Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Auditoria de Segurança da Informação em Sistemas e Aplicações

Carlos Magno Bispo Rosal da Cruz

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Orientador
Prof. Dr. André Costa Drummond

Brasília
2017

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

M957a

Magno Bispo Rosal da Cruz, Carlos
Auditoria de Segurança da Informação em Sistemas e Aplicações / Carlos Magno Bispo Rosal da Cruz; orientador André Costa Drummond. -- Brasília, 2017.
68 p.

Dissertação (Mestrado - Mestrado Profissional em Computação Aplicada) -- Universidade de Brasília, 2017.

1. Segurança da Informação. 2. Análise de Vulnerabilidade. 3. Teste de Penetração. 4. Ataque Cibernético. I. Drummond, André Costa, orient. II. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Auditoria de Segurança da Informação em Sistemas e Aplicações

Carlos Magno Bispo Rosal da Cruz

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Prof. Dr. André Costa Drummond (Orientador)
CIC/UnB

Prof. Dr. João José Costa Gondim
CIC/UNB

Dr. Robson de Oliveira Albuquerque
ABIN

Prof. Dr. Marcelo Ladeira
Coordenador do Programa de Pós-graduação em Computação Aplicada

Brasília, 03 de agosto de 2017

Dedicatória

A minha família : Conceição (Mãe), Alessandra (irmã), Katia (irmã), Eduardo (irmão), Alex (irmão) e a minha querida esposa Laura.

Agradecimentos

Agradeço ao meu grande e poderoso Deus e ao seu filho Jesus Cristo, por mais está conquista. Aos professores Dr. André Drummond e Dr. João Gondim pela confiança e apoio. Agradeço aos amigos e colegas de trabalho em especial: Renata Canuto, João Pinheiro, Fabio Tabosa, Glayson, Jeysel, Jefferson e Daniel Pessoa, agradeço a vocês por possibilitarem às pesquisas, pela confiança e apoio. Também a minha esposa Laura pela paciência, compreensão, ajuda e motivação.

Resumo

Os serviços de tecnologia da informação estão cada vez mais presentes nas rotinas diárias de pessoas e instituições, mas com a disponibilidade destes serviços surgem também susceptibilidades a ataques cibernéticos, que podem causar danos e indisponibilidade de sistemas e serviços de tecnologia da informação, prejudicando instituições publicas, organizações e seus usuários. A fim de contribuir de modo proativo com a segurança da informação e segurança cibernética, algumas estrategias podem ser utilizadas, com este intuito, propomos neste trabalho o uso da auditoria de segurança da informação em sistemas e aplicações web, usando o processo de Analise de Vulnerabilidade para identificar, enumerar e classificar vulnerabilidades e seus graus de criticalidade e o Teste de Penetração (*Penetration Testing, Pentest*) para testar a segurança de um sistema, gerando evidências sobre potenciais riscos e consequências provenientes da exploração de uma vulnerabilidades, que poderiam acontecer em casos reais de ataques cibernéticos. A metodologia proposta foi aplicada como estudo de caso, no data center de um órgão publico. Os experimentos foram realizados em sistemas em produção e os resultados obtidos foram apresentados aos gestores responsáveis para possibilitar a mitigação das vulnerabilidades detectadas.

Palavras-chave: Segurança da Informação, Análise de Vulnerabilidade, Teste de Penetração, Ataque Cibernético.

Abstract

Information technology services are increasingly present in the daily routines of people and institutions, but their availability make them susceptible to cyber attacks, which can cause damage and unavailability of information technology systems and services, harming public institutions, organizations and their users. In order to contribute proactively to information security and cyber security, some strategies can be used. For this purpose, we propose in this study the use of security audit in systems and web applications, using the Vulnerability Assessment process to identify, enumerate and classify vulnerabilities and their criticality degrees and Penetration Testing to test and measure the security of a system, generating evidence about the potential risks and consequences of exploiting vulnerabilities that could occur in real-life cyber attacks. The methodology was applied as a case study in systems under production, the experiments were carried out in real systems and applications and the obtained results were presented to the responsible managers to enable the mitigation of detected vulnerabilities.

Keywords: Information Security, Vulnerability Assessment, Penetration Testing, Cyber Attack.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AV Analise de Vulnerabilidade.

CCE Common Configuration Enumeration.

CPE Common Platform Enumeration.

CVE Common Vulnerabilities and Exposures.

CVSS Common Vulnerability Scoring System.

DBMS Data Base Management System.

DOS Denial of Service - ataque de negação de serviço.

HTTP Hyper text Transfer Protocol.

HTTPS Hyper text Tranfer Protocol Secure.

NIST National Institute of Standards and Technology.

NVD National Vulnerability Database.

OVAL Open Vulnerability and Assessment Language.

SCAP Security Content Automation Protocol.

SegCiber Segurança Cibernética.

SI Segurança da informação.

TI Tecnologia da Informação.

XCCDF Extensible Configuration Checklist Description Format.

Capítulo 1

Introdução

À medida que serviços de Tecnologia da Informação surgem nas diversas áreas e ficam disponíveis para melhorar a vida das pessoas, também ficam vulneráveis a ataques cibernéticos das mais diferentes formas [12], estes ataques visam causar a indisponibilidade de sistemas e serviços, obter acesso não autorizado, roubo de dados, propagação de códigos maliciosos em rede de computadores, desfiguração (*defacement*) - modificação da página de um site [13] e podem causar prejuízo para: pessoas, empresas, entidades governamentais e demais usuários que utilizam serviços de tecnologia da informação.

Nesse contexto, as organizações buscam constantemente formas de proteção contra ataques cibernéticos, como ferramentas de segurança, exemplos:

- Firewall é um filtro de pacotes que é colocado em um ponto entre uma rede privada de computadores e a Internet [14]. O firewall intercepta cada pacote que é trocado entre a rede privada e a Internet, examina os campos dos cabeçalhos de pacotes e toma uma decisão de descartar o pacote ou aceitá-lo de acordo com um conjunto definido de regras de segurança [15].

- Sistema Detector de Intrusão (IDS) é um software ou tecnologia de hardware que automatiza o processo de monitoramento e análise de eventos, estes eventos devem ser analisados para verificar se são sinais de intrusão [16] [17]. Intrusão é qualquer conjunto de ações que fazem tentativas no intuito de comprometer a integridade, confidencialidade ou disponibilidade de um recurso. Sistema Detector de Intrusão (IDS) é um software ou tecnologia de hardware que automatiza o processo de monitoramento e análise de eventos, estes eventos devem ser analisados para verificar se são possíveis sinais de intrusão [16] [17]. Intrusão é qualquer conjunto de ações que fazem tentativas com intuito de comprometer a integridade, confidencialidade ou disponibilidade de um recurso.

- Registro de Log é uma tecnologia de Segurança da informação (SI), usada para armazenar informações sobre determinadas ações ou eventos, o objetivo do registro é gerar trilhas de auditoria que possam ser rastreadas ou consultadas.

Mas nem sempre ferramentas como essas são suficientes para minimizar os riscos e possíveis consequências provocadas por ataques cibernéticos, principalmente em casos de exploração de vulnerabilidades (falha existente em um software, que quando descoberta, pode possibilitar a exploração desta permitindo um acesso não autorizado) [5]. Por isso a adoção de outros mecanismos são importantes para prevenção, mitigação ou minoração de ataques, esses mecanismos podem ser: uso de boas praticas (ISOs, Normas e politicas de segurança) e estrategias proativas como detecção de vulnerabilidades e testes de segurança.

Os testes de segurança são fundamentais para verificar se os mecanismos de proteção incorporados a um sistema vão de fato protegê-lo contra acesso indevido [18], pois qualquer software ou sistema de computador que trabalhe com informações sensíveis ou que cause ações que possam inadequadamente prejudicar ou beneficiar indivíduos, é um alvo para acesso impróprio ou ilegal. Os testes podem ser aplicados em formas de metodologias e técnicas especificas para verificação de vulnerabilidade e exploração destas. Neste trabalho propormos o uso da auditoria de sistemas e aplicações utilizando os processos de Analise de Vulnerabilidade e Teste de Penetração (*Penetration Testing*, Pentest) também foi aplicado a metodologia do Guia Técnico de Segurança da Informação Testes e Avaliação [19] (*Technical Guide to Information Security Testing and Assessment*) de forma adaptada. Este trabalho pode ser aplicado em varias categorias de software, as principais categorias de software são apresentadas na Tabela 1.1 [6], dentre as categorias apresentadas, os sistemas web e aplicações web são o escopo deste trabalho.

Tabela 1.1: Categorias de Software [6]

Categoria	Descrição
Sistema desktop e web	Coleção de programas escritos para servir outros programas. Exemplos: compiladores, editores e utilitários para gerenciamento de arquivos, componentes de sistema operacional, drivers, software de rede etc. Sistemas web são sistemas disponíveis na internet acessados por meio de um *browser*.
Aplicação	Consiste de programas isolados que resolvem uma necessidades específica do negócio.
Site web	Páginas da web recuperadas por um *browser*, podendo ser dinâmicas ou estáticas.
Aplicação web	Também chamada de Web App, é centralizada em redes e pode ser usada em uma vasta gama de aplicações, desde um conjunto de arquivos de hipertexto interconectados que apresentam informações por meio de texto e gráficas até sofisticados ambientes computacionais integrados a bancos de dados corporativos.
Legado	São softwares antigos, com pouca ou sem manutenção corretiva ou evolutiva.

1.1 Objeto de estudo

A proposta deste trabalho foi aplicada como estudo de caso no data center de uma unidade governamental, os experimentos e a coleta de dados foram autorizados pelos gestores do data center. Os serviços prestados por este órgão publico são:

1. Hospedagem de sistemas e portais governamentais,
2. Serviços de atendimento e suporte a usuários e unidades governamentais.

1.2 Problema

Detectar e identificar vulnerabilidades em sistemas e aplicações para possibilitar correções, testar a segurança de sistemas e aplicações para avaliar possíveis riscos e consequências provocados por ataques cibernéticos.

1.3 Justificativa

Segundo a Diretoria de segurança da informação da unidade governamental, seu data center tem sido alvo de ataques cibernéticos constantes, estes ataques visam causar: a

indisponibilidade de serviços, acesso não autorizado, propagação de códigos maliciosos em rede, modificação de paginas de web site (desfiguração), protestos, críticas ao governo, alusão a cibercriminosos, ou obtenção de vantagens ilícitas, o que e pode prejudicar os serviços prestados a população e órgãos públicos. Quando esses ataques ocorrem a equipe de segurança da informação tem que identificar e classificar o tipo do ataque cibernético ocorrido para que possibilite a realização de medidas preventivas e ou corretivas de uma vulnerabilidade, porém não existe um processo estabelecido para fazer analises ou verificações para descobrir quais aplicações ou sistemas estão vulneráveis, o mapeamento de vulnerabilidades é fundamental para definição de prioridades e implementação de pontos de controle no planejamento de segurança da informação [20].

1.4 Objetivos

1.4.1 Objetivo Geral

Demonstrar os benefícios da Auditaria de Segurança da Informação, para realizar a detecção de vulnerabilidades e testes de segurança em sistemas e aplicações.

1.4.2 Objetivos Específicos

- Realizar uma revisão da literatura com os temas utilizados na pesquisa para demonstrar potenciais benefícios do uso de Auditoria de Segurança da Informação e aplicar metodologia especifica, para realização da Auditoria de Segurança da Informação em Aplicações e Sistemas.
- Identificar, enumerar e classificar as vulnerabilidades detectadas em sistema e aplicações.
- Realizar testes de segurança para avaliar riscos e atestar a criticidade de uma vulnerabilidade.
- Apresentar os resultados da Auditoria de Segurança da Informação em Aplicações e Sistemas para auxiliar na tomada de decisão e correções das vulnerabilidades detectadas;

1.5 Organização do Trabalho

Este trabalho está divido em cinco capítulos. O primeiro capitulo apresenta: o contexto da pesquisa, problema, justificativa e os objetivos do trabalho; o segundo capitulo aborda a

revisão de literatura que visa identificar o estado da arte sobre Analise de vulnerabilidade e Teste de penetração (*Penetration testing*, Pentest), também aborda os seguintes tópicos complementares: segurança da informação e taxonomia de ataques web, e ainda traz as principais fontes de referências utilizadas, no desenvolvimento deste trabalho; o terceiro capitulo apresenta a metodologia utilizada neste trabalho; a quarto capitulo apresenta a aplicação da Auditoria de Segurança da Informação, que foi realizada com o estudo de três casos que são sistemas governamentais; o quinto capitulo apresenta a conclusão do trabalho.

Capítulo 2

Revisão do Estado da Arte

Na revisão da literatura a seguir buscou-se destacar conceitos de Segurança da informação, Ataque cibernético, Analise de Vulnerabilidade e Teste de penetração (*Penetration Testing*, Pentest). A revisão contemplou ainda Taxonomias de ataques cibernéticos e metodologia para Pentest.

2.1 Segurança da Informação

A Segurança da Informação visa a proteção dos dados e informações de indivíduos ou organizações e para isso possui propriedades como: confidencialidade, integridade e disponibilidade, outras propriedades também podem ser consideradas importantes [21] como: Autenticidade e não repudio a ISO/IEC 2700(2014) [22], define essas cinco propriedades da segurança da informação:

- Confidencialidade (*Confidentiality*) - a informação não é disponibilizada ou divulgada a indivíduos, entidades ou processos não autorizados;
- Integridade (*Integrity*) - propriedade que garante a exatidão, corretude e completude.
- Disponibilidade (*Availability*) - propriedade de estar acessível e utilizável sob demanda de uma entidade autorizada;
- Autenticidade (*Authenticity*) - propriedade de que uma entidade é o que a mesma diz ser;
- Não repúdio (*Non-repudiation*) - capacidade de comprovar a ocorrência de uma reivindicação de um evento ou ação e suas entidades originárias, quando uma mensagem é enviada, o destinatário pode provar que está realmente foi enviada por determina origem ou o processo inverso.

No intuito de prover ou melhorar a segurança da informação, surgiram referências normativas importantes [23], como a família 27000 da *International Organization for Standardization* (ISO) - que são específicas para gestão da segurança da informação e foram adotadas pela Associação Brasileira de Normas Técnicas (ABNT). A norma ABNT ISO/IEC 27001(2013) [24] - é uma tradução da ISO 27001(2013) - está norma especifica os requisitos para o estabelecimento, implementação, manutenção e melhoria contínua de um sistema de gestão da segurança da informação no contexto da organização [25]; a norma ABNT ISO/IEC 27002(2013) [26] - é uma tradução da ISO 27002(2013) [27], cujo o objetivo é fornecer diretrizes para padrões de segurança de informações organizacionais e práticas de gerenciamento de segurança de informações, incluindo: a seleção, implementação e gerenciamento de controles levando em consideração o ambiente de risco de segurança da informação da organização [25].

As orientações fornecidas por estas famílias de ISOs possibilitam uma organização alcançar a segurança da informação adequada possibilitando até obtenção de certificação em Sistema de Gestão de segurança da informação (*Information Security Management System - ISMS*) [28]. A Segurança Cibernética (SegCiber) é uma coleção de ferramentas, políticas, Conceitos, diretrizes de segurança, gerenciamento de riscos, ações e melhores práticas que podem ser utilizadas para proteger um *cyber* espaço (espaço cibernético), este espaço cibernético corresponde a ambientes compostos por ativos como: computadores em rede, internet, armazenamento, infra-estrutura, serviços de Tecnologia da Informação (TI), telecomunicações entre outros. Para [29] o objetivo da Segurança da informação (SI) é garantir a continuidade dos negócios e minimizar os danos aos negócios, limitando os incidentes de segurança.

2.2 Taxonomia de ataques Web

Com o crescente uso da internet pela sociedade para diversos fins como: comercio, informação, entretenimento, educação entre outros. Também aumenta paralelamente os ataques cibernéticos direcionados a internet, com base neste contexto um estudo elaborado por [1], propõe uma taxonomia para ataques web, esses ataques cibernético exploram os protocolos *Hyper text Transfer Protocol* (HTTP) - protocolo de transferência de hipertextos e *Hyper text Tranfer Protocol Secure* (HTTPS) - protocole de transferência de hipertexto Seguro [30], estes protocolos são usados por sistemas web. Segundo [1] a classificação tem o intuito de auxiliar a compreensão destes ataques e a construção de aplicações mais seguras.

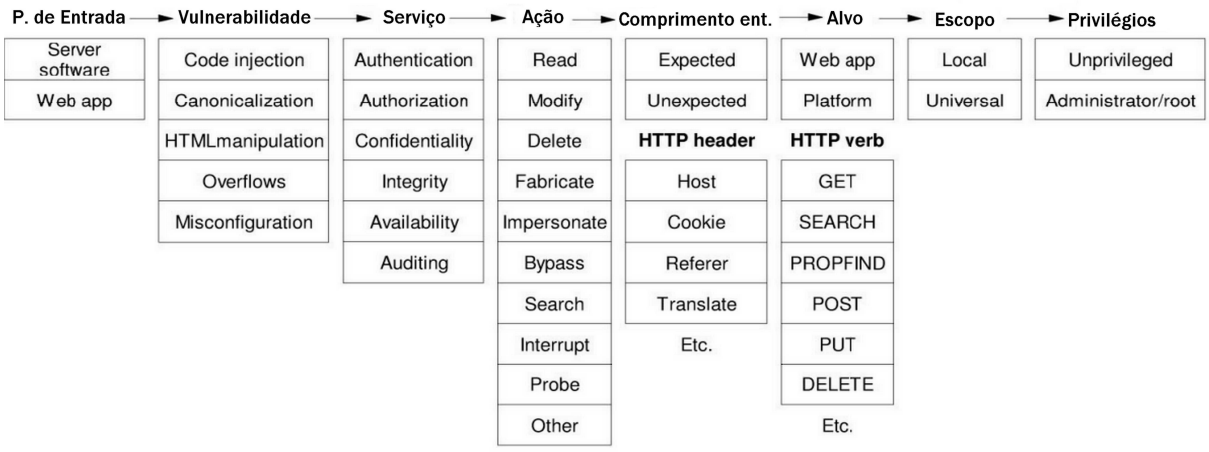

Figura 2.1: Taxonomia de ataques Web (adaptada [1]).

1. Ponto de entrada: onde o ataque passa.

    - *Server Software Web* (Software do servidor web): independentemente da plataforma ou do fabricante, os servidores web podem esconder involuntariamente uma série de vulnerabilidades, que por sua vez, também podem ser exploradas.
    - *Web Aplication Attacks* (Ataque em Aplicações web): referem-se a exploração das vulnerabilidades inerentes ao código de um aplicativo da web em si, independentemente da tecnologia na qual ele é implementado ou a segurança do banco de dados do servidor web no qual ele é construído [31], a origem destas vulnerabilidades podem ser erros em formulários HTML e scripts lado cliente (codificação visível no navegador) ou scripts do lado do servidor (ASP, JSP, PHP e etc).

2. Vulnerabilidade: uma fraqueza em um sistema que permite ação não autorizada.

    - *Code Injection* (Injeção de Código): permitem a injeção arbitrária de código malicioso em paginas web, essas vulnerabilidades surgem de falhas em rotinas de validação de entrada no lado do servidor, exemplo: XSS (*cross site scripting*) e *SQL Injection*.
    - *Canonicalization* (Canonização): é uma vulnerabilidade que pode ocorrer quando um aplicativo faz uma decisão baseada em nome (nome de arquivo, nome de pasta, endereço Web), porém sem levar em conta o fato de que o nome poderia ser expresso de varias maneiras [32], a maneira mais comum de explorar as questões de canonização é sob a forma de travessia de diretório, que permite um usuário sem o devido privilegio visualize dados fora do caminho de destino autentico.
    - *HTML manipulation* (Manipulação de HTML): é uma vulnerabilidade que permite a um usuário mal intencionado, modificar dados enviados entre o cliente (web browser) e o servidor (aplicação web). Exemplos: Inserção de *scripts* ou *querys* em URLs e formulários.
    - *Buffer Overflows*: ataque conhecido como estouro de Buffer, é quando um programa grava além dos limites de um *buffer* de tamanho fixo previamente alocado na memória, isto é chamado um estouro de buffer. Consiste em ler ou escrever, ultrapassando o tamanho total de um *buffer*, este ataque pode causar comportamentos anormais em programas ou até levar a falha completa.
    - *Misconfiguration* (Configuração Incorreta): se o servidor ou plataforma de uma aplicação Web, não estiverem configurados corretamente (atribuindo permissões de arquivos, caminho diretório e etc), podem ocasionar o surgimento de outras vulnerabilidades que levariam a exploração do servidor web comprometido.

3. Serviço: ameaçado pelo ataque.

    - *Authentication* (Autenticação): é um processo usado para determinar se alguém ou algo, é na verdade quem ou o que ele diz ser. Autenticação é comumente realizada através do uso de senhas. Exemplo de ataques prejudiciais para autenticação: fixação de sessão [33], falsificação de identidade, roubo de credenciais válidas e força bruta.
    - *Authorization* (Autorização): tem como objetivo permitir que usuários autenticados tenham os devidos direitos, para fazer algo ou acessar alguma coisa, como: acesso a arquivos, pastas, módulos de sistema. Exemplos de ataques

de autorização mais comuns que ignoram o controle de acesso: *buffer Overflows* (estouro de buffer), visto no item 2 (vulnerabilidade) desta classificação; o ataque de escalada de privilégios (*bypass*), que pode resultar no aumento não autorizado no domínio do acesso (aumento de privilegio de acesso), podendo um atacante obter privilégios de administrador ou *root*..

