TROUBLESHOOTING DE REDES

SEMANA DE CAPACITAÇÃO - EDIÇÃO ONLINE 7

NETWORK EDUCATION

https://network.education/ treinamentos@network.education

SEMANA DE CAPACITAÇÃO
23 a 27 de Outubro de 2023
Edição *Online* 7

INSTRUTOR:

ELIZANDRO PACHECO

Profissional que atua exclusivamente no ramo de provedores regionais há mais de 21 anos.

Estudou Engenharia em Sistemas Digitais na Universidade Estadual do Rio Grande do Sul e possui diversas certificações no ramo.

É fundador da **Network Education® (http://network.education)** e palestra nos maiores eventos do ramo por todo o Brasil.

É diretor na empresa **NextHop Solutions® (http://nexthop.solutions)**, empresa especializada no ramo de consultoria para pequenos e médios provedores.

É autor do livro **Docker Para Provedores** (http://www.dockerparaprovedores.com.br) e planeja novos lançamentos na área.

Nas raras horas vagas, se dedica aos carros e a programação, sempre buscando evoluir seus conhecimentos.

Troubleshooting

Quando falamos em redes, talvez a dificuldade mais recorrente que vemos em nosso dia a dia é a falta de entendimento acerca de procedimentos de troubleshooting aliados a falta de gestão de gerência.

Não raramente redes inteiras ficam inacessíveis por descuidos com soluções simples, que por muitas vezes, de tão simples, acabam sendo negligenciadas.

E é por isso, que neste tutorial traremos dicas e rotinas simples, mas extremamente importantes para todos aqueles que administram redes.

E é claro, vamos começar pelo começo...

O ICMP

O ICMP (Internet Control Message Protocol) definido na RFC792 é basicamente um protocolo de sinalização de redes.

A utilização mais conhecida é através do comando **ping**.

Mas o ICMP vai muito além disso...

Existem diversos tipos e códigos de mensagens que podem ser bastante úteis durante os procedimentos de troubleshooting.

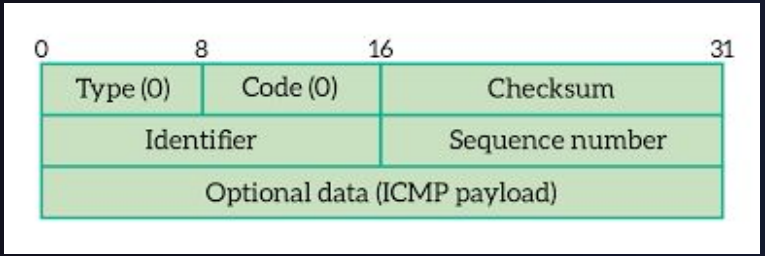
O ICMP

Conforme a própria RFC define: O Protocolo IP não foi projetado para ser **absolutamente** confiável, e o propósito do ICMP é justamente fornecer "feedback" sobre alguns problemas que podem acontecer no ambiente de comunicação.

O ICMP é humilde, ele não tem a intenção de tornar o IP confiável, deixando essa responsabilidade para os demais protocolos que o utilizam.

O ICMP

É importante saber que dentro do ICMP existem mensagens de diversos **tipos** e esses tipos, por sua vez, possuem **códigos**.



ICMP Message Types		
Type	Codes	Description
0/8	0	Echo Reply/Echo Request
3	0-15	Destination Unreachable
4	0	Source Quench
5	0-3	Redirect
9/10	0	Router Advertisement
11	0-1	Time Exceeded
12	0	Parameter Problem
13/14	0	Timestamp Request/Timestamp Reply
17/18	0	Address Mask Request/Address Mask Reply

Lista completa:
https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml#icmp-parameters-codes-0

Sniffing: Captura e Análise de Pacotes

Sniffing é a técnica de captura e análise de pacote que nos permite entender o que está acontecendo "por dentro da rede".

É fundamental que os administradores de rede conheçam a técnica e dominem as principais ferramentas do mercado.

Em geral, chamamos esse passo de "**escovar bit**", pois é um processo que faz uma imersão real em tudo que é trafegado em uma rede.

https://tcpdump.org

https://wireshark.org

Sniffing: Captura e Análise de Pacotes

Existem diversas formas de "sniffar" um ponto ou rede desejada, e essa técnica pode ser usada para realizar troubleshooting em praticamente qualquer cenário.

Dentre as mais utilizadas estão:

Sniffer Local: Você pode realizar a análise de conexões diretamente do dispositivo onde você instalou suas ferramentas. Ou ainda, gerar um arquivo com o dump das conexões que você filtrou e então transferir o arquivo para análise em qualquer outro dispositivo (muito utilizado em servidores e roteadores).

