mesmo será redirecionado para a página onde estava anteriormente.

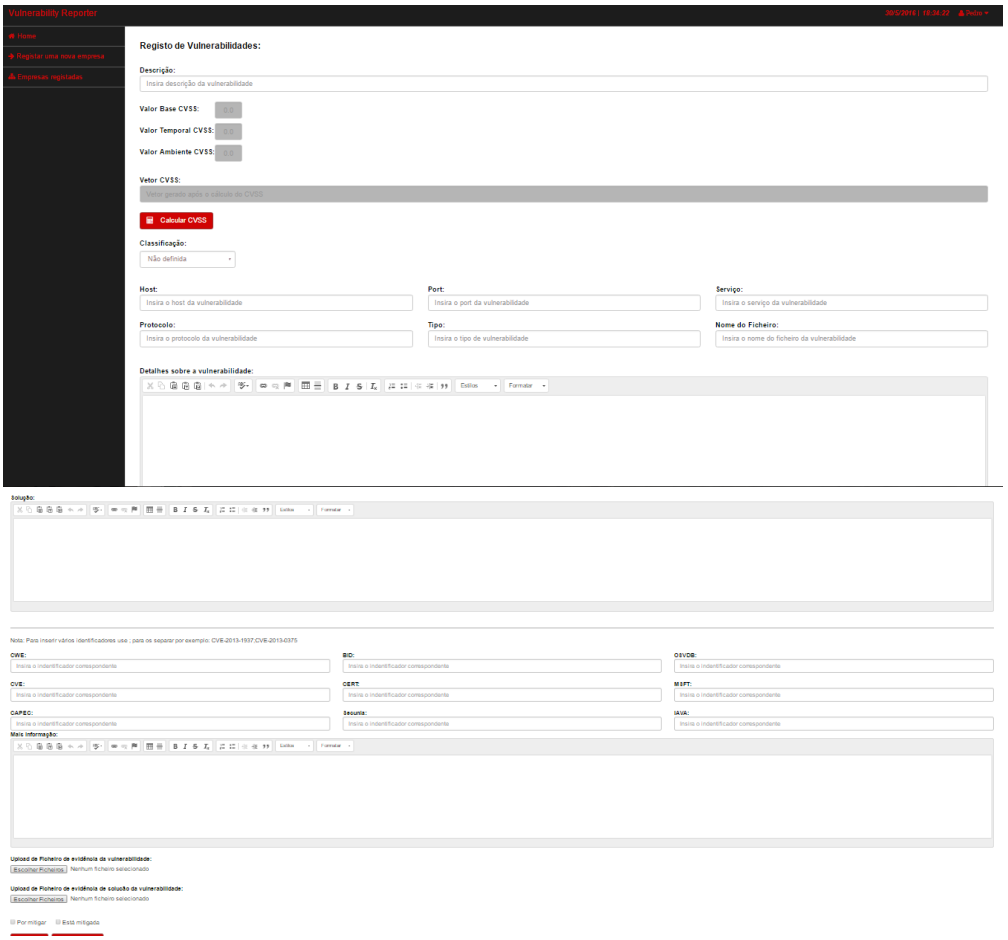


FIGURA 3.14: Página de registo de vulnerabilidades

3.5.12 Página da auditoria

Nesta página o utilizador, apresentada na figura 3.15, podem observar-se as informações sobre a auditoria bem como um *dashboard* que indica o número de vulnerabilidades que são classificadas como *critical* (crítica), *high* (alta), *medium* (média) e *low* (baixa). Esta página apresenta também um gráfico que mostra o risco que as vulnerabilidades demonstram, sendo que este gráfico usa a classificação final do CVSS para criar o mesmo e assim termos um gráfico que mostra o risco geral. O gráfico ao lado do de risco corresponde ao risco da confidencialidade, no qual vai usar a métrica do CVSS para classificar o risco entre 1, 5 e 10, sendo que 1 corresponde a *None* , 5 a *Partial* e 10 a *Complete*, sendo que ainda está incorporado no mesmo gráfico o risco para assim se poder comparar um gráfico com

o outro. O mesmo vai acontecer para os gráficos do risco de disponibilidade e de integridade. Sendo que para a integridade o gráfico vai tomar os valores tendo em conta a métrica do CVSS de *Integrity Impact* e para o gráfico de disponibilidade vai ter em conta a métrica *Availability Impact*. Em seguida vamos ter os gráficos referentes às classificações, em que dois se referem às classificações da OWASP, uma para o Top Ten da OWASP, outro para o Top Ten da OWASP mobile, em que ambos têm como propósito apresentar ao utilizador as 10 taxonomias que estes tops oferecem e contá-las, de modo a que seja possível verificar qual é a mais comum na auditoria. Por último, irá ser criado um gráfico que demonstra qual a classificação mais dominante na auditoria, sendo que para além dos dois tops do OWASP também vamos ter o não definida.

FIGURA 3.15: Página das auditorias

Para finalizar a página de auditorias temos a figura 3.16, onde observamos a

uma tabela onde vão estar expostas as vulnerabilidades que já foram reportadas na auditoria pelos utilizadores, na qual temos o nome das vulnerabilidades, o seu número identificador, a sua classificação, a sua taxonomia, o host onde foram reportadas e o resultado do seus CVSS. No final vamos ter os critérios da auditoria que foram inseridos pelos utilizadores quando registaram a auditoria.

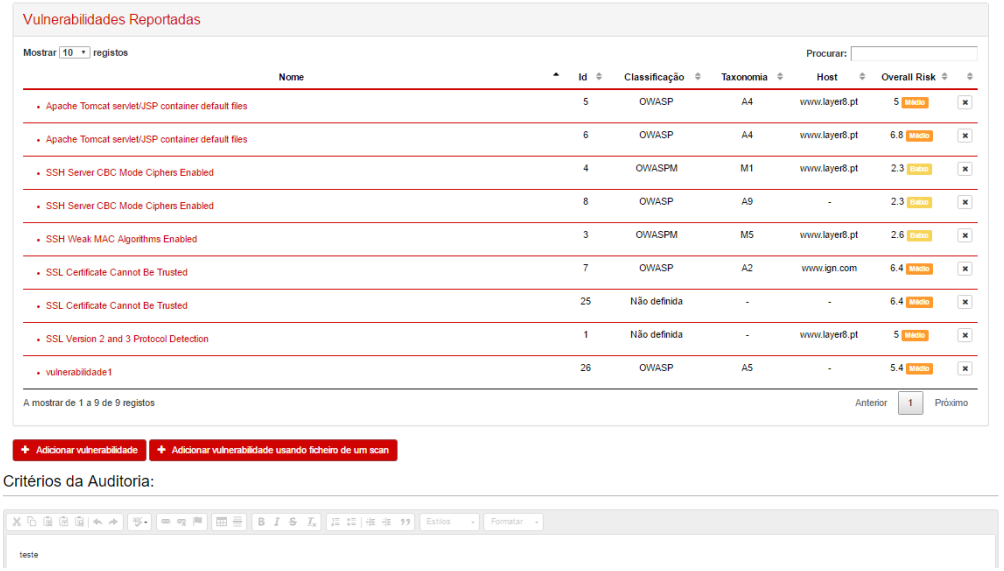


FIGURA 3.16: Página das auditorias

O utilizador nesta página pode ainda editar as informações da auditoria, escolher a opção de geração de um relatório da auditoria em PDF e gerir as vulnerabilidades que foram inseridas na auditoria.

