

# A ANÁLISE DE TRÁFEGO EM REDES COMO MÉTODO DE COLETA DE EVIDÊNCIAS DIGITAIS: ESTUDO DE CASOS COM SOFTWARES DE ANÁLISE DE TRÁFEGO

Cristiano Monteiro Nunes<sup>1</sup>

**Resumo:** O presente trabalho teve como objetivo geral demonstrar como as ferramentas de análise de tráfego em redes podem contribuir para a obtenção de achados que consolidem uma evidência digital para a perícia forense. O trabalho inicialmente ambienta o leitor ao assunto com as apresentações das ferramentas mais usuais na análise de tráfego para sistemas *GNU/Linux*, *Microsoft Windows* e baseados no *Android*. A explicação sobre como executar a análise de tráfego foi pautada em exemplos práticos básicos representados pelo autor através de imagens de tela e descrições detalhadas em ambientes simulados. O trabalho contextualizou com casos de testes experimentais e extratos de manuais de *softwares* de análise e monitoramento de tráfego com a finalidade de justificar e demonstrar o emprego de tais ferramentas em forense digital. Como conclusão foram apresentados pelo autor algumas das informações que podem ser obtidas.

**Palavras-Chave:** Perícia. Forense. Análise de Tráfego. Redes *TCP/IP*. Monitoramento.

## 1. INTRODUÇÃO

Com o advento de novos sistemas de comunicações e a vertiginosa evolução da Tecnologia da Informação nasceram outras formas de interação entre os seres humanos, essas formas proporcionaram o surgimento de um espaço que trafega muitas informações. A análise adequada através da coleta das informações deste espaço respeitando os dispositivos legais, pode fornecer informações úteis para a resolução de problemas.

Segundo o Manual de Patologia Forense do Colégio de Patologistas Americanos (1990), a Ciência Forense é a aplicação de princípios das ciências físicas ao direito na busca da verdade em questões cíveis, criminais e de comportamento social para que não se cometam injustiças contra qualquer membro da sociedade. Conforme o conceito apresentado, podemos sintetizar que através dos princípios das ciências físicas é possível buscar a solução de questões de direito, corroborando com isso a Forense Computacional é o ramo que se utiliza de métodos científicos para preservação,

---

<sup>1</sup> Pós-graduando, *lato sensu*, em Gestão de Segurança da Informação.  
E-mail: [postcristiano@gmail.com](mailto:postcristiano@gmail.com)

coleta, restauração, identificação, documentação e apresentação de evidências digitais. Existem muitas peculiaridades na forense Digital, comparando as demais ciências forenses, maioria delas são originárias da heterogeneidade de softwares, hardwares, uso de padrões distintos e as constantes mudanças tecnológicas. Por isso a Perícia Digital se utiliza de métodos e procedimento que são capazes de objetivar a análise. A coleta da evidência Digital deve ser realizada utilizando-se dos métodos e procedimentos e ela se divide, de acordo com o vocabulário utilizado pelos peritos, em análise viva e análise morta. Na análise viva a busca é por interações que remetam a acontecimentos no tempo real do fato investigado, assim temos as análises de memória e de rede, esta última realizada através da análise de tráfego em rede. Uma precisa análise do tráfego em rede do sistema ou equipamento com provável envolvimento na relação que se investiga pode descobrir vestígios e evidências que auxiliem o investigador a elucidar o caso.

Para a realização da análise de tráfego em rede é necessário a utilização de algumas ferramentas que não são em sua essência de desenvolvimento voltadas para a Perícia Forense Computacional, nesse contexto o escopo do trabalho é explicar como essas ferramentas aliadas aos métodos e procedimentos forenses podem ser poderosas e capazes de produzir informações preciosas para a obtenção da evidência digital mesmo através de dados tão voláteis.

## **2. PERÍCIA FORENSE DIGITAL**

Definida no Código de Processo Civil, Lei nº 5.869, de 11 de janeiro de 1973, art. 420, a prova pericial, por sua vez, consiste em exame, vistoria ou avaliação, sendo considerada, no decorrer do processo civil, um dos momentos mais cautelosos na estrutura de composição do texto. O profissional perito, independente da área em questão, utiliza técnicas e ferramentas desenvolvidas para tal, fazendo-se imprescindível o esclarecimento de detalhes técnicos para que fique evidente a imparcialidade com fé no processo.

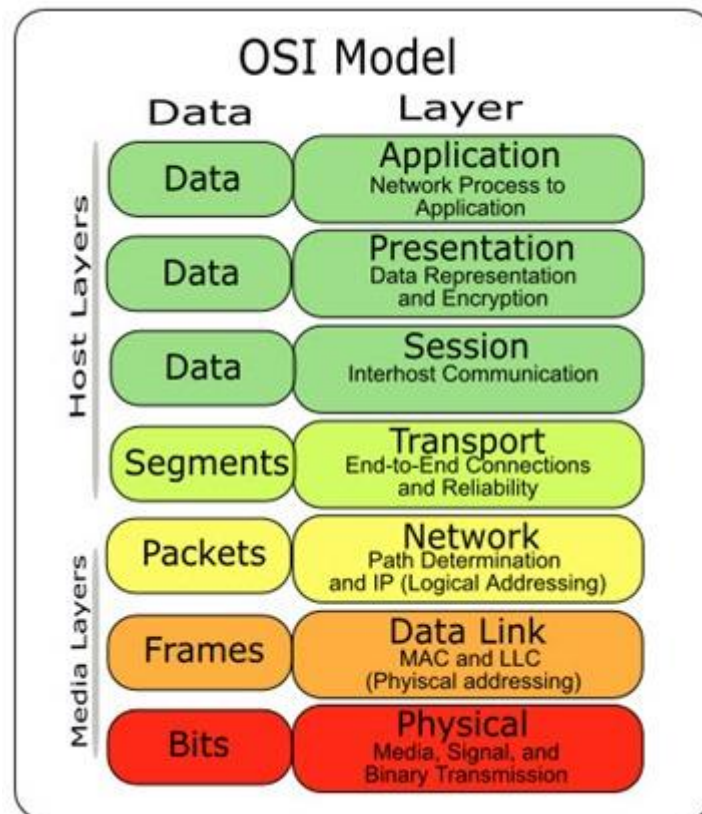
Do latim *forense*, a palavra é definida por Ferreira (2009) como o que se refere ao foro judicial, sendo relativo aos tribunais. A técnica forense, portanto, diz respeito à aplicação de recursos e métodos científicos em um processo jurídico, sendo que essas práticas se valem da perícia para alcançar os objetivos de prova, visto que muitas provas não são perceptíveis a ‘olho nu’ nem estão disponíveis a pessoas desprovidas de conhecimentos técnicos específicos.

A Perícia Forense Computacional ou Computação Forense tem a finalidade de auxiliar na solução de casos onde são cometidos crimes com o auxílio de dispositivos computacionais. Segundo Eleutério e Machado (2011), a perícia forense computacional é destinada a determinar a dinâmica, a materialidade e autoria de ilícitos ligados à área de informática, tendo como questão principal a identificação e o processamento de evidências digitais em provas materiais de crimes, por meio de métodos técnico-científicos, conferindo-lhe validade probatória em juízo. O relatório da Perícia é o documento comprobatório de sua execução, deve fazer parte dos autos da investigação e por isso segue um modelo contendo informações importantes para possuir validade.

Segundo Bustamante (2006), a perícia forense pode ser definida como coleção e análise de dados de um computador, sistema, rede ou dispositivos de armazenamento, de forma que sejam admitidos em juízo, sendo que as evidências que um perito encontra geralmente não podem ser vistas de forma direta, dependendo de ferramentas e meios para a sua obtenção. Nesse contexto, a coleção e análise de dados provenientes das interações de rede é exatamente onde se concentra o estudo elucidativo deste trabalho, pois para a extração desse tipo de dado é necessária a utilização de ferramentas e *softwares* específicos. Após a coleta do material, entra a experiência e o olhar crítico do perito para selecionar o que é ou não importante.

### **3. ANÁLISE DE TRÁFEGO EM REDES TCP/IP**

A análise de tráfego é muito importante para o entendimento do funcionamento de uma rede e seus serviços. “A análise de tráfego é um assunto essencial para administradores de redes de computadores. Com tal análise, um administrador realmente dominará a sua rede. Muitas vezes, administradores não conhecem os conceitos básicos sobre protocolos de redes e terminam não resolvendo, conscientemente, os problemas técnicos encontrados. Isso sem mencionar os problemas que existem, mas nunca foram vistos. [...]” (MOTA FILHO, 2013, p. 37). Além das ferramentas de análise de tráfego temos também o próprio sistema operacional, isto é, o conhecimento mais aprofundado e avançado do sistema operacional utilizado no dispositivo em questão pode ser também por si só uma ferramenta capaz de ajudar na coleta de informações, pois os próprios sistemas implementam uma série de recursos técnicos para a resolução problemas e verificação da rede. A figura abaixo ilustra as sete camadas do Modelo *OSI*:

Figura 01. Modelo *OSI*

Fonte: Internet < <https://pplware.sapo.pt/tutoriais/networking/redes-sabe-o-que-e-o-modelo-osi/> >

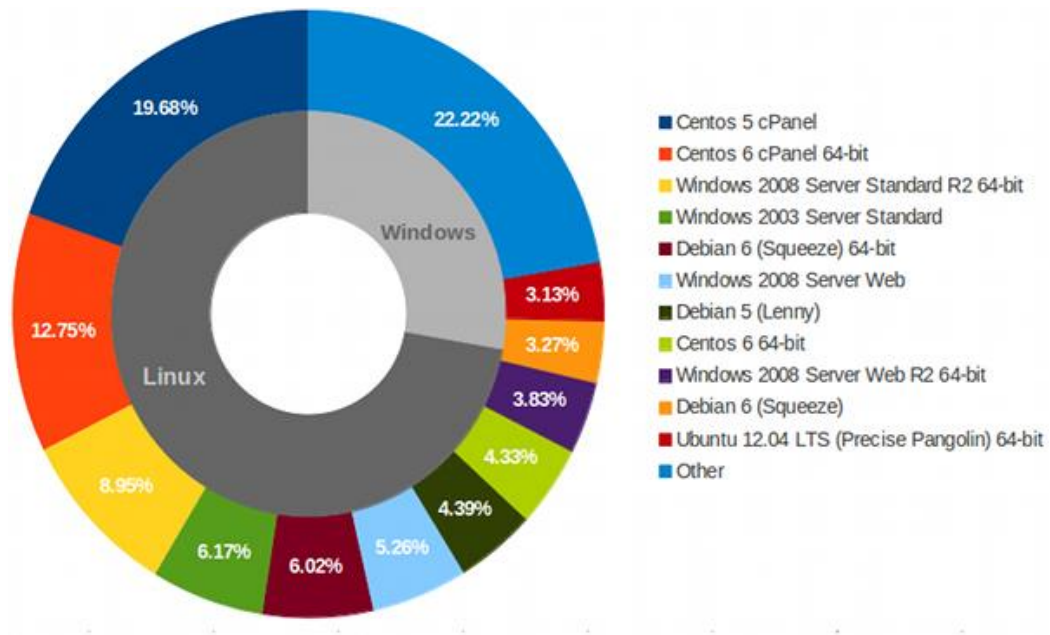
O conceito base do Modelo *OSI*, modelo de rede de computador dividido em sete camadas de funções com objetivo de ser um padrão para protocolos de comunicação entre os mais diversos sistemas, é base para a compreensão do funcionamento dos mecanismos que criam o tráfego na rede. Segundo Tanenbaum o Modelo *OSI* não é uma arquitetura de rede, pois não especifica os serviços e protocolos exatos que devem ser usados em cada camada. Ele apenas informa o que cada camada deve fazer. Na análise de tráfego para especificação de alguns serviços recorrer-se-á com frequência às *RFCs*, documentos técnicos desenvolvidos e mantidos pelo *IETF* (*Internet Engineering Task Force*), instituição que especifica os padrões que serão implementados e utilizados em toda a *internet*.