Sniffer Remoto: Existem diversas formas de você realizar análises de forma remota, até mesmo em tempo real, um bom exemplo é do próprio RouterOS!

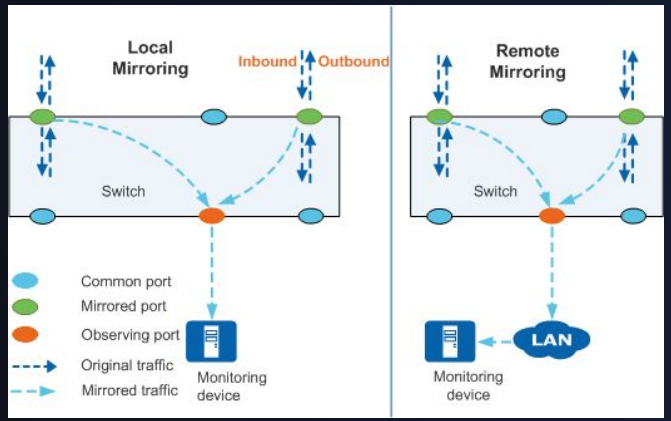
Tutorial no BPF: https://wiki.brasilpeeringforum.org/w/Como_capturar_pacotes_no_Mikrotik

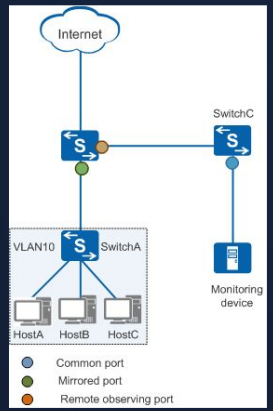
Sniffing: Captura e Análise de Pacotes

Sniffer Local ou Remoto com Espelhamento de Portas:

Em alguns fabricantes é possível realizar o espelhamento de portas e também espelhamento de tráfego, inclusive para dispositivos remotos, não necessariamente no mesmo dispositivo.

Mirroring Huawei: https://support.huawei.com/enterprise/en/doc/EDOC1100088115





Troubleshooting com ICMP

Já que agora sabemos como sniffar um enlace/interface, podemos voltar ao ICMP.

E começaremos inicialmente pelo básico, o comando Ping (Packet Internet or Inter-Network Groper).

O Ping é baseado em duas mensagens, o echo request e echo reply.

É muito comum, principalmente pro pessoal de N1, não ter clareza em como ele realmente funciona ou então fazer interpretações erradas de seus resultados.

Em geral, o ping é utilizado para medir a **latência** entre um ponto e outro de uma rede, mas ele pode te oferecer muito mais informações que isso...

Troubleshooting com ICMP

Algumas das tarefas onde o **ping** pode te ajudar:

- Medir latência entre dispositivos
- Medir estabilidade da conexão
- Validar MTU de caminhos
- Medir perda de pacotes
- Descobrir quando um pacote IP não consegue chegar ao seu destino
- Descobrir congestionamentos na rede
- Descobrir melhores rotas/caminhos
- Descobrir problemas e/ou má formação de cabeçalhos
- E muitos outros...

Mas é necessário usá-lo da forma correta...

Troubleshooting com ICMP

A primeira dica é: sempre utilize contagem de pacotes e nunca, **absolutamente nunca**, encerre o comando com control + c.

Siga a dica do Pai Ayub do BGP.

Para ter proporções exatas, utilize números de pacotes que facilitem o cálculo...

100, 1000, etc...

Ao utilizar o control + c ou interromper o processo, você pode gerar valores não confiáveis.

Em geral, o parâmetro **-c** é o parâmetro que define a quantidade de pacotes a serem enviados nos testes.

Troubleshooting com ICMP

```
~ [ ping -i 0.3 -c 1000 nytimes.com
PING nytimes.com (151.101.1.164): 56 data bytes
64 bytes from 151.101.1.164: icmp_seq=0 ttl=60 time=21.360 ms
64 bytes from 151.101.1.164: icmp_seq=1 ttl=60 time=21.267 ms
64 bytes from 151.101.1.164: icmp_seq=2 ttl=60 time=21.758 ms
64 bytes from 151.101.1.164: icmp_seq=3 ttl=60 time=21.746 ms
64 bytes from 151.101.1.164: icmp_seq=4 ttl=60 time=22.012 ms
64 bytes from 151.101.1.164: icmp_seq=5 ttl=60 time=23.669 ms
64 bytes from 151.101.1.164: icmp_seq=6 ttl=60 time=21.828 ms
64 bytes from 151.101.1.164: icmp_seq=7 ttl=60 time=24.219 ms
64 bytes from 151.101.1.164: icmp_seq=8 ttl=60 time=23.385 ms
64 bytes from 151.101.1.164: icmp_seq=9 ttl=60 time=21.668 ms
64 bytes from 151.101.1.164: icmp_seq=10 ttl=60 time=23.688 ms
```

```
--- nytimes.com ping statistics ---
100 packets transmitted, 100 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 21.105/23.174/28.645/1.418 ms
```

