Traceroute

UPE/POLI/Ecomp

Semestre 2021.1

Equipe

André Filipe Menezes

Elias Amaro

Guilherme Novaes

Jean Felipe

Introdução ao traceroute

O traceroute é uma ferramenta de diagnóstico que permite ver a rota que os datagramas IP seguem quando são enviados de um host a outro. O traceroute faz uso do protocolo ICMP e do campo TTL no cabeçalho IP do datagrama. O campo TTL (Time to Live) é um campo de 8 bits que o dispositivo de origem inicializa com um valor específico.

O propósito do TTL é evitar que datagramas entrem em um loop de roteamento, o que pode ocorrer devido a algum tipo de falha durante o roteamento dos pacotes. Quando um roteador recebe um datagrama cujo TTL é igual a 0 (zero), ele não o encaminhará mais. Em vez disso, o roteador irá descartar o pacote e enviar de volta ao host que o originou uma mensagem ICMP do tipo Tempo Excedido. Essa mensagem contém o endereço IP do roteador como endereço de origem – e esse é o segredo do traceroute.

Funcionamento do traceroute

O traceroute envia um datagrama com um TTL igual a 1 ao host de destino. Ao chegar ao primeiro roteador no caminho, o TTL é decrementado, ficando com o valor zero, de modo que o datagrama é descartado e a mensagem ICMP “Tempo Excedido” é enviada de volta à origem. Desta forma, o primeiro roteador no caminho é identificado. Então, o traceroute envia um novo datagrama, desta vez com o TTL igual a 2, que passará pelo primeiro roteador, que decrementar o TTL para 1, e então será descartado no segundo roteador – e assim, descobrimos o endereço IP deste roteador também. Esse processo continua até que um datagrama chegue ao host de destino.

Porém, ao chegar no host de destino, mesmo que o TTL do datagrama seja igual a 1, ele não será descartado e, portanto, a mensagem ICMP Tempo Excedido não será gerada. Neste caso, como sabemos que o pacote chegou ao seu destino? Isso vai depender do sistema operacional utilizado (mais precisamente, do utilitário presente); No Unix e derivados, o traceroute envia pacotes UDP, escolhendo portas de um valor elevado, que muito provavelmente não são utilizadas por nenhuma aplicação no host de destino. Desta forma, o host de destino irá gerar um erro ICMP do tipo “Porta Inalcançável” – e, então, tudo o que o traceroute tem a fazer é diferenciar o recebimento de mensagens Tempo Excedido da mensagem Porta Inalcançável para saber quando parar.

Já no Windows, são enviados datagramas ICMP do tipo Echo Request – o famoso ping – e quando o dispositivo de origem recebe uma resposta Echo Reply, ele sabe que o pacote chegou ao seu destino.

```
C:\Users\boson>tracert www.bosontreinamentos.com.br

Rastreando a rota para bosontreinamentos.com.br [186.202.153.82]
com no máximo 30 saltos:

  1     2 ms     1 ms     1 ms  192.168.1.1
  2     *       35 ms    27 ms  b18c5c01.virtua.com.br [177.140.92.1]
  3    11 ms    11 ms    12 ms  c9062201.virtua.com.br [201.6.34.1]
  4    11 ms    12 ms    14 ms  c9062a45.virtua.com.br [201.6.42.69]
  5    14 ms    14 ms    16 ms  c9062a41.virtua.com.br [201.6.42.65]
  6    12 ms    15 ms    11 ms  200.160.197.209
  7    15 ms    15 ms    11 ms  200.160.197.138
  8    17 ms    16 ms    14 ms  200.160.195.158
  9    12 ms    22 ms    22 ms  186.202.158.6
 10    14 ms    15 ms    15 ms  dist-aita20-b.locaweb.com.br [186.202.191.89]
 11    15 ms    18 ms    30 ms  186.202.158.126
 12    19 ms    22 ms    20 ms  hm7075.locaweb.com.br [186.202.153.82]

Rastreamento concluído.
```

No Windows usamos a palavra “**tracert**” em vez de “traceroute” para executar o utilitário. São mostrados os endereços dos roteadores por onde os pacotes passaram até chegar ao destino especificado. A primeira coluna da saída mostrada contém o número do salto, e as três colunas seguintes mostram os **RTT**s (Round Trip Times) das mensagens enviadas – o traceroute envia três mensagens por padrão para cada salto.

Código Fonte

```python
import sys
import socket
import random
from datetime import datetime
tempos = []

def traceroute(endereco_destino, pulos_maximos=30, tempo_limite=0.5):
    proto_icmp = socket.getprotobyname('icmp')
    proto_udp = socket.getprotobyname('udp')
    porta = random.choice(range(33434, 33535))
```

```
    for ttl in range(1, pulos_maximos+1):

        inicioTime = datetime.now()
        receiver = socket.socket(socket.AF_INET, socket.SOCK_RAW, proto_icmp)
        receiver.settimeout(tempo_limite)
        receiver.bind(('', porta))
        sender = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, proto_udp)
        sender.setsockopt(socket.SOL_IP, socket.IP_TTL, ttl)
        sender.sendto(b'', (endereco_destino, porta))

        try:
            dados, endereco_atual = receiver.recvfrom(512)
            endereco_atual = endereco_atual[0]

        except socket.error:
            endereco_atual = None

        finally:
            tempoTotal = datetime.now() - inicioTime
            tempos.append(tempoTotal)
            receiver.close()
            sender.close()

        yield endereco_atual

        if endereco_atual == endereco_destino:
            break
```

```
if __name__ == "__main__":

    url_destino = sys.argv[1]
    endereco_destino = socket.gethostbyname(url_destino)
    print("Traceroute para %s (%s)" % (url_destino, endereco_destino))
    print('\nhop\taddr\t\ttotal_time')

    for i, v in enumerate(traceroute(endereco_destino)):

        if v is None:
            print(str(i+1) + '\tEsgotado o tempo limite do pedido.')
        else:
            print(str(i+1) + "\t" + str(v) + "\t" + str(tempos[i]).split('.')[1] + ' ms')
```