### 3.1 PECULIARIDADE DAS DISTRIBUIÇÕES GNU/LINUX

Existem milhares de comandos e combinações de comandos funcionalmente nativas do *GNU/Linux* para a análise de tráfego e estudo das interações de rede, no trabalho tratamos de alguns, afim de atingir o objetivo com máximo de simplicidade. As ferramentas *Tcpdump*, *Tshark* e

*Wireshark* todas compartilham a biblioteca *libpcap* que foi originalmente desenvolvida para sistemas *Unix*. Entre as três ferramentas o *Tcpdump* se destaca por sua maneira concisa e simples de expor as informações, porém o *Wireshark* com sua interface gráfica explora uma outra forma de se executar a análise de tráfego, navegando em seus diversos recursos, abas e filtros.

Figura 02. Estimativa dos sistemas operacionais mais utilizados em servidores em 2012



Fonte: <<http://www.memset.com/blog/which-are-most-popular-operating-systems-our-cloud/>>

Os casos de testes foram executados em sistemas *GNU/Linux*, uma vez que eles são responsáveis pela maioria dos sistemas operacionais utilizados pelos administradores de redes e nos servidores corporativos. É coeso verificar que os sistemas *GNU/Linux* são a engrenagem principal quando falamos em interações em rede. “Em um sistema de redes poderemos ter servidores e clientes. Servidor é uma máquina que oferece um ou mais serviços dentro da rede. Cliente é uma máquina da rede que utiliza os serviços oferecidos pelos servidores. [...]” (MOTA FILHO, 2012, p. 770). Um profundo conhecimento nos sistemas operacionais *GNU/Linux* facilita reconhecer e compreender tráfegos de rede gerados por aplicações do sistema.

### 3.2 PECULIARIDADES DO MICROSOFT WINDOWS

A análise de tráfego pode ser feita a partir de um sistema operacional *Microsoft Windows* através de suas ferramentas embarcadas no sistema e outras que podemos adicionar, como

o *Windump*, disponível em <[www.winpcap.org/windump](http://www.winpcap.org/windump)> e o *Wireshark*, disponível em <[www.wireshark.org](http://www.wireshark.org)>. O ‘motor’ destes dois softwares de análise é a *libpcap*, que foi portada para sistemas *Windows*, se chamando *winpcap*, disponível em <[www.winpcap.org](http://www.winpcap.org)>. Vale ressaltar que a maneira de uso do *Wireshark* em um sistema *Windows* é exatamente igual a versão do programa para plataformas *Unix*.

Em sistemas *Windows* notaremos sempre a existência de alguns tráfegos diferentes de sistemas *Unix*, na grande maioria remetem a recursos exclusivos da plataforma, que buscam facilitar a experiência do usuário comum, podemos constatar isso através da análise abaixo que apresenta o sistema em uma rede isolada.

Figura 03. Imagem da tela do *software Wireshark*

No.	Time	Source	Destination	Protocol	Length	Info
7	0.68168400	192.168.100.100	192.168.100.255	NBNS	92	Name query NB ISATAP<00>
8	1.07213700	192.168.100.100	192.168.100.255	BROWSER	218	Request Announcement w7-PC
9	1.08395200	192.168.100.100	192.168.100.255	BROWSER	243	Host Announcement w7-PC, workstation
10	1.43147000	192.168.100.100	192.168.100.255	NBNS	92	Name query NB ISATAP<00>
11	2.18132300	192.168.100.100	192.168.100.255	NBNS	92	Name query NB ISATAP<00>
12	2.57202300	192.168.100.100	192.168.100.255	BROWSER	218	Request Announcement w7-PC

Fonte: O autor

Podemos notar no tráfego o protocolo *NBNS*, o *NetBIOS* do *Windows*. Sempre que for encontrado um protocolo desconhecido é recomendável pesquisar sobre sua *RFC* normativa no site do *IETF*.

### 3.3 PECULIARIDADES DO ANDROID

Os sistemas *Android* e baseados nele, são dotados do *kernel linux*, compartilham muitas semelhanças com os sistemas *Unix*, porém é um sistema ‘orientado à interface gráfica’, isto é, otimizado a prover ao usuário a facilidade e usabilidade avançada, dessa maneira podemos entender que não é razoável a utilização de terminais *tty* neste sistema.

A proposta de utilização do sistema *Android* não é a mesma de *notebooks* e computadores *PC*, várias ferramentas e comandos técnicos não têm seu acesso facilitado, ou até mesmo não são implementados. Para adicionar funcionalidades deste cunho no sistema é necessário instalar aplicações que as adicionem e mesmo assim existirá grandes limitações. Alguns *softwares* para implementar essas funções estão disponíveis no repositório F-droid <[f-droid.org](http://f-droid.org)>.

- *Terminal Emulator*, implementa acesso à uma tela *tty* para uso do *shell*;

- *BusyBox*, adiciona uma série de funções no *shell*;
- *ConnectBot*, cliente *SSH*;
- *Port Authority*, possibilita obter informações sobre o *hardware* e a rede que o aparelho faz parte e faz varreduras com *ARP* para descobrir *hosts* na a rede;
- *Network Scanner*, realiza automaticamente uma varredura *ARP* para descobertas de *hosts* na rede local;
- *Multiping*, possibilita enviar múltiplos *echo requests* pelos pacotes *icmp*; e
- *DNS man*, possibilitar ver e modificar o servidor *DNS* utilizado no sistema.

O *Android* e suas variações não são as melhores opções para se usar como *host* para análise de tráfego, porém a sua rápida popularização e os baixos custos dos dispositivos embarcados, tornam possível que nos deparemos com um dispositivo deste para trabalhar. A solução mais viável é a instalação de uma aplicação de acesso remoto e utilizar o dispositivo da mesma forma que um *thinclient* ou realizar a adaptação do sistema instalando alguns dos *softwares* citados anteriormente.

#### 4. LABORATÓRIO CASOS DE TESTES

A seleção das ferramentas e comandos de análise de tráfego teve como critério principal a fácil acessibilidade e a amplitude a diversas plataformas, sistemas operacionais diferentes. A grande maioria das ferramentas utilizadas nos casos de testes são nativas dos sistemas operacionais *GNU/Linux* e possuem versões portadas para o *Microsoft Windows*, as que não são, podem ser obtidas facilmente na *internet* pois são *freeware*.

O trabalho utilizou os sistemas operacionais *Debian* e *Ubuntu*, a instalação de softwares do repositório nestas distribuições é realizada através dos seguintes comandos:

Tabela 01:

Comandos:	Efeito:
# apt-get update	Sincroniza a máquina local com os repositórios de <i>software</i> configurado.
# apt-get install tcpdump	Instala o <i>software</i> 'tcpdump'.

Fonte: O autor.

Para apresentação das ferramentas abordadas foi consolidada três tabelas, sendo que suas informações são oriundas das páginas de manual distribuídas com os *softwares*. Para a consulta da página do manual de uma *software* em *GNU/Linux* se utiliza o seguinte comando:

Tabela 02:

Comando:	Efeito:
<code>\$ man wireshark</code>	Mostra o manual do <i>software</i> 'wireshark' se estiver disponível (presente ou instalado no sistema).

Fonte: O autor.

Página inicial do manual do *software Wireshark*, instalado em uma distribuição *GNU/Linux*.

Figura 04. Imagem do Manual do *software Wireshark* no Terminal

```

WIRESHARK(1)                               The Wireshark Network Analyzer                               WIRESHARK(1)
NAME
wireshark - Interactively dump and analyze network traffic

SYNOPSIS
wireshark [ -a <capture autostop condition> ] ...
[ -b <capture ring buffer option> ] ... [ -B <capture buffer size> ]
[ -c <capture packet count> ] [ -C <configuration profile> ]
[ -d <layer type>==<selector>,<decode-as protocol> ] [ -D ]
[ --display=<X display to use> ] [ -f <capture filter> ]
[ -g <packet number> ] [ -h ] [ -H ] [ -i <capture interface>|- ] [ -I ]
[ -j ] [ -J <jump filter> ] [ -k ] [ -K <keytab> ] [ -l ] [ -L ] [ -m <font> ]
[ -n ] [ -N <name resolving flags> ] [ -o <preference/recent setting> ] ...
[ -p ] [ -P <path setting> ] [ -r <infile> ] [ -R <read (display) filter> ]
[ -s <capture snaplen> ] [ -S ] [ -t a|ad|adoy|d|dd|e|r|u|ud|udoy ] [ -v ]
[ -w <outfile> ] [ -X <eXtension option> ] [ -y <capture link type> ]
[ -Y <displaY filter> ] [ -z <statistics> ] [ <infile> ]

DESCRIPTION
Wireshark is a GUI network protocol analyzer. It lets you interactively
browse packet data from a live network or from a previously saved capture
file. Wireshark's native capture file format is pcap format, which is also
the format used by tcpdump and various other tools.

Manual page wireshark(1) line 1 (press h for help or q to quit)

```

Fonte: O autor.