Troubleshooting com ICMP

É importante saber que alguns dispositivos podem fazer **rate-limit** de pacotes icmp, então tenha como base essa "perda" apenas como um norte, não acredite fielmente nele.

Preste atenção também no stddev (desvio padrão).

O stdev é um cálculo realizado a partir dos tempos de resposta a fim de lhe oferecer informações sobre o **grau de estabilidade** entre os hosts.

$$STDEV(x_n) = \sqrt{\frac{n\sum x^2 - (\sum x)^2}{n(n-1)}}$$

Quanto menor, mais estável.

(Os gamers agradecem)

Troubleshooting com ICMP

Validação de MTU com o ping:

O MTU (Maximum transmission unit) é o tamanho máximo que um quadro ou pacote pode ser transmitido através de um link de dados.

Antes de entender o mtu, é necessário entender a fragmentação.

A fragmentação de IP é um procedimento cuja função é efetuar a segmentação dos pacotes em partes que obrigatoriamente necessitam ter tamanho igual ou menor ao limite do caminho, possibilitando executar a sua transferência por um link de transmissão que possua um **MTU menor** do que o pacote original.

Vídeo Recomendado: https://www.youtube.com/watch?v=5OtebbSnwoM

Troubleshooting com ICMP

Não raramente, vemos problemas em BNGs quando usuários utilizam PPPoE.

O sintoma mais "superficial" que o cliente sente é:

- Algumas coisas "abrem" outras não.

Seja para o cliente final, ou até mesmo dentro do seu backhaul/backbone, sempre valide o MTU, e use-o no máximo possível (as cpus agradecem).

O Throughput está diretamente ligado ao PPS (Pacotes por Segundo), quanto maiores os pacotes, menor quantidade de pps... consequentemente maior **Throughput**.

Troubleshooting com ICMP

Para validar um caminho você deve utilizar o **ping** aumentando o tamanho do pacote e setando a flag DF (Don't Fragment), indicando para o dispositivo que ele deve enviar aquele pacote "sem fragmentar".

Assim, você consegue descobrir a capacidade máxima de transmissão de um caminho.

A opção de DF pode variar de acordo com o sistema operacional.

Consulte o help do comando.

Troubleshooting com ICMP

Traceroute:

Comando que rastreia a rota que os pacotes IP seguem indo para um "host".

Muito utilizado para "descobrir" um caminho e até "medir perdas" nesse caminho, é útil em muitas situações do dia a dia de um administrador de redes.

Mas, assim como o ping, é importante saber interpretar seus resultados e entender como ele funciona.

Antes disso, é importante entender o **TTL (Time to Live)**.

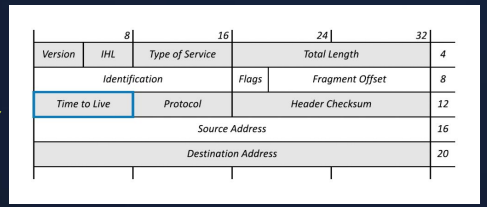
Troubleshooting com ICMP

TTL:

Quando um pacote de informações é criado e enviado pela internet, **há o risco de que ele continue a passar de um roteador para outro indefinidamente**. Para mitigar essa possibilidade, os pacotes são desenvolvidos com um prazo de validade chamado tempo de vida ou limite de salto. O TTL também pode ser útil para determinar há quanto tempo um pacote está em circulação e permitir que o remetente receba informações sobre o caminho percorrido por um pacote na internet.

Cada pacote tem um local no qual armazena um valor numérico que determina quanto tempo mais ele deve continuar a se movimentar pela rede. A cada vez que um roteador recebe um pacote, ele subtrai "um" da contagem do TTL e depois o transmite para o próximo local na rede. Se em um determinado ponto a contagem do **TTL for igual a zero** depois da subtração, o roteador descartará o pacote e **enviará uma mensagem ICMP de volta ao host de origem**.

