Referencial Teórico e Metodologia

**2. Fundamentação Teórica**

**2.1 Segurança da Informação**

O termo Segurança da Informação, normalmente associado a questões como Vulnerabilidade de Sistemas, se refere a todos os possíveis mecanismos de controle e proteção disponibilizados por uma instituição para defender sua Informação.

Informação é um ativo importante para a organização e os meios de defendê-la são bem variados podendo ser mecanismos lógicos, físicos ou até políticos.

**2.2 Norma ISO 27002**

É a norma definida pela ABNT para regulamentar a Segurança da Informação e quais seriam os procedimentos a serem adotados para garantir a segurança dos ativos.

A norma cita desde segurança da informação até controles, políticas, objetivos e até descrevendo ambientes para garantir que a informação esteja segura.

**2.3 Aplicações Web**

São sistemas desenvolvidos para responder requisições HTTP usando a arquitetura cliente-servidor. Esses sistemas são hospedados por servidores web que recebem e transferem todas as requisições recebidas para a aplicação.

Essas aplicações podem ser divididas em dois tipos: Estáticas, onde apenas são disponibilizados conteúdos estáticos em formato Html ou dinâmicas (interativas) onde para cada usuário que acesse, a aplicação comporta-se de forma diferente.

Exemplos de aplicações não faltam: E-commerces, sites de bancos, do governo, redes sociais etc.

**2.4 Tipos de ataques mais comuns em aplicações Web**

**2.4.1 Força Bruta**

É o tipo de ataque baseado na tentativa/erro onde as credenciais de acesso a dados protegidos, software ou sistema são encontradas baseadas em palpites Caso a tentativa consiga ser efetiva, dá ao atacante total acesso a informações, permissões e credenciais da vítima dentro do sistema alvo.

Como se trata de um longo processo de tentativas é comum que sejam utilizadas, no processo, ferramentas automatizadas aliadas ao uso de algumas técnicas que reduzam o espaço de amostras. Dessa forma o processo para descobrimento de credenciais torna-se mais rápido.

A técnica mais comum utiliza uso de dicionários que são amostras escolhidas pelo atacante baseado no prévio conhecimento dos padrões de senha ou de gostos pessoais da vítima. [citação cartilha]

**2.4.2 SQL Injection**

É um ataque que explora a falta de tratamento dos parâmetros textuais enviados a uma aplicação. A injeção de SQL pode ocorrer através da passagem de instruções SQL em parâmetros textuais utilizados por um sistema, a fim de executar ações não autorizadas no banco de dados.

O ataque só tem êxito devido ao interpretador SQL que todos os bancos possuem para converter o texto advindo das aplicações em instruções SQL. Como o nome indica, o interpretador apenas recebe uma informação e instantaneamente processa para transforma-la em consultas SQL, não havendo processo de compilação. Portanto, o trabalho do interpretador se limita a executar o que recebe causando uma brecha, caso os dados recebidos não sejam devidamente tratados.

Tendo como exemplo a consulta SQL abaixo:

select \* from usuarios where id = 1;

Se o parâmetro *“Id”* não for devidamente tratado, o texto enviado pela aplicação pode ser interpretado pelo interpretador de SQL do banco de dados e consultas maliciosas como essa conseguem ser executadas:

select \* from usuarios where id = 1 **and 1=1;--**;

Em caso de sucesso, o SQL injection dá ao atacante acesso direto ao banco podendo conseguir executar códigos maliciosos para: descobrir nomes e colunas das tabelas, descobrir dados cadastrados na tabela, apagar tabelas ou até mesmo o banco, etc.

**2.4.3 XSS**

É o acrônimo de Cross-Site Scripting, é a execução de scripts maliciosos dentro de um site confiável. As formas mais comuns de apresentação do ataque são: A Persistida e Refletida.

Persistida é quando um código javascript é inserido de forma permanente no banco de dados da aplicação devido à falta de validação da entrada de texto que contém um código javascript. No momento que o usuário acessa a página atacada, o código é trazido do banco e interpretado pelo navegador acreditando ser um código da aplicação. Um bom exemplo disso é o trecho abaixo:

*<script> alert(“Hello XSS”);</script>*

Caso o código consiga ser inserido no banco via aplicação, cada pessoa que acessar a página atacada verá um pop-up abrir com o texto *“Hello XSS”* sendo exibido, sem saber que foi realmente alvo de um ataque.