Tabela de comandos para auditoria local:

Tabela 03:

Comando	Função	Plataforma	Licença	Obtenção no trabalho
<b>ifconfig</b>	Mostra as configurações de todos os adaptadores de rede ativos na máquina.	Sistemas Unix	Código aberto e software livre.	Nativamente presente em sistemas <i>unix-like</i> .
<b>ipconfig</b>	Mostra configurações da rede local da máquina.	Microsoft Windows	Proprietário e freeware.	Nativamente presente no sistema desde a versão <i>Windows 95</i> .
<b>ethtool</b>	Mostra as características e as atividades de qualquer placa de	<i>Gnu/Linux</i>	Licença <i>GNU GPL v2</i>	Nos próprios repositórios oficiais de sistemas



	rede, incluindo sua situação de link			<i>debian-based.</i>
<b>route</b>	Mostra e edita as tabelas de roteamento de rede (roteamento estático).	Sistemas Unix e Microsoft Windows	Código aberto e software livre.	Nativo dos sistemas.
<b>netstat</b>	Mostra diversos dados sobre a rede, dentre eles as portas TCP e UDP que estão em operação na máquina.	Sistemas Unix e Microsoft Windows	Código aberto e software livre.	Nativo dos sistemas.

Fonte: O autor.

Tabela de Comandos para levantamento de dados:

Tabela 04:

Comando	Função	Plataforma	Licença	Obtenção no trabalho
<b>ping</b>	Possibilita saber se o pacote está chegando em seu destino.	Sistemas <i>Unix</i> e <i>Microsoft Windows</i> e outros.	Código aberto e software livre.	Nativo em praticamente todos os sistemas operacionais existentes.
<b>tracert (windows)</b> <b>tracert (unix-like)</b>	Possibilita descobrir toda rota entre dois pontos em uma rede.	Sistemas <i>Unix</i> e <i>Microsoft Windows</i> .	Código aberto e software livre.	Nativo em sistemas Windows. Pode ser instalado através dos repositórios oficiais em sistemas <i>debian-based</i> .
<b>mtr</b>	Semelhante ao <i>tracert</i> , porém ele mostra constantemente a rota até a determinada máquina	Sistemas <i>Unix</i>	Licença GNU GPL v2	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> .
<b>whois</b>	Mostra dados sobre domínios e blocos de IP na Internet, útil para pesquisas sobre redes ou entidades.	Sistemas <i>Unix</i> e <i>Microsoft Windows</i> .	Código aberto e software livre.	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> . Em sistemas Windows a ferramenta pode ser obtida no site da Microsoft.
<b>dig</b>	Realiza pesquisas em servidores DNS para verificar endereços IP.	Sistemas <i>Unix</i> .	Código aberto e software livre.	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> .
<b>netdiscover</b>	Possibilita realizar o levantamento de máquinas da rede, pois lista o MAC e o IP das mesmas. Pode atuar como ativo ou passivo.	Sistemas <i>Unix</i> .	Código aberto e software livre.	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> .

Fonte: O autor.

Tabela de ferramentas para análise de tráfego:

Tabela 05:

Ferramenta	Função	Plataforma	Licença	Obtenção no trabalho
<b>tcpdump</b>	Analisador de tráfego em modo texto. Mostra as conexões estabelecidas e o tráfego correspondente.	Sistemas <i>Unix</i> ..	Licença <i>GNU GPL v2</i> .	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> .
<b>windump</b>	É a versão do tcpdump para MS Windows.	<i>Microsoft Windows</i> .	Licença <i>GNU GPL v2</i> .	Download no <i>site</i> : <a href="https://www.winpcap.org/">&lt;https://www.winpcap.org/&gt;</a>
<b>wireshark</b>	Analisador de tráfego com interface gráfica. Capaz de ler arquivos de tráfego gerados pelo tcpdump.	Sistemas <i>unix</i> e <i>Microsoft Windows</i> .	Licença <i>GNU GPL v2</i> .	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> . Para sistemas <i>windows</i> no <i>site</i> : <a href="https://www.wireshark.org/">&lt;https://www.wireshark.org&gt;</a>
<b>tshark</b>	É uma implementação do wireshark em modo texto, também é similar ao tcpdump.	Sistemas <i>unix</i> .	Licença <i>GNU GPL v2</i> .	Nos próprios repositórios oficiais de sistemas <i>debian-based</i> .

Fonte: O autor.

#### 4.1 CASO DE TESTE 01: DESCOBRINDO MODIFICAÇÕES EM CONFIGURAÇÕES LOCAIS NO PRÓPRIO *HOST*

Este é o caso de teste mais simples, nele temos uma situação hipotética em que o empregado de uma empresa de pequeno porte modifica as configurações de rede de sua própria estação de trabalho para obter privilégios. Os privilégios desta modificação podem ser diversos, desde se aproveitar de uma má configuração de rede do administrador da rede, até se passar por um outro usuário. Uma ação simples como essa só terá efeitos em pequenas redes onde o administrador da rede é inexperiente, ou não existe, e utiliza sistemas de controle de acesso extremamente rudimentares.

Para descobrirmos a configuração de rede em utilização aplicamos os seguintes comandos de auditoria local:

Tabela 06:

Comandos:	Efeito:
<b>\$ ifconfig</b>	Apresenta as configurações de rede das interfaces ativas. Observação, algumas distribuições <i>GNU/Linux</i> necessitam ser o usuário <i>root</i> para aplicar este comando.
<b>\$ ethtool enp3s0</b>	Após se descobrir a interface de rede ativa, que é a 'enp3s0', se utiliza este comando para saber mais características da interface de rede.

Fonte: O autor.

Resultado da utilização do comando *ifconfig*:

Figura 05. Imagem de retorno do comando no Terminal

```

enp3s0  Link encap:Ethernet  HWaddr 10:bf:48:75:7a:2c
        inet addr:192.168.7.200  Bcast:192.168.7.255  Mask:255.255.255.0
        inet6 addr: 2804:d45:182c:6600:5e4b:6c62:a60:e59e/64  Scope:Global
        inet6 addr: fe80::921b:c2ff:7ad:bb5c/64  Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:622508 errors:0 dropped:3397 overruns:0 frame:0
        TX packets:952393 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:731262950 (731.2 MB)  TX bytes:66929636 (66.9 MB)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128  Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:22606 errors:0 dropped:0 overruns:0 frame:0
        TX packets:22606 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:2682310 (2.6 MB)  TX bytes:2682310 (2.6 MB)

```

Fonte: O autor.

Resultado da utilização do comando *ethtool*:

Figura 06. Imagem de retorno do comando no Terminal

```

Settings for enp3s0:
  Supported ports: [ TP MII ]
  Supported link modes:   10baseT/Half 10baseT/Full
                        100baseT/Half 100baseT/Full
                        1000baseT/Half 1000baseT/Full
  Supported pause frame use: No
  Supports auto-negotiation: Yes
  Advertised link modes:  10baseT/Half 10baseT/Full
                        100baseT/Half 100baseT/Full
                        1000baseT/Half 1000baseT/Full
  Advertised pause frame use: Symmetric Receive-only
  Advertised auto-negotiation: Yes
  Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                       100baseT/Half 100baseT/Full
  Link partner advertised pause frame use: Symmetric Receive-only
  Link partner advertised auto-negotiation: Yes
  Speed: 100Mb/s
  Duplex: Full
  Port: MII
  PHYAD: 0
  Transceiver: internal
  Auto-negotiation: on
Cannot get wake-on-lan settings: Operation not permitted
  Current message level: 0x00000033 (51)
                        drv probe ifdown ifup
  Link detected: yes

```

Fonte: O autor.

Através desse simples par de comandos extraímos as seguintes informações:

- A máquina possui apenas uma interface de rede ativa que é a **'enp3s0'** e trata-se provavelmente de uma rede cabeada do padrão *fast ou gigabit ethernet*,
- O *MAC Address* da máquina é **10:bf:48:75:7a:2c**,
- O endereço *IPv4* da máquina é **192.168.7.200** e máscara de subrede é **255.255.255.0**,
- O endereço *IPv6* é **2804:d45:182c:6600:5e4b:6c62:a60:e59e/64**,
- Ocorre transferência de dados (pacotes) na rede,
- O endereço de *broadcast* é **192.168.7.255**.

#### 4.2 CASO DE TESTE 02: MUDANÇAS EM TABELAS DE ROTEAMENTO LOCAL COM POSSÍVEIS FINS MALICIOSOS

Neste Caso de teste o cenário também ocorre em uma empresa de pequeno porte que possui duas rotas de saída, sendo uma de uso da alta administração e outra dos demais funcionários. Em uma verificação de rotina que pode ser feita inclusive de forma automatizada através *scripts* foi verificado que existia uma máquina usando uma rota errada, possivelmente com objetivo de obter vantagens quanto a política de acesso. Esse caso também é muito rudimentar, pois existem várias maneiras de se isolar redes, uma delas é através de *switches layer 3* que são capazes de criar *VLANs*.

Tabela 07:

Comando:	Efeito:
<b>\$ route</b>	Mostra as tabelas de roteamento, sendo encontrado na primeira máquina a rota <b>192.168.7.1</b> e na segunda máquina <b>192.168.7.7</b> .

Fonte: O autor.

Resultado encontrado na primeira máquina:

Figura 07. Imagem de retorno do comando no Terminal

```
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 192.168.7.1 0.0.0.0 UG 100 0 0 enp3s0
link-local * 255.255.0.0 U 1000 0 0 enp3s0
192.168.7.0 * 255.255.255.0 U 100 0 0 enp3s0
```

Fonte: O autor.

Resultado encontrado na segunda máquina:

Figura 08. Imagem de retorno do comando no Terminal

```
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 192.168.7.7 0.0.0.0 UG 100 0 0 enp3s0
link-local * 255.255.0.0 U 1000 0 0 enp3s0
192.168.7.0 * 255.255.255.0 U 100 0 0 enp3s0
```

Fonte: O autor.

Em nosso caso de teste a divergência encontrada claramente evidencia um problema, principalmente se existir um servidor *dhcp* que atribui as configurações de rede, incluindo logicamente a rota padrão, através de um *MAC Address* cadastrado. Com mais este comando simples e presente em quase todos os sistemas operacionais podemos levantar mais algumas informações clarividentes sobre problemas reais na administração da redes e/ou violações de usuários.

#### 4.3 CASO DE TESTE 03: DESCOBRINDO CONEXÕES SUSPEITAS EM ATIVIDADE

Neste caso de teste abordaremos uma máquina em utilização de nossa organização fictícia, veremos todas as conexões ativas no momento, isso pode ser feito remotamente pelo administrador da rede. O *netstat* está disponível praticamente em todos os sistemas operacionais. Realizando uma breve análise podemos levantar algumas conclusões interessantes.