Fonte: https://www.cloudflare.com/pt-br/learning/cdn/glossary/time-to-live-ttl/



Troubleshooting com ICMP

Voltando ao traceroute...

O comando traceroute tenta rastrear a rota que um pacote IP segue para um host, enviando os pacotes de análise UDP com um ttl pequeno (e incremental).

Assim, quando o TTL estoura, ele recebe uma mensagem ICMP TIME_EXCEEDED.

O TTL é incrementado até que uma mensagem ICMP PORT_UNREACHABLE seja retornada.

A mensagem ICMP PORT_UNREACHABLE indica que o host foi localizado ou que o comando atingiu o número máximo de saltos permitidos para o rastreio.

```
345             conf->first_ttl = 1;
346             conf->proto = IPPROTO_UDP;
347             conf->max_ttl = IPDEFTTL;
348             conf->nprobes = 3;
349             conf->expected_responses = 2; /* icmp + DNS */
350
351             /* start udp dest port # for probe packets */
352             conf->port = 32768+666;
353
354             memset(&rcvmhdr, 0, sizeof(rcvmhdr));
355             memset(&rcviov, 0, sizeof(rcviov));
```

Troubleshooting com ICMP

Ainda sobre traceroute...

Os pacotes que são chamados de "sondas" são enviados com destino à portas UDP que começam em 33434 e são incrementadas uma a uma.

Se o destino tiver qualquer bloqueio para essas portas, o traceroute não obterá resposta, mas isso não significa que o host está inacessível.

Você pode tentar outro conjunto de portas no comando (o parâmetro varia de acordo com cada sistema operacional).

Ou ainda... apelar para o bom e velho ICMP.

Source	Destination	Protoco	Lengt	Info
100.64.64.3	151.101.1.164	UDP	74	38289 → 33434 Len=32
100.64.64.3	151.101.1.164	UDP	74	53026 → 33435 Len=32
100.64.64.3	151.101.1.164	UDP	74	54223 → 33436 Len=32
100.64.64.3	151.101.1.164	UDP	74	59363 → 33437 Len=32

Troubleshooting com ICMP

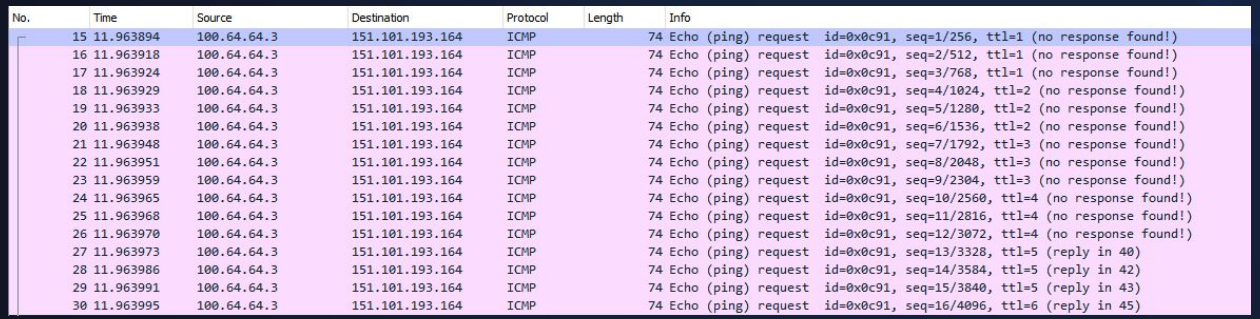
Ainda sobre traceroute...

Quando você não obtiver respostas com o conjunto de portas padrão do traceroute você tem 2 alternativas...

1 - Alterar, através de parâmetro do comando, a porta inicial.
2 - Utilizar o ICMP ao invés do UDP (geralmente parâmetro -I)

Faça o teste:

traceroute nytimes.com
e depois
traceroute -I nyimes.com



No.	Time	Source	Destination	Protocol	Length	Info
15	11.963894	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=1/256, ttl=1 (no response found!)
16	11.963918	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=2/512, ttl=1 (no response found!)
17	11.963924	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=3/768, ttl=1 (no response found!)
18	11.963929	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=4/1024, ttl=2 (no response found!)
19	11.963933	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=5/1280, ttl=2 (no response found!)
20	11.963938	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=6/1536, ttl=2 (no response found!)
21	11.963948	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=7/1792, ttl=3 (no response found!)
22	11.963951	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=8/2048, ttl=3 (no response found!)
23	11.963959	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=9/2304, ttl=3 (no response found!)
24	11.963965	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=10/2560, ttl=4 (no response found!)
25	11.963968	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=11/2816, ttl=4 (no response found!)
26	11.963970	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=12/3072, ttl=4 (no response found!)
27	11.963973	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=13/3328, ttl=5 (reply in 40)
28	11.963986	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=14/3584, ttl=5 (reply in 42)
29	11.963991	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=15/3840, ttl=5 (reply in 43)
30	11.963995	100.64.64.3	151.101.193.164	ICMP	74	Echo (ping) request id=0x0c91, seq=16/4096, ttl=6 (reply in 45)