3.5.13 Página de pré-visualização de PDF

Nesta página, que está apresentada na figura 3.17, o utilizador vai ver uma pré-visualização do relatório que vai ser gerado pela plataforma acerca da auditoria. Neste é possível observar como todos os dados que foram inseridos na auditoria, bem como as suas vulnerabilidades vão ser apresentadas no relatório final.

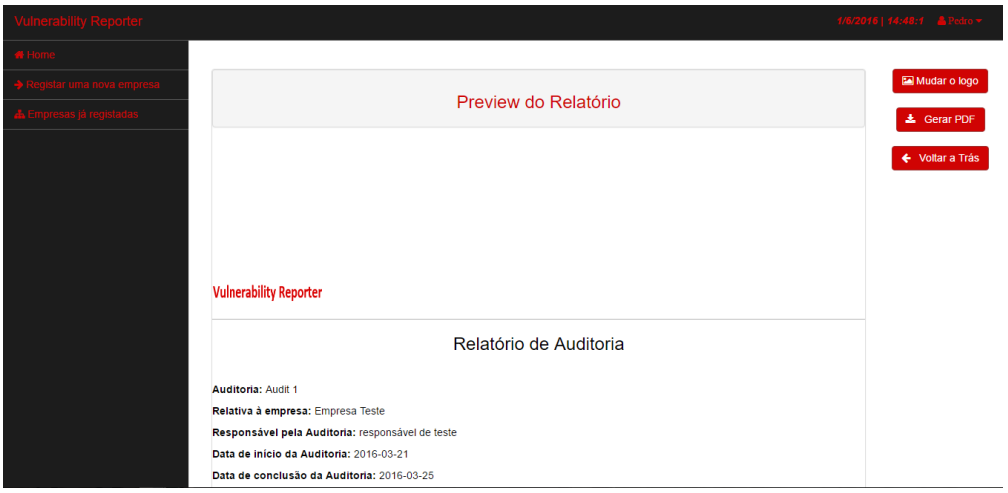


FIGURA 3.17: Página de pré-visualização de PDF

Como se pode observar na figura 3.17, do lado direito o utilizador vai ter 3 botões para interagir, um para regressar à auditoria, outro para fazer o download do PDF e finalmente um para fazer a alteração do logo que se quer inserir na auditoria. Ao decidir mudar o logo irá surgir uma janela para o utilizador poder efetuar o *upload* de um logotipo novo ou então usar um que já está presente, como se pode ver na figura 3.18.

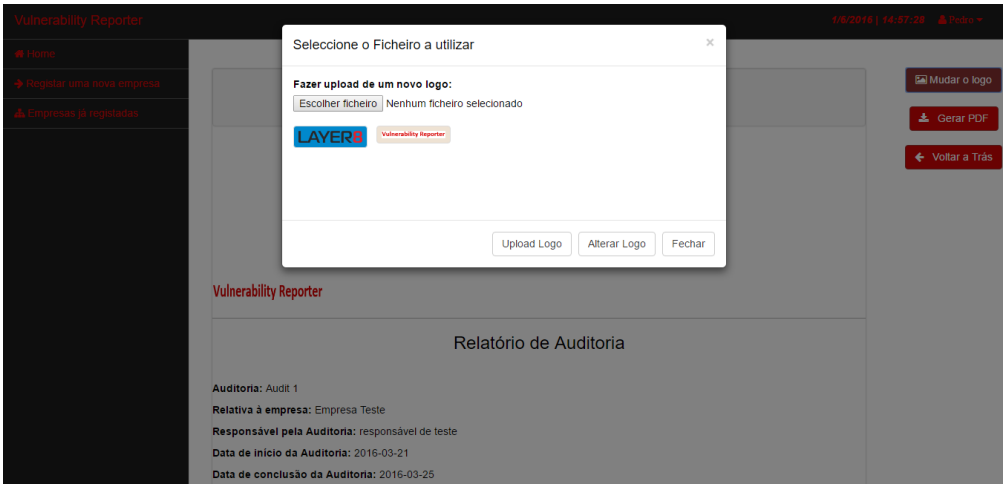


FIGURA 3.18: Exemplo da mudança de logótipo

3.5.14 Página de vulnerabilidades

A página representada na figura 3.19, vai apresentar a informação que o utilizador inseriu quando a registou. Aqui vamos poder ver as métricas que foram inseridas para o cálculo do CVSS relativo à vulnerabilidade bem como um gráfico que mostra os resultados para os cálculos dos 3 tipos de métricas e em baixo vamos observar os seus detalhes. Neste vamos ter informações sobre a vulnerabilidade, referências sobre a mesma que tenham sido inseridas durante o seu registo, bem como as imagens de evidências que o utilizador tenha em relação à existência das mesmas e a sua solução. Tendo ainda também um campo onde o utilizador pode inserir mais informação relativa à vulnerabilidade que deseje.