- *Confidentiality* (Confidencialidade): o objetivo é proteger a informação de ser divulgada ou revelada, a entidades não autorizadas a ter acessas a essas informações. Os ataques de confidencialidade mostram informações privadas, dependendo da vulnerabilidade um cibercriminoso poderia explora-lá para obter informações. Exemplo: ataques de Injeção de código visto no item 2 (vulnerabilidade) desta classificação.
- *Integrity* (Integridade): tem o objetivo de resguardar dados ou informações contra manipulação acidental ou maliciosa, para evitar a modificação indesejada. Os ataques comuns contra a integridade de dados incluem: manipulação de registros de banco de dados e o *Defacement* - que é um ataque que faz alteração em uma página web ou desfiguração.
- *Availability* (Disponibilidade): tem por objetivo garantir que um determinado ativo como: serviços de TI, sistemas ou acesso a internet estejam disponíveis para uso, por seus utilizadores. Exemplo de ataque que pode causar indisponibilidade: Denial of Service - ataque de negação de serviço (DOS).
- *Auditing* (Auditoria): proporciona ao administrador do sistema registar informações relevantes para a segurança, para que possam ser analisadas em verificações de possíveis violações.

4. Ação: ataque real contra o servidor web.

    - *Read* (Leitura): é uma ação para obter o conteúdo dos dados contidos em um arquivo, registro de banco de dados ou qualquer outro suporte de dados, armazenados no servidor web. A leitura não altera a integridade dos dados lidos. Exemplos incluem a visualização de código-fonte.
    - *Modify* (Modificar): é uma ação para alterar dados. Exemplo: alteração de um registro de banco de dados ou alteração do conteúdo de um arquivo.
    - Delete (Exclusão): é uma ação para remover ou apagar. Exemplo: excluir objetos de um banco de dados ou arquivos.
    - *Fabricate* (Fabricar): é uma ação para inserir objetos falsificados em um sistema. Exemplo: adicionar kits de ferramentas de hackers ao sistema de arqui-

vos de um servidor, criando novas contas ou inserindo registros em uma tabela de banco de dados.

- *Impersonate* (Representar): é uma ação para mascarar um usuário ilegal como legítimo. Exemplo: fraudes de autenticação.
- *Bypass* : ataque de escalada de privilégios, pode resultar em acesso não autorizado no domínio do acesso, podendo um atacante obter privilégios de administrador ou *root*.
- *Search* (Pesquisa): é uma ação para encontrar informações válidas de autenticação do usuário. Exemplo: ataques de força bruta, que tentam combinações diferentes de login e senha aos pares ou repetidamente, visando conseguir autenticação como um usuário valido.
- Interrupt (Interromper): é uma ação para fazer com que um servidor pare de operar ou oferecer um serviço. Exemplo: ataque Denial of Service - ataque de negação de serviço (DOS).
- *Probe* (Sonda): é uma tentativa de determinar as características ou vulnerabilidades de um alvo específico. Os atacantes geralmente começam coletando informações dos alvos, antes de entrar em atividades mais invasivas. Por exemplo: o uso de cabeçalhos HTTP incomuns, como *HEAD*, ou ferramentas de espelhamento de sites.
- *Other* (Outros): quando um invasor executa o código, mas é impossível determinar, qual é o comando executado, consideramos a ação executada como outra. Exemplo: estouro de *buffer* com uma carga de execução de comando, geralmente essas ações são consideradas uma ameaça contra a autorização, porque embora o resultado da execução não seja conhecido, assumimos que não é autorizado.

5. Comprimento dos argumentos passados para a solicitação HTTP. Os ataques de *buffer overflows*, são a forma mais comum de ataque de segurança hoje, não apenas nas aplicações web, mas também em todos os aplicativos da internet e serviços. Eles são os mais fáceis de explorar e podem causam consequências, como falha de um sistema. Os estouros de *buffer* geralmente precisam de dados de entrada muito longos para trabalhar. Assim, com base no comprimento do *String* de ataque, distinguimos entre: *Common lengt* (Comprimento Comum) e *Unusually long* (Excepcionalmente Longos):

   - *Common length* (Comprimento Comum): são ataques que não são baseados em explorar um *buffer overflow*, raramente apresentam argumentos de um com-

primento superior a um certo limite, que pode ser definido experimentalmente para um servidor web específico. Assim a maioria dos ataques se encaixam nesta categoria.

- *Unusually long* (Excepcionalmente Longo): os estouros de *buffer* requerem argumentos de comprimentos suficientes, para encher um buffer alocado em memória, para uma determinada entrada, juntamente com alguns dados adicionais, que são escritos na memória fora do *buffer*, quando esses dados adicionais estão faltando, o estouro do *buffer* geralmente resulta em um Denial of Service - ataque de negação de serviço (DOS).

6. HTTP verb, segundo a taxonomia proposta por [1], um ataque web vai usar o protocolo HTTP ou HTTPS. Uma solicitação HTTP, consiste em um verbo e um grupo de cabeçalhos. O verbo pode ser qualquer um dos métodos do protocolo HTTP: *GET, POST, HEAD, SEARCH, PROPFIND, PUT, DELETE* e etc.

7. HTTP Header, ataques diferentes, usam diferentes cabeçalhos e está categoria não é mutuamente exclusiva, mas [1], considera-lhe válida, para ser incluída em sua taxonomia, pois para ele, está fornece informações úteis. Cabeçalhos mais comum visto em ataques são: *Host, Cookie, Referer, Translate* e etc.

8. Alvo: objeto do ataque,

   - *Web Application* (Aplicação Web): este alvo recebe ataques normalmente destinados a páginas da Web, para: obtenção e ou modificação do código fonte de um site, captura de *cookies*, roubo de senhas, visualização, alteração ou exclusão de informações de registros em banco de dados.
   - *Platform* (Plataforma) : neste caso, o alvo está além da aplicação Web. É visado a Plataforma, que hospeda uma aplicação ou sistema. O invasor geralmente busca conseguir acesso não autorizado, após a execução de comandos arbitrários como: manipulação de contas de uma máquina, verificação de serviços para obter informações e o uso de *exploits* para explorar vulnerabilidades. No contexto deste ataque, um servidor Web, como exemplo : Apache, JBOSS, podem ser usados para exploração da aplicação ou até mesmo como um meio para acessar a rede interna de uma Organização.

9. Escopo: local ou abrangência do ataque no servidor web.

   - Local - neste ataque o escopo atinge somente um usuário, ou um pequeno grupo de usuários. Um exemplo, é o roubo de dados pessoais de um usuário, por meio de um ataque de *cross-site scripting*.

- Universal - no escopo deste ataque todos os usuários serão afetados, geralmente os ataques universais buscam: desfigurar pagina uma web (*defacement*), manipulação de registros de banco de dados e Denial of Service - ataque de negação de serviço (DOS).

10. Privilégios: obtidos pelo atacante após a conclusão de um ataque bem sucedido.

    - *Unprivileged User* (Usuário sem Privilégios): quando um invasor obtêm acesso de um usuário com poucos privilégios, os ataques são executados sob a identidade deste usuário. Esses usuários geralmente são de: aplicação Web, banco de dados e ou usuário de sistema. O invasor só pode acessar os mesmos recursos, permitidos ao usuário do qual ele obteve o acesso, por consequência o impacto do ataque é limitado.
    - *Root* (Administrador): se um atacante conseguir ganhar acesso com privilégios de Administrador (*Root*) - que detêm privilégios máximos em um sistema, a maquina atacada é completamente comprometida. Este é o mais alto nível de realização de um ataque.

2.3 Analise de Vulnerabilidade

Para [5], vulnerabilidade é uma falha em um sistema, e um cibercriminoso pode executar varreduras para localizar essa falha a fim de explora-la. Para [34] vulnerabilidade de software é a instância de um erro, que pode ocorrer na especificação, no desenvolvimento, ou configuração de um software, de tal maneira que sua execução pode violar diretivas de segurança explícita ou implícita. Neste contexto podemos chegar ao seguinte raciocínio se duas instâncias diferentes do mesmo erro podem ser vulnerabilidades diferentes ou não, essa resposta advêm do processo de Analise de Vulnerabilidade (AV). De acordo com [11] Analise de Vulnerabilidade - é o processo de analise de uma vulnerabilidade descoberta, para identificar características, propriedades, causas e riscos envolvidos.

2.3.1 Processo de Analise de Vulnerabilidade

O processo de Analise de Vulnerabilidade (AV) de um software crítico é repetidamente descrito, como uma das lacunas em estratégias nacionais e internacionais de segurança cibernética [35], neste processo de AV, um analista de segurança visa encontrar informações cruciais sobre o alvo testado para detectar e identificar vulnerabilidades [36]. A Analise de Vulnerabilidade (AV) é uma estratégia proativa de segurança cibernética [5].

Especialistas buscam formas de controle, para bloquear ataques cibernéticos e uma delas é a "Analise de Vulnerabilidade (AV) em conjunto com a remediação das vulnerabilidades detectadas " [37]. Para [5] a Analise de Vulnerabilidade (AV), propicia a correção proativa e mitigação das vulnerabilidades conhecidas em sistemas e ajuda a reduzir às oportunidades para exploração. Segundo [38] outro processo que auxilia a AV é o **gerenciamento de vulnerabilidades** que pode ter os seguintes ciclos no processo de gerenciamento:

- Varredura de Vulnerabilidade (*Vulnerability Scan*): processo usado para identificar as vulnerabilidades que afetam ativos do usuário, ou seja, produtos e sistemas.
- Análise de Risco (*Risk Analysis*): processo usado para estimar o impacto e a probabilidade de explorações de vulnerabilidade. O impacto é calculado a partir da gravidade de uma vulnerabilidade, número de sistemas afetados e a importância deles.
- Planejamento (*Planning*): processo usado para decidir às medidas, ou seja, decidir que ações devem ser tomadas para cada risco.
- Remediação (*Remediation*): processo usado para implementar contramedidas.

Analise de vulnerabilidade de segurança é o processo para detectar, identificar, avaliar e reportar para correção uma vulnerabilidade detectada em um sistema [2]. A Figura 2.2 apresenta as etapas do processo de AV, segundo [2]. Abaixo a descrição de cada etapa:

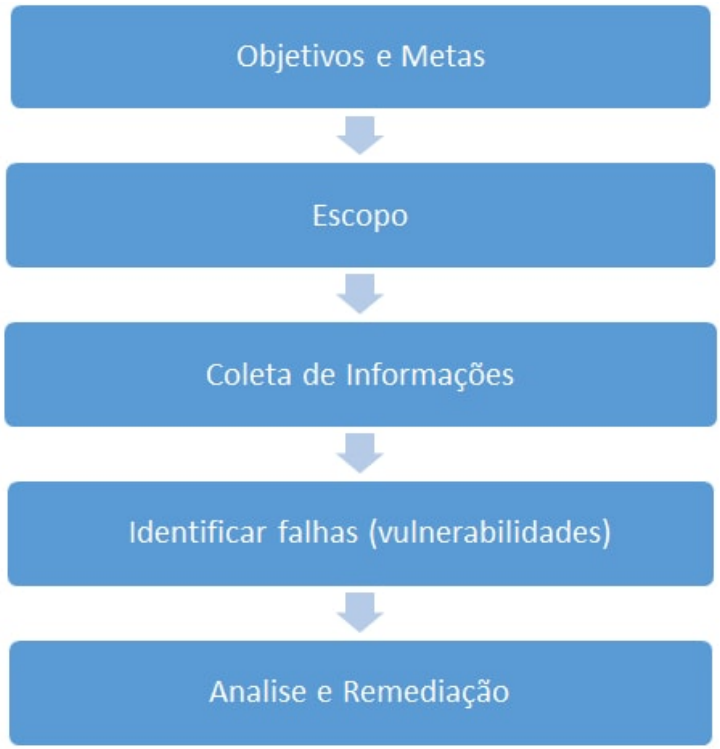

Figura 2.2: Etapas da Analise de Vulnerabilidade (adaptada [2]).

- Objetivos e metas é a etapa em que é definido qual o objetivo ou meta do processo de AV, por exemplo: usar o processo de AV para verificar se uma determinada aplicação possui vulnerabilidades.
- Escopo define a abrangência da AV, sobre um determinado alvo podendo ser total ou parcial. Exemplo: módulos de um sistema. Também pode-se definir se o escopo será: uma aplicação, um sistema ou rede.
- Coleta de informações é a principal tarefa para obter as informações mais detalhadas de um sistema, busca obter dados e informações relacionados infraestruturas de TI como redes, ambiente de desenvolvimento, linguagem de programação, versão do sistema operacional, componentes da aplicação etc.
- Identificar falhas (vulnerabilidades), está etapa busca detectar possíveis vulnerabilidades em um alvo, é realizada principalmente com analise intelectual (humana) e o uso de ferramentas de varreduras (*vulnerability scans*).

- Analise e remediação é a etapa que analisa as vulnerabilidades detectadas e verifica o grau de risco destas. Após a analise são geradas informações (relatório), para que possibilite a correção de vulnerabilidades, visando a prevenção contra ataques cibernéticos.

2.3.2 Repositórios e padrões

Vulnerabilidades de software são as ameaças de segurança criticas, para sistemas de software em rede, as vulnerabilidades encontradas são localizadas por mecanismos buscadores que utilizam relatórios comuns, disponíveis em repositórios abertos [39]. Os principais repositórios que armazenam informações sobre vulnerabilidades são: *National Vulnerability Database* (NVD) [40], mantido pelo Instituto Nacional de Padrões e Tecnologia (*National Institute of Standards and Technology*, NIST) - é uma agência governamental não regulatória da administração de tecnologia do Departamento de Comércio dos Estados Unidos; *Common Vulnerabilities and Exposures* (CVE) [41], mantido pelo MITRE *Corporation*, que é uma organização sem fins lucrativos e opera múltiplas pesquisas financiadas pelo governo federal dos Estados Unidos; *Open Vulnerability and Assessment Language* (OVAL) [42], mantido também pelo MITRE. Estes repositórios são usados no processo de AV e também são consultados pela comunidade interessada e também por ferramentas que são alimentadas com informações contidas nos mesmos.

Existem diversos padrões e repositórios disponíveis para utilização no processo de AV, gerenciamento de risco e correções de segurança, e por isso pode ocorrer divergência de informações ou falta de padronização, além também da possibilidade da realização da pesquisa, ser de forma: **manual** ou **automatizada**, podendo apresentar resultados não padronizados. Visando superar essas deficiências e padronizar o processo de AV, o NIST desenvolveu o *Security Content Automation Protocol* (SCAP)[7], que é um conjunto de especificações que padroniza a nomenclatura e o formato pelo qual produtos de segurança comunicam informações sobre: identificação de software, vulnerabilidades de software e configurações de segurança. Também oferece suporte para: protocolo multiuso, verificação de vulnerabilidade automatizada, controle técnico de atividades de conformidade e medições de segurança [7] [43]. A Tabela 2.1 apresenta os componentes do SCAP:

Tabela 2.1: Componentes SCAP adaptado de [7]

Categoria	Especificação	Definição
Linguagem	*Extensible Configuration Checklist Description Format* (XCCDF) [44].	Uma especificação XML para: coleções estruturadas de configurações de segurança, regras usadas por sistemas operacionais e plataformas de aplicativos.
Linguagem	*Open Vulnerability and Assessment Language* (OVAL) [42].	Uma especificação XML para troca de detalhes técnicos, sobre como verificar falhas de software relacionadas com: segurança, problemas de configuração e correções.
Enumeração	*Common Platform Enumeratio* (CPE) [45].	Uma convenção de nomenclatura para hardware, sistema operacional e classes de aplicações.
Enumeração	*Common Configuration Enumeration* (CCE) [46].	Um dicionário de nomes para, problemas de configuração de segurança de software, tais como: controle de acesso e configurações de diretiva de senha.
Vulnerabilidade	*Common Vulnerabilities And Exposures* (CVE) [41].	Um dicionário de nomes para vulnerabilidades de software relacionadas com segurança,sendo essas vulnerabilidades conhecidas publicamente.
Vulnerabilidade	*Common Vulnerability Scoring System* (CVSS) [8].	Um método para a classificação de características de vulnerabilidades de software e atribuição de escores (pontuação) de gravidade, baseada nas características das vulnerabilidades classificadas.

Conforme mostrado na tabela Tabela 2.1, o *Security Content Automation Protocol* (SCAP) é agrupado em três categorias [7], essas categorias são apresentadas abaixo:

1. Linguagem: é usada para especificar listas de verificações, gerar relatórios da lista de verificação e também especifica procedimento de teste.
2. Enumeração: possui nomenclaturas e dicionários de segurança da informação.
3. Vulnerabilidade: mede características e geram resultados de pontuação (escore) de risco de uma vulnerabilidade. A pontuação é atribuída ao grau de risco, que representa determinada vulnerabilidade de acordo com o risco e características mensuradas.

Pontuação (Escore) do grau de Severidade de uma vulnerabilidade

Outra taxonomia importante é a classificação de vulnerabilidades, por suas severidades o *Common Vulnerability Scoring System* (CVSS) (mais detalhes sobres os padrões ver Tabela 2.1), fornece um padrão para classificação das vulnerabilidade por gravidade de risco [39], atribuindo valores de 1 a 10, quanto maior o numero, maior é a gravidade [8]. O CVSS usa três grupos de métricas em suas classificação que são:

1. **Base:** representa as características intrínsecas e fundamentais de uma vulnerabilidade que são constantes ao longo do tempo e dos ambientes do usuário.
2. **Temporal:** representa as características de uma vulnerabilidade que muda ao longo do tempo, mas não entre os ambientes do usuário.
3. **Ambiental:** representa as características de uma vulnerabilidade que são relevantes e exclusivas para o ambiente de um determinado usuário.

A tabela Tabela 2.2 apresenta a classificação do grau de severidade baseado no CVSS Versão 2 [8] e também descreve possíveis consequências em caso de exploração [9][39].

Tabela 2.2: Classificação de vulnerabilidade por gravidade, CVSS V.2 [8] [9].

Gravidade	Escala de pontuação	Consequência
Alta	7 até 10	A exploração de uma vulnerabilidade desta classificação, tende ser explorada de forma uma pouco mais fácil, por um cibercriminoso remoto ou local, isso torna possível a violação da proteção de segurança de um sistema e um cibercriminoso poderia conseguir acesso de usuários comuns, até privilégios de Administrador de sistema como *Root*, o que permitiria controle total sobre um sistema. Em alguns casos a exploração de vulnerabilidades desta classificação, podem requerer mais complexidade no ataque e o sucesso deste pode ocasionar danos críticos. **Motivação para cibercriminosos: Altamente motivados. Valor para cibercriminosos: Alto.**
Media	4 até 6,99	Essa classificação é dada a falhas que podem ser mais difíceis de explorar, mas que ainda podem levar a algum comprometimento da confidencialidade, integridade ou disponibilidade de recursos, em determinadas circunstâncias. Estes são os tipos de vulnerabilidades que poderiam ter tido um impacto crítico ou impacto importante, mas são menos fáceis de serem exploradas com base em uma avaliação técnica da falha, ou afetam configurações improváveis [47]. **Motivação para cibercriminosos: gera interesse moderado. Valor para cibercriminosos: Médio.**
Baixa	1 até 3,99	A vulnerabilidade geralmente não render valiosas informações ou controle sobre um sistema, mas fornece ao invasor informações, que podem ajudá-lo a encontrar e explorar outras vulnerabilidades. **Motivação para cibercriminosos: Fraca. Valor para cibercriminosos: Baixo.**

Neste momento (escrita do trabalho) existe uma nova versão do CVSS, que é a versão 3. A comunidade de segurança, os fabricantes e organizações, estão em processo de migração da versão 2 para versão 3. Nesta nova versão, umas das principais mudanças foi a inserção de um novo nível que é o "Critico". Abaixo a nova disposição da classificação de gravidade das vulnerabilidades do CVSS versão 3 [40]:

- Vulnerabilidades classificadas como "Baixa" gravidade, são aquelas que têm pontuação base do CVSS v3 **de 0,0 até 3,9**.
- Vulnerabilidades classificadas como "Média" gravidade, são aquelas que têm pontuação base do CVSS v3 **de 4,0 até 6,9**.
- Vulnerabilidades classificadas como "Alta" gravidade, são aquelas que têm pontuação base do CVSS v3 **de 7,0 até 8,9**.
- Vulnerabilidades classificadas como "Critica" gravidade, são aquelas que têm pontuação base do CVSS v3 **de 9,0 até 10,0**.

2.4 Classificação das Técnicas de Varreduras de vulnerabilidade

Varredura de vulnerabilidade refere-se a tarefa de sondar redes corporativas ou serviços de internet, coletando informações, procurando vulnerabilidades ou formas para se infiltrar em ativos de TI. As atividades de varreduras, são usadas por cibercriminosos antes de lançar um ataque cibernético, por isso é de suma importância a investigação e adoção de métodos de detecção de varreduras de vulnerabilidades. Com intuito de fornecer uma classificação única, para técnicas para realização de varreduras e atualizar o ultimo estudo publicado em 2011 [48], foi proposto por [3], uma classificação de 19 (dezenove) técnicas, agrupadas em 5 (cinco) categorias, de acordo com suas características. A Figura 2.3 apresenta a classificação das técnicas de varredura de vulnerabilidade.

