**Resultados detalhados**

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.

1. **Iteração 1**

**1.1. Tela Login - Exploração via Força Bruta**

Tarefa realizada dia 01/04/16.

Duração 1hora e 20 minutos.

**Objetivo**

Utilizar a ferramenta Hydra para adivinhar, por tentativa e erro, um login de um usuário e senha válidos para efetuar login no Sistema de Vacinas para verificar se a tela de Login não é vulnerável a ataques de força bruta.

**Resultados**

**Tentativa 1**

Configurar a Hydra com 3 parâmetros para que ela realize os testes de acesso com força bruta na tela de Login.

**Impedimento**: Hydra não está reclamando de 3 parâmetros quando, a tela de Login só utiliza 2.

**Resultado**: Indefinido.

**Tentativa 2**

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

Configurar o número de 16 tentativas de acesso na ferramenta Hydra.

*Comando utilizado: hydra -l admin -P forcaBruta IP\_do\_servidor http-post-form “vacina/login.php:login=^USER^:senha=^PASS^:logado=false”*

Esses Trechos abaixo, segundo Marcos é uma duplicação necessária.

No comando acima:

1. “-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 de Vacinas;
2. “-P ” aponta para o arquivo contendo as senhas que a Hydra tentará utilizar para fazer o login;
3. IP\_do\_servidor é o IP onde está hospedada a aplicação alvo;
4. “http-post-form” é o método utilizado, no caso , POST do HTTP;
5. *“vacina/login.php:login=^USER^:senha=^PASS^:logado=false”* é a URL utilizada para fazer a requisição.
   1. USER é onde a Hydra deve inserir o login válido usado na requisição;
   2. PASS é onde a Hydra deve inserir a senha válida usada na requisição.

Não entendi a marcação aqui.

**Resultado:** Não passou no teste.

**Tentativa 3**

Configurar a ferramenta Hydra para executar tentativas utilizando o login “admin”, cadastrado na base de dados do sistema Espera-se que o sistema possua algum tipo de controle. Além disso, usar um dicionário de senhas retiradas da *Lista de Piores Senhas de 2015* da SplashData ¹.

*Comando utilizado: hydra -l admin -P forcaBruta IP\_do\_servidor http-post-form “vacina/login.php:login=^USER^:senha=^PASS^:logado=false” -V*

Esses Trechos abaixo, segundo Marcos é uma duplicação necessária.

No comando acima:

1. “ -l ” significa o login utilizado no momento. Deve ser sempre admin, pois é foi único usuário válido fornecido para a realização de testes no Sistema de Vacinas;
2. “ -P ” aponta para o arquivo contendo as senhas que a Hydra tentará utilizar;
3. IP\_do\_servidor é o IP onde está hospedada a aplicação;
4. “http-post-form” é o método utilizado, no caso , POST do HTTP;
5. “*vacina/login.php:login=^USER^:senha=^PASS^:logado=false”* é a URL utilizada para fazer a requisição;
   1. USER é onde a Hydra deve inserir o login válido usado na requisição;
   2. PASS é onde a Hydra deve inserir a senha válida usada na requisição.
6. “-V” é para exibir no console o resultado da operação.

Não entendi a marcação aqui.

**Resultado:** Não passou no teste. Entretanto, a execução do teste mostrou que o sistema é vulnerável ao acesso via força bruta. A Figura 1 mostra que embora várias tentativas tenham sido realizadas sem sucesso de login, o sistema permite que várias requisições sejam feitas ao servidor por tempo indeterminado. Como não há mecanismo de bloqueio para inúmeras requisições de login, o sistema abre uma brecha para que senhas de acesso ao sistema possam ser descobertas após várias tentativas.