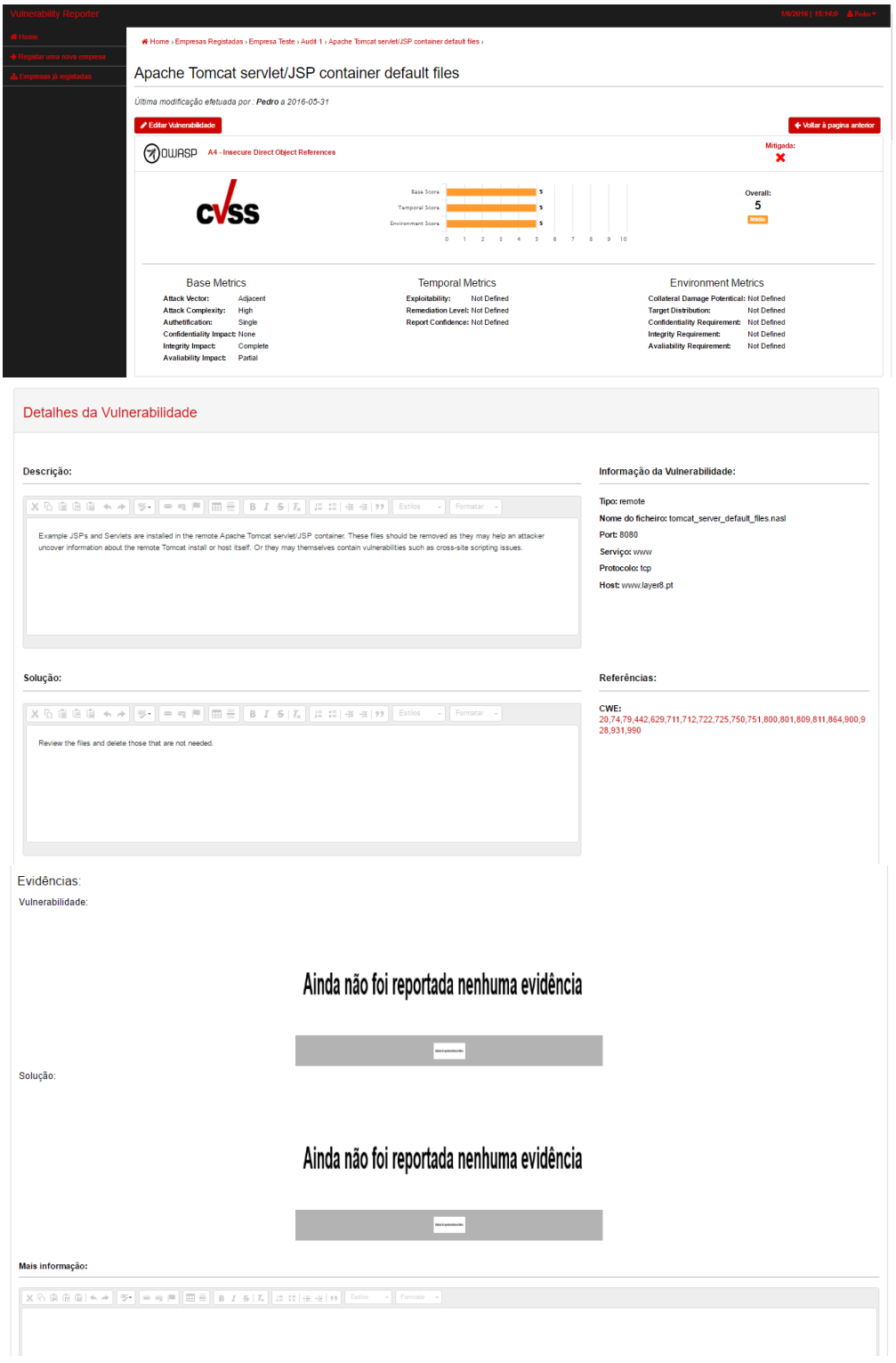


FIGURA 3.19: Página em que se apresentam as vulnerabilidades

Capítulo 4

Validação e Testes

Ao longo capitulo será abordada a validação levada a cabo ao sistema desenvolvido. Uma vez que esta é um sistema que foi desenvolvido especificamente para auditores e empresas que efetuam auditorias recorrendo a testes de intrusão, foi preparado um questionário especificamente desenhado para que esses profissionais e especialistas pudessem emitir a sua opinião.

Este questionário foi fornecido para que os mesmos pudessem revelar a sua opinião relativamente a experiência que tiveram ao testar a plataforma, bem como capturar as opinioes acerca de melhorias que se possam efetuar e ideias para o melhoramento da mesma.

Seguidamente será apresentado o questionario realizado, sendo que para testar a plataforma foram convidados essencialmente profissionais da area da seguranca de informação, auditores que recorrem a testes de intrusão. O questionário esteve disponível, por convite, entre o período compreendido entre Julho e Agosto de 2016. O sistema desenvolvimento esteve disponível em *http://vulnreporter.allofads.com/* e o questionario em *http://goo.gl/forms/aTdbb9vtZAhnsnMe2*.

Durante este período de testes a plataforma recebeu cerca de 30 registos de utilizadores, sendo que foram obtidas cerca de 10 respostas válidas por parte dos utilizadores. Destas 10 respostas 8 foram obtidas diretamente atraves de respostas ao questionario sendo que 2 delas foram obtidas via email com informação mais

descritiva. A plataforma continua ativa, e os utilizadores continuam a registar-se e a poderem usar a plataforma, no entanto, no contexto deste trabalho e da validação efetuada foram consideradas as respostas indicadas anteriormente.

4.1 Questionário

O questionário que foi apresentado aos utilizadores que testaram a plataforma foi dividido em 3 partes:

- Introdução - algumas questões que servem para reunir informação acerca do utilizador e da sua experiência na área de segurança.
- Questões relativas à experiência neste tipo de plataformas - umas questões que têm como propósito verificar a experiência do utilizador com este tipo de plataformas.
- Questões relativas à plataforma - questões para verificar a opinião do utilizador em relação à plataforma que foi desenvolvida.

As questões que foram apresentadas aos utilizadores foram as seguintes:

Introdução

1. Género do utilizador?
   (a) Masculino
   (b) Feminino
2. Idade do utilizador?
3. Profissão
4. Experiência na área de segurança informática

Questões relativas à experiência neste tipo de plataformas

1. Já teve alguma experiência com uma plataforma similar à que testou?
2. Atualmente usa uma plataforma semelhante a esta no seu emprego?

Questões relativas à plataforma

1. A informação que é pedida em relação à auditoria, é suficiente para a definir uma auditoria de segurança?
2. Os gráficos que são apresentados na página de auditoria ajudam na perceção do risco em que se encontra exposta a empresa?
3. A informação que é fornecida acerca do CVSS para efetuar o cálculo e a que está apresentada para todos os utilizadores observarem nas vulnerabilidades está explicita corretamente?
4. Os gráficos que são apresentados na página de vulnerabilidades são percetíveis?
5. A informação que é pedida para as vulnerabilidades é suficiente para definir a mesma?
6. As referências que são utilizadas de modo a fornecer mais informação à vulnerabilidade são suficientes?
    - Escolher o valor utilizando a escala de Likert de 1 a 5 (1- Insuficiente ; 5- Suficiente).
7. Em relação aos relatórios que são gerados pela plataforma, a informação que está contida nos mesmos é útil para um profissional da área?
8. Na página das empresas é percetível a existência de evolução de auditoria para auditoria que o gráfico tenta fornecer?
9. No envio do ficheiro de scan do nessus encontrou algum problema após a plataforma o ter analisado? (Por exemplo a plataforma falhou na análise do ficheiro, existiu vulnerabilidades que não foram identificadas pela plataforma)

10. É percetível nos gráficos do risco de confidencialidade, integridade e disponibilidade os valores que são usados para os medir?
11. Qual a sua opinião em relação ao design da plataforma?
    - Escolher o valor utilizando a escala de Likert (1- Mau ; 5- Bom).
12. Esta ferramenta seria algo que usaria no seu dia-a-dia profissional?
13. Encontrou algum problema durante o seu teste à plataforma?
14. Sugestões de melhoria ou de funcionalidades para acrescentar

4.2 Análise ao feedback recebido

Esta secção vai apresentar o balanço que foi realizado em relação ao feedback que foi enviado por parte dos utilizadores, bem como as respostas que foram obtidas em relação ao questionário.

Como podemos observar no gráfico da figura 4.1 todos os utilizadores que utilizaram a plataforma são do género masculino, bem como a maioria dos utilizadores tem a idade entre os 25 e 30 como se pode verificar na figura 4.2.