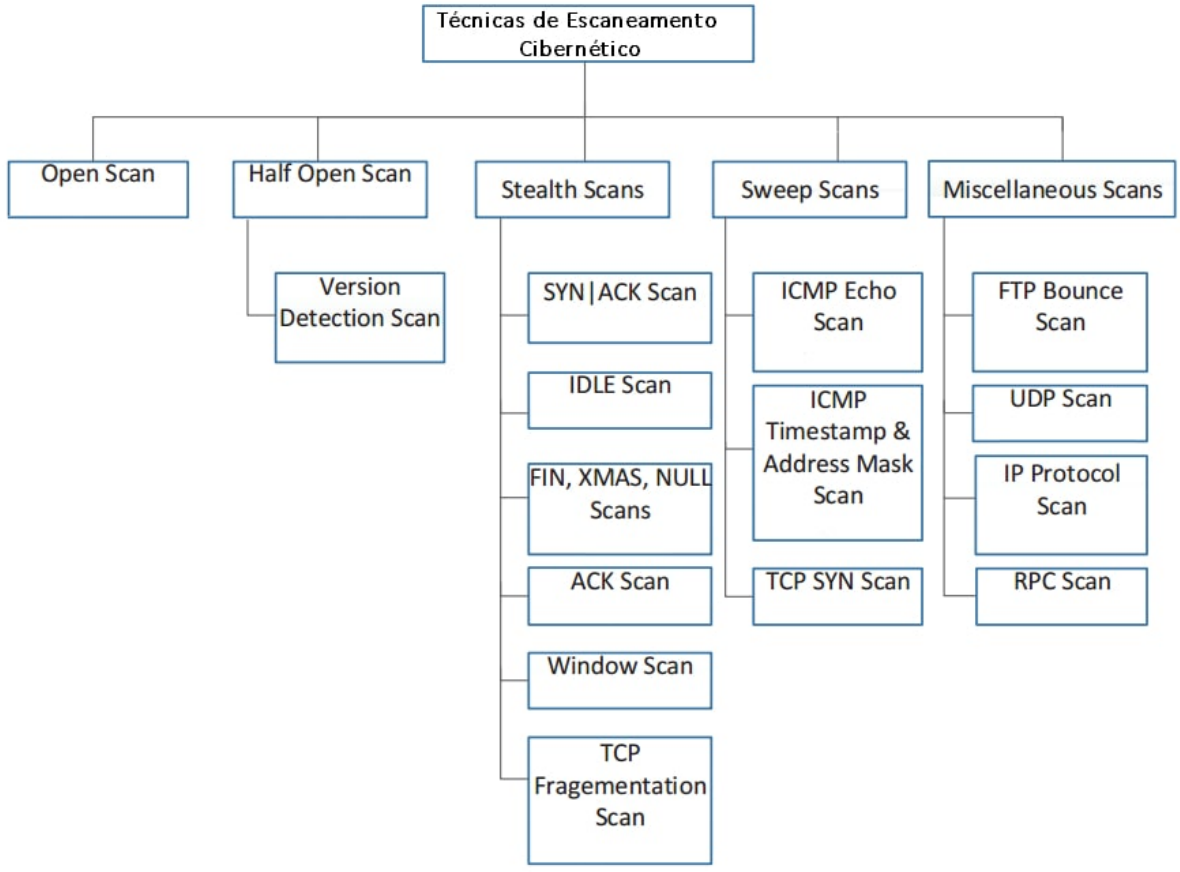

Figura 2.3: Classificação das técnicas para varredura de vulnerabilidade (adaptada [3]).

2.5 Teste de Penetração (*Penetration testing*)

O Teste de Penetração (*Penetration testing,* Pentest) segundo [2] é um processo que realiza tentativas de intrusão ou ataques cibernéticos a sistemas e aplicações ou até mesmo a rede de computadores, para avaliar a segurança de um determinado alvo. Segundo [4] o Pentest é um teste de segurança que realiza ataques e ou tentativas de invasão a sistemas, ou seja, o teste avalia a segurança de um sistema, realizando ataques de forma controlada e com autorização para avaliar se ataques reais seriam bem sucedidos. Para [49], o Pentest é uma analise de aspectos de falhas de segurança de um sistema, onde esses aspectos são acordados entres o demandante e o analista testador, os demandantes podem dizem aos analistas testadores o que esperam e ou determinar o que eles querem que seja testado, um objetivo genérico seria " A invasão de um sistema " para avaliar o quanto "fácil" ou "difícil" foi a quebra da segurança e quais às consequências e os impactos de uma invasão real. De acordo com [50] o Pentest também pode fornecer provas e evidências da segurança, de um sistema ou aplicação por meio de uma auditoria ética, essas evidências e provas podem contribuir com requerimentos de investimentos em segurança, solicitados ao alto escalão de empresas e ou organizações.

2.5.1 Áreas de realização do Pentest

Segundo [51], o Pentest pode ser aplicados nas seguintes áreas de TI:

1. **Rede de Computadores:** está área visa identificar falhas de segurança associadas a projeto, implementação e operação da rede de uma organização alvo. O analista verifica e analisa possíveis brechas de acesso não autorizado, acesso remoto, falhas de configuração de dispositivos, falhas em configuração de firewall, ou qualquer falha na rede, que possam servir de ponto de entrada para um cibercriminoso invadir a rede do alvo.

    - Rede sem fio - compreende todas as comunicações eletrônicas, sinais e emanações que ocorrem em rede sem fio.
    - Rede de dados - compreende todos os sistemas eletrônicos e redes de dados onde a interação ocorre através de cabo de rede com fio e as linhas estabelecidas com os sistemas eletrônicos [52].

2. **Sistemas e Aplicações:** está área visa avaliar os controles de segurança de uma aplicação, explorar vulnerabilidades para atestar riscos e consequências de ataques reais.

3. **Engenharia Social** é um campo da Segurança de sistemas de informação, para descrever casos onde pessoas são persuadidas a fornecerem informações que deveriam ser privadas [53]. O Pentest quando aplicado a está área, tem o intuito de realizar auditoria do fluxo de trabalho de uma organização, cujo o objetivo são as interações humanas e a coleta de informações confidenciais, sobre o alvo ou sobre seus sistemas e componentes.

4. **Telecomunicações:** compreende todas as redes de telecomunicações, digitais ou analógicas, onde a interação ocorre entre linhas telefônicas [52].

5. **Teste de segurança físico:** ocorre onde o canal é de natureza física e não-eletrônica, compreende um elemento tangível de segurança onde a interação requer esforço físico ou um transmissor de energia para manipulação [52].

Manual e Automatizado

O processo de Pentest, pode ser executado de forma manual e automatizado com uso de ferramentas especificas. O processo manual pode ser usado em tarefas rápidas, testes simplificados, para evitar: bloqueios no firewall ou deixar rastros; Automatizado: utiliza ferramentas e frameworks para facilitar ou agilizar um determinado teste. Algumas ferramentas necessitam de menos interversão humana, outras necessitam mais. A Tabela 2.3 apresenta às considerações, segundo [10].

Tabela 2.3: Considerações sobre Pentest manual e automatizado [10]

Evento	Manual	Automatizado
Processo de teste	Não padronizado, Atividade demanda mais esforço para tarefas mais complexas, Maior custo para personalização, Podem realizar atividades mas transparentes com maior domínio sobre o resultado.	Mais rápido em tarefas complexas, Processo padronizado, Repetição de testes são mais fáceis, Algumas ferramentas podem ser menos transparentes nos processos que executam.
Gerenciamento de vulnerabilidades e ataques	Gerenciamento, atualização e busca nos repositórios de vulnerabilidade mais lenta, Adaptação de códigos para plataformas diferentes.	Atualização, gerenciamento e busca nos repositório de vulnerabilidade são automáticas
Relatório	Necessita de coleta manual de dados	Pode -ser automatizado, em alguns casos pode ser customizado.
Limpeza	Caso precise o examinador tem que fazer limpeza de log e eventuais configurações.	Algumas ferramentas oferecem recurso para limpeza de logs e eventuais configurações.
Rastros	Em alguns casos pode ser usado para evitar bloqueios por firewall.	Ferramentas podem ser bloqueadas no firewall mais facilmente necessitando de técnicas de camuflagem ou evasão.
Treinamento	Não padronizado, Podem ser customizados.	Pode ser padronizado, Tende ser mais fácil, Porém pode ser limitado a determinadas soluções.

Vantagens e Desvantagens do Pentest

Segundo [11] o processo de Pentest, apresenta algumas características que podem ser classificadas como vantagens e desvantagens, conforme apresentado na Tabela 2.4:

Tabela 2.4: Vantagens e desvantagens do Pentest [11]

Vantagens	Desvantagens
1. Sem falso positivo, a vulnerabilidade descoberta é igual à vulnerabilidade a ser explorada. 2. Pode ser baseado na prática do usuário em um ambiente. 3. Capaz de expor vulnerabilidades não detectada por outras ferramentas. 4. Pode considerar fatores de engenharia social.	1. Depende fortemente do profissional testador 2. Os resultados dependem do conhecimento, capacidade e experiência do profissional testador. 3. A técnica pode afetar o sistema testado.

2.5.2 Processo de Pentest

O processo de Pentest é utilizado para realização de ataques cibernéticos de forma controlada, mediante autorização, observando possíveis e riscos e precauções (abordados em metodologia para Pentest) a fim de evitar causar danos ou a indisponibilidade do alvo a ser testado. O processo de Pentest pode ser constituído de algumas etapas como: coleta de informações, ataque, analise e relatório, estas etapas geralmente são alocadas em um processo composto de uma a três fases, dependendo da metodologia utilizada. A Figura 2.4 ilustra o processo de Pentest.

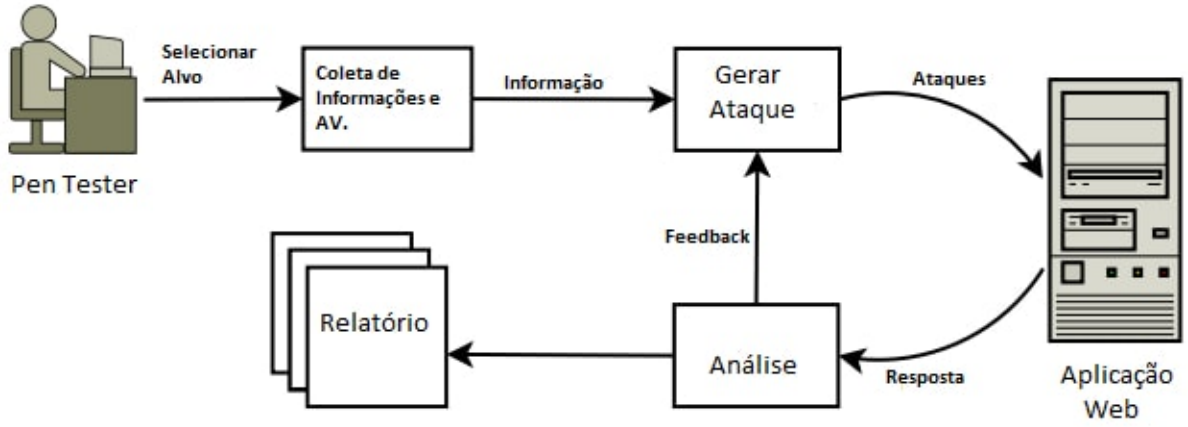

Figura 2.4: Processo de Pentest (adaptada [4]).

- Coleta de informações: nesta etapa do processo realiza-se a coleta de informações sobre um determinado alvo, exemplo: uma aplicação web. O intuito é conseguir informações e detectar vulnerabilidades, que auxiliem no teste exploratório, a coleta de informações geralmente e realizada no processo de Analise de Vulnerabilidade AV.

- Realização de ataque: está é a etapa mais importante deste processo, é nela que ocorre a realização dos ataques cibernéticos, para exploração das vulnerabilidades detectadas no processo de AV. Está etapa tem o intuito de atestar a existência de uma vulnerabilidade para mensurar a gravidade e riscos, que uma organização estaria exposta em caso de ataques reais.
- Análise: objetiva analisar às respostas e resultados dos ataques disparados contra um alvo, caso seja necessário o analista testador (*Pen Tester*) , pode repetir novamente os testes, modificando parâmetros e configurações, visando alcançar os resultados desejados.
- Relatório: produto final da auditoria, é o artefato que possui os resultados do processo de auditoria de segurança. Este é o produto que é entregue para o demandante. No relatório deve conter informações sobre às vulnerabilidades detectadas como: quantidade, tipo, grau de criticidade, etapas realizadas e explorações realizadas como: tipos de ataques realizados e resultados obtidos. Outros itens importantes que podem fazer parte do relatório são: print de telas, registro de log, vídeos, arquivos, scripts e outros elementos que forem considerados importantes.

2.5.3 Metodologia para Pentest

O Instituto Nacional de Padrões e Tecnologia (*National Institute of Standards and Technology*, NIST) fornece padrões, desenvolve testes e métodos para teste e analise técnicas visando promover o desenvolvimento e a produtividade da tecnologia da informação, o NIST por meio de seu laboratório de tecnologia da informação, lançou em 2008 a publicação especial 800-115 que é o Guia Técnico de Segurança da Informação Testes e Avaliação (*Technical Guide to Information Security Testing and Assessment*) [19], este é um guia para os aspectos técnicos básicos da realização de avaliações de segurança da informação, que tem como objetivo fornecer orientações para organizações sobre planejamento e condução de testes e avaliações técnicas de segurança da informação, podendo ser utilizado como parte de uma politica ou método de avaliação de uma organização.

De acordo com [19], uma metodologia de avaliação da segurança da informação pode trazer benefícios como:

- Documentação das atividades realizadas,
- Fornecer consistências aos testes de segurança para minimizar riscos,
- Registrar a definição de escopo e ou restrições,
- Fornece insumos para avaliação,

- Estruturar a execução da avaliação em fases ou etapas para:
  - Facilitar a execução,
  - Definir pontos de controle ou ruptura, registrando as fases ou estágios executados. Segundo a publicação especial 800-115 [19] uma metodologia de avaliação de segurança deve ter no minimo três fases,sendo : planejamento para definir a execução da auditoria; execução para identificar vulnerabilidades e ou validar vulnerabilidades (atestar riscos e consequências) usando Pentest ; avaliação ou pós-execução para realizar analise das vulnerabilidades detectadas, identificar as causas da vulnerabilidades para possibilitar a mitigação.

Para [54] o uso de uma metodologia permite dividir um processo complexo em tarefas menores e mais gerenciáveis, em geral as metodologias de avaliação de apresentam de quatro a sete fase ou etapas, variando apenas os nomes ou subdividindo em fase em outras, importando que o processo geral atenda as expectativas e necessidades dos utilizadores.

Fases do Guia Técnico de Segurança da Informação Testes e Avaliação

O Guia Técnico de Segurança da Informação Testes e Avaliação [19] (publicação especial 800-115), apresenta quatro etapas: Planejamento, Descoberta, Exploração, Relatório e mais uma etapa adicional especifica para Analise de vulnerabilidade, portanto totalizando cinco fases ou etapas, conforme descrição abaixo.

- Planejamento - é a etapa inicial, deve ser estruturada antes da execução da avaliação de segurança da informação, o intuito e planejar e definir como avaliação de segurança devera ser executada, tratando de itens como: autorização, escopo e ou objetivo (avaliar um sistema ou aplicação completamente ou módulos específicos), também defini os tipos de testes a serem realizados, agendamento (data de realização).
  - Escopo - critério importante, pois a partir deste, decidimos se um sistema será verificado integralmente ou apenas parte dele [55].
  - Forma de execução da Auditoria.
  - Realização: Interna ou Externa.
  - Tipos de Testes.
  - Período - data e hora (agendamento) o gestor deve definir o momento mais adequado para realização da auditoria.
  - Autorização - o gestor responsável deve autorizar a auditoria de segurança.

- Descoberta - depois de definir os objetivos, escopo, autorização e agendamento (período de execução) na fase de planejamento, o testes real começa pelo reconhecimento e escaneamento do alvo:
  - Reconhecimento: nesta etapa realiza-se a coleta de informações sobre o alvo, este processo é muito importante pois as informações coletadas podem contribuir para o sucesso das fases posteriores, por isso o analista testador busca coletar o máximo de informações possíveis sobre o sistema alvo como IP, endereço URL, arquitetura e componentes, linguagem de programação, sistema operacional [56]. O Reconhecimento pode ser dividido em duas formas [54]: Ativo - quando a realização da coleta interagi diretamente com o alvo, podendo o alvo registrar as atividades em log ; Passivo - busca informações de outras fontes como documentação ou internet.
  - Escaneamento de portas e serviços: consiste em identificar status de portas (aberta,fechada ou filtrada) e serviços em execução nessas portas, identificando detalhes nome e versão e o sistema operacional, os resultados devem ser listados e enumerados para utilização nas fases posteriores [57].
- Avaliação - as informações derivadas da fase de descoberta são a fonte de entrada para esta fase, com base nas informações coletadas são realizados escaneamentos e exames para detectar, identificar e classificar vulnerabilidades. Nesta fase realiza-se o processo de Analise de vulnerabilidade:
  - Analise de Vulnerabilidade - processo em que realiza-se varreduras em busca de vulnerabilidades para detectar possíveis falhas de segurança que possam ser aplicáveis a um determinado alvo [57], após a detecção de vulnerabilidades, realiza-se analises para identificar, listar, quantificar e classificar o grau de risco das vulnerabilidades detectadas [58].
- Exploração - tem como objetivo a realização de ataques cibernético de forma controlada mediante autorização.
  - Realizar ataques cibernéticos - neste processo a finalidade é ganhar controle sobre um sistema, explorando as vulnerabilidades ou falhas encontradas. O Pentest é uma excelente forma para identificar os pontos fortes e fracos de uma aplicação ou rede de computadores [59]. Nesta fase são realizados ataques de acordo, com as vulnerabilidade detectadas, exemplo de alguns ataques: *Code Injection* – injeção de códigos (SQL, PHP, LDAP); XSS/CSS (*cross site scripting*) – usado para obter sessões de usuário, obter cookies; Força bruta (*Brute Force*) – usado para quebra de senha, exploração de senhas fracas [60].

- Relatório - tem como objetivo apresentar os resultados obtidos nos testes realizados como vulnerabilidades detectadas, explorações realizadas e ainda apresentar as etapas ou fases realizadas, evidências coletadas (imagens, arquivos, logs). O relatório também pode apresentar sugestões ou recomendações para auxiliar a mitigação das vulnerabilidades detectadas. O relatório é o produto final do processo da auditoria de segurança, que é o artefato que será gerado e entregue ao gestores demandantes, este é uma peça critica, pois por meio dele os gestores (responsáveis) tomarão ciência dos resultados obtidos e terão informações para subsidiar suas decisões [61].

Auditoria usando Analise de Vulnerabilidade e Teste de Penetração

Manter ou aprimorar continuamente a segurança da informação é uma das questões enfrestadas pelas organizações, proteger adequadamente os ativos de informação e espaço cibernético é um desafio para organizações em todo o mundo, por isso pesquisadores, profissionais, empresas e governos têm buscando formas para resolver problemas relacionados a ataques cibernéticos. Para [5] [51] [62] [50] [2] o processo de Analise de Vulnerabilidade e o Pentest (*Penetration Testing*) são métodos que podem ser utilizados, para contribuir com ações proativas para a segurança da informação e proteção do espaço cibernético, promovendo os seguintes benefícios quando usado de forma periódica:

- Assegurar o nível de segurança de sistemas e aplicações.
- Identificar possíveis ameaças, por meio da detecção de vulnerabilidades cibernéticas em circunstâncias controladas, de modo a permitir a mitigação antes que ocorra um ataque cibernético com intuito de explorar as falhas detectadas.
- Avaliar riscos e danos em casos de taques cibernéticos reais.
- Pode auxiliar para evitar o acesso não autorizado a sistemas, aplicações e também informações confidências, desde que as vulnerabilidades detectadas sejam mitigadas.
- Pode contribui para evitar danos financeiros, perdas de dados e preservação da imagem de uma organização.

2.6 Síntese do Capitulo

A Segurança da informação objetiva proteger dados e informações de indivíduos ou organizações e para isso possui propriedades como: confidencialidade, integridade e disponibilidade, autenticidade e não repudio. Com intuito de promover ou melhorar a SI algumas normas ou padrões podem ser utilizados como a família da ISO/IEC 27000, que são específicas para gestão da segurança da informação.

Alguns desafios das organizações em proteger seus sistemas e informações são os ataque cibernéticos, um ataque cibernético é ato intencional ou não intencional que pode causar danos ou comprometer de outra forma, informações e ou os sistemas [63], objetivando entender um ataque cibernético pesquisadores promovem taxonomias, em que são definidas classificações baseadas no tipos e comportamento dos ataques, como exposto na taxonomia de ataques web apresentada na Figura 2.1, proposta por [1], cujo o intuito desta é auxiliar a compreensão destes ataques e a construção de aplicações mais seguras.

Alguns tipos de ataques visam causam indisponibilidade de sistemas e outros visam a exploração de vulnerabilidades, podendo um cibercriminoso obter acesso não autorizado. Como forma proativa de segurança para detectar, identificar e classificar uma vulnerabilidade de acordo com seu respectivo grau de risco é utilizado o processo de Analise de Vulnerabilidade (AV), em que os resultados deste processo podem ser utilizados para mitigações de falhas de segurança em sistemas, durante o processo de AV são realizado a coleta de informações sobre um determinado alvo e também realizados varreduras para detecção de vulnerabilidades.

Para atestar e avaliar possíveis riscos e consequências proveniente da exploração de uma vulnerabilidade, por meio de um ataque cibernético é utilizado o processo de Teste de penetração (*Penetration testing*, Pentest) que consistem na realização de ataque cibernético de modo controlado, mediante autorização para avaliar a segurança de um sistema. O processo de Pentest pode ser utilizado, seguindo metodologias e ou guias, o Instituto Nacional de Padrões e Tecnologia (*National Institute of Standards and Technology*, NIST) propõe o guia Técnico de Segurança da Informação Testes e Avaliação (*Technical Guide to Information Security Testing and Assessment*) [19] que é um guia especifico, para a realização do processo de Pentest, o qual foi utilizado neste trabalho de forma adaptada.