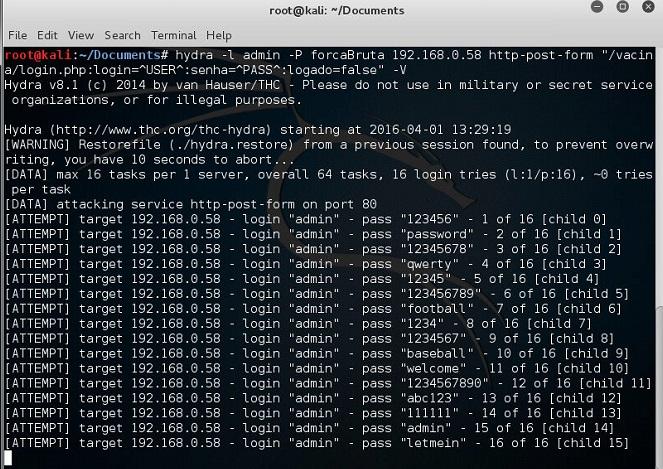


Figura 1 - Resultado da operação

**Referências**

1 - [Piores Senhas 2015](https://www.teamsid.com/worst-passwords-2015/) - <https://www.teamsid.com/worst-passwords-2015/>

<https://www.teamsid.com/worst-passwords-2015/>

**1.2. Tela Login - Exploração via Path Transversal**

Tarefa realizada entre os dias 31/03/16 e 01/04/16.

Duração: 2 horas e 30 minutos.

Um scanner foi realizado através da ferramenta Burp Suite para identificar a estrutura de pastas do sistema dentro do servidor.

Os resultados obtidos, conforme mostra a figura 2 [anexar figura], foram utilizados para executar os testes deste tópico.

.

**Objetivo**

Não entendi a marcação aqui.

Através dos parâmetros enviados pela URL e utilizados pela aplicação para realizar login, tentar navegar em diretórios da aplicação dentro do servidor. Dessa forma, conseguir acesso a locais não autorizados a um usuário do sistema. Foram utilizados os seguintes comandos “ ../../” para tentar realizar a navegação de pastas.

**Resultados**

**Tentativa 1**

Esses Trechos abaixo, segundo Marcos é uma duplicação necessária.

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

A primeira tentativa foi fazer uma requisição ao servidor alterando a url citada anteriormente para: v*acina/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 de Vacinas.

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 Windows serve para listar os arquivos e diretórios dentro da pasta atual.

**Resultado**: Passou no teste.

**Tentativa 2**

Fazer uma requisição ao servidor utilizando a url *vacina/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 de Vacinas.

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 o esperado seria um erro.

Não entendi a marcação sobre o windows.

**Resultado**: Passou no teste.

**Tentativa 3**

Fazer uma requisição ao servidor utilizando a url *vacina/login.php?logado=%2e%2e%2f%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 de Vacinas.

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

**Resultado**: Passou no teste.

**Tentativa 4**

Fazer uma requisição ao servidor utilizando a url *vacina/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 de Vacinas.

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

**Resultado**: Passou no teste.

**Tentativa 5**

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

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

**Resultado**: Passou no teste

**Tentativa 6**

Fazer uma requisição ao servidor utilizando a url *vacina/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 de Vacinas.

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

**Resultado**: Passou no teste

**Tentativa 7**

Fazer uma requisição ao servidor utilizando a url *vacina/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 de Vacinas.

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

**Resultado**: Passou no teste

**1.3 Tela Login - Mapeamento da hierarquia de pastas do sistema**

Tarefa realizada dia 30/03/16.

Duração: 1 hora e 30 minutos.

**Objetivo**

Utilizar de forma automatizada o módulo Burp Spider da ferramenta Burp Suite para realizar um reconhecimento da a hierarquia de diretórios do sistema armazenados dentro do servidor a partir da URL da tela de Login: IP\_do\_servidor/vacina/login.php. A partir dos 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.

**Resultados**

**Tentativa 1**

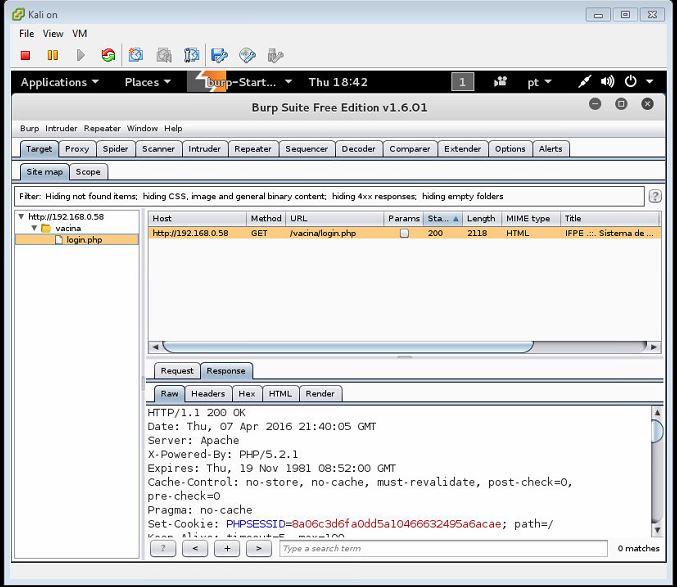
Acessar a página do Login utilizando a url

http:// IP\_do\_servidor/vacina/login.php.