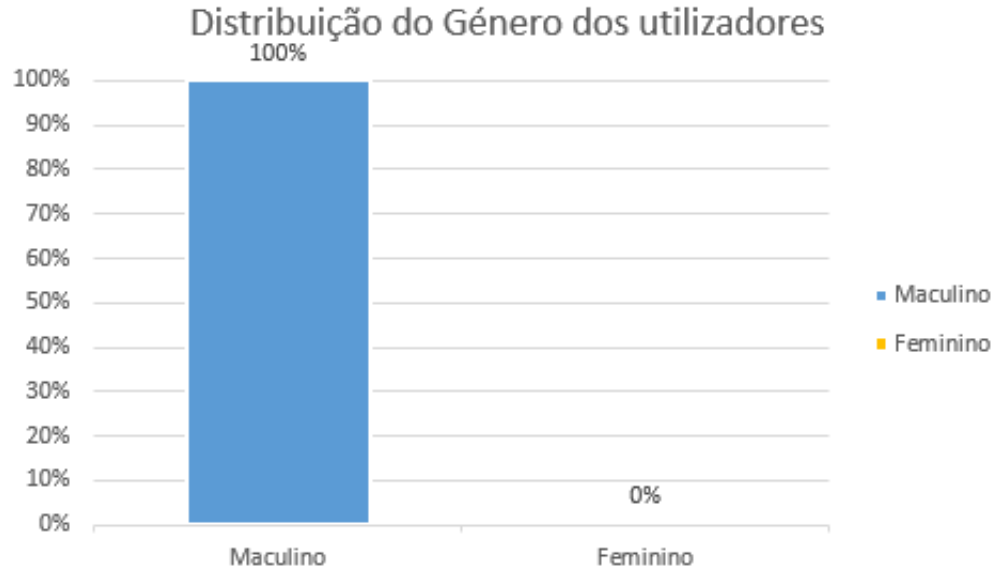


FIGURA 4.1: Género dos utilizadores registados

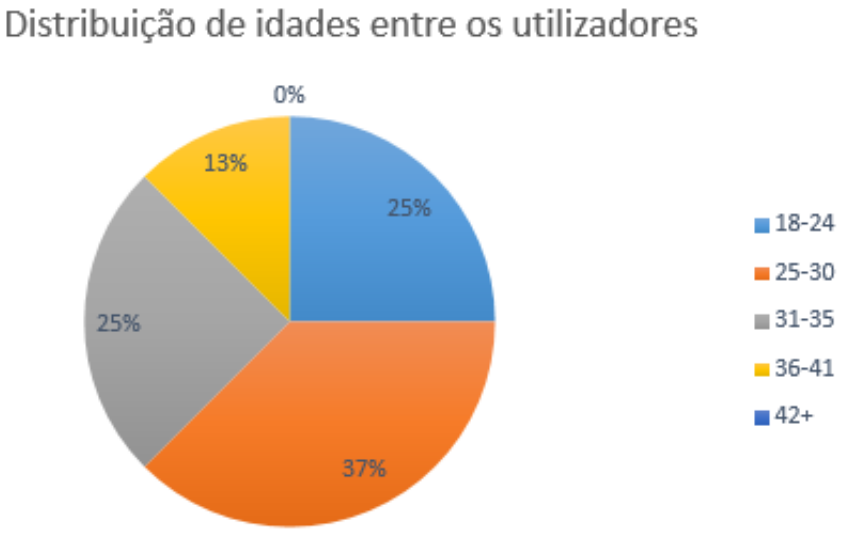


FIGURA 4.2: Distribuição de idades dos utilizadores

Relativamente à experiência que os utilizadores já possuíam em relação a este tipo de plataformas como podemos observar na figura 4.3 muitos já tinham tido um contato posteriormente com um plataforma deste tipo.

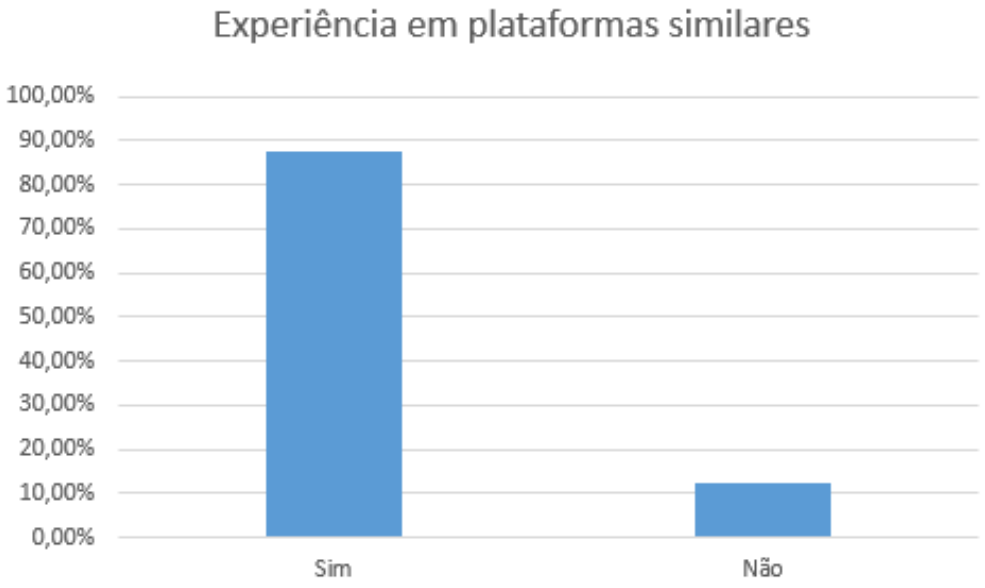


FIGURA 4.3: Contato que os utilizadores tiveram com plataformas deste tipo

Outras das questões relativas ao contato dos utilizadores com plataformas semelhantes foi se os mesmos utilizam atualmente na sua profissão algo similar, e

como podemos observar no gráfico da figura 4.4, a grande maioria não utiliza algo similar no dia-a-dia profissional.

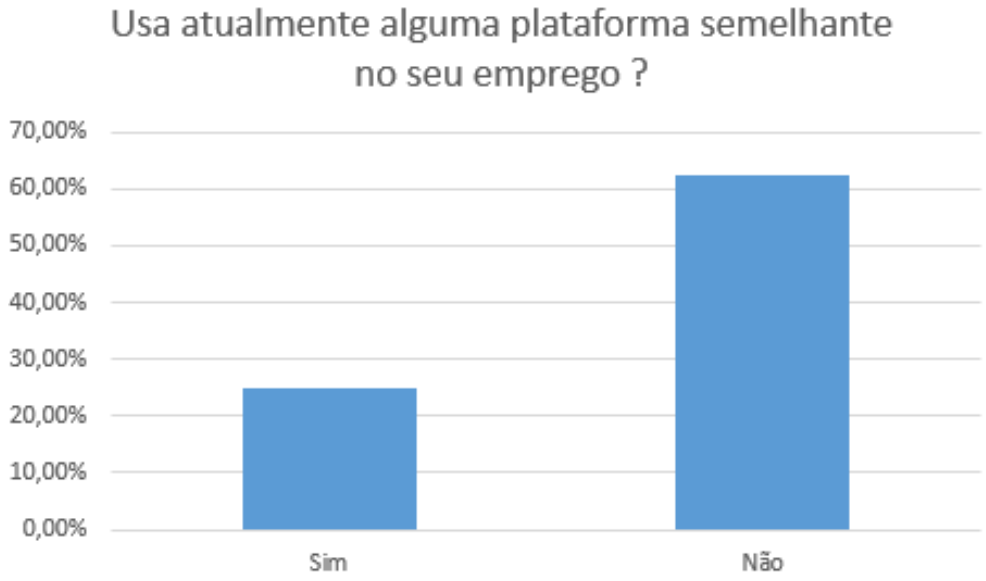


FIGURA 4.4: Utilização de plataformas similares no emprego

Seguindo para a avaliação das funcionalidades da plataforma, os utilizadores avaliaram a informação que é pedida pela mesma para fazer o registo de empresas, auditorias e vulnerabilidades. No aspeto da auditoria as respostas dos utilizadores foram muito positivas indicando sempre que sim vários utilizadores, alguns utilizadores também deram mais ideias de informação que também é importante definir-se para as mesmas, que no futuro poderão ser adicionadas.

Em relação à informação que são pedidas para as vulnerabilidades, podemos ver pelos gráficos das figuras 4.5, 4.6 e 4.7 em que se apresenta os resultados das perguntas aos utilizadores em relação à informação que a plataforma pede é suficiente para as definir, bem como verificar se o CVSS está bem caracterizado para os utilizadores e os gráficos que são gerados tendo em conta o mesmo modelo de classificação. Como podemos observar os utilizadores responderam sempre positivamente em relação a estes aspetos.