Tabela 08:

Comandos:	Efeito:
\$ <b>netstat -help</b>	Apresenta os principais parâmetros do comando. No caso selecionaremos o <b>t,u,n,a</b> e <b>p</b> .
\$ <b>netstat -tunap</b>	Mostra todas as conexões e portas TCP e UDP ativas (clientes e servidoras) do usuário 'logado', sem resolver nomes.
# <b>netstat -tunap</b>	Mostra todas as conexões e portas TCP e UDP ativas (clientes e servidoras), sem resolver. nomes.

Fonte: O autor.

Resultado encontrado utilizando o comando *netstat*:

Figura 09. Imagem de retorno do comando no Terminal

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 0.0.0.0:6667           0.0.0.0:*              LISTEN
tcp      0      0 0.0.0.0:6668           0.0.0.0:*              LISTEN
tcp      0      0 0.0.0.0:6669           0.0.0.0:*              LISTEN
tcp      0      0 127.0.1.1:53           0.0.0.0:*              LISTEN
tcp      0      0 0.0.0.0:22             0.0.0.0:*              LISTEN
tcp      0      0 0.0.0.0:3000           0.0.0.0:*              LISTEN
tcp      0      0 127.0.0.1:5432         0.0.0.0:*              LISTEN
tcp      0      0 127.0.0.1:25           0.0.0.0:*              LISTEN
tcp      0      0 192.168.7.200:39864    104.16.127.228:80      ESTABLISHED
tcp6     0      0 :::22                  :::*                    LISTEN
tcp6     0      0 :::1:25                 :::*                    LISTEN
tcp6     0      0 2804:d45:182c:660:51638 2800:3f0:4004:801::443 ESTABLISHED
tcp6     0      0 2804:d45:182c:660:53184 2800:3f0:4001:12::1:443 ESTABLISHED
tcp6     0      0 2804:d45:182c:660:58022 2800:3f0:4004:804::443 ESTABLISHED
tcp6     0      0 2804:d45:182c:660:53302 2800:3f0:4004:805::443 ESTABLISHED
tcp6     0      0 2804:d45:182c:660:58980 2606:2800:220:5c1:2:443 ESTABLISHED
tcp6     0      0 2804:d45:182c:660:49100 2607:f8b0:4004:80a::443 ESTABLISHED
udp      0      0 0.0.0.0:43858          0.0.0.0:*              LISTEN
udp      0      0 0.0.0.0:52193          0.0.0.0:*              LISTEN
udp      0      0 0.0.0.0:5353           0.0.0.0:*              LISTEN
udp      0      0 127.0.1.1:53           0.0.0.0:*              LISTEN
udp      0      0 192.168.7.200:123      0.0.0.0:*              LISTEN
udp      0      0 127.0.0.1:123          0.0.0.0:*              LISTEN
udp      0      0 0.0.0.0:123            0.0.0.0:*              LISTEN
udp      0      0 127.0.0.1:33140        127.0.0.1:33140        ESTABLISHED
udp      0      0 0.0.0.0:631            0.0.0.0:*              LISTEN
udp6     0      0 :::33512                :::*                    LISTEN
udp6     0      0 :::5353                  :::*                    LISTEN
udp6     0      0 fe80::921b:c2ff:7ad:123 :::*                    LISTEN
udp6     0      0 2804:d45:182c:6600::123 :::*                    LISTEN
udp6     0      0 :::1:123                 :::*                    LISTEN
udp6     0      0 :::123                   :::*                    LISTEN
```

Fonte: O autor.

A resposta do comando apresenta muitas informações esperadas e outras não. Analisando as informações esperadas verificamos interações do protocolo *NTP*, verificado pelas portas 123 aberta no computador, lembrando que o *IP* 0.0.0.0 refere-se a este computador, temos conexões de *DNS*, com as portas 53 e 5353, temos também um servidor *SSH* ativo, porta 22, maneira pela qual o administrador da rede poderá executar o *netstat* remotamente, temos portas altas abertas formando sockets de conexões com portas do serviço *HTTPS* e outras portas de serviços muito comuns. O que realmente chama a atenção de diferente são as três portas, 6667, 6668 e 6669 abertas na máquina, em nosso caso de ilustração abrimos elas propositalmente através de uma ferramenta chamada *netcat*, considerado o canivete suíço do *TCP/IP*. Ao se deparar em uma análise real com uma situação de portas desconhecidas abertas, é aconselhável consultar <<https://www.iana.org/assignments/service-names-port-numbers>>, pois conexões estranhas abertas

é um forte indício de um malware em execução na máquina. Pesquisando na internet pela porta aberta encontrada é possível na maioria das vezes descobrir qual é o *malware* e o que ele faz. Possuindo um firewall bem configurado na rede pode-se inviabilizar ataques externos.

#### 4.4 CASO DE TESTE 04: DESCOBRINDO UM SISTEMA OPERACIONAL A PARTIR DE QUALQUER *HOST*

Este caso de teste é na verdade a demonstração do poder de uma ferramenta tão simples e que muitos administradores não sabem interpretar o seu real potencial. Trata-se do conhecido comando *ping*, presente também em praticamente todos os sistemas operacionais existentes. Não vou apresentar um caso contextual, pois as possibilidades de uso são infinitas. Muitos softwares se utilizam do *ping* e de seus pacotes *echo request icmp* para mostrar informações mais elaboradas e varrer uma rede. A grande limitação ocorre quando o pacote *icmp* encontra no caminho um *firewall* que o ‘dropa’, descarta sem dar resposta, isso nos gera algumas dúvidas dependendo da situação.

Com o *ping* descobrimos se o *host* está *online*, como está a qualidade da conexão, a possível quantidade de roteadores e até chegar no *host* ‘pingado’ e até mesmo o possível sistema operacional que está sendo utilizado no *host* que está recebendo os pacotes *icmp*.

Tabela 09:

Comandos:	Efeito:
\$ ping 192.168.7.201	Comando <i>ping</i> executado no <i>host</i> <b>192.168.7.201</b> .
\$ ping 192.168.7.205	Comando <i>ping</i> executado no <i>host</i> <b>192.168.7.205</b> .
\$ ping 8.8.8.8	Comando <i>ping</i> executado no DNS público do Google na <i>internet</i> .

Fonte: O autor.

Para mostrar a diversidade de plataformas em que o *ping* pode ser utilizado, as figuras abaixo apresentam ele sendo executado a partir de um *smartphone* com o sistema operacional Android 6.0, lembrando que o sistema Android utiliza o kernel Linux.

Figura 10. Imagem de retorno do comando no *software Terminal Emulador* em dispositivo *Android*

```

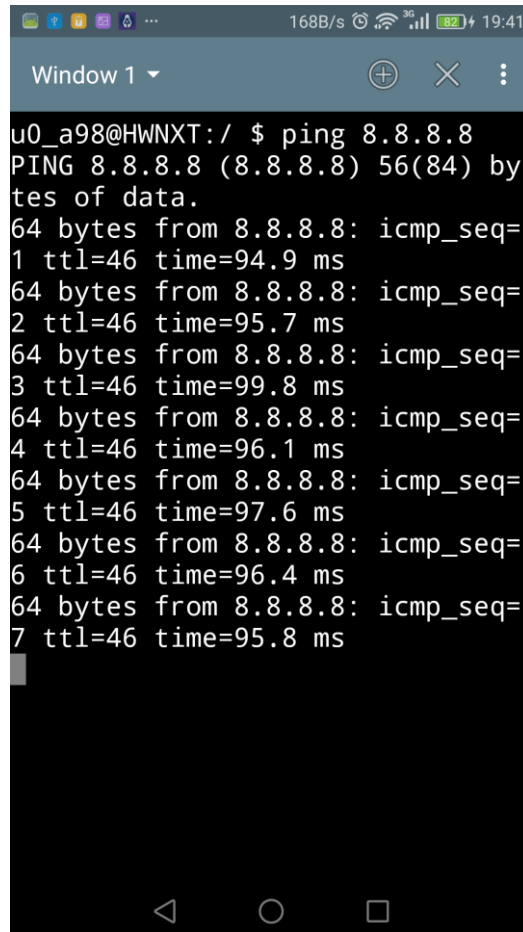
ing 192.168.7.201 <
PING 192.168.7.201 (192.168.7.201) 56(84) bytes of data.
64 bytes from 192.168.7.201: icmp_seq=1 ttl=128 time=8.84 ms
64 bytes from 192.168.7.201: icmp_seq=2 ttl=128 time=11.3 ms
64 bytes from 192.168.7.201: icmp_seq=3 ttl=128 time=8.29 ms
64 bytes from 192.168.7.201: icmp_seq=4 ttl=128 time=7.56 ms
64 bytes from 192.168.7.201: icmp_seq=5 ttl=128 time=10.2 ms
64 bytes from 192.168.7.201: icmp_seq=6 ttl=128 time=10.4 ms
64 bytes from 192.168.7.201: icmp_seq=7 ttl=128 time=9.79 ms
64 bytes from 192.168.7.201: icmp_seq=8 ttl=128 time=7.62 ms
64 bytes from 192.168.7.201: icmp_seq=9 ttl=128 time=13.3 ms
64 bytes from 192.168.7.201: icmp_seq=10 ttl=128 time=7.79 ms

ing 192.168.7.205 <
PING 192.168.7.205 (192.168.7.205) 56(84) bytes of data.
64 bytes from 192.168.7.205: icmp_seq=1 ttl=64 time=9.94 ms
64 bytes from 192.168.7.205: icmp_seq=2 ttl=64 time=9.64 ms
64 bytes from 192.168.7.205: icmp_seq=3 ttl=64 time=8.62 ms
64 bytes from 192.168.7.205: icmp_seq=4 ttl=64 time=15.1 ms
64 bytes from 192.168.7.205: icmp_seq=5 ttl=64 time=18.3 ms
64 bytes from 192.168.7.205: icmp_seq=6 ttl=64 time=11.9 ms
64 bytes from 192.168.7.205: icmp_seq=7 ttl=64 time=10.9 ms
64 bytes from 192.168.7.205: icmp_seq=8 ttl=64 time=10.0 ms
  
```

Fonte: O autor.

Os *hosts* que estão enviando o *echo reply* estão na mesma rede que nosso *smartphone*, neste caso podemos dizer que provavelmente o *IP* de final 201 é um *host MS Windows* pois seu *TTL* integral é 128 e o *host* de final 205 é um sistema *GNU/Linux* pois seu *TTL* integral é 64. O *TTL* é sempre decrementado ao passar por um roteador. O *TTL* inicial pode ser modificado tanto em sistemas *GNU/Linux* quanto *MS Windows*, mas isto raramente ocorre. Na figura abaixo disparamos o *ping* contra o *DNS* público do Google, que sabemos por ser uma informação pública que é um servidor *GNU/Linux*.



Figura 11. Imagem de retorno do comando no *software Terminal Emulator* em dispositivo *Android*