Troubleshooting com ICMP

Bugs:

Há alguns bugs conhecidos, então atente-se a eles para não errar a interpretação.

Um deles é o encaminhamento de pacotes mesmo quando o TTL é 0.

```
*     [yak 71]% traceroute nis.nsf.net.
*     traceroute to nis.nsf.net (35.1.1.48), 30 hops max, 56 byte packet
*      1  helios.ee.lbl.gov (128.3.112.1)  19 ms  19 ms  0 ms
*      2  lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  39 ms  19 ms
*      3  lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  39 ms  19 ms
*      4  ccngw-ner-cc.Berkeley.EDU (128.32.136.23)  39 ms  40 ms  39 ms
*      5  ccn-nerif22.Berkeley.EDU (128.32.168.22)  39 ms  39 ms  39 ms
*      6  128.32.197.4 (128.32.197.4)  40 ms  59 ms  59 ms
*      7  131.119.2.5 (131.119.2.5)  59 ms  59 ms  59 ms
*      8  129.140.70.13 (129.140.70.13)  99 ms  99 ms  80 ms
*      9  129.140.71.6 (129.140.71.6)  139 ms  239 ms  319 ms
*     10  129.140.81.7 (129.140.81.7)  220 ms  199 ms  199 ms
*     11  nic.merit.edu (35.1.1.48)  239 ms  239 ms  239 ms
```

Fonte (Literalmente): https://github.com/openbsd/src/blob/master/usr.sbin/traceroute/traceroute.c

Troubleshooting com ICMP

Portanto, o **traceroute** é uma ótima ferramenta que nos auxilia na hora de troubleshootings, mas em geral vemos análises equivocadas ou erros de interpretação dos resultados.

É importante ter esse conhecimento mais aprofundado para que você consiga identificar problemas e "caminhos a seguir".

Principalmente em cenários multi-homed, afinal de contas antes de encontrar o problema é necessário saber em qual caminho procurar.

E é justamente esse o papel do traceroute.

Troubleshooting com ICMP

MTR:

Outra ferramenta muito importante na hora do troubleshooting é o MTR. Ele combina as funcionalidades do **ping** e do **traceroute**.

GitHub: https://github.com/traviscross/mtr

A grande vantagem do MTR é que ele é sequencial, repete os testes por intervalos, facilitando a visualização.

As combinações de possibilidades que o MTR oferece o torna uma das ferramentas mais poderosas e produtivas no troubleshooting.

Troubleshooting com ICMP

MTR:

Por padrão, ao contrário do traceroute, o mtr utiliza o ICMP (por padrão).

Mas não se preocupe, você pode selecionar a forma de "sondar" o caminho todo através de parâmetros opcionais, tanto em TCP quanto UDP.

Dentre as opções que considero mais útil estão:

- Extensões mpls
- Resolução de AS do salto
- Possibilidade de definição de porta de destino
- Reports em diversos formatos (json, xml, csv ...)


```
{
    "count": 11,
    "host": "200-147-3-157-157.static.uol.com.br",
    "ASN": "AS7162",
    "Loss%": 0.0,
    "Snt": 5,
    "Last": 4.569,
    "Avg": 4.781,
    "Best": 4.569,
    "Wrst": 4.895,
    "StDev": 0.133
}
```


Utilize o conhecimento adquirido até aqui para explorá-lo.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Há muito o que se discutir em relação ao DNS, e também, muitas vezes, encontramos situações onde o troubleshooting é equivocado, mas antes de entrarmos em detalhes precisamos começar pelo começo:

NENHUM DNS EXTERNO A SUA REDE SERÁ MELHOR QUE UM DNS PRÓPRIO, DENTRO DELA.

Nota mental: A menos que você faça uma besteira MUITO grande.

Não, nem o do Google, nem o da CloudFlare, nem o OpenDNS e nem outro qualquer, fora da sua rede, pode te oferecer maior segurança, desempenho e confiabilidade que seu próprio DNS.