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

**Resultado**: Passou no teste

Conforme mostra figura 3, não foi possível mapear a estrutura de pastas do sistema apenas acessando a página de Login.

**Figura 3 - Resultado tentativa de mapeamento diretórios do sistema**

**1.4 Tela Login - Exploração Via SQL Injection**

Tarefa realizada dia 07/04/16.

Duração: 1 hora e 40 minutos.

Refatorar os textos sobre SQL Injection.

**Objetivo**

Submeter entradas maliciosas no sistema através de injeção de códigos SQL na página de Login na tentativa de forçar ações não autorizadas no sistema.

**Resultados**

**Tentativa 1**

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

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

Entrada: *admin’*

**Resultado**: Passou no teste. A aplicação está fazendo tratamento de dados de entrada e não permite a injeção de instruções SQL através do campo de Login.

**Tentativa 2**

Inserir uma string com uma distribuição adequada de aspa simples e comentários inline como entrada no campo Login, a fim de modificar a instrução SQL utilizada pelo sistema na consulta de dados referentes ao login de 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:

*admin’ or 1=1 #*

*admin’ or 1=1 --*

O envio desses dados como entrada no campo do Login, tenta forçar o sistema a consultar não apenas os dados do usuário *admin*, mas de todos os usuários cadastrados na base de dados, uma vez que a instrução “1=1” será sempre verdadeira.

Para que a consulta alcance o objetivo proposto, é necessário utilizar possíveis instruções de que comentam linhas de código SQL em determinados SGBDs: “#” e “--”. Essas instruções podem ignorar qualquer instrução posterior presente na mesma linha em que elas foram inseridas.

Espera-se que o sistema trate todos os caracteres informados como um parâmetro regular da consulta.

**Resultado**: Passou no teste. A aplicação está fazendo tratamento de dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

**Tentativa 3**

Inserir um texto fazendo uso de uma única aspa simples para completar a entrada de uma possível instrução SQL executada no momento do login no sistema. Nesse caso, é possível que uma instrução SQL de consulta semelhante a essa seja executada SELECT login from usuarios WHERE login = ‘admin’ or ‘a’=’a’.

Entradas:

admin’ or ‘a’=’a

O envio desses dados como entrada no campo do Login, tenta forçar o sistema a consultar não apenas os dados do usuário *admin*, mas de todos os usuários cadastrados na base de dados, uma vez que a instrução ‘a’=’a’ será sempre verdadeira.

Espera-se que o sistema trate todos os caracteres informados como um parâmetro regular da consulta.

**Resultado:** Passou no teste. A aplicação está fazendo tratamento de dados de entrada, não permitindo a mudança de comportamento da instrução SQL utilizada pelo sistema para consultar os dados dos usuários cadastrados.

1. **Iteração 2**

**2.1 Análise do Servidor - Exploração via Nmap**

Tarefa realizada no dia 09/04/16

Duração: 1 hora.

**Objetivo**

Utilizar a ferramenta Nmap para descobrir quais portas estão abertas no servidor e que atividade Maliciosa pode ser desenvolvida a partir disso.

**Resultados**

**Tentativa 1**

*Utilização do comando nmap -Ss -O IP\_do\_servidor.*

1. “-Ss” permite que a ferramenta tente fazer o scan SYN do TCP.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.
3. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Não vi isso no meu arquivo original, vc colocou isso?.

Esperava-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, os serviços que essas portas rodavam no momento do escaneamento.

**Resultado:** Erro na ferramenta.

**Tentativa 2**

*Utilização do comando nmap -sS -O IP\_do\_servidor.*

1. “-sS” permite que a ferramenta tente fazer o scan SYN do TCP.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.
3. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Esperava-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, os serviços que essas portas rodavam no momento do escaneamento.