```
Window 1
u0_a98@HWNXT: / $ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=46 time=94.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=46 time=95.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=46 time=99.8 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=46 time=96.1 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=46 time=97.6 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=46 time=96.4 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=46 time=95.8 ms
```

Fonte: O autor.

Podemos verificar que ocorreu um decremento de 18 *hops* (=64-46), portanto a partir do *smartphone* que executei o *echo request*, no caminho de volta do *reply* existem provavelmente 18 roteadores.

#### 4.5 CASO DE TESTE 05: MUDANÇAS EM TABELAS DE ROTEAMENTO LOCAL COM POSSÍVEIS FINS MALICIOSOS

O cenário deste Caso de teste ocorre em uma empresa de pequeno a médio porte que possui um endereço de *intranet* e conseqüentemente um *DNS* local para facilitar o acesso dos funcionários. O *DNS* como já é sabido é um serviço de rede responsável por ‘transformar’ endereços *IP* em ‘nomes fantasia’ e vice-versa. No caso da rede da nossa pequena empresa, o servidor *DNS* local serve apenas para dar um nome mais fácil para os serviços web locais utilizados, sendo que as requisições de endereços da internet são repassadas pelo *DNS* local para o *DNS* do *ISP* da empresa.

Recentemente um funcionário da empresa relatou para seus amigos que teve modificações estranhas na sua conta de *internet banking*. Para verificar a situação utilizaremos os seguintes comandos em duas máquinas da empresa, uma aleatória e outra a do funcionário que teve problema com seu *internet banking*.

Tabela 10:

Comandos:	Efeito:
\$ route	Mostra as tabelas de roteamento, sendo no caso iguais nas duas máquinas.
\$ traceroute -4 www.bb.com.br	Apresenta a rota entre o meu <i>host</i> e o <i>site</i> do Banco do Brasil na <i>internet</i> . Os resultados são diferentes nas máquinas testadas.
\$ mtr -4 www.bb.com.br	Apresenta a rota interativa entre o meu <i>host</i> e o <i>site</i> do Banco do Brasil na <i>internet</i> .

Fonte: O autor.

Retorno do comando *route* na máquina aleatória:

Figura 12. Imagem de retorno do comando no Terminal

```
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
default          192.168.7.7    0.0.0.0         UG    600    0      0 wlp3s0
192.168.7.0      0.0.0.0        255.255.255.0   U     600    0      0 wlp3s0
```

Fonte: O autor.

Retorno do comando *traceroute* na máquina aleatória:

Figura 13. Imagem de retorno do comando no Terminal

```
traceroute to www.bb.com.br (170.66.11.10), 30 hops max, 60 byte packets
 1 192.168.7.7 (192.168.7.7) 12.681 ms 14.212 ms 14.639 ms
 2 * * *
 3 100.122.49.40 (100.122.49.40) 37.661 ms 100.122.49.42 (100.122.49.42) 38.450 ms 100
.122.49.78 (100.122.49.78) 43.087 ms
 4 100.122.20.185 (100.122.20.185) 48.490 ms 50.884 ms 51.376 ms
 5 100.122.17.70 (100.122.17.70) 49.606 ms 49.606 ms 51.366 ms
 6 100.122.22.93 (100.122.22.93) 53.864 ms 32.111 ms 35.742 ms
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 186-230-140-178.ded.intelignet.com.br (186.230.140.178) 92.635 ms 83.404 ms 85.492
```

Fonte: O autor.

Retorno da aplicação *mtr*, semelhante ao *traceroute* porém apresenta a rota continuamente atualizada:

Figura 14. Imagem de retorno do comando no Terminal

```

My traceroute [v0.87]
rallow2 (0.0.0.0) Sun Jul 2 17:32:58 2017
Keys: Help Display mode Restart statistics Order of fields quit
          Packets
Host      Loss%  Snt   Last   Avg   Best  Wrst  StDev
1. myrouter.domain.name 0.0%   36    2.2   3.4   1.8   9.5   1.9
2. 187-79-184-1.user.veloxzone.com.br 86.1%   36 25261 25725 25261 26089 323.1
3. 100.122.49.42 94.3%   36  29.3  29.4  29.3  29.5   0.0
4. 100.122.20.193 94.1%   35  31.2  32.3  31.2  33.5   1.4
5. 100.122.17.180 94.1%   35  31.0  31.4  31.0  31.7   0.0
6. 100.122.22.93 71.4%   35  33.0  33.1  31.2  35.1   1.2
7. ???
8. ???
9. ???
10. ???
11. ???
12. 186-230-140-178.ded.intelignet.com.br 64.7%   35  70.6  69.6  61.7  75.0   3.7
13. ???

```

Fonte: O autor.

Através dos resultados obtidos e apresentados nas imagens, podemos concluir algumas coisas, a saída padrão do *host* é **192.168.7.7**, é notável também que a partir do primeiro *hop* os pacotes *icmp* já chegam a internet através de um roteador do *ISP* que fornece o *link*. Os pacotes seguem por roteadores da *internet* em busca de seu destino que é o *site* do Banco do Brasil. Naturalmente percebemos que alguns roteadores não respondem aos *echo requests* o que é algo normal, pois isso varia de empresa para empresa que administra o roteador.

Retorno do comando *route* na máquina do funcionário afetado:

Figura 15. Imagem de retorno do comando no Terminal

```

Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
default 192.168.7.7 0.0.0.0 UG 100 0 0 enp3s0
link-local * 255.255.0.0 U 1000 0 0 enp3s0
192.168.7.0 * 255.255.255.0 U 100 0 0 enp3s0

```

Fonte: O autor.

Retorno do comando *traceroute* na máquina do funcionário afetado:

Figura 16. Imagem de retorno do comando no Terminal

```

traceroute to www.bb.com.br (192.168.7.222), 30 hops max, 60 byte packets
1 192.168.7.222 (192.168.7.222) 0.231 ms 0.262 ms 0.289 ms

```

Fonte: O autor.

Na segunda imagem percebemos que não ocorreu alterações na tabela de rotas, porém o *traceroute* encaminha os pacotes para acessar o *site* do Banco do Brasil a um *host* local de IP **192.168.7.222**, obviamente o site oficial do Banco do Brasil não está hospedado em um *host* da rede local da empresa, portanto estamos diante de uma técnica conhecida como *DNS Poisoning*, ou em português envenenamento de *DNS*. No caso podemos perceber que não foram todas as máquinas da rede afetadas. O que provavelmente ocorreu foi a criação de um *DNS* paralelo na rede local e ele foi configurado estaticamente em alguns *hosts* da rede, assim evitando grandes alardes.

Com mais um par de comandos básicos podemos desvendar mais um problema de possível cunho criminoso, utilização da técnica de *phishing* sobre o *site* do Banco do Brasil associada a um envenenamento de *DNS*.

#### 4.6 CASO DE TESTE 06: DESCOBRINDO INFORMAÇÕES SOBRE DOMÍNIOS

Em uma organização nota-se nos registros do *proxy* acessos a determinados domínios que violam a política de segurança, porém o bloqueio de acesso deles ainda não foi implementado nos sistemas de controle de acesso.

Para adquirir mais informações sobre os domínios e subsidiar um registro formal do usuário que o acessou o conteúdo proibido podemos fazer consultas à servidores *DNS*, com isso obteremos dados para auxiliar em uma implementação eficiente de restrição no *proxy*. Os comandos utilizados para o levantamento de informações são:

Tabela 11:

Comandos:	Efeito:
\$ dig www.unisul.br @138.197.25.214	Buscou pelo endereço utilizando o servidor <b>138.197.25.214</b> .
\$ whois www.unisul.br	Mostra dados completos sobre o domínio www.unisul.br
\$ whois 200.237.249.66	Mostra dados sobre o IP <b>200.237.249.66</b> e seu domínio e bloco de IP.

Fonte: O autor.

Retorno de tela após a utilização do comando *dig*:

Figura 17. Imagem de retorno do comando no Terminal

```

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.unisul.br @138.197.25.214
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23616
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags;; udp: 4096
;; QUESTION SECTION:
;www.unisul.br.                IN      A

;; ANSWER SECTION:
www.unisul.br.                900     IN      A      200.237.249.66

;; Query time: 368 msec
;; SERVER: 138.197.25.214#53(138.197.25.214)
;; WHEN: Sun Jul 02 23:10:54 BRT 2017
;; MSG SIZE rcvd: 58

```

Fonte: O autor

Retorno de tela após a utilização do comando *whois* no domínio:

Figura 18. Imagem de retorno do comando no Terminal

```

domain:      unisul.br
owner:       Fundacao Universidade do Sul de Santa Catarina
ownerid:     86.445.293/0001-36
responsible: Tatiane dos Santos Leal
country:     BR
owner-c:     DBC46
admin-c:     DBC46
tech-c:      DBC46
billing-c:   DBC46
nsserver:    ns.unisul.br 200.237.249.100
nsstat:      20170629 AA
nslastaa:    20170629
nsserver:    ns1.unisul.br 200.237.249.101
nsstat:      20170629 AA
nslastaa:    20170629
nsserver:    ns2.unisul.br 200.237.249.102
nsstat:      20170629 AA
nslastaa:    20170629
nsserver:    adns1.pop-sc.rnp.br
nsstat:      20170629 AA
nslastaa:    20170629
nsserver:    adns2.pop-sc.rnp.br
nsstat:      20170629 AA
nslastaa:    20170629
created:     20000208 #246301
changed:     20170308
status:      published

nic-hdl-br: DBC46
person:      Daniel Bitencourt Cadorin
e-mail:      operacoes.ti@unisul.br
country:     BR
created:     20021217
changed:     20131216

```

Fonte: O autor

Retorno de tela após a utilização do comando *whois* no endereço *IP*:

Figura 19. Imagem de retorno do comando no Terminal

