Desenvolvimento

**Segurança da Informação**

Normalmente associada a questões como Vulnerabilidade de Sistemas, o termo Segurança da Informação se refere a todos os possíveis mecanismos de controle e proteção disponibilizados por uma instituição para defender sua 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 ~~seria~~ é um ativo importante para a organização e os meios de defendê-la são bem variados podendo ser mecanismos lógicos, físicos ou até políticos.

**Norma ISO 27002 [Acho q devemos falar um pouco mais da norma. Posso ajudar nessa parte]**

É a norma definida pela ABNT para regulamentar ~~o que seria~~ 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.

**Pentest**

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

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 ~~da~~ 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 ataques do tipo engenharia social.

Ferramentas Utilizadas

**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 ~~por aqui~~ 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: [Achei essa parte um pouco desnecessária.]

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

**Hydra**

É uma das ferramentas disponibilizadas no Kali e pode ser utilizada pra 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.

**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.

**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 porta e suas versões, e o sistema operacional utilizado na máquina que hospeda o sistema alvo.

~~Neste trabalho foi utilizado a versão 6.49 do Nmap enquanto que a ferramenta encontra-se na versão 7.12.~~

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

**Burp Suite**

É um grande framework de testes de intrusão que contém vários módulos interessantes para o uso em testes ~~de intrusão~~ intrusivos em sistemas web. Ao contrário das demais ferramentas descritas, o Burp é um software proprietário onde a pessoa que deseje utilizar todos os módulos disponíveis no programa em seus testes precisa adquirir uma licença com seus criadores.

Também é possível utilizá-lo de forma livre mas não nesse caso não tem acesso a todas as técnicas disponibilizadas pelo Burp. [Não entendi direito =( ]

Neste trabalho, o Burp foi utilizado ~~de forma free~~ 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.

~~Neste trabalho foi utilizado a versão 1.6.01 enquanto que a versão atual do Burp Suite é a 1.7.03.~~

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

**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 ~~free~~  livre disponibilizada no Kali. É uma ferramenta que trabalha com dois conceitos: Exploits e Payloads.

Exploit é o nome que recebe o código mal-intencionado ~~que~~ e é feito ~~com o fim~~ com 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 ~~que se recebe~~ fornecido 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: ~~Ainda~~ Com base 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.[projeto Metasploit].

**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.

Técnicas Utilizadas

**Força Bruta**

É o tipo de ataque baseado na tentativa/erro onde um login e/ou senha são encontrados baseados em palpites [é utilizado apenas para campos de login e senha? PQ a forma como foi escrito dá a impressão que só é utilizado nesses casos]. 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 ferramentas automatizadas no processo aliados ao 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] [Rescrever essa parte a oração tá muito grande tá ruim pra compreender bem as coisas]

**SQL Injection**

É um ataque que ~~tira vantagem~~ explora a falta de tratamento dos parâmetros textuais enviados a uma aplicação. ~~Parâmetros que manipulam a consulta SQL executada no banco efetuando procedimento ações não autorizadas.~~ 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

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.

**XSS**

É o acrônimo de Cross-Site Scripting, é a execução de scripts maliciosos dentro de um site confiável. Ele pode ser executado de duas formas: 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 da 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;

**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 /vulnerable.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.

**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.