Já a forma Refletida, faz uso de uma chamada realizada ao servidor para incluir um código malicioso que também é executado no lado cliente. Tomando como exemplo uma aplicação que receba um parâmetro sem tratamento pela URL e exiba-o na tela como indicado abaixo:

*www.meusitevulneravel.com.br?parametroInjetavel=teste*

E sendo exibido na tela assim:

*echo ‘<h1>‘ + getParameter(‘parametroInjetavel’) + ‘</h1>';*

Caso o parâmetro *parametroInjetavel* não receba o tratamento adequado, um código malicioso pode ser injetado por um atacante e ser interpretado e executado pelo browser no lado do cliente. Como abaixo:

*www.meusitevulneravel.com.br?parametroInjetavel=<script> alert(“Hello XSS”);</script>*

O código acima, exibiria o mesmo pop-up do exemplo de XSS refletido. Devido à falta de registros armazenados o XSS Refletido é bem mais difícil de ser executado, porém é bem trabalhoso para identificar o ataque. Já houveram registros desse tipo de ataque no Brasil ao site do Santos FC, porém com o intuito de fazer uma brincadeira. [citação não salvo]

No caso, uma falsa notícia sobre venda de um conhecido jogador do Santos conseguiu ser injetada no site devido à falta de tratamento dos parâmetros enviados à tela.

Com o XSS o atacante pode:[citação Josh Pauli]

1. Usar pop-ups (utilizado em provas de conceito),
2. Sequestrar Identificadores de Sessão,
3. Fazer download e instalar programas danosos como keyloggers,
4. Redirecionar a página para uma URL diferente;

**2.4.4 Path Transversal**

É um ataque executado também na tentativa/erro onde o atacante tenta atingir arquivos que estão fora da pasta onde o sistema está hospedado. Para isso ele usa expressões como “../../” para conseguir subir dentro da hierarquia de pastas do sistema.

Em caso de sucesso o atacante consegue acesso a informações não autorizadas e sensíveis como: o código compilado da aplicação, arquivos do Sistema Operacional, Arquivos de Senhas, etc.

Um exemplo de código seria a execução da seguinte chamada HTTP:

*GET /sistemaVulneravel.php HTTP/1.0*

*Cookie: TEMPLATE=../../../../../../../../../etc/passwd*

Se fosse bem executado, o código acima conseguiria realizar o download do arquivo de senhas do Linux.

**2.5 Pentest**

Também conhecido como Teste de Intrusão, é o nome dado a auditoria de segurança feita por um profissional de TI, denominado Pentester. O Pentest busca simular a ação de indivíduos mal intencionados para avaliar como está a questão de segurança da empresa/sistema.

O alvo de um Pentest pode ser um software ou até mesmo a política de uma empresa. No primeiro caso, são utilizadas ferramentas para tentar burlar o esquema de segurança do software, coletar resultados e reproduzir ataques. No segundo caso, podem ser avaliadas: a política de segurança da empresa e o comportamento de seus funcionários frente a algumas situações como: recebimento de e-mails suspeitos e manipulação de informações sigilosas.

**2.6 Kanban**

É uma metodologia de desenvolvimento ágil que é pouco prescritiva [citação kanban] e por isso, consegue ser facilmente integrada a outras metodologias para melhorar o processo de desenvolvimento de software de equipes ágeis.

Basicamente, o Kanban tem como fundamento os pontos abaixo:

1. Fluxo de Trabalho (Workflow)

A maior preocupação do Kanban é com a transparência do processo de desenvolvimento, a todos os colaboradores. O motivo disso é que caso seja identificado um gargalo, existe a possibilidade dos colaboradores encontrarem processos que o resolvam rapidamente.

1. Limite de quantidade de trabalho paralelo (WIP)

Existe uma variável chamada WIP (Work In Progress, Trabalho em andamento) que é de conhecimento de todos e representa a vazão de trabalho que a equipe consegue suportar. A principal motivação dessa ideia é deixar transparente a quantidade de tarefas que a equipe suporta.

1. Gerenciamento e Medidas sobre o fluxo

Como todas as metodologias de desenvolvimento, é preciso se tirar métricas e avaliar números para saber como está o andamento da equipe. Para fazer melhorias, primeiro deve-se conhecer onde a equipe acerta e onde ela erra.

1. Oportunidade de Melhorias

Como foi dito previamente, através das medidas é que se encontram os pontos de melhoria. O kanban foca no processo contínuo de aprimoramento do processo.