**Resultado:** Passou no teste. Conforme a Figura 4, a ferramenta conseguiu escanear todas as portas abertas e exibir os serviços que estavam rodando nessas portas.

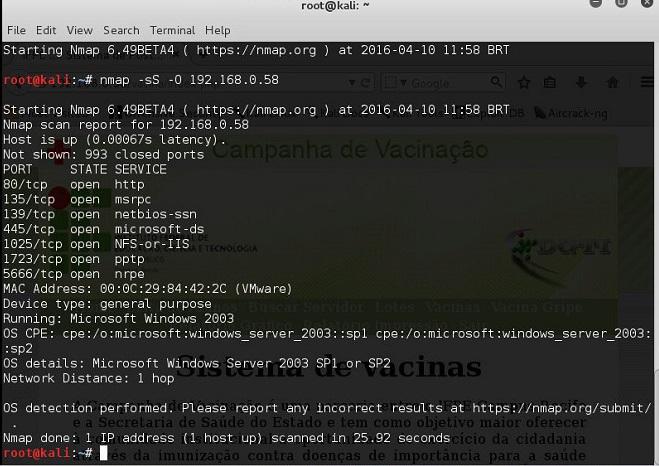


Figura 4 - Portas encontradas e os serviços que estavam rodando nelas no momento do escaneamento.

**Tentativa 3**

*Utilização do comando: nmap -sV -O IP\_do\_servidor.*

1. “-sV” permite que a ferramenta realize o SYN e busque a versão do software rodando no momento.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.
3. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Esperava-se descobrir o sistema operacional utilizado pelo servidor explorado, as portas abertas, e as versões dos sistemas que essas portas rodavam no momento do escaneamento.

**Resultados:** Passou no teste. Conforme a Figura 5 foi possível escanear as portas abertas e identificar a versão dos sistemas que estavam rodando nelas.

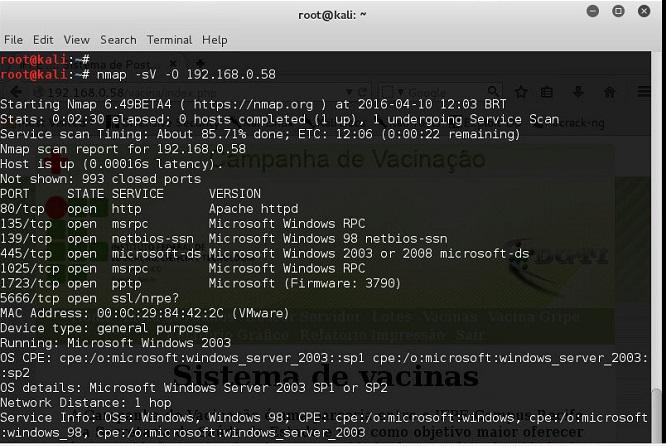


Figura 5 - Portas escaneadas e versões dos sistemas que elas estavam rodando.

Serviços:

* 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.

**Tentativa 4**

*Utilização do comando nmap -sU -O IP\_do\_servidor.*

1. “-sU” permite que a ferramenta utilize o detector de portas UDP.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.

IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Esperava-se descobrir as portas UDP abertas, e as versões dos sistemas que essas portas rodavam no momento do escaneamento.

**Resultado:** Passou no teste. Conforme a Figura 6, foi possível identificar as portas UDP abertas e os serviços que elas estavam rodando no momento do escaneamento.

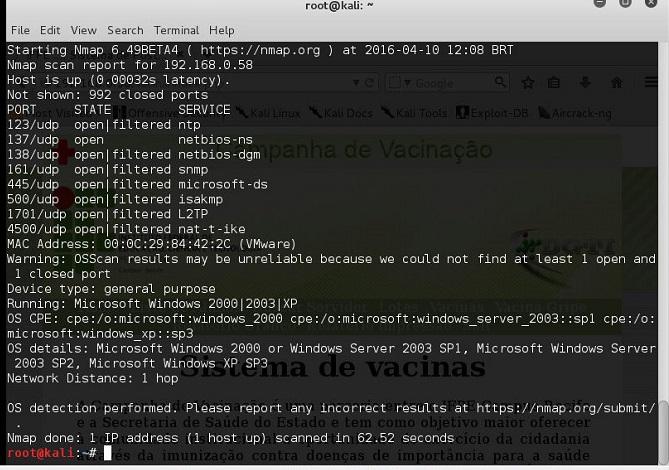
.