Não se iluda.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Os erros mais comuns são:

- NAT em direção ao DNS
- Erros na metodologia de teste
- Erros de provisionamento (DNSSEC, Root Servers, Cache, etc...)

E exploraremos alguns deles...

Mas para isso, precisamos avançar no quesito **sniffing**.

Troubleshooting de DNS

Voltando aos sniffers...

Voltamos a nossa dupla preferida:

Mas desta vez para realizar **filtros**.

Troubleshooting de DNS

Filtros de Captura:

Em geral, e quanto maior a banda do enlace e/ou interface que estamos sniffando, é praticamente impossível realizar análises sem a utilização de filtros.

Algumas vezes é humanamente impossível fazer uma leitura em tempo real sem eles.

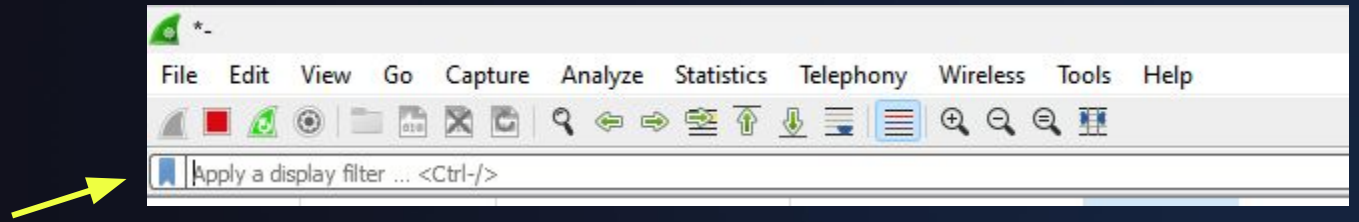
Tanto o wireshark quanto o tcpdump possuem um poderoso sistema de filtragem que torna nosso trabalho mais "fácil".

https://wiki.wireshark.org/CaptureFilters

Troubleshooting de DNS

Filtros de Captura:

Os filtros podem ser dos mais simples ao mais complexo, incluindo (inclusive) encadeamento de filtros. **É importante estudar sobre eles**.



TCPDUMP & LIBPCAP

WIRESHARK

Troubleshooting de DNS

Exemplos de Filtros de Captura:

Os filtros podem ser dos mais simples ao mais complexo, incluindo (inclusive) encadeamento de filtros. **É importante estudar sobre eles**.

Capturar tráfego de apenas um host:

```
ip.addr == 100.64.64.1
```

Capturar tráfego de apenas uma rede:

```
ip.addr == 100.64.64.0/24
```

Capturar tráfego de apenas uma origem:

```
ip.src == 100.64.64.0/24
```

Capturar tráfego de apenas uma rede de destino:

```
ip.dst == 100.64.64.0/24
```

Troubleshooting de DNS

Exemplos de Filtros de Captura:

Capturar tráfego de apenas uma porta:

```
tcp.port == 80
udp.port == 53
```

Possui também a opção de srcport ou dstport

Capturar tráfego de apenas um host e uma porta:

```
ip.addr == 100.64.64.1 && tcp.port == 80
```

E muitos outros... uma busca rápida no google lhe traz muitas combinações, inclusive a documentação oficial:
https://www.wireshark.org/docs/wsug_html_chunked/ChWorkBuildDisplayFilterSection.html

Troubleshooting de DNS

TCPDUMP:

Muitas vezes não temos, de forma rápida, como executar um sniffer em determinado enlace, mas temos como utilizar o dispositivo final para realizar o mesmo procedimento.

Com o tcpdump você pode, tanto ver o tráfego em tempo real quanto **gerar um arquivo para análise posterior no wireshark**.

Troubleshooting de DNS

Exemplos de Filtros de Captura:

Assim como no wireshark, você pode facilmente encontrar exemplos de filtragem. Eles também permitem encadeamento.

Capturar tráfego de apenas um host:
```
tcpdump -i eth0 host 100.64.64.1
```

Capturar tráfego de apenas uma rede:
```
tcpdump -i eth0 net 100.64.64.0/24
```

Capturar tráfego de apenas uma origem:
```
tcpdump -i eth0 src host 100.64.64.0/24
```

Capturar tráfego de apenas uma rede de destino:
```
tcpdump -i eth0 dst net 100.64.64.0/24
```

Troubleshooting de DNS

Exemplos de Filtros de Captura:

Capturar tráfego de apenas uma porta:

```
tcpdump -i eth0 port 53
```

Lembre-se que você pode encadear filtros:

```
tcpdump -i eth0 dst 8.8.8.8 and icmp
```

Vamos então filtrar pacotes de um DNS...