Capítulo 3

Metodologia

A solução proposta neste trabalho é a Auditoria de Segurança da informação em Sistemas e Aplicações, usando o processo de Analise de Vulnerabilidade e Pentest (*Penetration Testing*) de acordo com: [5] [51] [62] [50] [2], são técnicas estratégicas para contribuir de modo proativo com a segurança da informação e segurança cibernética.

O processo de auditoria é divido em cinco fazes: Planejamento, Descoberta, Analise, Exploração e Relatório, conforme apresentado no Capítulo 2, adaptado da publicação especial 800-115 [19] do *National Institute of Standards and Technology* (NIST), pois é uma metodologia consolidada e reconhecida mundialmente. Requisitos - profissional com conhecimentos em: Gestão de segurança da informação, Analise de Vulnerabilidade e Pentest, outros conhecimentos complementares são: banco de dados, redes de computadores e linguagem de programação e sistemas operacionais. As atividades de Analise de Vulnerabilidades e Teste de Penetração, podem ser realizadas de forma: manual (uso de comandos, rotinas e scripts) ou automatizada (com apoio de ferramentas e frameworks).

3.1 Forma de execução da auditoria

O processo de Auditoria pode ser executado de duas formas [64], conforme descrição abaixo e ilustração na Figura 3.1

1. Parcial - executa-se somente a etapa referente ao processo de Analise de Vulnerabilidade (AV) (o processo de AV foi apresentado no Capítulo 2). Na forma Parcial, são executadas as seguintes fases: Planejamento, Descoberta, Avaliação e Relatório. Na execução da forma Parcial, ocorre a detecção, identificação, quantificação e classificação de uma vulnerabilidade, com isso é possível fazer a remediação ou mitigação das vulnerabilidades detectadas. Alguns fatores podem influenciar na decisão da escolha por esta forma de execução, são os riscos e precauções que envolvem o

Pentest, que serão abordados posteriormente neste capitulo e ainda decisões particulares de uma organização, como falta de acordo que forneçam a permissão ou aceite suficiente para a execução da forma completa.

2. Completa - executa-se a etapa do processo de Analise de Vulnerabilidade (AV) mais a etapa do processo de Pentest (apresentado no Capítulo 2). Na forma de Execução Completa, são executadas todas as fases: Planejamento, Descoberta, Avaliação, Exploração e Relatório. A execução deste processo tem o intuito de atestar, gerar provas e evidências, sobre potenciais riscos e consequências provenientes da exploração de vulnerabilidades, que poderiam acontecer em casos reais de ataques cibernéticos. Para [62] é importante ressaltar, que os testes de segurança realizados em copias, ambientes de testes, que não tenham acessos de usuários e saída para internet, podem ter seus resultados minorados, sendo que até mesmo uma única configuração diferente, do ambiente de teste em relação ao real, pode levar a resultados diferentes, do que ocorreria no ambiente legitimo em produção. Também [62] recomenda que os testes de segurança, sejam realizados preferencialmente por equipe interna, evitando assim, a exposição de uma organização (dados sigilosos, falhas, segredos industriais) a terceiros, falta de transparência na execução do processo e ainda a capacidade técnica dos contratados, que pode satisfazer ou não os clientes demandantes.

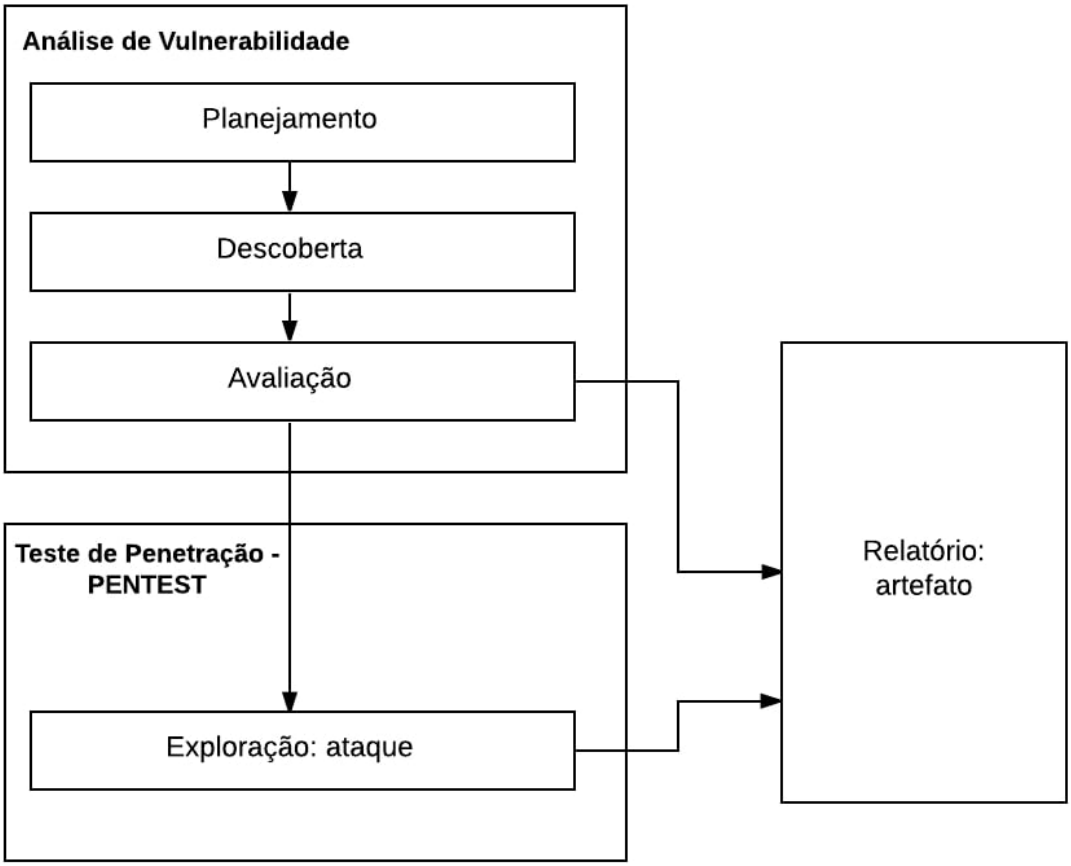

Figura 3.1: Processo de Auditoria de Sistemas e Aplicações, adaptado de [5].

3.2 Tipos de testes

3.2.1 Realização

Segundo [65] os testes de segurança podem ser realizado de duas maneiras em relação a localização do analista testador:

1. Realização INTERNA - dentro da rede da instituição,
2. Realização EXTERNA - fora da rede da instituição.

3.2.2 Tipos de teste

Antes de iniciar o processo de auditoria, pode-se escolher entre os três tipos, conforme ilustrado na Figura 3.2 e descrição abaixo [65] [51]:

- Caixa preta (*Black Box*) - neste método o testador não possui conhecimento da infraestrutura a ser testada. Os testadores devem primeiro determinar a localização e a

extensão dos sistemas, antes de iniciar sua análise. O teste de penetração de caixa preta executa ataques como alguém, que não está familiarizado com o sistema.

- Caixa branca (*White Box*) - neste método os testadores possuem conhecimento da infra-estrutura ou aplicação a ser testada, muitas vezes incluindo diagramas de rede, código fonte e informações de endereçamento IP. Este teste demonstra o que pode acontecer durante um "trabalho interno"ou após um "vazamento"de informações sensíveis, onde o atacante tem acesso ao código fonte, layouts de rede e possivelmente até senhas.
- Caixa cinza (*Gray Box*) - é uma variação entre os testes: caixa preta e caixa branca. No tipo caixa cinza, os testadores terão conhecimentos parciais sobre a infraestrutura ou sistema que serão testados.

Figura 3.2: Tipos de teste.

Riscos envolvendo o Pentest

A execução de testes de Segurança como o Pentest, pode trazer alguns riscos para o alvo testado, tendo em vista a natureza das técnicas utilizadas, que são invasivas e similares às técnicas usadas por cibercriminosos em ataques reais [66]. De acordo com [51] os riscos específicos do Pentest são categorizados como:

1. Risco Técnico: estes são os riscos causados diretamente pelas atividades do pentest. Alguns dos principais riscos técnicos que podem ser causados a um sistema alvo são: a interrupção de serviço, redução do desempenho ocasionando lentidão, modificação ou contaminação de dados e até exposição de dados.
2. Risco Organizacional: envolve alguns efeitos colaterais organizacionais, como disparos de processos de incidente, por exemplo: no caso do Pentest autorizado, por um

gestor, a fim de testar também a reação da sua equipe, mas sem informa-la sobre o teste, poderia desencadear procedimentos para contenção do ataque, o que poderia ser positivo ou não. Ainda pode ocorrer a interrupção de processos de negócios e perda da reputação, no caso da parada de serviços prestados a terceiros.

3. Risco Legal: quando relacionado a obrigações legais (leis), questões jurídicas, devido possíveis efeitos colaterais, como violação de obrigações legais pelos envolvidos, ou que causem prejuízo a terceiros. Exemplo: a queda de um sistema de uma entidade, durante a realização de um Pentest, executado sem autorização..

Precauções gerais

Com base nos fatores de risco envolvidos no Pentest, os testadores podem adotar algumas precauções a fim de evitar causar danos inesperados ao sistema alvo. De acordo com [51], as seguintes ações podem ser adotadas como precauções:

1. Teste indireto: visa coletar evidências suficientes, para concluir que uma vulnerabilidade está presente, em um determinado sistema ou aplicação e pode ser suscetível a ataque cibernético, uma forma de realizar essa estratégia é aplicar o processo de Analise de Vulnerabilidade, neste caso, não se efetua os testes (Pentest) que realizam ataques para exploração da vulnerabilidade.

2. Exploração limitada: neste caso usa-se testes (ataques) que demonstrem a existência da vulnerabilidade e a sua exploração, de forma que mostrem efeitos mensuráveis, mas sem causar efeitos colaterais graves.

3. Efeito retardado: Às vezes quando possível, os testes de exploração podem ser cancelados durante suas execuções, o intuito desta estrategia é iniciar um ataque e interromper em determinado ponto antes da conclusão, objetivando minorar possíveis riscos ou danos a um sistema. Está estratégia, quando possível de ser utilizada, é eficaz em casos que deseja-se mensurar os efeitos de um ataque cibernético real.

4. Teste interrompível: em alguns cenários, pode ser que os analistas testadores tenham que garantir, que possam interromper um determinado teste a qualquer momento, para que possam reagir imediatamente, caso seja observado alguma consequência não intencional.

5. Ferramentas de sobrecargas ou Estrangulamento: algumas ferramentas automatizadas, podem causar sobrecarga ao servidor alvo, isso pode resultar em interrupção dos serviços ou parada de um sistema ou aplicação. Exemplo: ferramenta DOS Slowloris.

6. Evitando bloqueios: às vezes a repetição de algum teste, pode desencadear o acionamento de funções projetadas para bloquear atacantes, como funções de login ou autenticação, que permitem um determinado numero de tentativas, ao exceder o numero permitido de tentativas a função bloqueara novas tentativas.

7. Ferramentas de confinamento: o Pentest envolve interação do analista testador com o sistema alvo, através de uma rede. Assim para evitar qualquer execução acidental de um teste de risco, contra outros sistemas que estão na mesma rede, mas que não são o alvo, o escopo deve ser delimitado nas ferramentas que serão usadas, portanto deve-se verificar se as ferramentas utilizadas, podem ser configuradas de modo a ficarem confinadas a determinado alvo.

8. Teste de teste: abordagens de novos testes exploratórios, como novas técnicas de ataques cibernéticos, podem ser mais ariscados do que técnicas ja conhecidas pelo analista testador. Por este motivo é recomendável treinar em um ambiente de teste ou laboratório, para experimentar novas técnicas, antes de disparar os novos testes contra alvos reais.

9. Isolamento parcial e replicação: subsistemas de um sistema alvo, às vezes podem ser configurados, para testes isolados ou utilizar uma réplica, para um subsistema com uma configuração diferente e assim reduzir os efeitos colaterais do teste.

10. Regras de engajamento: o analista testador e o seu cliente, devem estabelecer claramente regras inequívocas, especificando os objetivos, alvo, escopo, limites e qualquer outra informação, que deva estar acordada entre os envolvidos. Caso haja terceiros que possam ser afetados, pelo teste em questão, também deverão ser notificados.

3.3 Síntese do Capitulo

Este capitulo apresentou a metodologia utilizada neste trabalho que foi adaptada da metodologia disposta na publicação especial 800-115 [19] do NIST, que consiste em cinco fases: Planejamento, Descoberta, Avaliação e Relatório (apresentado no Capitulo 2).

A formas de execução da Auditoria que pode ser Completa - realiza-se execução do processo de Analise de vulnerabilidade (apresentado no Capitulo 2) em conjunto o processo de Pentest (apresentado no Capitulo 2) ou Parcial - realiza-se a execução somente do processo de Analise de vulnerabilidade; Tipos do teste : Caixa preta (sem conhecimento da estrutura), Caxa branca (com conhecimento da estrutura), Caixa Cinza (conhecimento parcial da estrutura); realização Interna - dentro da rede da organização ou Externa - fora da rede da organização. Também presentou-se os risco e precauções do processo de Pentest.

Capítulo 4

Casos de Estudo

Este capitulo apresenta o estudo de caso de três sistemas governamentais, que foram utilizados como experimento para aplicação pratica da Auditoria de Segurança da Informação em Sistemas e Aplicações. Conforme processo de metodologia apresentado no Capitulo 3, o processo de auditoria foi aplicado ao seguintes casos:

1. O primeiro caso (Caso I) é o Sistema de aquisição de produtos e serviços, auditoria aplicada na forma de execução: Completa (processo Av + Pentest).
2. O segundo caso (Caso II) é o Sistema colaborativo de informações eletrônicas, auditoria aplicada na forma de execução: Parcial (somente processo de AV).
3. O primeiro caso (Caso III) é o Sistema de aquisição de kits pedagógicos, auditoria aplicada na forma de execução: Completa (processo Av + Pentest).

4.1 Ferramentas utilizadas

Esta seção apresentação as principais ferramentas utilizadas durante os experimentos deste trabalho, a aplicação deste trabalho não esta limitada as ferramentas apresentadas, outras ferramentas para o mesmo proposito também podem ser utilizadas.

- Nmap [1] - é um utilitário livre e de código aberto (licença) para descoberta de rede e auditoria de segurança. Utilizado por administradores de rede e profissionais de segurança [67] para tarefas como o inventário de rede, gerenciando agendamentos de atualização de serviço e monitorando o tempo de atividade do host ou serviço [68]. O Nmap usa pacotes de IP em formas inovadoras para determinar quais hosts estão disponíveis na rede, quais serviços (nome e versão do aplicativo) que esses hosts estão oferecendo, quais sistemas operacionais (versões do sistema operacional) estão

[1] `https://nmap.org/`

sendo executados, que tipo de filtros de pacotes, firewalls estão em uso e dezenas de outras características. Ele foi projetado para varrer rapidamente grandes redes, mas funciona bem contra hosts únicos. O Nmap é executado nos principais sistemas operacionais de computadores, e os pacotes binários oficiais estão disponíveis para Linux, Windows e Mac OS X. O Nmap pode ser utilizado nas fases Descoberta e Avaliação.

- WhatWeb [2] - é uma ferramenta para varredura web, coleta informações sobre aplicações web e reconhece as tecnologias destas, incluindo sistemas de gerenciamento de conteúdo (CMS), plataformas de blogs, pacotes de estatística, análise, bibliotecas JavaScript, servidores web (inclusive números de versão) e dispositivos embutidos. possui mais de 1700 plugins, cada um para reconhecer algo diferente, também identifica endereços de e-mail, IDs de conta, módulos de estrutura web, erros SQL [69], o WhatWeb pode ser utilizado nas fases Descoberta e Avaliação.

- Vega [3] - é uma plataforma de código aberto gratuito para testar a segurança das aplicações web [70]. Vega é escrito em Java, e pode ser executado em diversas plataformas como Linux, OS-X e Windows, também pode ser facilmente expandido com módulos escritos em Javascript e pode encontrar e validar SQL Injection, Cross-Site Scripting (XSS), informações confidenciais divulgadas inadvertidamente, configurações de segurança TLS/SSL e ainda possui um varredor (scanner) automatizado para testes rápidos e um proxy de interceptação para inspeção tática. O Vega pode ser usado nas fases Avaliação e Exploração.

- Nessus [4] - é a ferramenta para analise de vulnerabilidade, realiza a detecção e identificação de vulnerabilidade [71], geralmente utilizados por profissionais como auditores e analistas de segurança, possui uma versão home com limitações (16 IPs) que pode ser utilizada sem custo e ainda possui outras versões profissionais mais completas que devem ser licenciadas (compradas). Os usuários podem agendar varreduras, usar assistentes para criar e criar políticas facilmente e rapidamente, agendar scans e enviar resultados via e-mail. O Nessus pode ser utilizado nas fases Descoberta e Avaliação.

- OpenVas [5] - é uma estrutura de vários serviços e ferramentas (framework) que oferecem uma abrangente e poderosa solução para detectar e realizar gerenciamento de vulnerabilidades [72]. O Sistema de Avaliação de Vulnerabilidade Aberta (OpenVAS) é um quadro de vários serviços e ferramentas que são capazes de encontrar

[2] `https://www.morningstarsecurity.com/research/whatweb`
[3] `https://subgraph.com/vega/`
[4] `https://www.tenable.com/products/nessus-vulnerability-scanner`
[5] `http://www.openvas.org/`

diversas vulnerabilidades automaticamente. OpenVas é um software livre e a maioria dos seus componentes está licenciada sob a licença pública geral GNU-GPL. Pode ser utilizado na fase Avaliação.

- W3af [6] - é um framework para realização de auditoria de aplicações web [73], pode detectar mais de 200 (duzentas) vulnerabilidades como *SQL Injection* e XSS, também pode realizar ataques para testes exploratívos, possui diversos plugins para extensão do framework, é escrito em Python e é licenciada sob o GPLv2.0. O w3af pode ser utilizado nas fases Reconhecimento, Avaliação e Exploração.
- Sqlmap [7]- Sqlmap é uma ferramenta de teste de penetração de código aberto que automatiza o processo de detecção e exploração de falhas de injeção de SQL e a tomada de servidores de banco de dados [74]. A ferramenta vem com um poderoso mecanismo de detecção, muitos recursos de nicho para o testador de penetração eainda possui mecanismo que podem até acessar o sistema de arquivos subjacente e executar comandos no sistema operacional. O Sqlmap pode ser utilizado na fase Exploração.

4.2 Caso I

O Caso I - apresenta o uso da metodologia de Auditoria de Segurança da informação aplicada ao Sistema de aquisição de produtos e serviços. Os gestores da demandantes escolheram este sistema, devido sua importância e também pelo fato de ser um sistema legado. Observação: devido a preservação e sigilo institucional, algumas informações e evidências foram discretizadas ou suprimidas.

4.2.1 Planejamento

Fase Planejamento do Caso I,

- Escopo: Sistema de aquisição de produtos e serviços.
- Realização: Interna.
- Período: Horário de expediente em junho de 2016.
- Forma de execução: Completa (Av + Pentest).
- Tipo: Caixa Preta (informação recebida somente URL do sistema)

[6] `http://w3af.org/`
[7] `http://sqlmap.org/`

4.2.2 Descoberta

Reconhecimento e Coleta de informações

Objetivo desta fase é a coleta de informações, sobre o sistema alvo da auditoria, delimitando o escopo, para que não ultrapasse para a fronteira de outros sistemas ou serviços, que não façam parte da estrutura do mesmo. São levantadas diversas informações como: IP, linguagem de programação, servidor web, serviços, portas, arquitetura, visando obter o máximo de detalhes possíveis como: a versão de componentes e serviços. A partir dessas informações são verificadas a existência de possíveis vulnerabilidades. Nesta etapa foram utilizados comandos de sistemas operacionais, pesquisas manuais e a ferramenta Whatwheb [69] para coleta de informações. Dentre as informações obtidas citamos: linguagem de programação ASP e servidor de Aplicação IIS. A Figura 4.1 apresenta a captura de informações do sistema Caso I.