Figura 6 - Portas UDP escaneadas e serviços que estavam rodando nelas no momento do escaneamento.

Serviços:

* ntp - protocolo de sincronização de relógios usando UDP;
* netbios-ns - responsável pelo registro de nomes de aplicação do NetBios;
* netbios -dgm - serviço de datagrams do NetBios¹ ;

Não consegui colocar no rodapé...essa bixiga fica repetindo em todas as páginas ¬¬

* snmp - gerenciamento de redes;
* microsoft-ds - protocolo windows para compartilhamento e impressões de arquivos;
* L2TP - protocolo de encapsulamento de camada 2, para criar redes vpn privadas;
* nat-t-like - protocolo de tradução de ips.

**Tentativa 5**

*Utilização do comando nmap -A -O IP\_do\_servidor.*

1. “-A” é para usar o Tracerouter na requisição.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.
3. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Acho desnecessário explicar aqui o que seria isso

Esperava-se descobrir a rota traçada pelo Tracerouter.

**Resultado:** Passou no teste. Conforme as Figura 7 e 8, foi possível identificar a rota traçada pelo Tracerouter.

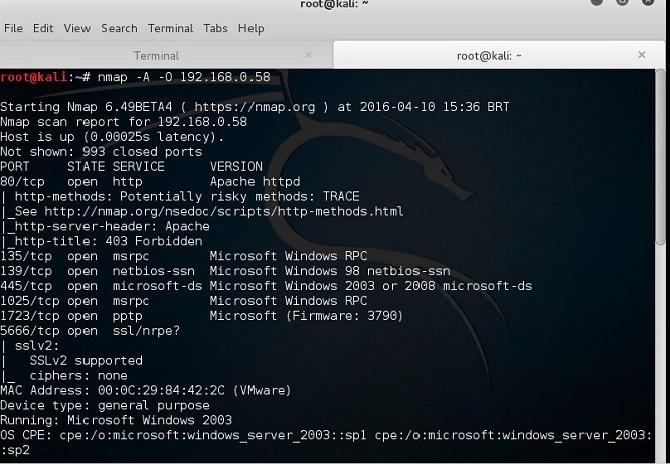


Figura 7 - Rota traçada pelo Tracerouter - Parte 1

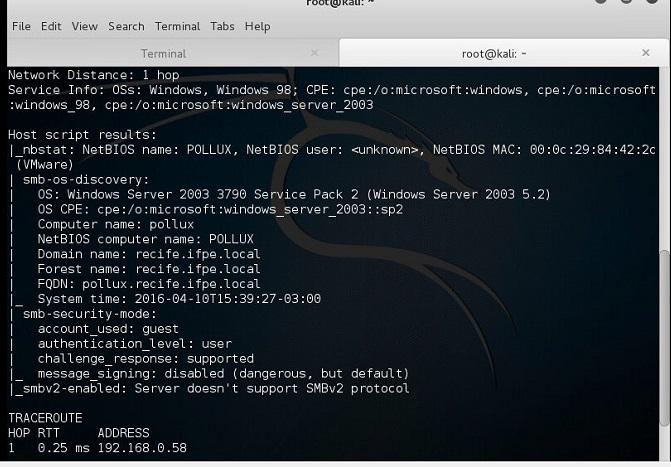


Figura 8 - Rota traçada pelo Tracerouter - Parte 2

**Tentativa 6**

*Utilização do comando nmap -sV -p 1-65535 -O IP\_do\_servidor.*

1. “-sV” permite que a ferramenta realize o SYN e buscar a versão do software rodando no momento.
2. “-p” indica quais portas devem ser usadas no teste, o delimitador é “-”.
3. “-O” permite que a ferramenta tente encontrar o S.O. rodando no momento.
4. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Esperava-se descobrir a versão dos softwares que estavam rodando nas portas escaneadas.

**Resultado:** Passou no teste. Conforme a Figura 9, foi possível identificar a versão dos softwares que estavam rodando nas portas escaneadas.