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Conforme mencionado anteriormente os erros mais comuns são:

- NAT em direção ao DNS

Não raramente encontramos cenários onde a origem é dentro da rede e o servidor dns também encontra-se dentro da rede, portanto alcançável via roteamento, JAMAIS use nat em direção ao servidor.

E ainda que tenha exceção no firewall, se for CGNAT prefira exceção no PBR, não deixe esse tráfego acessar sua caixa de CGNAT.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Conforme mencionado anteriormente os erros mais comuns são:

- NAT em direção ao DNS

Não raramente encontramos cenários onde a origem é dentro da rede e o servidor dns também encontra-se dentro da rede, portanto alcançável via roteamento, JAMAIS use nat em direção ao servidor.

E ainda que tenha exceção no firewall, se for CGNAT prefira exceção no PBR, não deixe esse tráfego acessar sua caixa de CGNAT.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Outro erro bastante comum é utilizar o **ping** para medir a qualidade de um DNS.

Esse procedimento até pode te dar um indicativo de velocidade até o IP do servidor, mas para saber qual é o mais rápido você precisa se atentar a outros parâmetros, dentre eles o **Query Time**, que pode ser obtido através do comando dig.

Esse comando faz parte do pacote **dnsutils** e é bastante completo.

Sua utilização básica é:

```
dig dominio.com.br @<ip do dns>

Ex: dig network.education @8.8.8.8
```

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

O Query Time é o resultado do tempo de envio e recebimento da resposta, ele não tem haver com a latência.

No próprio código fonte do comando é possível notar isso:

```
debug("sending a request");
TIME_NOW(&query->time_sent);
INSIST(query->sock != NULL);
```

```
if (query->lookup->stats && !short_form) {
        diff = isc_time_microdiff(&query->time_recv, &query->time_sent);
        if (use_usec)
                printf(";; Query time: %ld usec\n", (long) diff);
        else
                printf(";; Query time: %ld msec\n", (long) diff / 1000);
```

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

E justamente por isso, pode acontecer de o servidor que tem uma latência mais alta, ter uma resposta mais rápida.

Ou o servidor que tem latência melhor, ter uma resposta mais lenta.

E é por isso que você não deve levar em consideração apenas a latência para determinar se um servidor DNS é melhor que outro.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Se você não especificar um dns no comando dig, ele utilizará o configurado no sistema. Mas, mesmo sem alterar a configuração do dispositivo, você pode testar qualquer DNS recursivo aberto.

Exemplo:

```
dig network.education @8.8.8.8

dig network.education @1.1.1.1
```

Mas não acaba aí... o dig possui diversas outras funcionalidades.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Determinado então o melhor DNS para sua rede, você certamente passará por situações onde o cliente não consegue abrir determinado site.

E claro, que isso é genérico demais e pode estar ligado a diversos outros motivos, mas o DNS não pode ser descartado...

Com o comando dig você obterá, dentre outras informações, o status da consulta:

```
; <<>> DiG 9.10.6 <<>> uol.com.br
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46349
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
```

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

DNS Response Code	Description
NOERROR	DNS query successfully completed
NXDOMAIN	DNS query failed because the domain name queried does not exist
SERVFAIL	DNS query failed because an answer cannot be given
REFUSED	DNS query failed because the server refused to answer due to policy

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Nem sempre apenas o código é suficiente... muitas vezes você terá de ir além.

Ex: Com o servidor X resolve, com o Y não.

Muitas vezes um servidor de DNS não consegue resolver um endereço por um problema de alcance de roteamento que você ainda não identificou ou ainda por algum erro em servidores a frente do seu (inclusive o autoritativo do próprio domínio).

Nessas horas a opção +trace do dig pode te ajudar.