```
inetnum:      200.237.248.0/21
aut-num:      AS11242
abuse-c:      SIRTE8
owner:        Fundacao Universidade do Sul de Santa Catarina
ownerid:      86.445.293/0001-36
responsible:  Tatiane dos Santos Leal
country:      BR
owner-c:      DBC46
tech-c:       DBC46
inetrev:      200.237.249.0/24
nserver:      ns.unisul.br
nsstat:       20170701 AA
nslastaa:     20170701
nserver:      ns1.unisul.br
nsstat:       20170701 AA
nslastaa:     20170701
nserver:      ns2.unisul.br
nsstat:       20170701 AA
nslastaa:     20170701
nserver:      adns1.pop-sc.rnp.br
nsstat:       20170701 AA
nslastaa:     20170701
nserver:      adns2.pop-sc.rnp.br
nsstat:       20170701 AA
nslastaa:     20170701
created:      20080722
changed:      20130307
inetnum-up:   200.237.192.0/18

nic-hdl-br:   DBC46
person:       Daniel Bitencourt Cadorin
e-mail:       operacoes.ti@unisul.br
country:      BR
created:      20021217
changed:      20131216
```

Fonte: O autor.

Algumas informações importantes foram obtidas no retorno destes comandos, para exemplificar utilizamos o *site* da Unisul. Os achados são:

- O endereço *IP* do domínio <www.unisul.br> que é **200.237.249.66**,

- servidores *DNS* da organização, **200.237.249.100**, **200.237.249.101** e **200.237.249.102**,
- Nomes dos responsáveis pela página e pelo registro do domínio,
- Faixa de *IP* do *AS* do domínio **200.237.248.0/21**.

#### 4.7 CASO DE TESTE 07: LEVANTANDO TODOS OS *HOSTS* ATIVOS DE UMA REDE

Neste caso de teste trataremos de um problema em uma grande empresa. Esta empresa possui mais de quinhentos dispositivos de rede, desde *switches*, *access points*, servidores e máquinas dos usuários. Para aumentar a segurança e facilitar a administração a Seção de Tecnologia da Informação fez uma segmentação lógica da rede por meio de *VLANs* por departamentos internos. A política de segurança da empresa não permite que nenhum funcionário utilize dispositivos particulares. Suspeita-se que dentro da *VLAN* de um departamento, **192.168.124.0/24**, está ocorrendo violações da política de segurança e alguns funcionários estão utilizando dispositivos particulares. Para descobrir os *hosts* ativos do departamento foi utilizado os seguintes comandos:

Tabela 12:

Comandos:	Efeito:
<b># netdiscover -help</b>	Retorna as opções de utilização do netdiscover.
<b># netdiscover -r 192.168.124.0/24</b>	Realiza um <i>scan</i> ativo, retornando os endereços MAC e IP dos <i>hosts</i> .
<b># netdiscover -r 192.168.124.0/24 -p</b>	Realiza um <i>scan</i> passivo, retornando os endereços MAC e IP dos <i>hosts</i> .

Fonte: O autor.

Na imagem abaixo foi realizado um *scan* ativo, limitando a execução do *netdiscover* à subrede do departamento, em casos em que não se conhece a faixa de *IPs* da subrede pode-se utilizá-lo sem o *range* de *IPs*.

Figura 20. Imagem de retorno do comando no Terminal

```
Currently scanning: Finished! | Screen View: Unique Hosts
121 Captured ARP Req/Rep packets, from 26 hosts. Total size: 7170
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.124.2	74:d4:35:f0:50:48	46	2760	GIGA-BYTE TECHNOLOGY CO.,LTD.
192.168.124.1	f8:b1:56:26:c8:be	8	480	Dell Inc
192.168.124.6	48:5b:39:b1:bc:f8	2	120	ASUSTek COMPUTER INC.
192.168.124.7	40:61:86:ff:70:68	1	60	MICRO-STAR INT'L CO.,LTD
192.168.124.9	ac:22:0b:bc:2f:cd	1	60	ASUSTek COMPUTER INC.
192.168.124.13	bc:5f:f4:6c:c2:47	1	60	ASRock Incorporation
192.168.124.16	00:1b:b9:9e:1e:20	1	60	Elitegroup Computer System Co.
192.168.124.17	00:1b:a9:bf:16:c7	2	120	BROTHER INDUSTRIES, LTD.
192.168.124.20	00:16:ec:23:39:36	1	60	Elitegroup Computer Systems Co., Ltd.
192.168.124.22	00:1b:b9:9e:1f:65	1	60	Elitegroup Computer System Co.
192.168.124.23	00:16:ec:23:37:ed	3	180	Elitegroup Computer Systems Co., Ltd.
192.168.124.26	00:1b:b9:9e:1e:6c	1	60	Elitegroup Computer System Co.
192.168.124.27	a6:0e:68:17:df:fc	1	60	Unknown vendor
192.168.124.28	00:1a:4d:a9:b2:d1	1	60	GIGA-BYTE TECHNOLOGY CO.,LTD.
192.168.124.29	e0:cb:4e:22:1e:4b	1	60	ASUSTek COMPUTER INC.

Fonte: O autor

Retorno do *netdiscover* sendo utilizado de maneira passiva na rede:

Figura 21. Imagem de retorno do comando no Terminal

```
Currently scanning: (passive) | Screen View: Unique Hosts
131 Captured ARP Req/Rep packets, from 27 hosts. Total size: 7734
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.124.2	74:d4:35:f0:50:48	51	3060	GIGA-BYTE TECHNOLOGY CO.,LTD.
192.168.124.122	fc:aa:14:f5:ea:ef	33	1980	GIGA-BYTE TECHNOLOGY CO.,LTD.
192.168.124.100	4a:74:52:0c:a8:9a	5	300	Unknown vendor
192.168.124.44	00:23:14:e8:14:60	6	252	Intel Corporate
192.168.0.1	c8:3a:35:54:42:10	1	42	Tenda Technology Co., Ltd.
192.168.124.1	f8:b1:56:26:c8:be	8	480	Dell Inc
192.168.124.6	48:5b:39:b1:bc:f8	2	120	ASUSTek COMPUTER INC.
192.168.124.7	40:61:86:ff:70:68	1	60	MICRO-STAR INT'L CO.,LTD
192.168.124.9	ac:22:0b:bc:2f:cd	1	60	ASUSTek COMPUTER INC.
192.168.124.13	bc:5f:f4:6c:c2:47	1	60	ASRock Incorporation
192.168.124.16	00:1b:b9:9e:1e:20	1	60	Elitegroup Computer System Co.
192.168.124.17	00:1b:a9:bf:16:c7	2	120	BROTHER INDUSTRIES, LTD.
192.168.124.20	00:16:ec:23:39:36	1	60	Elitegroup Computer Systems Co., Ltd.
192.168.124.22	00:1b:b9:9e:1f:65	1	60	Elitegroup Computer System Co.
192.168.124.23	00:16:ec:23:37:ed	3	180	Elitegroup Computer Systems Co., Ltd.

Fonte: O autor

Lembrando que sempre que possível deve se utilizar um *scan* passivo, o *scan* ativo envia pacotes ARP em *broadcast* e prejudica o desempenho da rede enquanto estiver em execução.

Após a descobertas dos *hosts* ativos na tabela retornada pelo *netdiscover* o trabalho é apenas de confrontá-la com uma relação de *hosts* realmente previstos para o departamento.



#### 4.8 CASO DE TESTE 08: VISUALIZANDO UM ESCANEAMENTO ATIVO DE REDE COM POSSÍVEL FINALIDADE MALICIOSA

Este caso de teste será realizado no mesmo departamento da grande organização do caso de teste anterior. Na situação existem suspeitas de que um funcionário interno mal-intencionado realiza *scan* na rede com a ferramenta *Tcpdump* afim de descobrir os *hosts* do departamento. Para verificarmos como ocorre a varredura e quem está realizando, utilizaremos os seguintes comandos para análise:

Tabela 13:

<b>Comandos:</b>	<b>Efeito:</b>
<b>\$ man tcpdump</b>	Retorna em tela o manual do tcpdump para escolha de parâmetros adequados para realizar a análise de forma objetiva.
<b># tcpdump -nvi eth1</b>	Apresenta de forma contínua a escuta do tráfego da rede sem resolver nomes, aplicando o modo verboso (mostra mais informações) se utilizando da interface de rede ' <i>eth1</i> ' da máquina.
<b>\$ sudo wireshark</b>	Inicia o software de análise de tráfego que possui interface gráfica Wireshark com permissão de superusuário. A permissão de superusuário é necessária para o correto funcionamento da ferramenta

Fonte: O autor.

Imagem da captura realizada com o *tcpdump*:

Figura 22. Imagem de retorno do comando no Terminal

```

11:40:12.575842 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.1 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.575858 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.1 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.577050 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.2 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.577058 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.2 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.577890 ARP, Ethernet (len 6), IPv4 (len 4), Reply 192.168.124.1 is-at f8:b1:56:26:c8:be, length 46
11:40:12.578214 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.3 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.578221 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.3 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.579199 ARP, Ethernet (len 6), IPv4 (len 4), Reply 192.168.124.2 is-at 74:d4:35:f0:50:48, length 46
11:40:12.579352 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.4 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.579360 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.4 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.580555 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.5 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46
11:40:12.580561 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.124.5 (ff:ff:ff:ff:ff:ff) tell 192.168.124.67, length 46

```

Fonte: O autor

Imagem da captura realizada com o *software Wireshark*:

Figura 23. Imagem da tela do *software Wireshark*

70	5.945875926	PcsCompu_f2:6b:0c	Broadcast	ARP	60 who has 192.168.124.1? Tell 192.168.124.67
71	5.945888632	IntelCor_15:5e:8c	Broadcast	ARP	60 who has 192.168.124.1? Tell 192.168.124.67
72	5.947059939	PcsCompu_f2:6b:0c	Broadcast	ARP	60 who has 192.168.124.2? Tell 192.168.124.67
73	5.947065409	IntelCor_15:5e:8c	Broadcast	ARP	60 who has 192.168.124.2? Tell 192.168.124.67
74	5.947787086	Dell_26:c8:be	IntelCor_15:5e:8c	ARP	60 192.168.124.1 is at f8:b1:56:26:c8:be (dupl
75	5.948222869	PcsCompu_f2:6b:0c	Broadcast	ARP	60 who has 192.168.124.3? Tell 192.168.124.67
76	5.948229456	IntelCor_15:5e:8c	Broadcast	ARP	60 who has 192.168.124.3? Tell 192.168.124.67
77	5.949368492	Giga-Byt_f0:50:48	IntelCor_15:5e:8c	ARP	60 192.168.124.2 is at 74:d4:35:f0:50:48 (dupl
78	5.949617909	PcsCompu_f2:6b:0c	Broadcast	ARP	60 who has 192.168.124.4? Tell 192.168.124.67
79	5.949625142	IntelCor_15:5e:8c	Broadcast	ARP	60 who has 192.168.124.4? Tell 192.168.124.67
80	5.950821486	PcsCompu_f2:6b:0c	Broadcast	ARP	60 who has 192.168.124.5? Tell 192.168.124.67

Fonte: O autor

Imagem do detalhamento do pacote *ARP*, fornecida pelo *Wireshark*:

Figura 24. Imagem da tela do *software Wireshark*