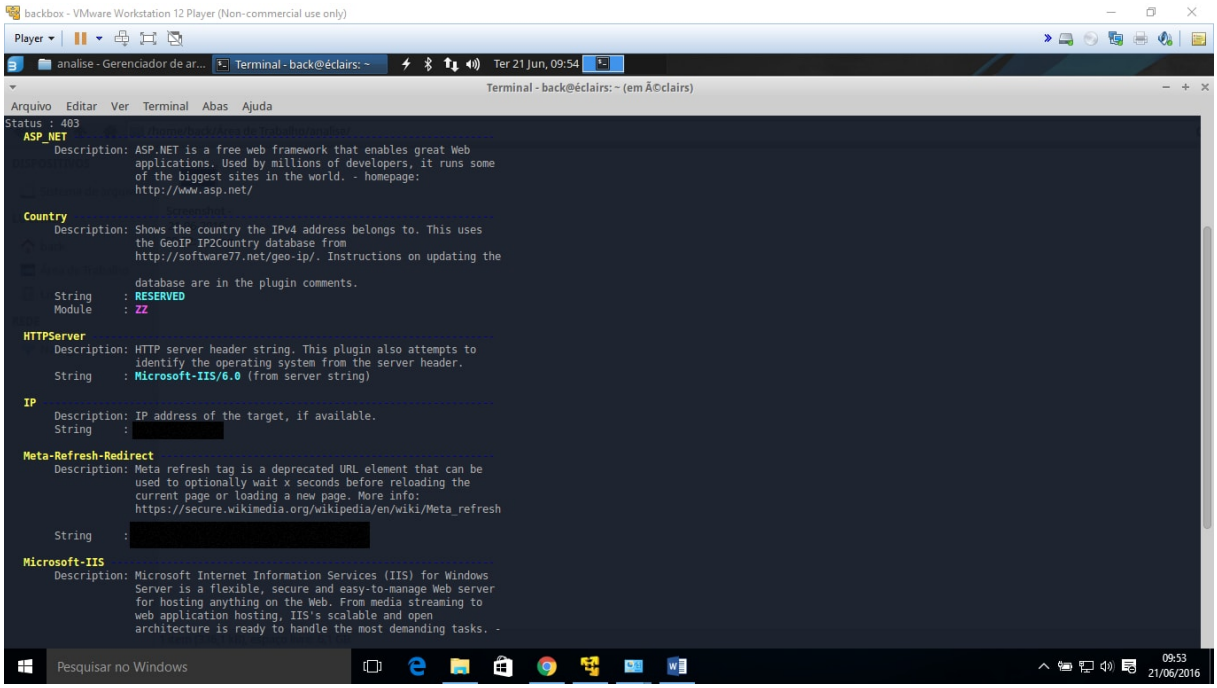

Figura 4.1: Levantamento de informações Caso I.

Varredura de portas e serviços

Foi realizado a execução da varredura de portas, verificando status das portas, serviços e protocolos e sistema operacional utilizado. A figura 4.2 mostra: as principais portas, serviços e protocolos em execução no servidor de aplicação Caso I.

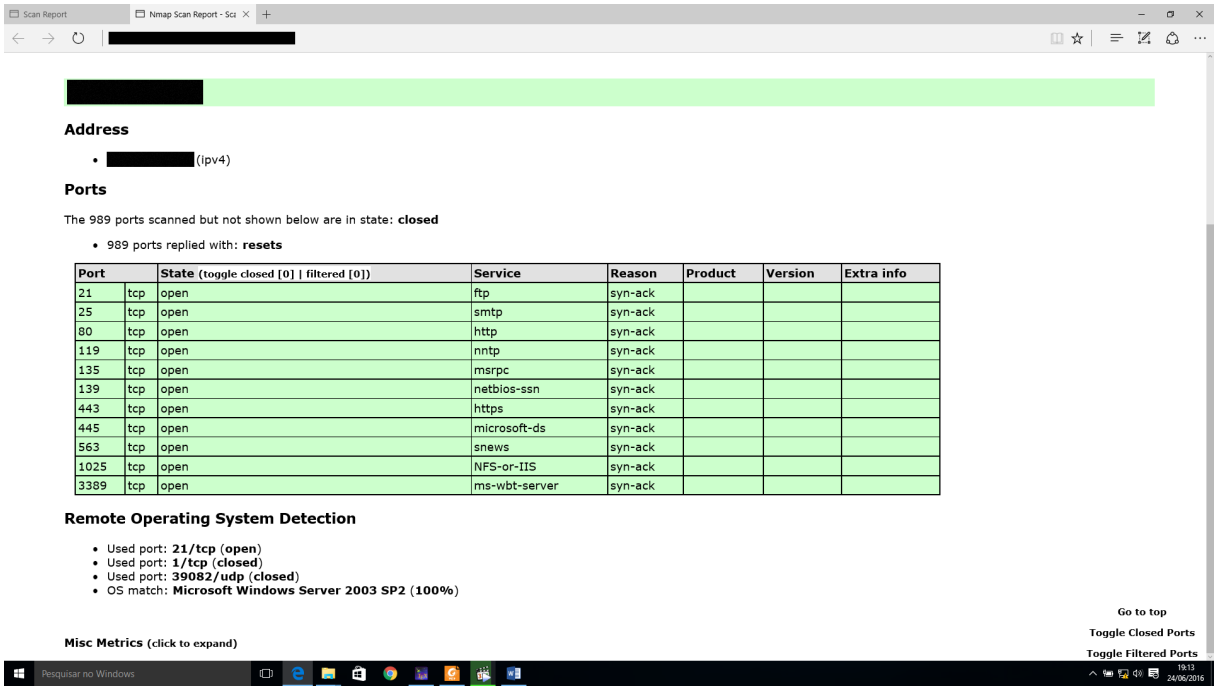

Figura 4.2: Escaneamento de portas, serviços e protocolos - Caso I.

4.2.3 Avaliação

Esta seção mostra as vulnerabilidades detectadas, por meio do processo de Analise de Vulnerabilidade. As vulnerabilidades detectadas são classificadas em grau de risco de criticidade como: Alto, Médio e Baixo. No caso das vulnerabilidades classificadas com risco de criticidade Alto (*High*), requerem atenção especial, pois podem ser exploradas com mais facilidade, e os danos e impactos, podem ser severos [75]. A Figura 4.3 apresenta as vulnerabilidades detectadas neste processo, ferramenta utilizada Vega[70].

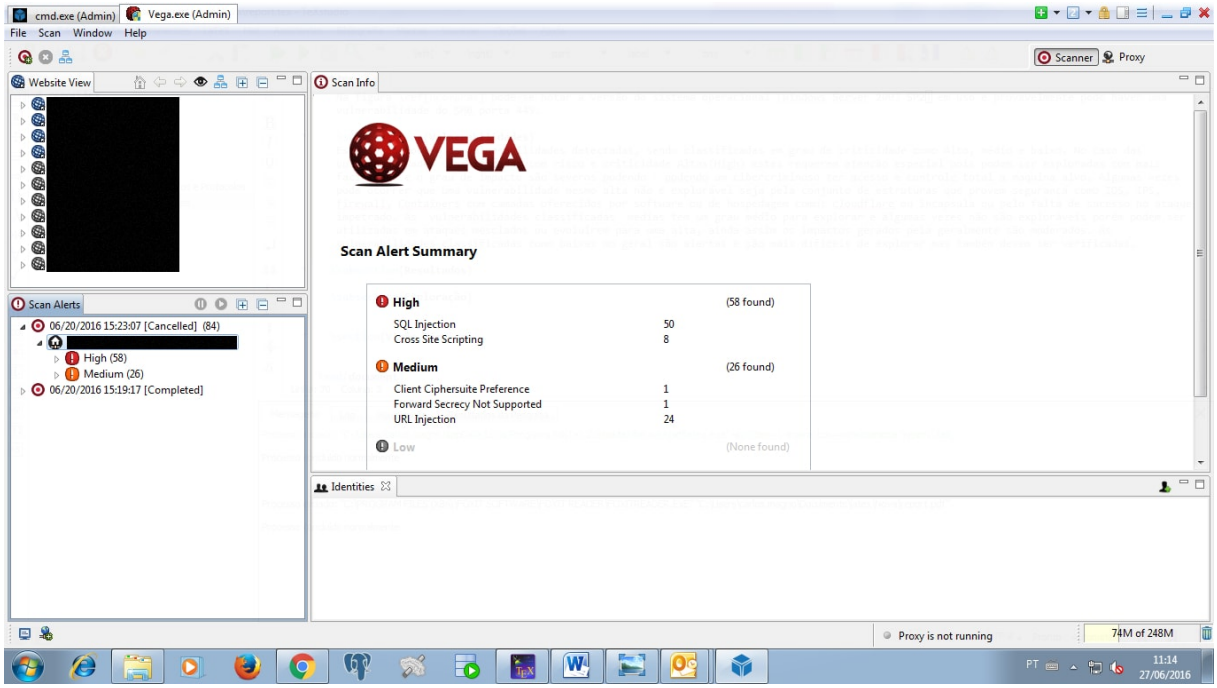

Figura 4.3: Vulnerabilidades detectadas - sistema Caso I.

Analise de Vulnerabilidade

No Caso I, foi detectado um numero expressivo de vulnerabilidades no sistema, conforme mostrado na Figura 4.3 e exposto abaixo:

- Cinquenta (50) vulnerabilidades de *SQL Injection* - grau de risco Alto, está vulnerabilidade consiste em realizar a inserção ou injeção de uma consulta SQL através dos dados de entrada do cliente para a aplicação ANLEY [76].
- Oito (08) vulnerabilidades de *Cross Site Scripting*, vulnerabilidade conhecida também como XSS - grau de risco Alto, está vulnerabilidade consiste em realizar injeções de códigos maliciosos do lado cliente, também podem fazer o direcionamento de acesso para outro site ou executar scripts maliciosos [77].
- Vinte e quatro (24) vulnerabilidades, relacionadas a *Url Injection* (injeção em url) - grau de risco Médio, essas vulnerabilidades consistem em fazer a injeção de strings na url de um site.
- Outras duas (02) vulnerabilidades, também grau de risco Médio: *cipher suite preference* (preferencia em conjunto de codificação) e Perfect Forward Secrecy (Segurança Futura "Para Frente" Perfeita), ambas estão relacionadas a falhas de comunicação e criptografia, que podem ocorrer na interação com os protocolos SSL((*Secure Soc-*

kets Layer) e TSL (*Transport Layer Security*), que são protocolos de criptografia projetados para internet [78].

Varredura da Plataforma e componentes

Conforme apresentado na taxonomia de ataques web do Capítulo 2, a plataforma que suporta uma aplicação, também deve ser verificada, pois no caso de uma exploração um cibercriminoso, poderia obter controle da maquina e componentes onde a aplicação está hospedada. Neste processo foram verificados os componentes, que são usados pela aplicação como: serviços, protocolos e sistema operacional, com suas respectivas versões, utilizando a ferramenta NMAP que é uma ferramenta open source, utilizada para coleta de informações e mapeamento de redes [68].

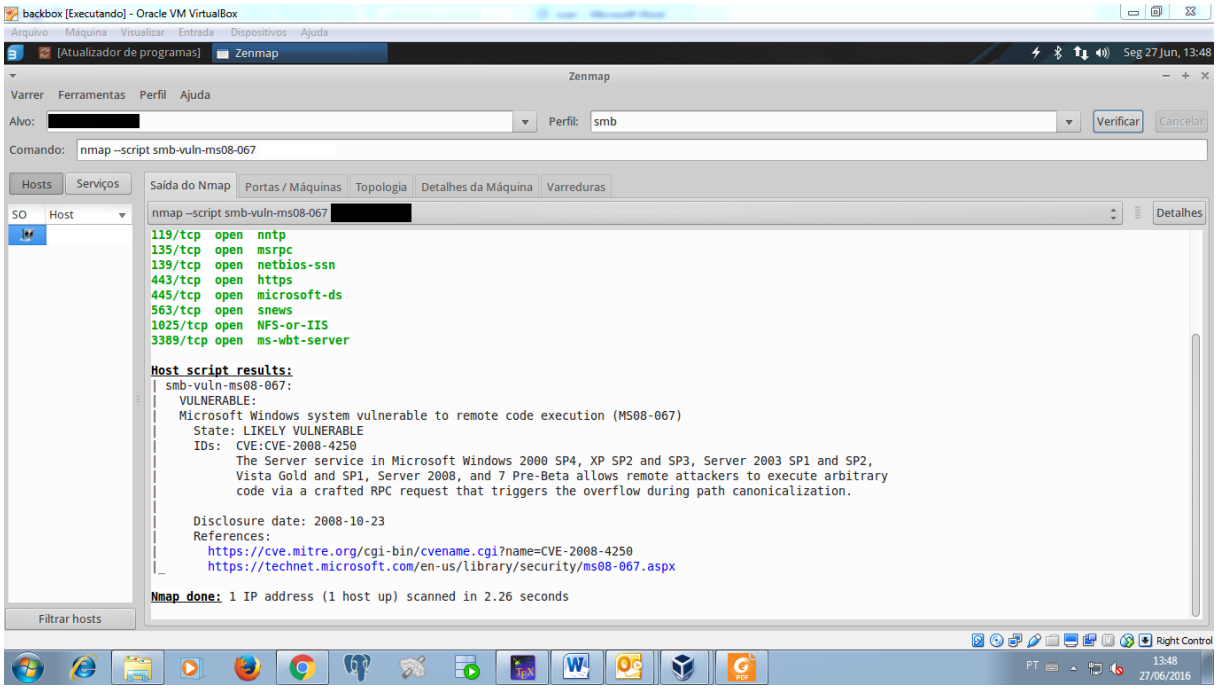

Figura 4.4: Vulnerabilidade detectadas plataforma do sistema Caso I.

Nesta etapa foi detectado uma vulnerabilidade de SMB, conforme mostra a Figura 4.4, também podemos visualizar na imagem apresentada o CVE 2008-4250 (conceito exposto no Capítulo 2), que descreve a vulnerabilidade detectada do protocolo SMB (*Server Message Block*), que é um protocolo de compartilhamento de arquivos em rede, para permitir que os aplicativos de um computador, leiam e gravem em arquivos e solicitem serviços dos programas do servidor, em uma rede de computadores Windows [79]. O protocolo SMB pode ser usado, sobre o protocolo TCP/IP ou outros protocolos de rede, está vulnerabilidade possui grau de risco Alto e pode ser explorada, por um cibercriminoso com

uso de *exploit* - é um *malware* utilizado para explorar uma vulnerabilidade [80]. Como consequência, um cibercriminoso poderia obter acesso não autorizado.

4.2.4 Exploração

A fim de atestar a suscetibilidade à ataques cibernéticos do sistema Caso I, devido a grande quantidade de vulnerabilidades detectadas e conforme acordado na fase de Planejamento deste Caso I, foi realizado um ataque cibernético de *SQL Injection* - este ataque foi executado de forma controlada e cuidadosa, a fim de evitar causar indisponibilidade ou danos ao sistema, conforme exposto na Seção: Precauções gerais, apresentada no Capitulo 2. Neste ataque de *SQL Injection* foi utilizado a ferramenta SQLMAP que é uma ferramenta *open source*, utilizada para automatizar testes de penetração. A Figura 4.5 apresenta o SGBD, utilizado pelo sistema Caso I, que é o *Data Base Management System* DBMS: Oracle.

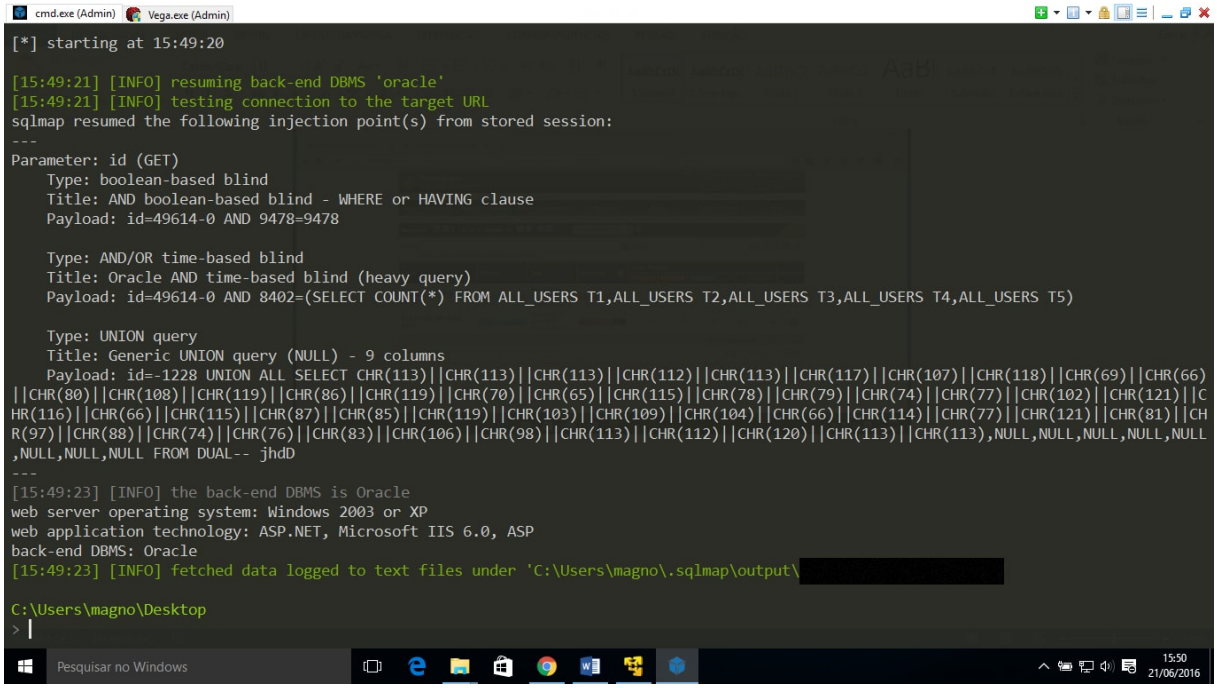

Figura 4.5: Descobrindo o SGBD - sistema Caso I.

Riscos e Consequências do ataque

O ataque disparado contra o alvo (Caso I), foi bem sucedido, é por causa disso, poderia ocorrer as seguintes consequências:

- Listar e acessar schemas do banco de dados, conforme apresentado na Figura 4.6.

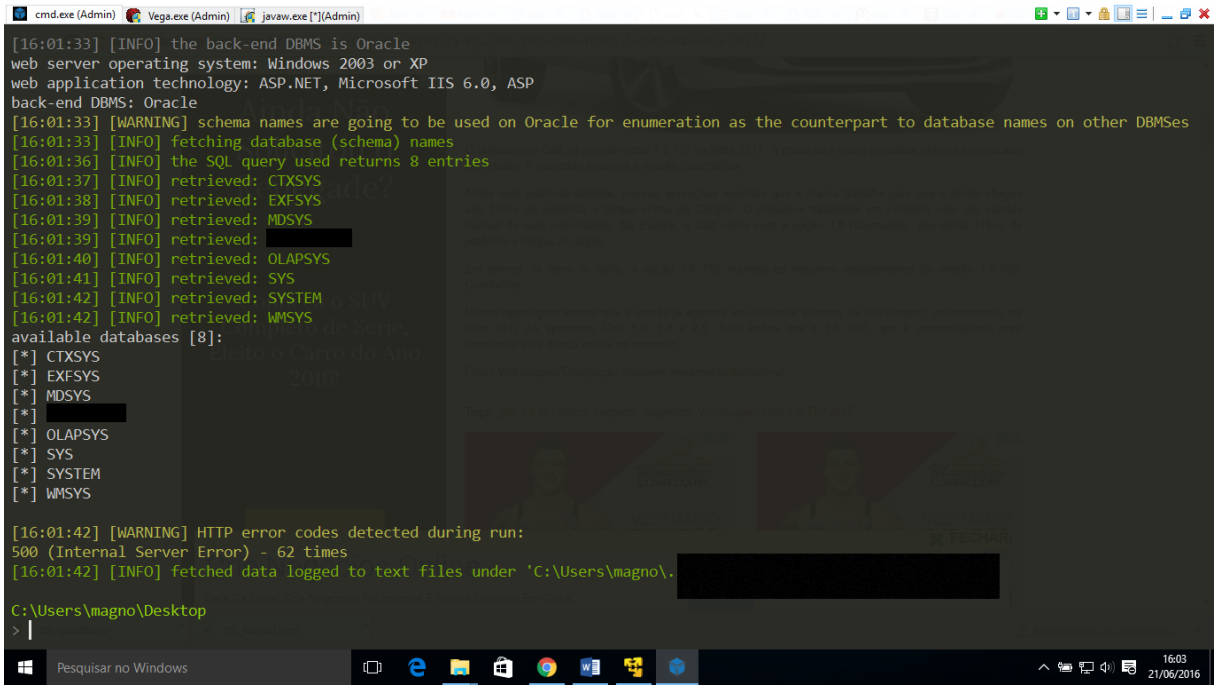

Figura 4.6: Listagem dos esquemas (*schemas*) da base de dados - sistema Caso I.

- Listar e acessar as tabelas e dados do sistema Caso I, conforme apresentado na Figura 4.7.
- Realizar o backup do banco de dados.
- Caso um cibercriminoso, tivesse realizado esse ataque, ele poderia: explorar os esquemas (*schemas*), tabelas e dados armazenados; alterar ou deletar dados; realizar a copia dos dados; também poderia até danificar o sistema, fazendo uso de *malwares*.