**2.7 Scrum**

É uma metodologia de desenvolvimento ágil bem conhecida no meio de desenvolvimento. Para o Scrum, as funcionalidades são chamadas de Product Backlog e o desenvolvimento é dividido em ciclos de tempo denominado Sprints. [citação Scrum]

Elas são normalmente divididas em semanas e no inicio ocorre uma reunião chamada Planning, cujo nome já informa trata-se do planejamento de quais atividades serão inseridas no pipeline da Sprint. Passada essa etapa, o acompanhamento é feito em reuniões diárias rápidas (cerca de 5 minutos) chamadas Daily. Nela todos os membros dão feedback do que foi feito, do que será feito e impedimentos. O motivo disso é manter a comunicação da equipe e resolver problemas que atrapalhem o progresso da Sprint.

Ao final da Sprint, acontecem duas reuniões chamadas Sprint Review e Sprint Retrospective. Review é onde o time apresenta as funcionalidades desenvolvidas no curso da Sprint. Enquanto que a Sprint Retrospective, é a reunião para avaliar como foi a Sprint e apontar pontos problemáticos e possíveis melhorias a serem feitas na próxima Iteração.

**3. Metodologia**

**3.1 Kali Linux**

É um sistema operacional open source baseado no Debian que é sucessor do Back-Track. O Kali foi desenvolvido para auxiliar o trabalho do pentest e já vem pré-configurado com uma série de ferramentas para testes de exploração, dentre as quais se destacam:

1. Nmap;
2. Hydra;
3. Metasploit;
4. Burp Suite;
5. John the Ripper;

O Kali é mantido atualmente pela empresa Offensive Security Ltd, uma empresa americana de segurança da informação que gerencia além do Kali outros projetos na área.

O Kali pode ser encontrado na página do projeto [citação projeto Kali] e diversas e lá ISSO’s podem ser adquiridas para a instalação. Durante os testes realizados neste trabalho, foi utilizada a versão 1.1.0.

Para instalar e executar o Kali, a configuração mínima é:

|  |  |
| --- | --- |
| Arquitetura do Processador | i386 ou AMD64 |
| Espaço em Disco | 8 GB disponíveis |
| Memória | 512 MB |

**3.2 Ferramentas Utilizadas**

**3.2.1 Hydra**

É uma das ferramentas disponibilizadas no Kali e pode ser utilizada para realizar ataques do tipo força bruta. Ela foi escrita em C e é uma ferramenta bem versátil para testes de força bruta, sendo capaz de realizar tentativas de intrusão usando vários protocolos como, por exemplo:

1. HTTP GET e POST;
2. FTP;
3. SSH;
4. Mysql;
5. Cisco;
6. LDAP.

Como é uma ferramenta open source e colaborativa, existe um projeto hospedado no repositório Github onde qualquer desenvolvedor interessado pode baixar o código fonte, abrir tickets e submeter correções. [citação Projeto Hydra].

Para os testes descritos neste trabalho, foi utilizada a versão 8.1 que é a versão atual do software.

**3.2.2 SqlMap**

É uma ferramenta escrita em Python que automatiza os processos de detecção e intrusão usando SQL Injection. Como a grande maioria das ferramentas disponibilizadas no Kali, é uma ferramenta open source e colaborativa sendo bem famosa entre os pentesters profissionais. Ela consegue interagir com os principais bancos de dados mais conhecidos e fazer inúmeras tentativas de intrusão.

O SqlMap possui uma página própria e um projeto no Github [citação projeto SQLMAP] de onde é possível baixar o código do projeto, abrir tickets e submeter correções e melhorias.

A versão mais atual do SqlMap é a 1.0, mesma versão utilizada durante os testes descritos neste trabalho.

**3.2.3 Nmap**

É o acrônimo de Network Mapper (Mapeador de Redes), outra ferramenta de grande uso da comunidade pentester. Trata-se de um flexível scaneador de portas utilizado na etapa de conseguir informações do sistema alvo. O software foi escrito em C, C++, Python e Lua.

Além de ser de fácil uso, possui grande documentação disponível e muitos tutoriais online [citação Nmap].

Quanto ao seu funcionamento, o Nmap utiliza pacotes IP brutos para tentar conexões com portas de um sistema e a partir disso descobrir se estão abertas ou fechadas, os serviços disponíveis nessas portas e suas versões, e o sistema operacional utilizado na máquina que hospeda a aplicação alvo.