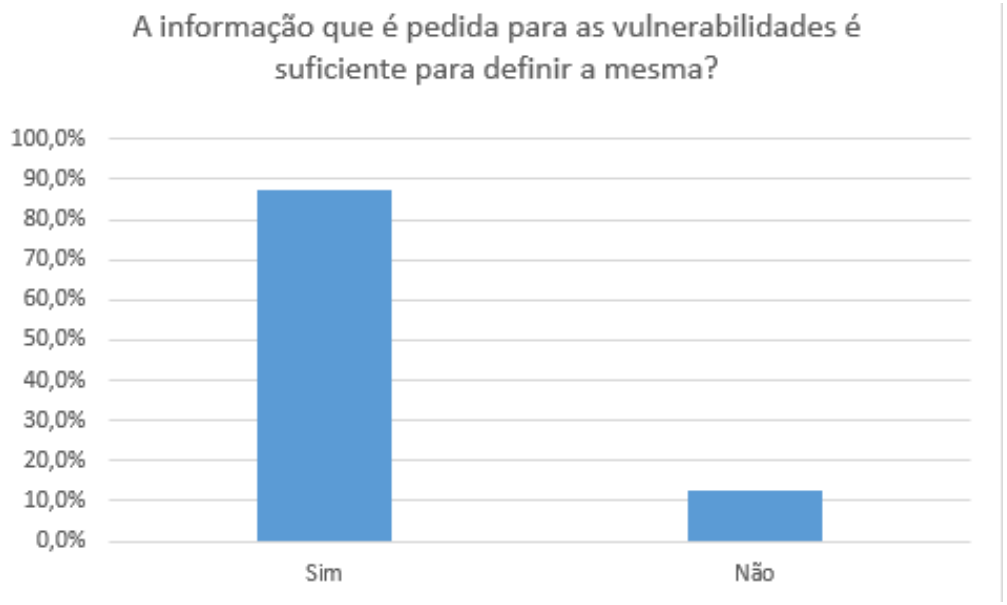


FIGURA 4.5: Informação guardada pela plataforma para registar uma vulnerabilidade

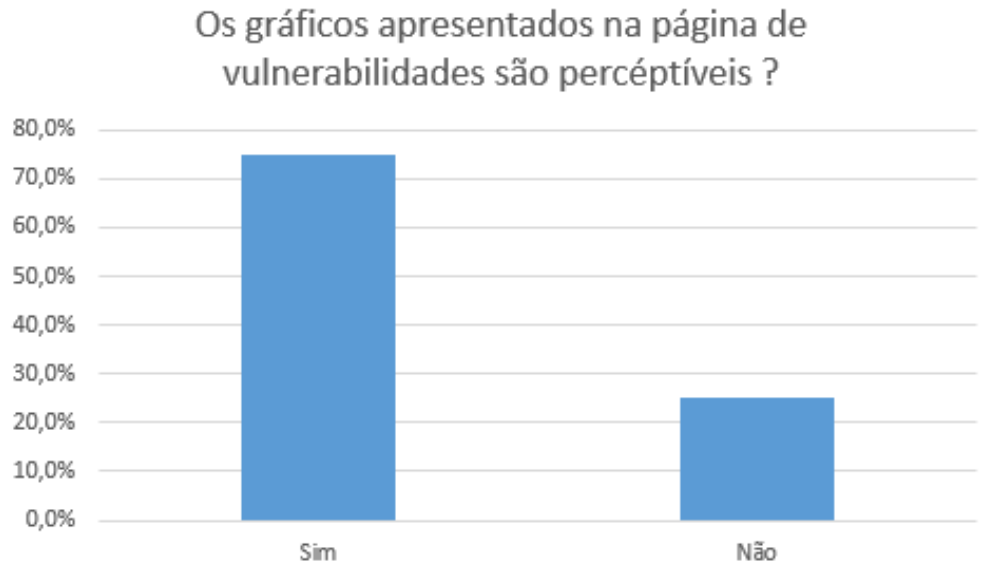


FIGURA 4.6: Gráficos de resposta à pergunta relativa aos gráficos apresentados na página das vulnerabilidades

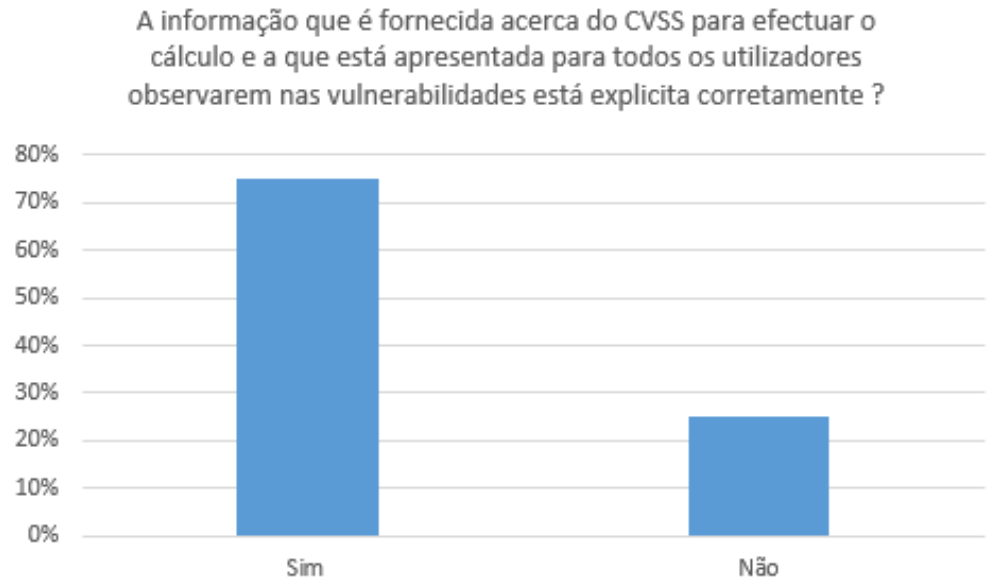


FIGURA 4.7: Gráfico que apresenta as respostas dadas pelos utilizadores relativo à informação do CVSS

Ainda relativamente a informação sobre as vulnerabilidades, existiu uma questão que se colocou aos utilizadores se as referências que são pedidas de modo a complementar a informação das vulnerabilidades podemos observar que servem para introduzir mais informação como podemos observar no gráfico da figura 4.8, os utilizadores em grande parte acharam que as mesmas servem de complemento para as vulnerabilidades.

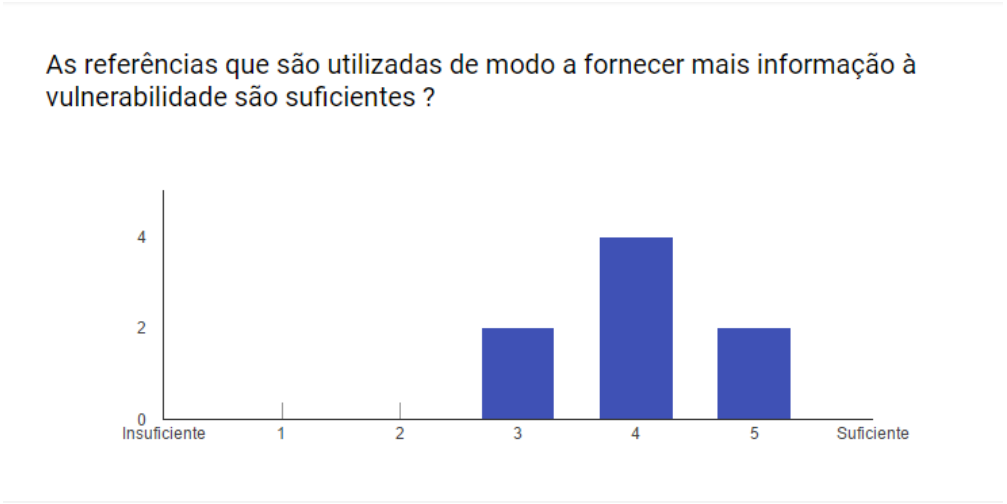


FIGURA 4.8: Gráfico que apresenta as respostas dadas pelos utilizadores relativo à informação que as referências oferecem para as vulnerabilidades

Tendo em conta os gráficos que são gerados com a informação das métricas do CVSS das vulnerabilidades relativos ao risco, à confidencialidade, integridade e disponibilidade os utilizadores indicam que os mesmos trazem informação, mas existiu alguns a que os gráficos não adicionaram mais informação, como se pode ver no gráfico da figura 4.9.