Com a opção +trace o dig irá utilizar traçar todo o caminho fazendo as requisições de acordo com a hierarquia.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Source	Destination	Protocol	Length	Info
100.64.64.3	8.8.8.8	DNS	82	Standard query 0x7518 NS <Root> OPT
8.8.8.8	100.64.64.3	DNS	567	Standard query response 0x7518 NS <Root> NS i.root-servers.net NS a.root-servers.net NS j.root-servers.net NS
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x27f5 A i.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x61fd AAAA i.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x27f5 A i.root-servers.net A 192.36.148.17
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x61fd AAAA i.root-servers.net AAAA 2001:7fe::53
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x5603 A a.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0xa505 AAAA a.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x5603 A a.root-servers.net A 198.41.0.4
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0xa505 AAAA a.root-servers.net AAAA 2001:503:ba3e::2:30
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x8b65 A j.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x646e AAAA j.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x8b65 A j.root-servers.net A 192.58.128.30
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x646e AAAA j.root-servers.net AAAA 2001:503:c27::2:30
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x1a83 A d.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x1b98 AAAA d.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x1a83 A d.root-servers.net A 199.7.91.13
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x1b98 AAAA d.root-servers.net AAAA 2001:500:2d::d
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x9e2e A e.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x4d22 AAAA e.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x9e2e A e.root-servers.net A 192.203.230.10
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x4d22 AAAA e.root-servers.net AAAA 2001:500:a8::e
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x93e1 A f.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x93ea AAAA f.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x93e1 A f.root-servers.net A 192.5.5.241
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x93ea AAAA f.root-servers.net AAAA 2001:500:2f::f
100.64.64.3	8.8.8.8	DNS	78	Standard query 0xe485 A g.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x9378 AAAA g.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0xe485 A g.root-servers.net A 192.112.36.4
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x9378 AAAA g.root-servers.net AAAA 2001:500:12::d0d
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x55a2 A l.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0xf1a6 AAAA l.root-servers.net
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x55a2 A l.root-servers.net A 199.7.83.42
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0xf1a6 AAAA l.root-servers.net AAAA 2001:500:9f::42
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x0a71 A h.root-servers.net
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x8b7d AAAA h.root-servers.net
8.8.8.8	100.64.64.3	DNS	106	Standard query response 0x8b7d AAAA h.root-servers.net AAAA 2001:500:1::53
8.8.8.8	100.64.64.3	DNS	94	Standard query response 0x0a71 A h.root-servers.net A 198.97.190.53
100.64.64.3	8.8.8.8	DNS	78	Standard query 0x5d42 A b.root-servers.net

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

E é justamente aqui, que você pode não ter alcance a algum servidor, por um problema de roteamento ou outro problema qualquer.

Você ainda poderá descobrir em que "salto" ou qual servidor apresentou um problema, validando todo o caminho.

Atente-se: Um recursivo não responder, ou retornar uma falha, não significa necessariamente que o problema está nele.

Troubleshooting de DNS

DNS: O Herói (ou vilão) da rede.

Existem diversas ferramentas que podem lhe auxiliar no troubleshooting de DNS.

Do lado servidor, o dnstop é uma delas. Com ela é possível fazer uma análise rápida sobre as requisições que chegam no servidor e identificar anomalias rapidamente.

```
usage: dnstop [opts] netdevice|savefile
        -4      Count IPv4 packets
        -6      Count IPv6 packets
        -Q      Count queries
        -R      Count responses
        -a      Anonymize IP Addrs
        -b expr BPF program code
        -i addr Ignore this source IP address
        -n name Count only messages in this domain
        -p      Don't put interface in promiscuous mode
        -P      Print "progress" messages in non-interactive mode
        -r      Redraw interval, in seconds
        -l N    Enable domain stats up to N components
        -X      Don't tabulate the "source + query name" stats
        -f      filter-name
```

Recomendação de Leitura:
https://www.cyberciti.biz/faq/dnstop-monitor-bind-dns-server-dns-network-traffic-from-a-shell-prompt/

E o IPv6?

Já não é novidade que quem possui uma rede apenas IPv4 está no passado e possui uma rede legada.

Todos os procedimentos de troubleshooting apresentados aqui servem para ambos protocolos com pequenas alterações.

Um ponto especial de atenção não IPv6 é o ICMP.

Apenas alguns tipos de mensagens icmp podem ser bloqueadas no IPv6.

JAMAIS BLOQUEIE TODO ICMPv6 NA REDE, ELA VAI PARAR!

E o IPv6?

Comandos para IPv6:

Em alguns sistemas operacionais o comando **ping** é o mesmo! Em outros, há o comando **ping6** assim como o **traceroute6**.

Em algumas versões você pode forçar e/ou escolher o protocolo desejado, como no caso do **mtr**...

Basta você especificar o protocolo desejado com: -4 ou -6

Ex: `mtr -6 google.com`

OBRIGADO!

Agradecemos, mais uma vez, a oportunidade de novamente poder contribuir um pouco para uma internet melhor!

Que este seja apenas mais um passo na jornada de evolução de todos.

Em especial a toda equipe do Nic.BR e da NextHop Solutions !

Elizandro Pacheco
+55 51 99871-8111
@elizandropacheco