A versão mais atual do Nmap é a 7.12. Neste trabalho foi utilizada a versão 6.49.

**3.2.4 Burp Suite**

É um grande framework de testes de intrusão que contém vários módulos interessantes para o uso de testes intrusivos em sistemas web. Ao contrário das demais ferramentas descritas anteriormente, o Burp é uma ferramenta privada e licenciada onde para utilizar todos os módulos disponíveis no programa é necessário adquirir uma licença com seus criadores.

Os módulos disponibilizados pelo Burp são:

1. Burp Target
2. Burp Intruder
3. Burp Spider
4. Burp Decoder
5. Burp Proxy
6. Burp Scanner
7. Burp Repeater
8. Burp Comparer

Apesar de ser uma ferramenta privada e licenciada, é possível utiliza-la de forma gratuita. Porém essa versão não possui acesso ao módulo de Scanner.

Neste trabalho, foi utilizado a versão gratuita do software, para tal foi possível utilizar somente os módulos:

* Target - responsável por fazer um mapeamento completo na hierarquia de diretórios do sistema atacado;
* Proxy - capaz de interceptar todas as requisições ao sistema alvo trazendo informações como cabeçalho das requisições HTTP e parâmetros passados entre cliente e servidor.

A versão mais atual do Burp Suite é a 1.7.03. Neste trabalho foi utilizada a versão 1.6.01.

**3.2.5 Metasploit**

Trata-se de um grande framework automatizado para testes de intrusão escrito em Ruby. Embora ele seja propriedade da empresa Rapid7, existe uma versão gratuita disponibilizada no Kali Linux. É uma ferramenta que trabalha com dois conceitos: Exploits e Payloads.

Exploit é o nome dado ao código mal-intencionado que é feitocom o propósito de executar uma ação não autorizada a fim de causar prejuízos. Exemplo: Caso um programa possua uma falha que seja conhecida, o exploit é o código que se aproveita dessa falha para conseguir um acesso não autorizado.

Payload é o nome normalmente dado à transmissão de dados, pois significa “Carga Paga”. Para o Metasploit, payload é o código executado depois que um exploit conseguiu sucesso, ou seja, a ação nociva dentro do sistema que será executado. Exemplo: Baseando-se na falha citada no trecho anterior, após a execução do exploit, um possível payload seria a busca das senhas criptografadas dos usuários do sistema alvo ou até a inserção de um novo usuário para fins danosos ao sistema.

O Metasploit possui uma grande base de dados de exploits e payloads. Além disso, é possível que um pentester também possa escrever os próprios exploits que se integrem aos já existentes.[citação projeto Metasploit].

**3.2.6 Iceweasel**

É um navegador web de código aberto para sistemas baseados em Debian idêntico ao Mozilla. Ele é o navegador padrão configurado no Kali Linux.

**3.3 Processo de Testes**

**Referências**:

<https://www.kali.org> , acessada 19/06/16.

<http://www.itnews.com.au/news/backtrack-successor-kali-launched-336420> , acessado 19/06/16.

<http://tools.kali.org/tools-listing> , acessada 19/06/16.

<https://github.com/vanhauser-thc/thc-hydra> , acessada 17/06/16.

<https://github.com/sqlmapproject> , acessada 17/06/16.

<http://insecure.org/> (Nmap) , acessada 17/06/16.

Testes de Invasão, Uma introdução prática ao Hacking , Georgia Weidman

<https://portswigger.net/burp>, acessada 17/06/16.

<https://www.metasploit.com/> , acessada 17/06/16.

<https://github.com/rapid7/metasploit-framework> , acessada 17/06/16.

<http://cartilha.cert.br/ataques/> , acessada 17/06/16.

<https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)>, acessada 19/06/16.

<https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)>, acessada 17/06/16.

<http://www.redesegura.com.br/2012/01/saiba-mais-sobre-o-cross-site-scripting-xss>, acessada 17/06/16.

<http://www.naosalvo.com.br/o-dia-em-que-o-ganso-foi-para-o-corinthians-eu-virei-um-hacker-e-o-santos-decidiu-me-processar/> , acessada 19/06/16.

Introdução ao Web Hacking, Josh Pauli

Norma ISSO 27002.

<http://www.culturaagil.com.br/kanban-do-inicio-ao-fim/> , acessado em 21/06/16.

<http://www.desenvolvimentoagil.com.br/scrum/>, acessado em 21/06/16.