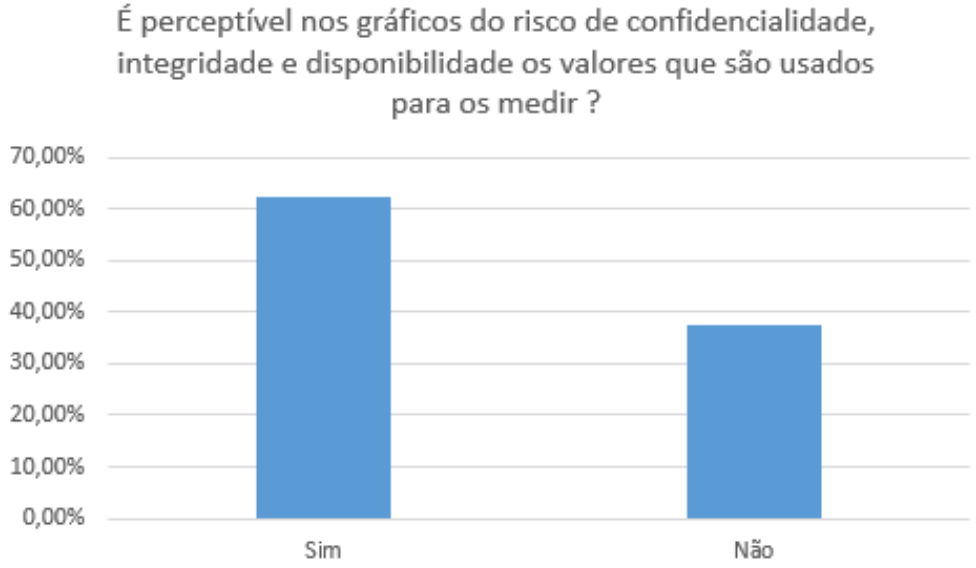


FIGURA 4.9: Gráfico que apresenta as respostas dadas pelos utilizadores relativo à perceção dos gráficos de risco, confidencialidade, integridade e disponibilidade

Em relação à questão acerca do relatório que é gerado pela plataforma, muitos utilizadores acharam que a informação que os mesmos possuem é útil para eles, mas houve outros utilizadores que acharam que a informação fornecida não chegava, como se pode ver no gráfico da figura 4.10, onde se nota que as opiniões são equilibradas. Existiram uns utilizadores que sugeriram que para melhorar, o relatório poderia começar a apresentar informação relativa à evolução da segurança de auditoria para auditoria.

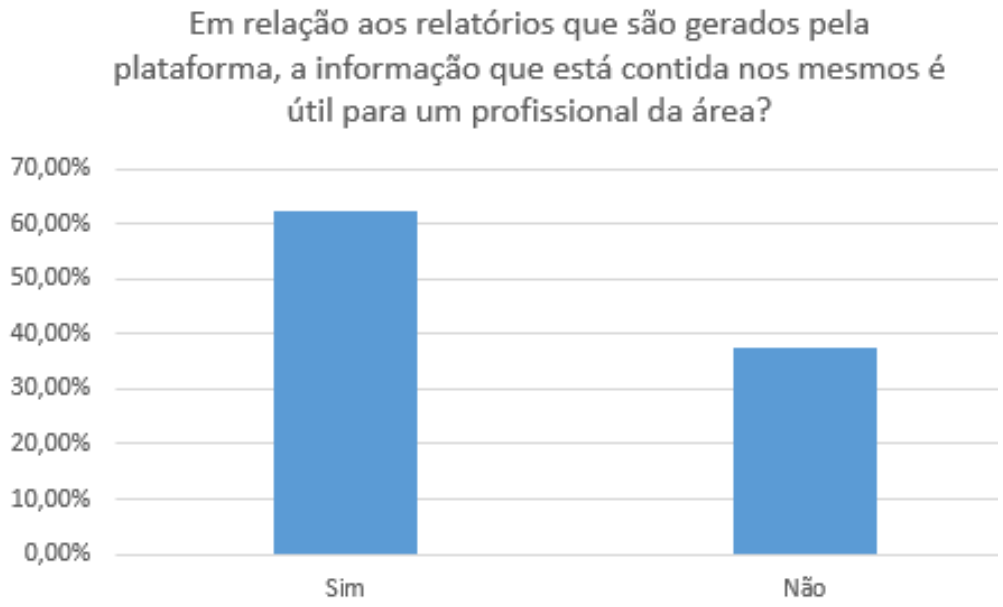


FIGURA 4.10: Gráfico que apresenta as respostas dadas pelos utilizadores em relação à informação que é apresentada nos relatórios gerados pela plataforma

Na questão da funcionalidade de submeter um ficheiro de Nessus para a plataforma extrair as vulnerabilidades do mesmo, apenas 2 utilizadores é que a testaram indicando que a mesma funcionava sem problemas, sendo que os restantes não possuíam nenhum ficheiro de Nessus para efetuar o teste.

Relativamente à informação que a plataforma pede para as auditorias os utilizadores acharam que o que a plataforma pede como base é suficiente, apenas se o campo dos critérios for preenchido corretamente, e ainda sugeriram em algumas melhoras neste aspeto, como por exemplo, inserir a equipa que está associada à auditoria, bem como o âmbito da mesma de modo a ser possível ter mais detalhes. Outra questão que foi colocada aos utilizadores em relação às auditorias foi em relação aos gráficos que são apresentados na página das mesmas, se estes ajudariam a melhorar a perceção do risco em que uma empresa se encontra exposta, grande parte dos utilizadores responderam que sim, sendo que um teve problemas a visualizar os mesmos e outro indicou que seria melhor se estes apresentam-se uma evolução a nível temporal, bem como a nível da mitigação das vulnerabilidades que foram reportadas nestas auditorias.

À questão do *design* da plataforma que se colocou aos utilizadores, ao observarmos o gráfico na figura 4.11 pode-se concluir que os utilizadores não acharam o mesmo mau mas também não é assim tão bom, e este pode ser um dos aspetos a melhorar no trabalho futuro.

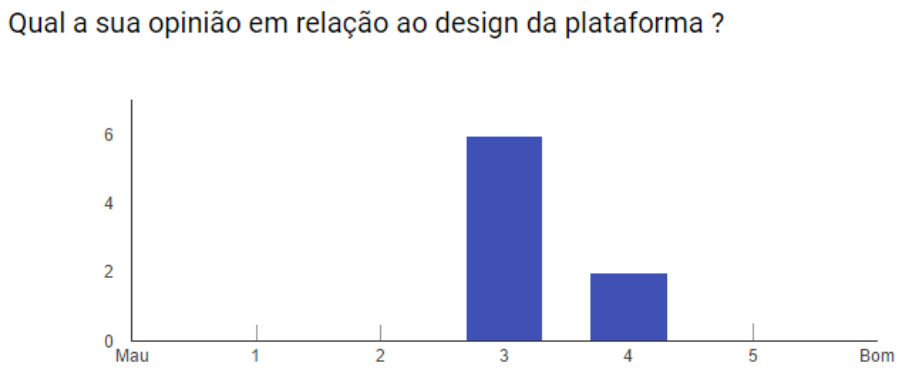


FIGURA 4.11: Gráfico que apresenta as respostas dadas pelos utilizadores relativamente ao *design* da plataforma

Grande parte dos utilizadores, como podemos observar no gráfico da figura 4.12, quando foram questionados se utilizariam esta plataforma no seu dia-a-dia profissional todos responderam que sim, apenas um respondeu negativamente.