```

▶ Frame 78: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▶ Ethernet II, Src: PcsCompu_f2:6b:0c (08:00:27:f2:6b:0c), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: PcsCompu_f2:6b:0c (08:00:27:f2:6b:0c)
  Sender IP address: 192.168.124.67
  Target MAC address: Broadcast (ff:ff:ff:ff:ff:ff)
  Target IP address: 192.168.124.4

```

Fonte: O autor

Podemos concluir que tanto o *Tcpdump* quanto o *Wireshark* são muito eficientes para análise de tráfego. Conseguimos ver nitidamente os pacotes *ARPs* utilizados na varredura realizada pelo funcionário mal-intencionado. O *Wireshark* implementa uma facilidade a mais na extração de dados devido possuir uma interface gráfica intuitiva e organizada. Na Figura 23 podemos verificar o *MAC Address* da máquina de onde partiram as requisições, **08:00:27:f2:6b:0c**, com esta informação fica muito fácil descobrir o responsável.

#### 4.9 CASO DE TESTE 09: VISUALIZANDO UM ATAQUE DE NEGAÇÃO DE SERVIÇO INTERNO ATRAVÉS DA ANÁLISE DE TRÁFEGO

Neste caso de teste a rede de uma organização possui um servidor *web* para disponibilizar acesso a um sistema de interesse de um determinado departamento interno. Foi verificado que repentinamente o acesso ao sistema estava extremamente lento e frequentemente fora do ar. Resolvemos realizar uma análise nos cabeçalhos de todos os pacotes direcionados ao servidor *web* com o problema. Para isso utilizamos os seguintes comandos com o *tcpdump*.

Tabela 14:

Comando:	Efeito:
<b>\$ man tcpdump</b>	Retorna em tela o manual do <i>tcpdump</i> para escolha de parâmetros adequados para realizar a análise de forma objetiva.
<b># tcpdump -ne host 192.168.124.100 and tcp port 80</b>	Apresenta de forma contínua a escuta do tráfego da rede sem resolver nomes, que envolva o <i>host 192.168.124.100</i> , que seja TCP e tenha como destino a porta 80.

Fonte: O autor.

Imagem do *tcpdump* em execução:

Figura 25. Imagem de retorno do comando no Terminal

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
15:31:46.400071 4a:74:52:0c:a8:9a > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 24
3: 192.168.124.100.138 > 192.168.124.255.138: NBT UDP PACKET(138)
15:36:27.405706 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 98.202.14.61.3290 > 192.168.124.100.80: Flags [none], seq 0:12, win 19327, length 12: H
TTP
15:36:27.405790 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 22.54.151.35.33876 > 192.168.124.100.80: Flags [none], seq 0:12, win 42250, length 12:
HTTP
15:36:27.405867 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 92.27.50.119.10640 > 192.168.124.100.80: Flags [none], seq 0:12, win 17218, length 12:
HTTP
15:36:27.405914 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 221.160.235.46.28792 > 192.168.124.100.80: Flags [none], seq 0:12, win 62815, length 12
: HTTP
15:36:27.405933 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 84.22.125.65.53702 > 192.168.124.100.80: Flags [none], seq 0:12, win 62101, length 12:
HTTP
15:36:27.405972 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
: 128.64.132.39.51897 > 192.168.124.100.80: Flags [none], seq 0:12, win 2890, length 12:
HTTP
15:36:27.405989 dc:53:60:15:5e:8c > 4a:74:52:0c:a8:9a, ethertype IPv4 (0x0800), length 66
```

Fonte: O autor

Através do *tcpdump* conseguimos observar algumas informações elucidativas. Realmente está ocorrendo uma série de requisições ao *host* do servidor web, no *socket* **192.168.124.100:80**. Podemos observar pelo *tcpdump* que os pacotes possuem endereços *IPs* diferentes, porém o mesmo *MAC Address*, o que mostra que é uma tentativa de enganar *IDS/IPS* na rede. A grande quantidade de requisições forjadas encaminhadas a máquina servidora dificulta o envio de respostas legítimas pelo servidor, apresentando o *site* para o usuário como indisponível ou extremamente lento.

#### 4.10 CASO DE TESTE 10: SALVAR UM *LOG* DE CAPTURA PARA POSTERIOR ANÁLISE

Este caso não trata-se exatamente de um caso de teste e sim da apresentação de uma função extremamente útil do *Tcpdump*. Com um parâmetro podemos salvar todo o tráfego de um determinado período de tempo em um arquivo de extensão '*.pcap*', '*.cap*', '*.pcapng*' e etc.

Tabela 15:

Comando:	Efeito:
# tcpdump -w trafego.dump	O parâmetro <b>-w</b> do <i>tcpdump</i> permite armazenar os datagramas

<pre># tcpdump -i wlan0 -w captura.pcap port 80</pre>	capturados em um arquivo binário para posterior análise.
	O comando abaixo fará uma captura na interface <i>wlan0</i> por todo pacote que seja destinado ou tenha como origem a <b>porta 80</b> .

Fonte: O autor.

#### 4. CONSIDERAÇÕES FINAIS

Ao longo do presente trabalho, foi observado que os conceitos que permearam no âmbito da pesquisa bibliográfica contribuíram no processo de desenvolvimento das ideias apresentadas. Com o estudo dos casos de testes concluímos que as informações de interesse na análise de tráfego normalmente são: endereço *IP* inválido ou suspeito, tráfego em portas desconhecidas, tráfego em serviços ou portas que não deveria estar ocorrendo, datagramas com opções, *flags* ou tamanhos que não respeitam os padrões do protocolo previsto na *RFC*, *flood* de datagramas na rede, tráfego *TCP* em portas incomuns, entre outras. Verificamos também que as ferramentas de coleta de informações em redes são *softwares* que podem ser usados para obter dados para uma análise forense. Estas ferramentas geralmente oferecem a funcionalidade de um *sniffer* (*Tcpdump* e *Wireshark*).

Os *sniffers* operam na camada de enlace do Modelo *OSI*. Isso significa que eles não têm que se submeter às regras que as aplicações e serviços que residem nas camadas superiores. Eles podem capturar tudo e gravar para posterior análise, permitindo que o perito forense analise todos os dados que estão contidos nos pacotes e datagramas em um segundo momento, conforme verificado no Caso de Teste número 10, podendo ainda utilizar filtros personalizados para otimizar o trabalho.

No Caso de Teste 01 encontramos configurações sensíveis sobre interfaces redes, modos de operação, tipos de interfaces e outros parâmetros típicos de adaptadores de rede. As configurações utilizadas na rede conectada pela máquina analisada também foram expostas. O caso da utilização de um comando de administração de rede comum e sua maior limitação é a necessidade de possuir um acesso de *root* a máquina em estudo.

No Caso de Teste 02 foi descoberta a tabela de roteamento ativa, podendo trazer ao perito o entendimento da direção do fluxo de dados na rede facilitando também a descoberta de

possíveis fraudes em tabelas de roteamento com objetivo de desviar a informação do seu caminho desejável.

No Caso de Teste 03 foi feita a descoberta dos serviços clientes e servidores, sockets de rede e possíveis portas indevidas abertas em um firewall.

No Caso de Teste 04 percebemos o poder de um comando tão trivial, presente em quase todos os sistemas operacionais modernos, o *ping*, através dele medimos a eficiência da conexão e dos protocolos de roteamento utilizados, analisamos se a rota prevista está sendo utilizada e descobrimos o possível sistema operacional de um *host*.

No Caso de Teste 05 notamos que o entendimento de uma rota é fundamental pois possíveis ataques envolvendo a legitimidade na resolução DNS e mudanças suspeitas no funcionamento de protocolos de roteamento são fortes indícios de que algo não está certo.

No Caso de Teste 06 foi evidenciada a facilidade de se obter diretamente da Internet informações sobre, blocos de redes de entidades, informações técnicas, contatos sobre administradores de redes e resoluções de nomes. Essas informações associadas a Engenharia Social podem poupar muito trabalho do perito.

No Caso de Teste 07 ocorreu o escaneamento de uma rede sem roteamento, foi possível o levantamento de informações físicas, números de *MAC Address*, endereços *IPs*, marcas de equipamentos, segmentação de redes e organização física e lógica da rede. Todos esses dados são praticamente a identificação de um usuário.

No Caso de Teste 08, com monitoramento contínuo de pacotes/datagramas, *flags* de comunicação, comportamento das interações cliente-servidor de um segmento de rede podemos extrair qualquer informação sobre os usuários legítimos ou não, a única limitação é o conhecimento do perito, pois é fundamental que ele entenda profundamente as especificações dos protocolos utilizados.

No Caso de Teste 09, uma extensão do Caso de Teste 08, verificamos um recurso básico possibilitando restringir o monitoramento a um *host* e/ou serviço (porta) específico aumentando a produtividade na análise, porém o risco é de não se monitorar algo importante que foi ofuscado.

No Caso de Teste 10, uma outra extensão do Caso de Teste 08 fica demonstrado a possibilidade de se armazenar um *sniff* em um arquivo para uma posterior análise. A desvantagem dessa possibilidade é que dependendo do contexto do problema procurado pode-se não encontrar a informação desejada na ‘fatia’ de tempo armazenada.

Como conclusão final as análises das informações, datagramas e pacotes realizadas nos casos de testes de fato apresentam evidências digitais, portanto a partir do tráfego de rede é possível compreender toda a comunicação que ocorre entre a máquina do possível atacante e a máquina vítima, estabelecendo uma sequência de eventos e comparando com as outras evidências encontradas.

## **THE NETWORK TRAFFIC ANALYSIS AS A DIGITAL EVIDENCE COLLECTION**

### **METHOD: STUDY OF CASES WITH TRAFFIC ANALYSIS SOFTWARE**

**Abstract:** The present work had as general objective to demonstrate how the tools of analysis of traffic in networks can contribute to the obtaining of findings that consolidate a digital evidence for the forensic expertise. The work initially enlightens the reader to the subject with the presentations of the most usual tools in traffic analysis for GNU / Linux, Microsoft Windows and Android based systems. The explanation of how to perform the traffic analysis was based on basic practical examples represented by the author through screen images and detailed descriptions in simulated environments. The work contextualized with cases of experimental tests and extracts of software manuals of analysis and traffic monitoring to justify and demonstrate the use of such tools in digital forensics. As conclusion, the author presented some of the information that can be obtained.

**Keywords:** Expertise. Forensic. Traffic Analysis. TCP / IP networks. Monitoring.

## **REFERÊNCIAS**

MOTA FILHO, João Eriberto. **Descobrendo o Linux: entenda o sistema operacional GNU/Linux**. 3ª Ed., São Paulo: Novatec Editora, 2012.

ELEUTÉRIO, Pedro Monteiro da Silva; MACHADO, Marcio Pereira. **Desvendando a Computação Forense**. São Paulo: Novatec Editora, 2011

TANENBAUM, A. S. **Redes de Computadores**. 5ª Ed., Editora Campus (Elsevier), 2011.