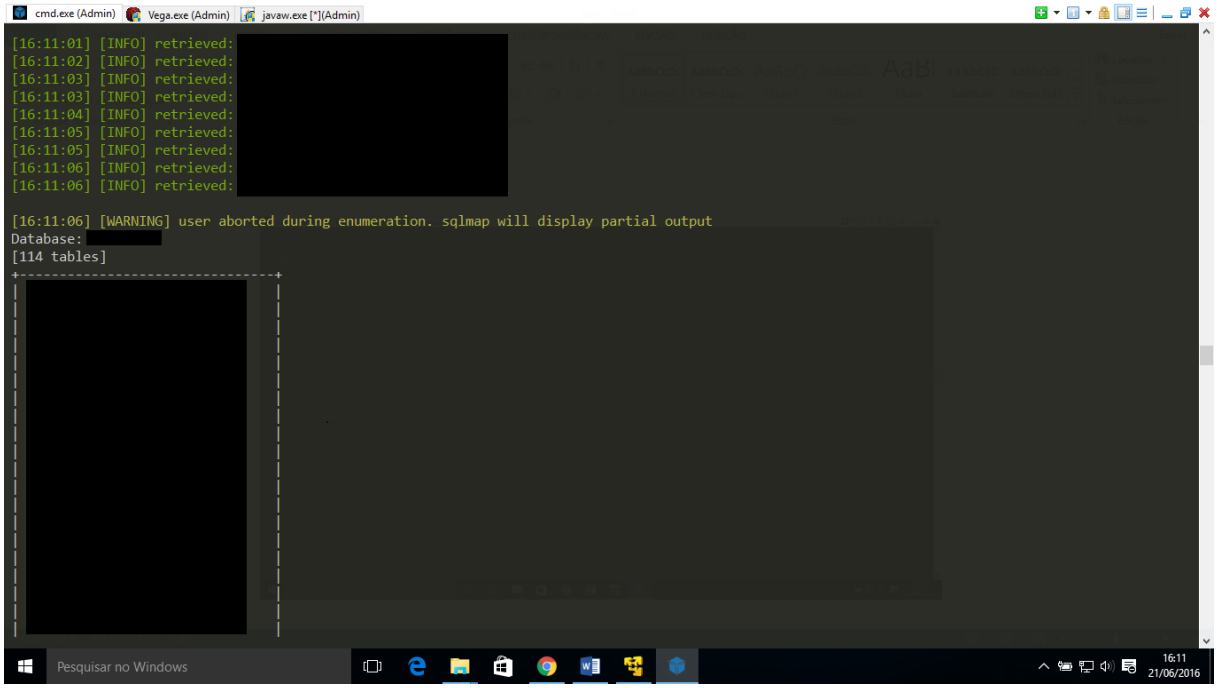

Figura 4.7: Exploração de tabelas e dados - Caso I.

4.2.5 Relatório

O Relatório é o produto final da auditoria, é o artefato que é entregue aos demandantes, nele consta às atividades executadas durante o processo de auditoria deste Caso I, inclusive às evidências coletadas, durantes o processo de auditoria. O relatório original foi entregue aos Gestores responsáveis pela unidade governamental, para ciência dos resultados e tomada de decisão, algumas informações extraídas do relatório original, foram discretizadas ou suprimidas, devido preservação e sigilo institucional. Os resultados obtidos deste Caso I, são apresentados na seção Resultados obtidos deste Capitulo.

4.3 Caso II

O Caso II - apresenta o uso da metodologia de Auditoria de Segurança da informação aplicada ao Sistema colaborativo de informações eletrônicas. Os Gestores demandantes escolheram este sistema, devido sua importância para a instituição, pois trata-se de um sistema novo, que está fase de implantação. Observação: devido a preservação e sigilo institucional, algumas informações e evidências foram discretizadas ou suprimidas.

4.3.1 Planejamento

Fase Planejamento do Caso II,

- Escopo: Sistema colaborativo de informações eletrônicas.
- Realização: Interna.
- Período: Horário de expediente em junho de 2016.
- Forma de Execução: Parcial (Análise de Vulnerabilidades)
- Tipo: Caixa Preta (informação recebida somente URL do sistema)

4.3.2 Descoberta

Reconhecimento e Coleta de informações

Objetivo desta fase é levantar e coletar de informações sobre o sistema alvo da auditoria, delimitando o escopo para que não ultrapasse para a fronteira de outros sistemas ou serviços, que não façam parte da estrutura do mesmo. Nesta fase são levantadas diversas informações como: IP, linguagem de programação, servidor web, serviços, portas, arquitetura, visando obter o máximo de detalhes possíveis como: a versão de componentes e serviços. A partir dessas informações são verificadas a existência de possíveis vulnerabilidades. No Caso II, foram utilizados os mesmos mecanismos e ferramentas do Caso I, para obtenção de informações. Dentre as informações coletadas citamos: a linguagem PHP e o servidor de aplicação web Apache. A Figura 4.8 apresenta, o processo de coleta de informações do sistema Caso II.

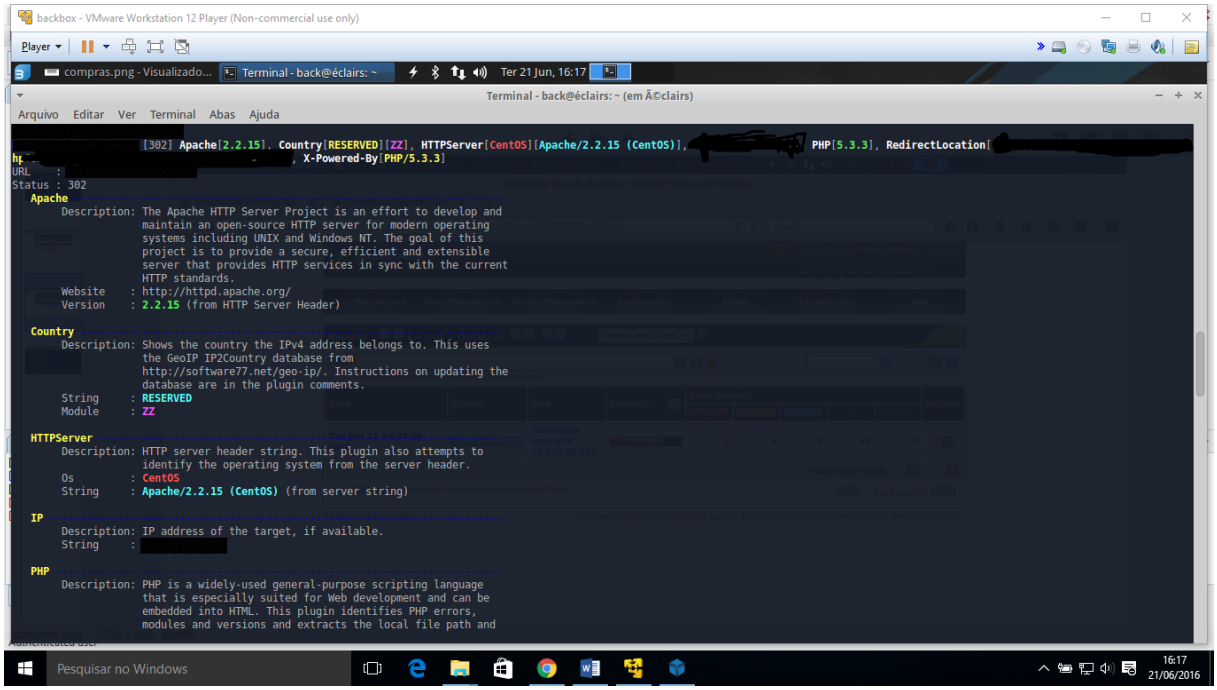

Figura 4.8: Coleta de Informações Caso II.

Varredura de portas e serviços

Este processo contempla a varredura de portas, verificando status das portas (aberta, fechada ou filtrada), serviços e protocolos em uso no servidor de aplicação do sistema [68]. A obtenção dessas informações, são de suma importância para as fases seguintes, pois possibilita a descoberta e identificação de possíveis vulnerabilidades de acordo com a versão dos componentes em uso [81]. A Figura 4.9 apresenta: a varredura de portas, serviços e protocolos e sistema operacional do sistema Caso II.

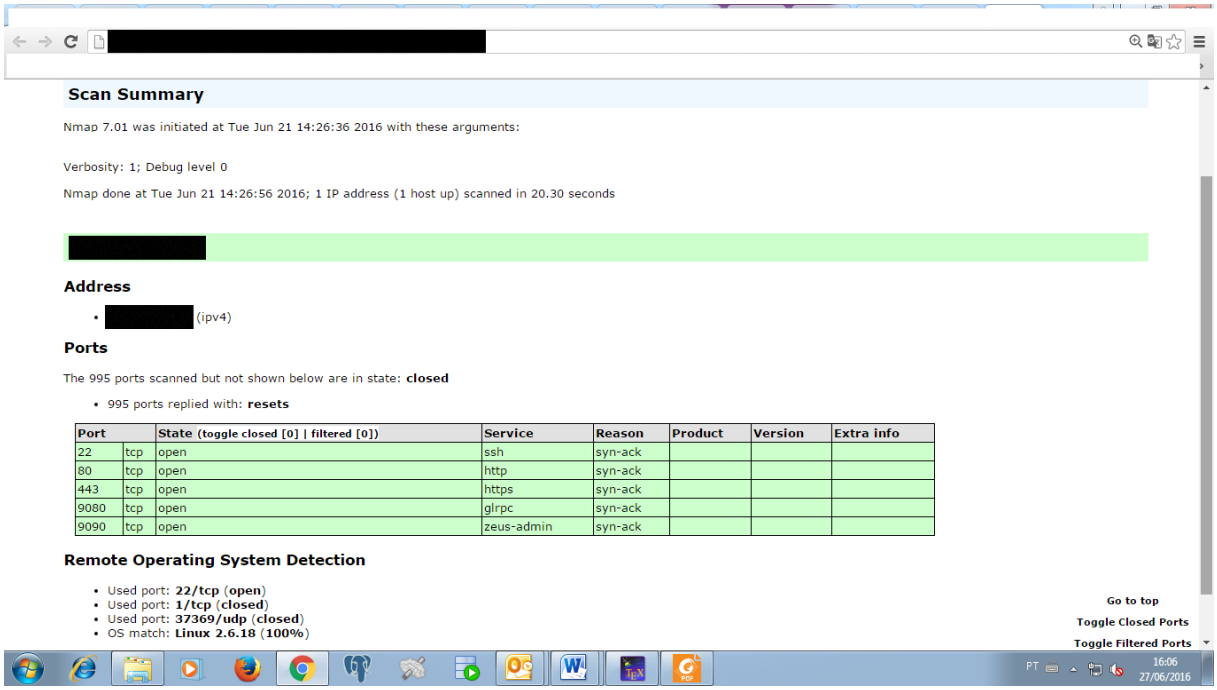

Figura 4.9: Varredura de portas, serviços e protocolos - Caso II.

4.3.3 Avaliação

Esta etapa apresenta as vulnerabilidades detectadas no sistema Caso II, sendo classificadas em grau de criticidade como: Alto, Médio e Baixo. Em casos de vulnerabilidades classificadas com grau de risco Alto (High), requerem atenção especial, pois podem ser exploradas com mais facilidade e o grau de impacto são severos, podendo um cibercriminoso ter acesso e controle total sobre a maquina alvo. Algumas vezes, pode ocorrer que uma vulnerabilidade mesmo com grau de risco Alto, não seja explorável, devido o conjunto de estruturas que promovem a segurança como: IDS, IPS, *firewall* [82], ou pela falta de sucesso no ataque impetrado. A ferramenta utilizada foi o *scanner* de vulnerabilidades VEGA [70], está ferramenta é *open source* e possui opções de execução manual, automatizada e modo hibrido.

Analise de Vulnerabilidade

Nesta etapa do Caso II, foi detectado apenas uma (01) vulnerabilidade no sistema, a qual possui grau Médio de risco. A vulnerabilidade detectada foi: suporte ao HTTP *trace* - esse método simplesmente ecoa ao cliente, qualquer sequência de caracteres, que foram enviados para o servidor e é usado principalmente para fins de depuração (diagnósticos). Este método, originalmente era considerado inofensivo, mas pode ser usado para montar um ataque conhecido como : *Cross Site Tracing* (XST) - que opera na requisição do pro-

tocolo HTTP enviada para o servidor, mas ao invés de enviar a requisição com o método *GET*, envia com método TRACE, o navegador (*browse*) devolve em uma janela javascript, todos os dados do cabeçalhos, como exemplo: *cookies* e credencias que deveriam ser enviados somente ao servidor, essa técnica de ataque foi descoberta pelo pesquisador Jeremiah Grossman em 2003 [83], que segundo o pesquisador, essa vulnerabilidade também pode ser explorada em conjunto com o ataque XSS. A Figura 4.10 apresenta a vulnerabilidade detectada, no sistema Caso II.

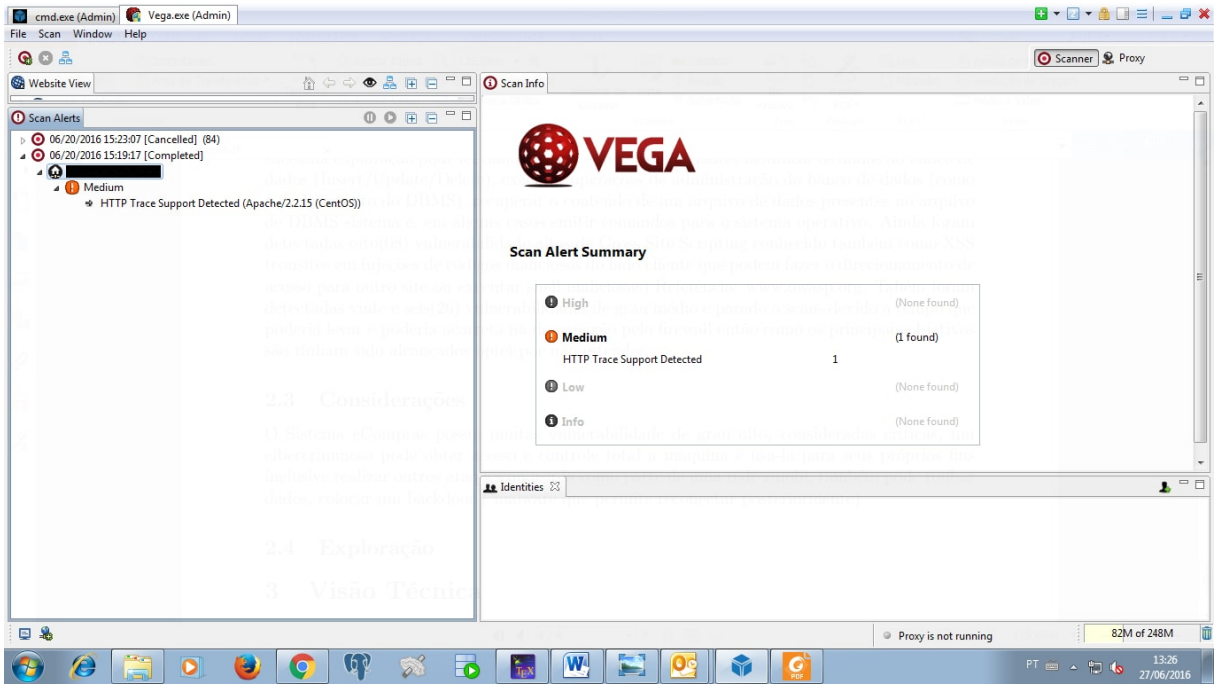

Figura 4.10: Vulnerabilidade detectada no sistema Caso II.

Varredura da Plataforma e componentes

Conforme foi apresentado no Capitulo 2, na seção de taxonomia de ataques Web, que inclui também a verificação da plataforma de aplicações web, portanto também verificamos a plataforma e componentes do sistema Caso II. Nesta etapa realizou-se o escaneamento do servidor de aplicação, estruturas e componentes como: servidor web, contêineres, serviços, linguagens e sistema operacional. Utilizou-se também procedimentos manuais e automatizados e a ferramenta OPENVAS [72] que é um *framework open source* e foi utilizado nos procedimentos de verificação de vulnerabilidade. Na varredura da plataforma foram detectadas quatorze (14) vulnerabilidades, sendo: **duas (02) com grau de risco Alto (High)**, referente as configurações do PHPInfo; **dez (10) com grau de risco Médio (*Medium*)**, referente as configurações do PHPConfig; **duas (02) com grau de**

risco Baixo (*Low*), estas duas ultimas são referente a alertas de configurações. A Figura 4.11 mostra as vulnerabilidades detectadas: na plataforma, estrutura e componentes - sistema Caso II.

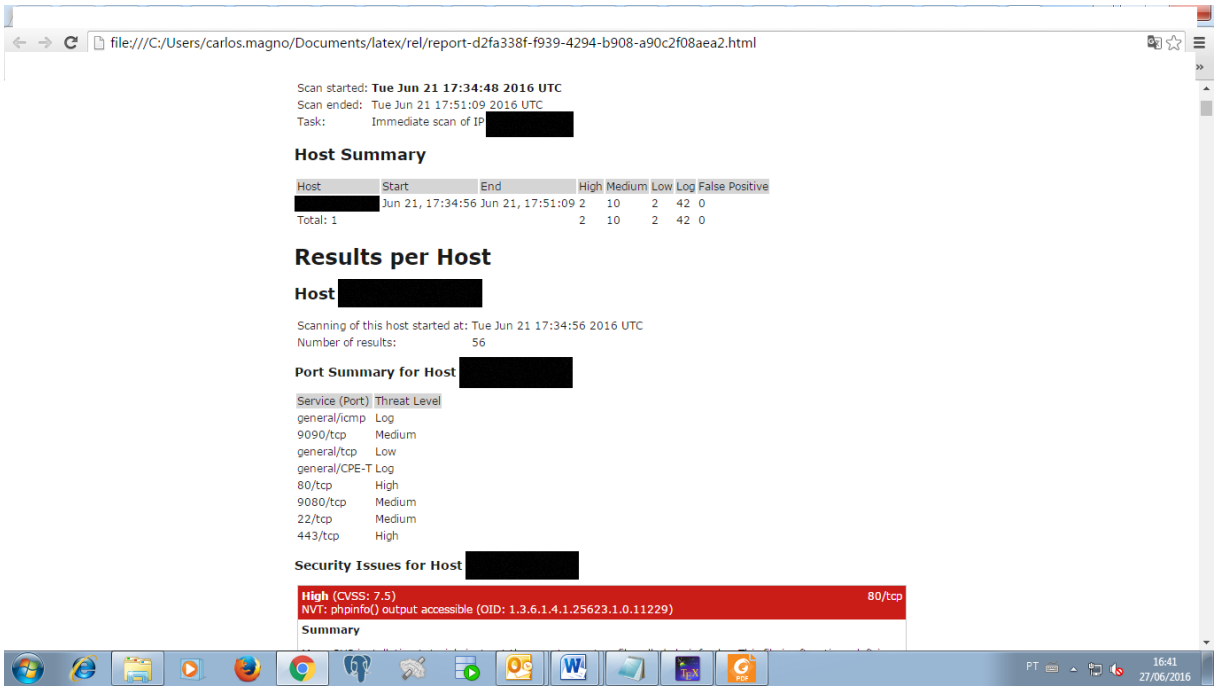

Figura 4.11: Vulnerabilidades plataforma, estrutura e componentes - sistema Caso II.

4.3.4 Relatório

O Relatório é o produto final da auditoria, é o artefato que é entregue aos demandantes, nele consta às atividades executadas durante o processo de auditoria deste Caso II, inclusive às evidências coletadas, durante o processo de auditoria. O relatório original foi entregue aos Gestores responsáveis pela unidade governamental, para ciência dos resultados e tomada de decisão, algumas informações do extraídas do relatório original, foram discretizadas ou suprimidas, devido preservação e sigilo institucional. Os resultados obtidos deste Caso II, são apresentados na seção Resultados obtidos deste Capitulo.

4.4 Caso III

O Caso III - apresenta o uso da metodologia de auditoria de segurança da informação aplicada ao Sistema de aquisição de kits pedagógicos. Observação: devido a preservação e sigilo institucional, algumas informações e evidências foram discretizadas ou suprimidas.

4.4.1 Planejamento

Fase Planejamento do Caso III,

- Escopo: Sistema de aquisição de kits pedagógicos.
- Realização: Externa,
- Período: Entre Janeiro e Fevereiro 2017.
- Forma de Execução: Completa
- Tipo: Caixa Preta (informação recebida somente URL do sistema)

4.4.2 Descoberta

Reconhecimento e Coleta de informações

Objetivo desta fase é a coleta de informações, sobre o sistema alvo da auditoria, delimitando o escopo para que não ultrapasse para a fronteira de outros sistemas ou serviços, que não façam parte da estrutura do mesmo. Nesta fase são levantadas diversas informações como: IP, linguagem de programação, servidor web, serviços, portas, arquitetura, visando obter o máximo de detalhes possíveis como: a versão de componentes e serviços. A partir dessas informações são verificadas a existência de possíveis vulnerabilidades. Na coleta de informações deste Caso III, realizamos o mapeamento do site para descoberta de paginas existentes, inclusive identificar paginas com formulários, o processo para coleta de links e paginas de um web site é chamado de *Crawler*. A Figura 4.12 apresenta o mapeamento das paginas do sistema Caso III.

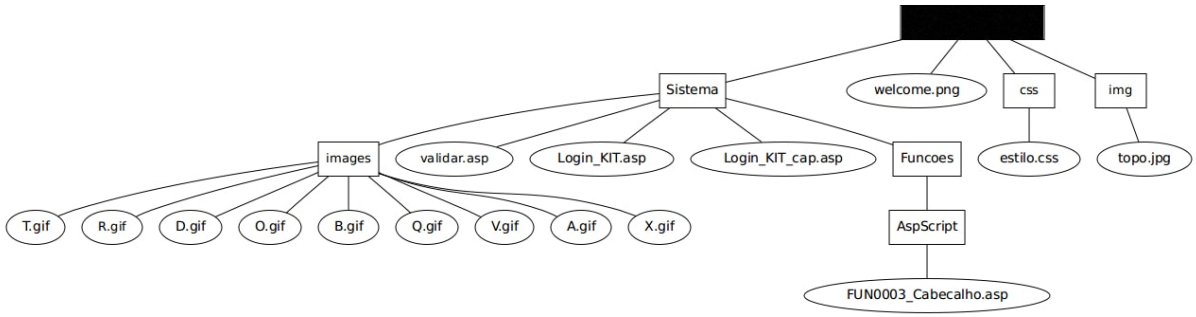

Figura 4.12: Mapeamento do sistema Caso III.