FIGURA 4.12: Gráfico que apresenta as respostas dadas pelos utilizadores em relação à utilização da plataforma no seu dia-a-dia profissional

Como podemos ver em relação ao feedback reportado pelos utilizadores, a plataforma é capaz de corresponder às expectativas esperadas para responder às questões que foram levantadas no início da investigação. Tendo em conta que foram apenas encontrados alguns *bugs* a nível visual, bem como um utilizador que encontrou alguns problemas ao testar e algumas vulnerabilidades que foram reportadas que em trabalho futuro irão ser mitigadas, mas os restantes não apontaram mais nada. Foram também recebidas ideias por parte dos utilizadores, para o melhoramento da plataforma, estas foram tidas em conta e algumas delas encontram-se presentes no capítulo 5.

Capítulo 5

Conclusão e Trabalho Futuro

Neste capítulo vão ser abordadas as conclusões que foram retiradas ao longo desta investigação, bem como o trabalho futuro poderá se realizar de modo a que ajude a plataforma a evoluir e a fazer com que a mesma fique mais completa.

5.1 Conclusão

Esta dissertação tem como foco em agregar por auditorias as vulnerabilidades que são descobertas, quando são efetuados testes de intrusão e auditorias a nível da segurança nos sistemas de informação, de modo a que os utilizadores consigam juntar toda a informação que vão recolhendo ao logo dos mesmos.

Para tal foi necessário analisar e fazer um levantamento das principais funcionalidades e requisitos da plataforma, de modo a que se pudesse comparar com as plataformas já existentes e verificar o que estava em falta nas mesmas. Outro dos pontos que foram analisados, foi o da necessidade de um sistema para classificar as vulnerabilidades, bem como a classificação das mesmas e a informação que é necessária para que as mesmas fiquem bem definidas.

A plataforma esteve disponível para que um grupo de utilizadores que foram convidados a testar a mesma para assim se puder validar. Durante o período de

tempo que a mesma esteve disponível existiu o registo ao todo de 30 utilizadores, aos mesmos foi pedido que respondessem a um questionário para que se pudesse validar a plataforma e esta dissertação.

Relembrando as questões de investigação que foram feitas para que esta investigação conseguisse corresponder, "É possível fornecer aos auditores uma solução que lhes permita registar os resultados que vão obtendo dos testes de intrusão que são efetuados?", outra questão era se "É possível uma plataforma suportar receber resultados de ferramentas de deteção automáticas de vulnerabilidade?". Finalmente a última questão consistia em saber se "É possível fornecer ao auditor responsável um *dashboard*, onde o mesmo pode observar os vários riscos que as vulnerabilidades apresentam à empresa em causa?".

Para se puder responder às questões foram definidos os seguintes objetivos para servirem como ponto de guia ao longo do desenvolvimento desta investigação, o desenvolvimento de uma plataforma que seja capaz de agregar os resultados das auditorias realizadas pelos auditores, com estes resultados ter a capacidade de gerar *dashboards* e gráficos evolutivos que facilitem a perceção tanto dos auditores, bem como gestores de topo. E por último fornecer a geração de relatórios personalizáveis tendo em conta a informação que foi submetida.

Para determinar se foi possível concluir os objetivos inicialmente propostos para esta tese, foi realizado o questionário aos utilizadores tendo como foco criar questões para os utilizadores responderem que abordassem estes pontos como se pode observar em 4, bem como demonstrar que a plataforma ia de encontro às questões efetuadas inicialmente. Isso podemos observar pelo feedback que foi recebido em relação à primeira questão em que o *feedback* obtido pelos utilizadores que testaram a plataforma foi bastante positivo, e assim foi possível determinar que a plataforma é capaz de responder corretamente à primeira questão, bem como satisfazer o primeiro objetivo imposto.

Em relação à segunda questão foi a mais difícil de responder, pois a plataforma suporta receber os resultados de uma ferramenta automática, mas como podemos verificar na secção 4.2, poucos foram os utilizadores que conseguiram validar esse

ponto, sendo que os que conseguiram testar disseram que a funcionalidade estava a fazer o proposto, esta entra também um pouco dentro do primeiro objetivo pois é mais uma forma de recolher informação de vulnerabilidades oferecida pela plataforma.

De modo a responder à última questão, esta mesma que está enquadrada com o segundo objetivo que foi proposto, foram criados vários *dashboards* com o propósito de apresentar a informação que é cedida pelos utilizadores ao registarem as vulnerabilidades, fornecendo assim aos utilizadores informação sobre o risco e outros parâmetros importantes a nível de segurança que as vulnerabilidades.

Para se conseguir concluir o último objetivo que foi proposto a plataforma oferece uma funcionalidade de geração de relatórios personalizáveis aos utilizadores por auditoria de modo a que possam gerar um relatório com a informação que inseriram, esta última função os utilizadores acharam que era útil para quem trabalha na área de segurança.

Com toda esta informação é possível afirmar que conseguimos responder com sucesso ás questões, bem como chegar aos objetivos que foram propostos inicialmente. Como hoje em dia cada vez mais verifica-se a necessidade de existência de plataformas deste tipo, pois com o crescimento a nível dos sistemas de informação torna-se primordial que os mesmos estejam seguros para que não haja nenhuma falha que possa afetar uma empresa.

Esta plataforma foi desenvolvida com o intuito de ser *Open-source*, sendo que o código pode ser utilizado por todos e está aberto sendo que se encontra no seguinte link *https://github.com/Cananex/VulnerabilityReporter*. A plataforma encontra-se neste momento online, no seguinte link *http://vulnreporter.allofads.com/*.

5.2 Trabalho Futuro

Nesta secção vão ser registadas ideias de como se poderá evoluir a plataforma no futuro.

Todos os pontos que vão ser aqui apresentados vêm de sugestões dadas pelos utilizadores que testaram a plataforma, bem como ideias que sugeriram depois de vários testes à plataforma. Fazer um *hardening* à plataforma de modo a mitigar as vulnerabilidades que possam existir, bem como outras que possam aparecer mais tarde, efetuar um polimento a nível da interface de modo a que seja mais percetível por parte dos utilizadores, manter um registo de todas as modificações que foram efetuadas às vulnerabilidades, bem como quem as modificou, possibilitar a importação de vulnerabilidades a partir de outras plataformas para além do Nessus, adaptar a plataforma para refletir melhor a realidade em que vai ser utilizada, agregar as vulnerabilidades por subnet, categoria, entre outras.

Algo que pode também ser criado com base nesta é um *scanner* automático de vulnerabilidades, no qual seja possível criar a auditoria, deixar o mesmo correr e ele próprio encontra as vulnerabilidades e regista as mesmas na plataforma, tornando assim o processo de testes intrusão e registo dos mesmos totalmente automático.

Outra extensão que pode ser feita tendo esta plataforma como base seria utilizar a informação que a plataforma recebe das auditorias que a mesma recebe e poder fazer uma avaliação relativamente à maturidade das organizações, de modo a que avalie tendo em conta a capacidade de resolução de vulnerabilidades por parte das organizações que sofreram as auditorias.