SHIMONSKI, Robert. **The Wireshark Field Guide Analyzing and Troubleshooting Network Traffic**. New York, NY: Elsevier Inc., 2013.

MORIMOTO, Carlos Eduardo. **Redes, guia prático: ampliada e atualizada**. 2ª Ed. Porto Alegre: Sul Editores, 2011.

MADEIRA, Mauro Notarnicola. **Forense computacional**. Livro digital. Palhoça. Unisul Virtual, 2012.

GUIMARÃES, Cap QCO Marcello Fernandes de Berredo. **A Aviação do Exército no Processo de Modernização do Ensino**. Revista Pegasus, Taubaté SP, v. 21, p. 1-3, jun. 2015. Disponível em: <[http://www.ciavex.eb.mil.br/pegasus/pegasus21/artigo\\_007.html](http://www.ciavex.eb.mil.br/pegasus/pegasus21/artigo_007.html)>. Acesso em: 20 maio 2017.

FARMER, Dan. VENEMA, Wietse. **Perícia Forense Computacional**. 1ª Ed. São Paulo: Pearson Prentice Hall, 2007.

CARDOSO, Thiago Xavier. **Trabalho de Redes de Computadores I do Curso de Engenharia de Computação e Informação**. UFRJ, 2018. Disponível em: <[https://www.gta.ufrj.br/grad/08\\_1/forense/AnaliseViva.html](https://www.gta.ufrj.br/grad/08_1/forense/AnaliseViva.html)>. Acesso em: 18 mar. 2017.



## ANEXO 'A' – GLOSSÁRIO

**Access Point** – Ponto de acesso. Hotspot. Ponto de rede sem fio criado por transmissor rádio.

**Android** – Sistema Operacional (SO) baseado no núcleo Linux desenvolvido atualmente pela empresa de tecnologia Google.

**ARP** – Address Resolution Protocol. Protocolo de Resolução de Endereços. É um protocolo de telecomunicações usado para resolução de endereços da Camada de Internet em endereços da Camada de Enlace. Definido pela RFC 826.

**AS** – Sistema Autônomo. Coleção de redes IP e roteadores sob controle de uma entidade.

**Broadcast** – É um método de transferência de mensagem para todos os receptores simultaneamente.

**Datagrama** – É uma entidade de dados completa e independente que contém informações suficientes para ser roteada da origem ao destino sem precisar confiar em trocas anteriores entre essa fonte, a máquina de destino e a rede de transporte.

**Debian** – Distribuição Linux.

**Debian-based** – Refere-se as distribuições Linux baseada na distribuição Debian.

**DHCP** – Dynamic Host Control Protocol. Protocolo de Configuração Dinâmica de Host. É um protocolo de serviço TCP/IP que oferece configuração dinâmica de terminais, com concessão de endereços IP de host, máscara de sub-rede, default gateway, número IP de um ou mais servidores DNS.

**DNS** – Domain Name System. É um sistema de gerenciamento de nomes hierárquico e distribuído para computadores, serviços ou qualquer recurso conectado à Internet ou numa rede privada.

**Echo Request** – Requisição 'echo' do protocol ICMP.

**Engenharia Social** – No contexto de segurança da informação, refere-se a manipulação psicológica de pessoas para execução de ações ou divulgar informações confidenciais.

**Fast Ethernet** – Dispositivo de rede de velocidade de 100 Mb/s.

**Flags** – Opções sobre informações. Parâmetros.

**Flood** – Inundar. No contexto do trabalho refere-se a congestionar algum serviço ou sistema, buscando prejudicar o funcionamento e/ou indisponibilizá-lo

**F-droid** – Repositório de aplicativos FOSS (Free and Open Source Software) para sistemas Android.

**Firewall** – Dispositivo de rede de computadores que tem por objetivo aplicar uma política de segurança a um determinado ponto da rede.

**Forense Computacional** – Ciência que abrange todas as questões relacionadas aos crimes praticados na Internet ou fora dela. Estuda como coletar evidências de crimes e violações, analisar e documentar casos, segue as principais metodologias internacionais usadas e adotadas na investigação de crimes comuns.

**Freeware** – Software sem custo de licença de uso. Software grátis.

**Gigabit Ethernet** – Dispositivo de rede de velocidade de 1000 Mb/s.

**GNU** – Sistema operacional tipo Unix cujo o objetivo desde a sua concepção é oferecer um sistema operacional completo totalmente composto de software livre.

**GNU/Linux** – Sistema operacional unix-like baseado no GNU e no Kernel Linux. É comum usar o nome Linux para se referir aos sistemas GNU/Linux, embora seja um termo mais amplo.

**Hop** – No contexto do trabalho refere-se a passagem por um roteador, ocorrendo a subtração do TTL.

**Host** – Dispositivo de rede.

**HTTPS** – Protocolo de transferência de hipertexto com segurança.

**ICMP** – Internet Control Message Protocol. Protocolo de Controle de Mensagem de Internet

**IDS** – Sistema de detecção de intrusos em rede de computadores.

**IETF** – Internet Engineering Task Force, entidade responsável pela edição e normatização dos padrões que serão implementados na Internet.

**Internet** – É um Sistema global de redes de computadores que utilizam um conjunto próprio de protocolos (TCP/IP).

**Internet Banking** – Atendimento bancária através da Internet, possibilitando realizar transações.

**Intranet** – É uma rede de computadores privada, de uso exclusivo de um determinado local, por exemplo uma empresa.

**IP** – Internet Protocol. Protocolo de Internet.

**IPS** – Sistema de prevenção de intrusos em redes de computadores.

**IPv4** – Internet Protocol Version 4. Protocolo de Internet Versão 4.

**IPv6** – Internet Protocol Version 6. Protocolo de Internet Versão 6.

**ISP** – Internet Service Provider. Provedor de acesso a Internet.

**Kernel** – Núcleo do Sistema.

**Layer** – Camada de abstração.

**Link** – Enlace. Enlace de dados.

**Linux** – Referência ao núcleo do Sistema desenvolvido por Linus Torvalds.

**Log** – É uma expressão utilizada para descrever o processo de registro de eventos relevantes num sistema computacional. Esse registro pode ser utilizado para restabelecer o estado original de um sistema ou para que um administrador conheça o seu comportamento no passado. Um arquivo de log pode ser utilizado para auditoria e diagnóstico de problemas em sistemas computacionais.

**Mac Address** – Endereço físico de uma interface de rede.

**Microsoft Windows** – Sistema Operacional para computadores.

**NBNC** – NetBIOS Naming Service.

**Netbios** – É um protocolo da camada de apresentação que utiliza o UDP/TCP e porta 137. É implementado pela Microsoft através do Windows Internet Name Service (WINS).

**Netcat** – É uma ferramenta de rede disponível para sistemas operacionais Unix, Linux, Microsoft Windows e Macintosh que permite, por intermédio de comandos e com sintaxe muito sensível, abrir portas TCP/UDP.

**NTP** – Network Time Protocol. Protocolo de sincronização de relógios em rede de computadores.

**Online** – Ativo. Em funcionamento. Ligado.

**Obfuscado** – Que foi escondido e/ou ocultado.

**Open source** – Código fonte de software aberto.

**Pacote** – Abstração para fazer referência a determinada quantidade de dados separa para sem transportada em uma rede de computadores.

**PC** – Computador pessoal.

**Perícia** – É a análise técnica de uma situação, fato, ou estado redigida por um especialista numa determinada disciplina.

**Phishing** – É o empréstimo que designa as tentativas de obtenção de informação pessoalmente identificável através de uma suplantação de identidade por parte de criminosos no contexto de informática.

**Poisoning** – Envenenamento. No contexto do trabalho trata-se de uma técnica que fraudas o acesso a um sistema legítimo.

**Proxy** – **Sistema** de controle de acesso e cache de uma rede para acessar a Internet ou outra rede.

**Range** – Faixa de números IPs.

**Root** – Usuário administrador de um sistema operacional.

**Reply** – No contexto do trabalho refere-se a resposta do protocolo ICMP após um ‘echo request’.

**RFC** – Request For Comments. São documentos técnicos desenvolvidos e mantidos pelo IETF.

**Scan** – No contexto do trabalho significa varredura, pesquisa por informação.

**Shell** – É o interpretador de linha de comando que fornece uma interface semelhante ao Unix tradicional.

**Site** – É um conjunto de páginas Web, isto é, hipertextos acessíveis geralmente pelo protocolo HTTP na Internet.

**Smartphone** – Telefone que combina recursos de computadores pessoais, com funcionalidades avançadas que podem ser estendidas por meio de programas aplicativos executados pelo seu sistema operacional.

**Sniffer** – No contexto do trabalho, refere-se a quem realiza um sniff, isto é, monitoramento, escuta de tráfego de dados de uma rede de computadores.

**Socket** – É um ponto final de um fluxo de comunicação entre computadores é baseado no Protocolo de Internet. Trata-se do número IP e porta.

**Software** – Programa de computador.

**SSH** – Security Shell. Popular serviço de acesso remoto para sistemas Unix em interface texto.

**Switch** – Dispositivo utilizado em redes de computadores para reencaminha pacotes (frames) entre os diversos nós.

**TCP/IP** – Transfer Control Protocol/ Internet Protocol. Refere-se ao modelo para Internet, intermediação entre o transporte e a rede de acordo com o modelo OSI

**Tcpdump** – Software para análise e monitoramento de tráfego.

**Thinclient** – Cliente Magro. Cliente ‘burro’. Computador com hardware mínimo para acessar uma máquina indiretamente e/ou remotamente.

**Tshark** – Software para análise e monitoramento de tráfego.

**TTL** – Time to live. Tempo de vida máximo de um pacote de comunicação em rede de computadores.

**Tty** – é todo equipamento disponibilizado ao usuário que serve de interface com um sistema de informação. Em Linux faz referência a janela de terminal de interface texto para inserção de comandos diretos no sistema.

**Ubuntu** – Sistema operacional baseado na distribuição Debian, mantido pela empresa Canonical.

**Unix** – Sistema operacional portátil, multitarefa originalmente criado por Ken Thompson e Dennis Ritchie. Deu origem a outros sistemas como GNU/Linux, Minix, FreeBSD, OpenBSD e etc.

**Unix-like** – Sistemas operacionais baseados no Unix. Exemplo: GNU/Linux.

**Vlan** – Virtual Local Area Network. Utilizada para segmentar uma rede local por setores e/ou departamentos de uma entidade.

**Web** – Sistema hipertextual que opera através da Internet.

**Wireshark** – Software para análise e monitoramento de tráfego.