Escaneamento de portas e serviços

Foi realizado a execução da varredura de portas, verificando status das portas, serviços e protocolos e sistema operacional. Foram detectadas varias portas abertas, sistema operacional e o servidor de aplicação IIS da Microsoft. Abaixo apresentamos parte do log refente a coleta de informações do sistema Caso III.

```
Host is up (0.0023s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE       VERSION
80/tcp    open  http          Microsoft IIS httpd 7.5
| http-methods: Potentially risky methods:
|_See http://nmap.org/nsedoc/scripts/http-methods.html
|_http-title: IIS7
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn
445/tcp   open  netbios-ssn
3389/tcp  open  ms-wbt-server Microsoft Terminal Service
8010/tcp  open  http          Microsoft IIS httpd 7.5
| http-methods: Potentially risky methods:
|_See http://nmap.org/nsedoc/scripts/http-methods.html
|_http-title: Site doesn't have a title (text/html).
49152/tcp open  msrpc         Microsoft Windows RPC
49153/tcp open  msrpc         Microsoft Windows RPC
49154/tcp open  msrpc         Microsoft Windows RPC
Device type: general purpose
Running: Microsoft Windows Vista|2008|7
OS details: Microsoft Windows Vista SP0 or SP1, Windows Server 2008 SP1,
```

or Windows 7, Microsoft Windows Vista SP2 or Windows Server 2008

4.4.3 Avaliação

Analise de Vulnerabilidade

Esta seção mostra as vulnerabilidades detectadas, por meio do processo de Analise de Vulnerabilidade. No caso III, a classificação das vulnerabilidades detectadas foi baseada no CVSS versão 3, conforme apresentado na Seção Analise de vulnerabilidade do Capítulo 2. A principal diferença é que padrão de classificação CVSS v2, possui três categorias de Risco: Alto, Médio, Baixo; e o padrão CVSS v3, possui quatro categorias de Risco: Critico, Alto, Médio, Baixo [8] [40]. As vulnerabilidades mostradas na Figura 4.13, já contemplam: o sistema, sua plataforma e componentes, conforme apresentado na taxonomia de ataques web do Capítulo 2, a plataforma que suporta uma aplicação, também deve ser verificada, pois no caso de uma exploração um cibercriminoso poderia obter controle da maquina e componentes onde a aplicação está hospedada.

Summary

Critical	High	Medium	Low	Info	Total
0	2	6	2	30	40

Details

Severity	Plugin Id	Name
High (7.5)	11139	CGI Generic SQL Injection
High (7.5)	43160	CGI Generic SQL Injection (blind, time based)
Medium (5.1)	18405	Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness
Medium (5.0)	44670	Web Application SQL Backend Identification
Medium (5.0)	57608	SMB Signing Disabled
Medium (4.3)	47831	CGI Generic Cross-Site Scripting (comprehensive test)
Medium (4.3)	57690	Terminal Services Encryption Level is Medium or Low
Medium (4.3)	58453	Terminal Services Doesn't Use Network Level Authentication (NLA)
Low (2.6)	26194	Web Server Uses Plain Text Authentication Forms
Low (2.6)	30218	Terminal Services Encryption Level is not FIPS-140 Compliant

Figura 4.13: Vulnerabilidades detectadas - sistema Caso III.

Conforme apresentado na Figura 4.13, foram detectas dez (10) vulnerabilidades, sendo: três (03) relacionadas ao sistema e sete (07) relacionada a plataforma, nenhuma vulnerabilidade critica foi detectada, em relação ao dado "30 info", são informações direcionadas para o Administrador de sistema e ou rede. Descrição das vulnerabilidade abaixo:

Software: - três (03), sendo: duas Alto risco e uma risco Médio:

- Duas (02) Alta - *Sql Injection*
- Uma (01) Media - CGI *Generic XSS (comprehensive test)* - paginas e formulários do servidor Web, não sanitizam adequadamente os strings de requisição de JavaScript. Ao alavancar essa falha, um invasor pode ser capaz de usar código HTML ou javascript arbitrário,

Plataforma - cinco (05) com risco Médio, descrição segundo [71]:

- Uma (01) - *Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness* - a versão do terminal *service* (ferramenta windows para acesso remoto) é vulnerável ao ataque man-in-the-middle, este ataque também é conhecido como homem no meio, consiste na interceptação de dados por um cibercriminoso, em que este intercepta os dados trocados, entre duas entidades, como cliente e servidor.
- Uma (01) - *Web application SQL backend identification* - um aplicativo *back-end* SQL, pode ser identificado, esta informação é útil para um cibercriminoso ajustar seu ataque contra um alvo.
- Uma (01) - *Signing SMB Disabled* - assinatura do SMB está desabilitada, a assinatura não é necessária no servidor SMB remoto, um invasor remoto não autenticado, pode explorar essa falha, para realizar ataque *man-in-the-middle*.
- Uma (01) - *Terminal Services Encryption Level is Medium or Low* - Nível de criptografia do terminal *service* é médio ou baixo. O serviço do terminal remoto, não está configurado para usar criptografia forte.
- Uma (01) - *Terminal Services Doesn't Use Network Level Authentication* (NLA) - o terminal *service*, não está configurado, para usar autenticação de nível de rede NLA (*Network Level Authentication*), o NLA usa o protocolo CredSSP (*Credential Security Support Provider*) para realizar uma autenticação forte.

Plataforma - Duas (02) com Baixo risco, descrição segundo [71]:

- Uma (01) - *Web Server Uses Plain Text Authentication Form*s - refere-se a um servidor web remoto, que contém vários formulários HTML com campos, que contêm entrada do tipo "password", que transmitem suas informações para um servidor web em texto puro.

- Uma (01) - *Terminal Services Encryption Level is not FIPS-140 Complian* - O servidor não é compatível, com o padrão estipulado pelo FIPS (*Federal Information Processing Standard*), padrão de criptografia estipulado na publicação 140.

4.4.4 Exploração

A fim de atestar a suscetibilidade à ataques cibernéticos e mensurar os possíveis riscos e consequências destes, para o Caso III, foi realizado um ataque cibernético, visando explorar a vulnerabilidade de *SQL injection*, que possui grau de risco Alto. Os ataques de injeções de SQL ocorrem geralmente, por meio de injeções de expressões lógicas ou comandos SQL em: formulários de pagina da internet ou URL, associados a um script CGI, que não realiza a validação de entrada adequadamente [84]. Como mostra a Figura 4.14, ao invés de inserir o nome do usuário (carlos) e a senha de usuário (minha senha), foi inserida a expressão: " OR '1'='1' - vazio ou um, é igual a um, o que torna e expressão verdadeira porque 1=1 (um é igual a um). Conforme exemplo apresentado na Figura 4.14, os cibercriminosos podem usar diversas expressões e operações em seus ataques, a fim de obter acesso não autorizado,

O mesmo código apresentado na Figura 4.14, poderia ser reescrito com boas praticas de programação para evitar o ataque. Como exemplo: usando parametrização em tempo de execução, parâmetros SQL, são valores adicionados a uma consulta SQL, em tempo de execução [85]. O código abaixo foi reescrito em ASP.NET Razor - é uma *view engine* que já está incluída no WebMatrix (IDE), com ele temos a possibilidade de inserir a lógica da aplicação diretamente na camada de visualização do projeto. É possível inserir a *sintax razor*, junto com os códigos HTML dentro da mesma página [86]. No código abaixo os parâmetros são representados na instrução SQL, por um marcador "@", com isso o mecanismo SQL, verifica cada parâmetro para garantir que ele está correto para sua coluna e são tratados literalmente e não como parte do SQL a ser executado, sendo assim, os dados de entrada inseridos em uma query, serão lidos como texto e não como operações, que causariam ações inesperadas.

```
txtUserId = getRequestString("UserId");
txtPassID = getRequestString("PassID");
txtSQL = "SELECT * FROM Users WHERE UserId= @0 AND PassID=@1 ";
db.Execute(txtSQL,txtUserId,txtPassID);
```

Usando a logica apresentada na Figura 4.14, executamos varias sequencias, de inserções manuais, em um formulário do sistema Caso III. O teste de exploração foi executado com sucesso, obtivemos acesso a aplicação, usando injeção de SQL, realizado manualmente. A Figura 4.15 mostra a exploração bem sucedida, consequentemente o acesso não autorizado.

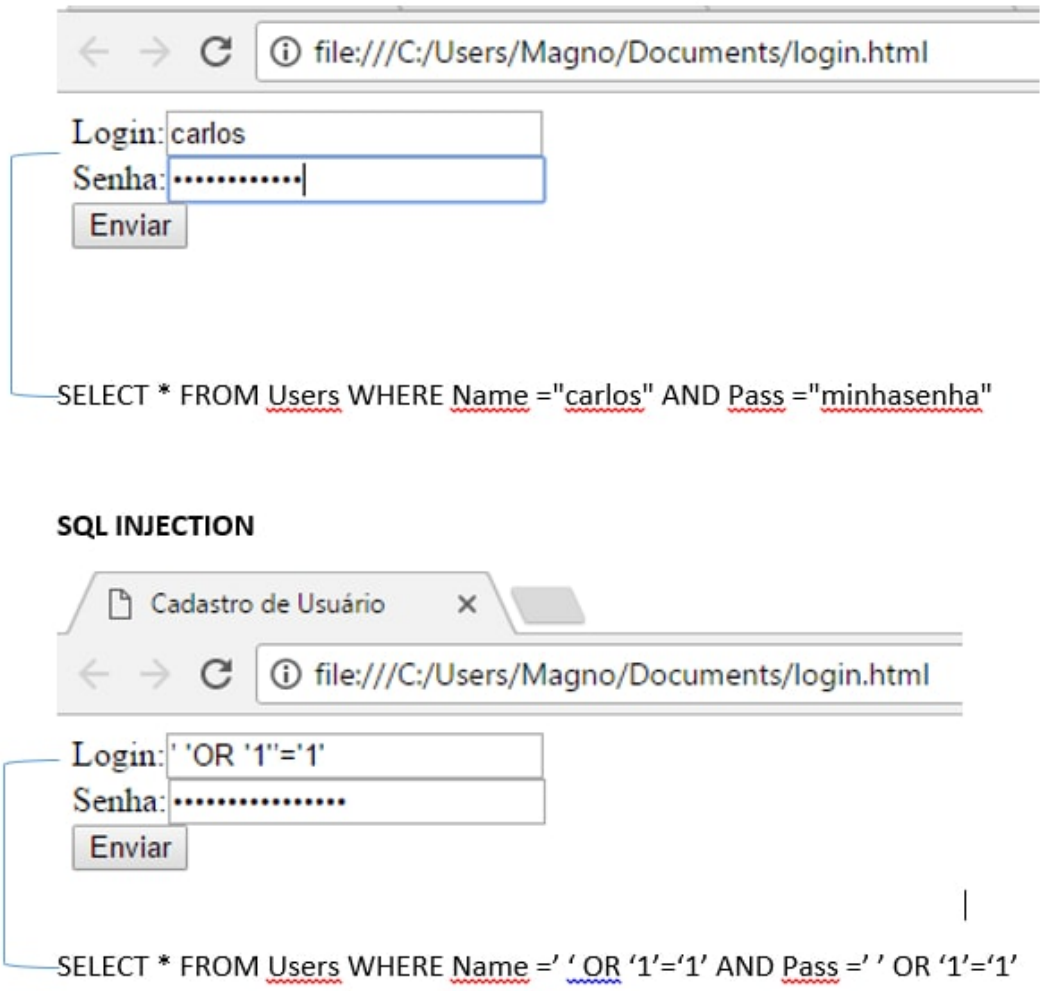

Figura 4.14: Exemplo SQL Injection manual.

Riscos e Consequências do ataque

Depois de obter acesso não autorizado, também foi possível acessar os dados dos usuários, conforme mostrar a Figura 4.16. Caso um cibercriminoso tivesse explorado está vulnerabilidade, o ataque poderia acarretar nas seguintes consequências:

- Obter acesso não autorizado ao sistema, conforme mostrado na Figura 4.15;
- Obter acesso aos dados do usuário, conforme mostrado na Figura 4.16;
- Copiar dados;
- Alterar ou apagar dados, inclusive poderia danificar o sistema.

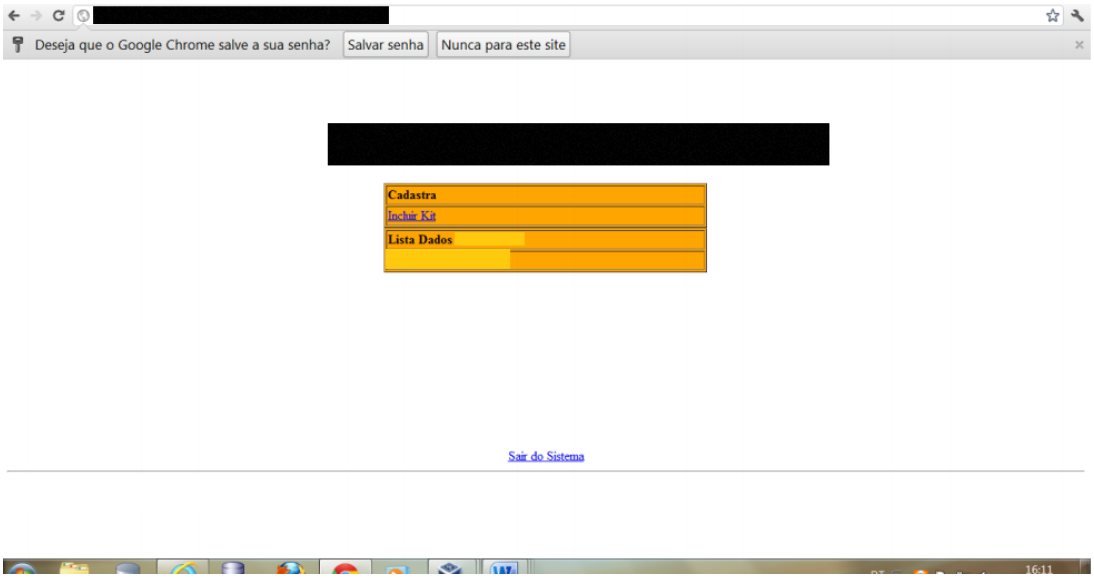

Figura 4.15: Exploração Caso III, obtenção de acesso não autorizado.

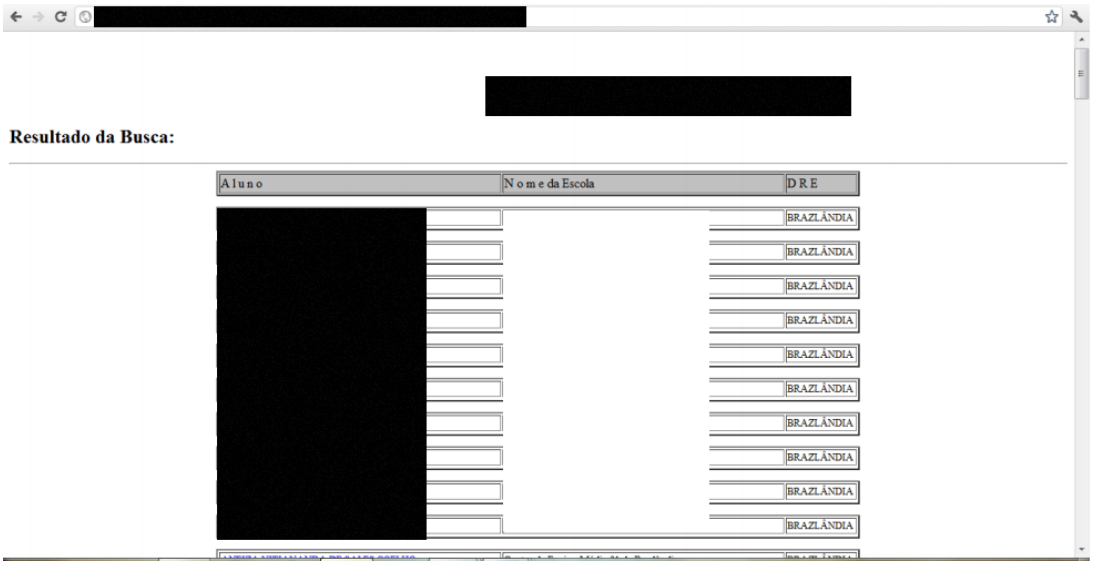

Figura 4.16: Exploração Caso III, Listagem de dados dos usuários.

4.4.5 Relatório

O Relatório é o produto final da auditoria, é o artefato que é entregue aos demandantes, nele consta às atividades executadas durante o processo de auditoria deste Caso III, inclusive às evidências coletadas, durantes o processo de auditoria. O Relatório original foi entregue aos Gestores responsáveis pela unidade governamental, para ciência dos resultados e tomada de decisão, algumas informações do relatório original, foram discretizadas ou suprimidas, devido preservação e sigilo institucional. Os resultados obtidos deste Caso III, são apresentados na seção Resultados obtidos deste Capitulo.

4.5 Resultados obtidos

4.5.1 Resultados obtidos Caso I

Tabela 4.1: Vulnerabilidades Caso I

Vulnerabilidade	Quantidade	Risco	Categoria
SQL Injection	50	Alto	Software
XSS	10	Alto	Software
SMB	1	Alto	Plataforma
URL Injection	26	Médio	Software
Forward Secrecy	1	Médio	Software
Cipher Suite	1	Médio	Software

A Tabela 4.1, apresenta às vulnerabilidades detectadas, no sistema Caso I, com suas respectivas classificações. No Caso I, ainda foi realizada a fase de Exploração, com a execução de um ataque cibernético o *SQL injection*, o qual foi executado com sucesso, para constatar a severidade das vulnerabilidades detectadas e avaliar as possíveis consequências, em caso de ataques reais, conforme apresentado na seção Exploração Caso I - item Riscos e Consequências do ataque. Os Gestores responsáveis pela unidade governamental, já estão tomando providências, para solucionar às vulnerabilidade expostas, neste Caso I.

4.5.2 Resultados obtidos Caso II

Tabela 4.2: Vulnerabilidades Caso II

Vulnerabilidade	Quantidade	Risco	Categoria
PHPInfo	02	Alto	Plataforma
HTTP Trace	01	Médio	Software
PHPConfig	10	Médio	Plataforma
Alerts Config	02	Baixo	Plataforma

A Tabela 4.2 apresenta as vulnerabilidades detectadas no sistema Caso II, com suas respectivas classificações. Conforme fase de Planejamento este caso não demandou execução da fase de Exploração, a intenção era obtenção de informações sobre o grau de segurança do mesmo e por se tratar de um sistema novo e estar em fase de implantação, foi considerado que somente a execução da etapa de Analise de Vulnerabilidade (forma de execução Incompleta) seria suficiente para avaliar a segurança do sistema Caso II, possibilitando as correções necessárias para sanar as vulnerabilidades detectadas.

4.5.3 Resultados obtidos Caso III

A Tabela 4.3 apresenta às vulnerabilidades detectadas no sistema Caso III, com suas respectivas classificações. Neste Caso III, também da foi realizado a etapa de Pentest, para realizar a

Tabela 4.3: Vulnerabilidades Caso III

Vulnerabilidade	Quantidade	Risco	Categoria
SQL Injection	02	Alto	Software
XSS	01	Médio	Software
Terminal Service	03	Médio	Plataforma
backend identification	01	Médio	Plataforma
Signing SMB	01	Médio	Plataforma
Text Authentication Forms	01	Baixo	Plataforma
Terminal Services	01	Baixo	Plataforma

exploração de uma vulnerabilidade de *SQL Injection*, com grau de risco Alto. A Exploração foi realizada com sucesso, para constatar a severidade do riscos das vulnerabilidades detectadas, sendo possível acessar o sistema sem autorização e ter acessos aos dados dos usuários, possibilitando inclusive a copia não autorizada dos dados e ou alteração, conforme apresentado na seção Exploração Caso III - item Riscos e Consequências do ataque. Os Gestores responsáveis pela unidade governamental, já estão tomando providências, para solucionar às vulnerabilidade expostas, neste Caso III.

4.6 Síntese do Capitulo

Este capitulo apresentou o estudo de caso de três sistemas governamentais, a aplicação da auditoria de segurança da informação foi realizada da seguinte forma: Caso I - Sistema de aquisição de produtos e serviços, Forma de Execução da Auditoria: Completa, resultados apresentados na Tabela 4.1; Caso II - Sistema colaborativo de informações eletrônicas, Forma de Execução da Auditoria: Parcial, resultados apresentados na Tabela 4.2; Caso III - Sistema de aquisição de kits pedagógicos, Forma de Execução da Auditoria: Completa, Resultados apresentados na Tabela 4.3.

Capítulo 5

Conclusões

Os serviços de tecnologia da informação, estão cada vez mais presentes nas rotinas diárias das pessoas, empresas e entidades governamentais, porém os ataques cibernéticos, também estão aumentando e alcançando diversas áreas onde a tecnologia da informação está presente. Os ataques cibernéticos, visam causar a indisponibilidade de sistemas e serviços, obter acesso não autorizado, roubo de dados, protestos, alusão a cibercriminosos, entre outras vantagens ilícitas. Por isso autoridades, entidades, profissionais, pesquisadores e governos, têm buscado soluções, para promover a segurança da informação e a segurança do espaço cibernético. Com este intuito este trabalho, apresentou o uso da Auditoria de Segurança da Informação em Sistemas e Aplicações, usando os processos de: Analise de Vulnerabilidade e Pentest (Teste de penetração de segurança), como estrategia proativa, visando: detectar, identificar, classificar vulnerabilidades e avaliar seus respectivos riscos, para possibilitar a remediação ou mitigação das vulnerabilidades detectadas. Os experimentos, foram realizados em três sistemas governamentais, como estudo de caso.