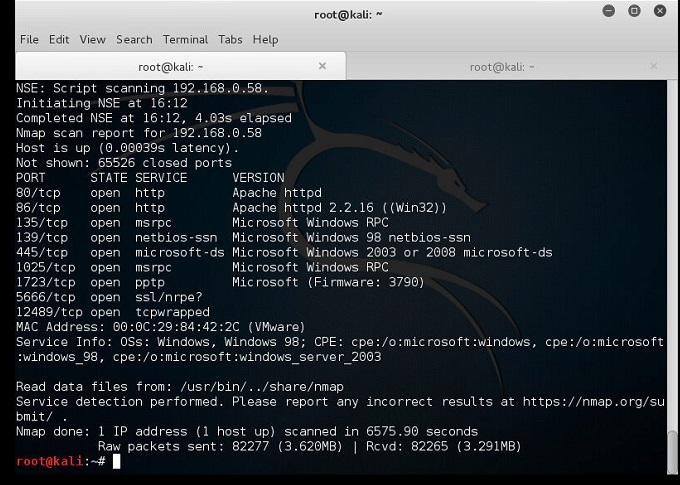


Figura 9 - Versão do software rodando nas portas escaneadas

**Tentativa 7**

*Utilização do comando nmap -sU -p 1-65535 IP\_do\_servidor.*

1. “-sU” permite que a ferramenta use o detector de portas UDP.
2. “-O” permite que a ferramenta tente encontrar o sistema operacional rodando no momento.
3. IP\_do\_servidor é o IP onde está hospedada a aplicação em teste.

Esperava-se descobrir a versão dos softwares que estavam rodando nas portas UDP escaneadas.

**Resultado:** Passou no teste.

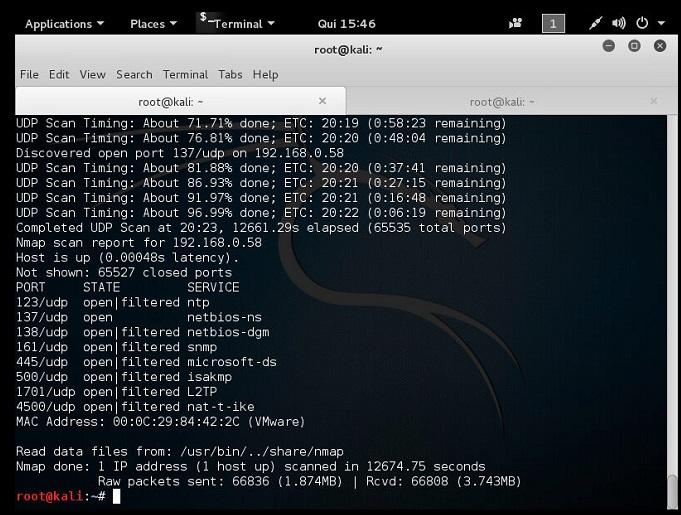


Figura 10 - Versão do software rodando nas portas UDP escaneadas.

Referências

<NTP> - <https://pt.wikipedia.org/wiki/Network_Time_Protocol>

[NetBios](https://pt.wikipedia.org/wiki/NetBIOS) - <https://pt.wikipedia.org/wiki/NetBIOS>

[SNMP](https://pt.wikipedia.org/wiki/Simple_Network_Management_Protocol) -<https://pt.wikipedia.org/wiki/Simple_Network_Management_Protocol>

**2.2 Ataque à porta RPC descoberta na análise de servidor - NMAP**

Tarefa realizada dia 10/04/16.

Duração: 1 hora.

**Objetivo**

Ataque à porta RPC descoberta na análise de servidor - NMAP.

Tentar se conectar à porta 135 usando o framework Metasploit para tentar conseguir privilégios dentro do Windows.

**Resultados**

**Tentativa 1**

Tentar configurar algum exploit que ataque a porta 135.

**Resultado:** A ferramenta é de difícil configuração e possui documentação muito ruim para iniciantes. Não foi possível obter grandes resultados.

**Tentativa 2**

Não entendi a marcação

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**: O exploit foi enviado mas não trouxe resultado. Após verificação, foi constatado que o exploit é pra Windows NT.

**Tentativa 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*. O payload é o utilizado na tentativa 2: *windows/meterpreter/reverse\_tcp.* Esse payload força a máquina atacada a abrir uma conexão com a máquina atacante via TCP.

**Resultado**: Passou no teste.

**Tentativa 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**: Passou no teste.