Anexos

Anexo A

Equações CVSS

Retirado de [6]

A.1 Equações de Base

Resultado de Base: ((0.6 * Impact) + (0.4 * E)-1.5) * f(Impact)

Impact: 10.41*(1 -(1 - Confidentiality Impact)*(1 - Integrity Impact) * (1 - Avaliability Impact))

E = 20* AV * AC * Au

f(Impact)=0 se Impact = 0, caso contrário f(Impact)=1.176

AV - Se for L=0.395, se for A=0.646, se for N=1.0

AC - Se for H=0.35, se for M=0.61, se for L=0,71

Au - Se for M=0.45, se for S=0.56, se for N=0,704

Confidentiality Impact= Se for N=0.0, se for P=0.275, se for C=0.660

Integrity Impact= Se for N=0.0, se for P=0.275, se for C=0.660

Avaliability Impact= Se for N=0.0, se for P=0.275, se for C=0.660

A.2 Equação Temporal

Resultado Temporal: Resultado de Base * E * RL * RC

E - Se for U=0.85, se for POC=0.9, se for F=0.95, se for H=1.0, se for ND=1.0

RL - Se for OF=0.87, se for TF=0.90, se for W=0.95, se for U=1.0, se for ND=1.0

RC - Se for U=0.90, se for UC=0.95, se for C=1.0, se for ND=1.0

A.3 Equação do Ambiente

Resultado do Ambiente: (AdjustedTemporal + (10-AdjustedTemporal) * CDP) * TD)

AdjustedTemporal: Corresponde ao TemporalScore recalculado com o Impact do resultado de base substituido com a seguinte equação.

min(10, 10.41 * (1 - (1 - C * CR) * (1 - I * IR) * (1 - A * AR)))

CDP - Se for N=0.0, se for L=0.1, se for LM=0.3, se for MH=0.4, se for H=0.5, se for ND=0.0

TD - Se for N=0.0, se for L=0.25, se for M=0.75, se for H=1.0, se for ND=1.0

CR, IR, AR - Se for L=0.5, se for M=1.0, se for H=1.51, se for ND=1.0

Bibliografia

[1] A. H. Chatterjee and S., *Design Research in Information Systems*. Springer, 2010.

[2] A. J. d. S. Brandão, "Uso de ontologia para a classificação de vulnerabilidades em sistemas computacionais," Master's thesis, ICMC-USP, 2004.

[3] "Common vulnerabilities and exposures — cve." `https://cve.mitre.org/docs/cve-intro-handout.pdf`. Accessed: 2016-03-21.

[4] Y. Z. Qixu Liu, "Vrss: A new system for rating and scoring vulnerabilities," *Computer Communications*, vol. 34, pp. 264–273, 2011.

[5] Y. K. Q. W. Qixu Liu, Yuqing Zhanga, "Improving vrss-based vulnerability prioritization using analytic," *The Journal of Systems and Software*, vol. 85, pp. 1699–1708, 2012.

[6] S. R. Peter Mell, Karen Scarfone, *A Complete Guide to the Common Vulnerability Scoring System Version 2.0*. Forum of Incident Response and Security Teams, 2 ed., 2007.

[7] K. S. Peter Mell, Sasha Romanosky, "Common vulnerability scoring system," *IEEE SECURITY & PRIVACY*, vol. 4, pp. 85–89, 2006.

[8] S. R. Peter Mell, Karen Scarfone, "The common vulnerability scoring system (cvss) and its applicability to federal agency systems," tech. rep., National Institute of Standards and Technology, 2007.

[9] *An Analysis of CVSS Version 2 Vulnerability Scoring*, 2009.

[10] "Common vulnerabilities and exposures (cve)." `https://cve.mitre.org/`. Accessed: 2016-04-07.

[11] "Common weakness enumeration (cwe)." `https://cwe.mitre.org/`. Accessed: 2016-04-08.

[12] R. Rogers, *Nessus Network Auditing*. Syngress Inc., 10 2011.

[13] "Securityfocus." `http://www.securityfocus.com/`. Accessed: 2016-04-08.

[14] "Vulnerability notes database." `https://www.kb.cert.org`. Accessed: 2016-04-08.

[15] "Cert." `http://www.cert.org/`. Accessed: 2016-04-08.

[16] "Exploit database." `https://www.exploit-db.com/`. Accessed: 2016-04-08.

[17] "Microsoft security bulletin." `https://technet.microsoft.com/en-us/security/bulletin/`. Accessed: 2016-04-09.

[18] "Secunia." `https://secunia.com/`. Accessed: 2016-04-10.

[19] C. on National Security Systems, *Committee on National Security Systems (CNSS) Glossary*. National Security Agency, 4 2015.

[20] "Information assurance vulnerability alert (iava)." `https://www.saintcorporation.com/solutions/IAVA.html`. Accessed: 2016-04-11.

[21] Enforcive, "Bsafe/cross-plataform audit," 2005.

[22] FireEye, "Cm series: Real-time exchange of dynamic threat intelligence and unified management of enterprise deployments," 2015.

[23] D. Software, "Analyzer," 2001.

[24] T. Limited, *Nipper Studio – Beginner's Guide*. Titania Limited, 2015.

[25] "Faraday." `https://www.faradaysec.com/`. Accessed: 2016-05-9.

[26] "Penteston." `https://penteston.com/`. Accessed: 2016-05-9.

[27] "Dradis-effective information sharing." `http://dradisframework.org/`. Accessed: 2016-07-26.

[28] T. O. Foundation, *OWASP Top 10 - 2013*. The OWASP Foundation, 2013.

[29] "Owasp mobile top ten." `https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks`. Accessed: 2016-04-19.

[30] "Cwe/sans top 25 2011." `http://cwe.mitre.org/top25/`. Accessed: 2016-07-14.

[31] "Wasc threat classification." `http://projects.webappsec.org/w/page/13246927/FrontPage`. Accessed: 2016-07-14.

[32] "Php: Hypertext preprocessor." `http://php.net/`. Accessed: 2016-04-25.

[33] J. Niederauer, *Desenvolvendo Websites com PHP*. Novatec Editora, 2 ed., 2011.

[34] M. S. Silva, *Criando Sites com HTML: Sites de Alta Qualidade com HTML e CSS*. Novatec Editora, 2008.

[35] Y. Fan, "Cascading style sheets," Master's thesis, KEMI-TORNIO UNIVERSITY OF APPLIED SCIENCES TECHNOLOGY, 2010.

[36] M. L. Alexys Leon, *SQL: A Complete Reference*. McGraw-Hill Education (India) Pvt Limited, 1999.

[37] "Javascrip.com." `https://www.javascript.com/`. Accessed: 2016-04-27.

[38] G. Richards, S. Lebresne, B. Burg, and J. Vitek, "An analysis of the dynamic behavior of javascript programs," in *ACM Sigplan Notices*, vol. 45, pp. 1–12, ACM, 2010.

[39] "Netbeans ide." `https://netbeans.org/`. Accessed: 2016-04-28.

[40] "Xampp apache + mariadb + php + perl." `https://www.apachefriends.org`. Accessed: 2016-04-28.

[41] "Highcharts." `http://www.highcharts.com/`. Accessed: 2016-04-28.

[42] "Ckeditor." `http://ckeditor.com/`. Accessed: 2016-04-29.

[43] "Securimage php captcha." `https://www.phpcaptcha.org/`. Accessed: 2016-04-29.

[44] "Php mailer." `https://github.com/PHPMailer/PHPMailer`. Accessed: 2016-04-30.

[45] "Cvss 2.0 calculator." `https://github.com/BitSentinel/CVSS2-Calculator`. Accessed: 2016-04-30.

[46] "Bootstrap." `http://getbootstrap.com/`. Accessed: 2016-04-30.