5.1 Métodos utilizados para atingir os objetivos específicos:

Para atingir o **primeiro objetivo especifico**, realizou-se pesquisa bibliográfica sobre ataque cibernético e taxonomia de ataques web, analise de vulnerabilidade e teste de penetração. Também realizou pesquisa em literatura e guias de referência ISO/IEC 2700(2014) [22], CVE [41], SCAP [7], CVSS [8]. Quanto a metodologia base utilizou-se como referencia o Guia Técnico de Segurança da Informação Testes e Avaliação (*Technical Guide to Information Security Testing and Assessment*) [19], o primeiro objetivo especifico foi atendido no Capítulo 2. Para atingir o **segundo objetivo especifico** utilizou-se o processo de Analise de Vulnerabilidade, que é o processo utilizado para detectar, identificar e classificar uma vulnerabilidade, bem como seu respectivo grau de risco, a Analise de vulnerabilidade foi aplicada nos Casos I, II e III. Em todos os casos de estudo, foram detectadas vulnerabilidades as quais foram identificadas, classificadas

de acordo com o seus respectivos graus de risco, utilizando como referencia CVE [41], SCAP [7], CVSS [8], OVAL [42], CPE [45], o segundo objetivo especifico foi atendido no Capítulo 4. No **terceiro Objetivo especifico**, utilizou-se o processo de Teste de Penetração (*Penetration Testing*, Pentest), que é o processo que consiste na realização de ataques cibernéticos em condições controladas e com consentimento (autorizado) para exploração de uma vulnerabilidade com intuito de avaliar a segurança de um sistema, para atingir este objetivo o Pentest foi aplicado ao Casos I e II Capítulo 4, em que verificou-se que as vulnerabilidades detectadas com grau de risco alto, podem de fato serem exploradas em casos de ataques cibernéticos reais. O **quarto objetivo especifico** foi atendido no Capítulo 4, na seção Resultados obtidos e conforme exposto nas tabelas: Tabela 4.1 resultado obtido Caso I, Tabela 4.2 resultado obtido Caso II e Tabela 4.3 resultado obtido Caso III, os resultados apresentados contribuem para tomadas de decisão dos gestores e pode auxilar a mitigação das vulnerabilidades detectadas.

5.2 Conclusões sobre os Casos de Estudo

No Caso I um numero expressivo de vulnerabilidades (conforme o exposto no Capítulo 4) principalmente com alto grau de risco. Uma solução para o problema, seria melhorar às técnicas de programação, aplicando boas praticas, exemplo: usar parametrização e escape de aspas simples, para entrada de dados fornecida pelo usuário, filtragem e limitação do comprimento dos caracteres de entrada [84]. Correções de grandes proporções no código da aplicação, podem demandar muito esforço e ou tempo da equipe técnica, por isso, enquanto não é corrigido ou substituído, sugerimos algumas remediações:

- Aplicar proteção baseada em camadas, como exemplo: uso de um WAF - web application firewall, que intercepta o fluxo vindo da internet para o servidor web, realizando monitoramento, filtragem e bloqueio de solicitações HTTP/HTTPS, para depois enviar as solicitações tidas como seguras ao servidor web [87], este tipo de proteção pode contribuir para mitigação de ataques *SQL injection*, XSS, Injecções em URl entre outros.
- Upgrade do Sistema operacional, se possível, pois sistemas operacionais antigos, tendem a perder suporte para correção de vulnerabilidade. E as versões mais novas geralmente apresentam aprimoramentos relacionados a segurança.
- Isolar a maquina vulnerável, monitorando, fechando ou filtrando a saída, para outras maquina da rede interna ou externa, pois se a maquina vulnerável for invadida, a mesma não dará acesso a outras maquinas da rede interna, e o tratamento de saída pode evitar, por exemplo: o download de malware, e também que a maquina vulnerável seja utilizada em ataques externos.
- Aplicar Hardening ao servidores do Caso I, este processo usa boas praticas, como exemplo: a ISO 27001, para efetuar: procedimentos de ajustes ou configurações, visando a correção ou mitigação de falhas de segurança.

No Caso II a maior parte das vulnerabilidades detectadas a maior parte das vulnerabilidades detectadas são relacionadas a configurações e ajustes da plataforma (conforme o exposto no Capítulo 4), sendo somente uma vulnerabilidade relacionada a aplicação. Para a vulnerabilidade detectada na aplicação (suporte ao método HTTP Trace), recomendamos a desativação do suporte a este método. Para as vulnerabilidade detectadas na plataforma, recomendamos o uso do processo de Hardening nos servidores do Caso II, este processo usa boas praticas, como exemplo: a ISO 27001, para efetuar: procedimentos de ajustes ou configurações, visando a correção ou mitigação de falhas de segurança.

O Caso III apresentou duas vulnerabilidades de software com grau de risco Alto, que podem ser exploradas e no caso de uma invasão, poderia ocorrer serias consequências, conforme demostrado na fase de Exploração. Portanto a correção ou mitigação das vulnerabilidades detectadas, para este Caso III, são fundamentais para preservar os dados dos usuários e a disponibilidade dos serviços oferecidos. Sugerimos algumas remediações:

- Aplicar boas praticas de programação, principalmente para validar os dados de entrada nos formulários, exemplo: usar parametrização e escape de aspas simples, para entrada de dados fornecida pelo usuário, filtragem e limitação do comprimento dos caracteres de entrada. Em caso de não haver possibilidade de correção imediata, outra solução seria utilizar proteção baseada em camadas, como exemplo: uso de um WAF - web application firewall, que intercepta o fluxo vindo da internet, para o servidor web, realizando monitoramento, filtragem e bloqueio de solicitações HTTP/HTTPS, para depois enviar as solicitações tidas como seguras ao servidor web [87], este tipo de proteção pode contribuir para mitigação de ataques SQL injection, XSS, Injeções em URl, entre outros.
- Isolar a maquina vulnerável, monitorando, fechando ou filtrando a saída, para outras maquina da rede interna ou externa, pois se a maquina vulnerável for invadida, a mesma não dará acesso a outras maquinas da rede interna, e o tratamento de saída pode evitar, por exemplo: o download de malware, e também que a maquina vulnerável seja utilizada em ataques externos.
- Aplicar Hardening ao servidores do Caso III, este processo usa boas praticas, como exemplo: a ISO 27001, para efetuar: procedimentos de ajustes ou configurações, visando a correção ou mitigação de falhas de segurança.

Este trabalho contribuiu, para identificação de vulnerabilidades e falhas de segurança apresentadas nos sistemas experimentados Casos I, II e III, os artefatos gerados (Relatórios) possibilitaram a tomada de decisão dos gestores responsáveis pela unidade governamental e também forneceu insumos para auxiliar a equipe técnica na mitigação ou remediação das vulnerabilidades detectadas. Concluirmos que atingimos os resultados esperados e os resultados obtidos, foram uteis e relevantes, para a unidade governamental objeto de estudo.

Referências

[1] Álvarez, Gonzalo e Slobodan Petrović: *A new taxonomy of web attacks suitable for efficient encoding.* Computers & Security, 22(5):435–449, 2003. x, 8, 12, 29

[2] Yadav, Sushilkumar e Chandrakant Navdeti: *Survey: Secured techniques for vulnerability assessment and penetration testing.* IJCSIT) International Journal of Computer Science and Information Technologies, 5(4):5132–5135, 2014. x, 15, 21, 28, 30

[3] Bou-Harb, Elias, Mourad Debbabi e Chadi Assi: *Cyber scanning: a comprehensive survey.* IEEE Communications Surveys & Tutorials, 16(3):1496–1519, 2014. x, 20

[4] Halfond, William GJ, Shauvik Roy Choudhary e Alessandro Orso: *Penetration testing with improved input vector identification.* Em *Software Testing Verification and Validation, 2009. ICST'09. International Conference on*, páginas 346–355. IEEE, 2009. x, 21, 24

[5] Shinde, Prashant S e Shrikant B Ardhapurkar: *Cyber security analysis using vulnerability assessment and penetration testing.* Em *Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), World Conference on*, páginas 1–5. IEEE, 2016. x, 2, 13, 14, 28, 30, 32

[6] PRESSMAN, Roger S: *Engenharia de Software.* 6ª Ed. Editora MCGrawHill: Porto Alegre, 2010. xi, 2, 3

[7] Radack, Shirley e Rick Kuhn: *Managing security: The security content automation protocol.* IT professional, 13(1):9–11, 2011. xi, 16, 17, 60, 61

[8] FIRST: *Common vulnerability scoring system (cvss).* Disponivel em:. `<https://www.first.org/cvss/>`, Acessado em: 07 mar. 2017. xi, 17, 18, 53, 60, 61

[9] Scarfone, Karen e Peter Mell: *The common configuration scoring system (ccss): Metrics for software security configuration vulnerabilities.* NIST Interagency Report, 7502, 2010. xi, 18

[10] Stefinko, Yaroslav, Andrian Piskozub e Roman Banakh: *Manual and automated penetration testing. benefits and drawbacks. modern tendency.* Em *Modern Problems of Radio Engineering. Telecommunications and Computer Science (TCSET), 2016 13th International Conference on*, páginas 488–491. IEEE, 2016. xi, 22, 23

[11] Liu, Bingchang, Liang Shi, Zhuhua Cai e Min Li: *Software vulnerability discovery techniques: A survey.* Em *Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on*, páginas 152–156. IEEE, 2012. xi, 13, 24

[12] Canongia, Claudia e Raphael Mandarino Junior: *Segurança cibernética: o desafio da nova sociedade da informação.* Parcerias Estratégicas, 14(29):21–46, 2010. 1

[13] Davanzo, Giorgio, Eric Medvet e Alberto Bartoli: *A comparative study of anomaly detection techniques in web site defacement detection.* Em *IFIP International Information Security Conference*, páginas 711–716. Springer, 2008. 1

[14] Mayer, Alain, Avishai Wool e Elisha Ziskind: *Fang: A firewall analysis engine.* Em *Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on*, páginas 177–187. IEEE, 2000. 1

[15] Elmallah, Ehab e Mohamed Gouda: *Hardness of firewall analysis.* IEEE Transactions on Dependable and Secure Computing, 2015. 1

[16] Bace, RG: *Intrusion detection: Defining intrusion detection.* Intrusion detection concepts, páginas 37–43, 2000. 1

[17] King, CM, CE Dalton e TE Osmanoglu: *Security architecture-design, development & operations, business and application drivers (case study).* Authorisation and Access Control, páginas 93–94, 2001. 1

[18] Pressman, Roger S: *Engenharia de software: uma abordagem profissional. 7ª ed.ª edição.* Ed: McGraw Hill, 2011. 2

[19] Scarfone, Karen, Murugiah Souppaya, Amanda Cody e Angela Orebaugh: *Technical guide to information security testing and assessment.* NIST Special Publication, 800:115, 2008. 2, 25, 26, 29, 30, 35, 60

[20] Fontes, Edison: *Praticando a segurança da informação.* Brasport, 2008. 4

[21] Parker, Donn B: *Toward a new framework for information security?* Computer Security Handbook, Sixth Edition, páginas 3–1, 2002. 6

[22] *Information technology - Security Techniques - Information security management systems - Overview and vocabulary.* 6, 60

[23] Rigon, Evandro Alencar e Carla Merkle Westphall: *Modelo de avaliação da maturidade da segurança da informação.* Revista Eletrônica de Sistemas de Informação ISSN 1677-3071 doi: 10.5329/RESI, 12(1), 2013. 7

[24] *ABNT NBR ISO/IEC 27001:2006: Tecnologia da informação – Técnicas de segurança – Sistemas de gestão de segurança da informação – Requisitos.* 7

[25] *Information technology - Security Techniques – Information security management systems - Requirements.* 7

[26] *ABNT. NBR ISO/IEC 27002:2005: Tecnologia da informação – Técnicas de segurança – Código de prática para a gestão da segurança da informação.* 7

[27] *Information technology - Security techniques - Code of practice for information security controls.* 7

[28] Disterer, Georg: *Iso/iec 27000, 27001 and 27002 for information security management.* 2013. 7

[29] Solms, Rossouw von: *Information security management (3): the code of practice for information security management (bs 7799).* Information Management & Computer Security, 6(5):224–225, 1998. 7

[30] Goldberg, Arthur, Robert Buff e Andrew Schmitt: *A comparison of http and https performance*. Computer Measurement Group, CMG98, 1998. 8

[31] Scott, David e Richard Sharp: *Abstracting application-level web security*. Em *Proceedings of the 11th international conference on World Wide Web*, páginas 396–407. ACM, 2002. 8

[32] LeBlanc, David e Michael Howard: *Writing secure code*. Pearson Education, 2002. 9

[33] Kolšek, Mitja: *Session fixation vulnerability in web-based applications*. Acros Security, página 7, 2002. 9

[34] Ozment, James Andrew: *Vulnerability discovery & software security*. Tese de Doutoramento, University of Cambridge, 2007. 13

[35] Borah, Chandra Kamal: *Cyber war: the next threat to national security and what to do about it? by richard a. clarke and robert k. knake*, 2015. 14

[36] Buja, Geogiana, Kamarularifin Bin Abd Jalil, Fakariah Bt Hj Mohd Ali e Teh Faradilla Abdul Rahman: *Detection model for sql injection attack: An approach for preventing a web application from the sql injection attack*. Em *Computer Applications and Industrial Electronics (ISCAIE), 2014 IEEE Symposium on*, páginas 60–64. IEEE, 2014. 14

[37] SANS, CSIS: *Critical security controls version 4.1*, 2009. 14

[38] Peltier, Thomas R, Justin Peltier e John A Blackley: *Managing a Network Vulnerability Assessment*. CRC Press, 2003. 14

[39] Alhazmi, Omar H, Sung Whan Woo e Yashwant K Malaiya: *Security vulnerability categories in major software systems*. Em *Communication, Network, and Information Security*, páginas 138–143, 2006. 16, 18

[40] NIST: *National vulnerability database*. Disponivel em:. `<https://nvd.nist.gov/>`, Acessado em: 07 mar. 2017. 16, 19, 53

[41] MITRE: *Common vulnerabilities and exposures (cve)*. Disponivel em:. `<http://www.cve.mitre.org/>`, Acessado em: 07 mar. 2017. 16, 17, 60, 61

[42] MITRE: *Open vulnerability and assessment language (oval)*. Disponivel em:. `<https://oval.mitre.org/>`, Acessado em: 07 mar. 2017. 16, 17, 61

[43] Nakamura, Akihito: *Towards unified vulnerability assessment with open data*. Em *Computer Software and Applications Conference Workshops (COMPSACW), 2013 IEEE 37th Annual*, páginas 248–253. IEEE, 2013. 16

[44] NIST: *Extensible configuration checklist description format (xccdf)*. Disponivel em:. `<https://scap.nist.gov/specifications/xccdf/>`, Acessado em: 07 mar. 2017. 17

[45] NIST: *Common platform enumeration (cpe)*. Disponivel em:. `<https://scap.nist.gov/specifications/cpe/>`, Acessado em: 07 mar. 2017. 17, 61

[46] MITRE: *Common configuration enumeration (cce)*. Disponivel em:. `<https://cce.mitre.org/>`, Acessado em: 07 mar. 2017. 17

[47] hat, Red: *Severity ratings*. Disponivel em:. `<https://access.redhat.com/security/updates/classification>`, Acessado em: 27 mar. 2017. 18

[48] Bhuyan, Monowar H, DK Bhattacharyya e Jugal K Kalita: *Surveying port scans and their detection methodologies.* The Computer Journal, página bxr035, 2011. 20

[49] Bishop, Matt: *About penetration testing.* IEEE Security & Privacy, 5(6), 2007. 21

[50] Kranthi Kumar, K. Srinivasa Rao: *A latest approach to cyber security analysis using vulnerability assessment and penetration testing.* International Journal of Emerging Research in Management and Technology, 3, 2014. 21, 28, 30

[51] Shah, Sugandh e BM Mehtre: *A modern approach to cyber security analysis using vulnerability assessment and penetration testing.* Int J Electron Commun Comput Eng, 4(6):47–52, 2013. 21, 28, 30, 32, 33, 34

[52] *OSSTMM - Open Source Security Testing Methodology.* 21, 22

[53] Hansson, Sven Ove: *A note on social engineering and the public perception of technology.* Technology in society, 28(3):389–392, 2006. 22

[54] Engebretson, Patrick: *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy.* Elsevier, 2011. 26, 27

[55] Harper, Allen, Shon Harris, Jonathan Ness, Chris Eagle, Gideon Lenkey e Terron Williams: *Gray Hat Hacking The Ethical Hackers Handbook.* McGraw-Hill Osborne Media, 2011. 26

[56] Allen, Lee: *Advanced Penetration Testing for Highly-Secured Environments: The Ultimate Security Guide.* Packt Publishing Ltd, 2012. 27

[57] Klevinsky, Thomas J, Scott Laliberte e Ajay Gupta: *Hack IT: security through penetration testing.* Addison-Wesley Professional, 2002. 27

[58] Wilhelm, Thomas: *Professional penetration testing: Creating and learning in a hacking lab, P.91-94.* Newnes, 2013. 27

[59] Rights, Retains Full: *Use offense to inform defense. find flaws before the bad guys do.* 2003. 27

[60] Stuttard, Dafydd e Marcus Pinto: *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws.* John Wiley & Sons, 2011. 27

[61] Beaver, Kevin: *Hacking for dummies.* John Wiley & Sons, 2013. 28

[62] Shah, Sugandh e BM Mehtre: *A reliable strategy for proactive self-defence in cyber space using vapt tools and techniques.* Em *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*, páginas 1–6. IEEE, 2013. 28, 30, 31

[63] Whitman, Michael E e Herbert J Mattord: *Principles of information security.* Cengage Learning, 2011. 29

[64] Xynos, Konstantinos, Iain Sutherland, Huw Read, Emlyn Everitt e Andrew JC Blyth: *penetration testing and vulnerability assessments: A professional approach.* 2010. 30

[65] ASSUNÇÃO, Marcos F: *Segredos do hacker ético.* 2014. 32

[66] Türpe, Sven e Jörn Eichler: *Testing production systems safely: Common precautions in penetration testing.* Em *Testing: Academic and Industrial Conference-Practice and Research Techniques, 2009. TAIC PART'09.*, páginas 205–209. IEEE, 2009. 33

[67] Project, Nmap: *The nmap security scanner*. Disponivel em:. <https://nmap.org/>, Acessado em: 14 agosto. 2017. 36

[68] Lyon, Gordon Fyodor: *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure, 2009. 36, 42, 47

[69] Horton, Andrew e Brendan Coles: *Whatweb*. Disponivel em:. <https://www.morningstarsecurity.com/research/whatweb>, Acessado em: 14 agosto. 2017. 37, 39

[70] Subgraph: *Vega*. Disponivel em:. <https://subgraph.com/vega/>, Acessado em: 14 agosto. 2017. 37, 40, 48

[71] Tenable: *Informação sobre vulnerabilidade relacionadas ao id plugin e cve*. Disponivel em:. <http://www.tenable.com/plugins/?view=all/>, Acessado em: 08 mar. 2017. 37, 54

[72] Greenbone: *Openvas*. Disponivel em:. <http://www.openvas.org/>, Acessado em: 14 agosto. 2017. 37, 49

[73] Riancho, Andrés: *W3af*. Disponivel em:. <http://w3af.org/>, Acessado em: 14 agosto. 2017. 38

[74] Bernardo Damele, Assumpcao Guimaraes e Miroslav Stampar: *Sqlmap*. Disponivel em:. <http://sqlmap.org/>, Acessado em: 14 agosto. 2017. 38

[75] OWASP, Top: *Top 10–2013*. The Ten Most Critical Web Application Security Risks, 2013. 40

[76] Anley, Chris: *Advanced sql injection in sql server applications*, 2002. 41

[77] Grossman, Jeremiah: *XSS Attacks: Cross-site scripting exploits and defense*. Syngress, 2007. 41

[78] Schum, Chris: *Correctly implementing forward secrecy*. Disponivel em:. <https://www.sans.org/reading-room/whitepapers/bestprac/correctly-implementing-secrecy-35842>, Acessado em: 08 março 2017. 42

[79] Ross, Julio: *Redes de computadores*. Julio Ross, 2008. 42

[80] Newson, Alan: *Network threats and vulnerability scanners*. Network Security, 2005(12):13–15, 2005. 43

[81] Panjwani, Susmit, Stephanie Tan, Keith M Jarrin e Michel Cukier: *An experimental evaluation to determine if port scans are precursors to an attack*. Em *2005 International Conference on Dependable Systems and Networks (DSN'05)*, páginas 602–611. IEEE, 2005. 47

[82] Akujuobi, C. M., N. K. Ampah e M. N. O. Sadiku: *Application of wavelets and self-similarity to enterprise network intrusion detection and prevention systems*. Em *2007 IEEE International Symposium on Consumer Electronics*, páginas 1–6, 2007. 48

[83] Grossman, Jeremiah: *Cross-site tracing (xst)*. WhiteHat Security White Paper, 2003. 49

[84] Boyd, Stephen W e Angelos D Keromytis: *Sqlrand: Preventing sql injection attacks*. Em *International Conference on Applied Cryptography and Network Security*, páginas 292–302. Springer, 2004. 55, 61

[85] W3SCHOOLS: *Sql injection*. Disponivel em:. <https://www.w3schools.com/sql/sql_injection.asp>, Acessado em: 09 abril. 2017. 55

[86] Sato, Alexandre Tadashi: *Introdução ao asp.net razor*. Disponivel em:. <https://msdn.microsoft.com/pt-br/library/gg675215.aspx>, Acessado em: 09 abril. 2017. 55

[87] Gupta, Namit, Abakash Saikia e D Sanghi: *Web application firewall*. Indian Institute of Technology, Kanpur, 2007. 61, 62