**Analýza správania sa TCP a UDP protokolu pri stave zahltenia a ako vie pomôcť frontová stratégia WFQ**

**Topológia**

Zapoj a nakonfiguruj sieť podľa schémy, over konektivitu koniec-koniec:

**R1**

**R2**

**10.0.1.0 / 24**

**10.0.2.0 / 24**

**10.0.12.0 / 24**

**128 k**

**clock rate**

**bandwidth**

**Sender**

**Reciever, WS**

**S0**

**S0**

**f0**

**f0**

**Požiadavky**

* Pri výbere smerovačov nevyberajte radu 2600 (kvôli nepodpore niektorých QoS mechanizmov)
* DCE kábel pripojte do "T" interfejsu (2-port serial WIC - WIC-2T), druhý koniec hoci do “A/S” interfejsu (2-port asynchronous/synchronous WIC (WIC-2A/S)) – tým získate prepoj s rýchlosťou až 8 Mbps, inak by sme mali linku len niekoľko kbps
* obmedzte rýchlosť sériovej linky na 128 kbps (aby sme nemuseli generovať veľkú záťaž)
* upravte rýchlosť linky aj v parametri „bandwidth“ na 128 kbps (pozor, údaj v BW sa zadáva v kbps a clockrate v bps)
* nastavte statické smerovanie
  + nepokúšajte sa o dynamické smerovanie, lebo pri zahltení linky vám nebudú prechádzať ani smerovacie informácie, takže záznamy z tabuľky vám môžu aj vypadnúť, a konektivitu v topo úplne stratíte
* nastavte heslo pre vzdialené prihlasovanie sa na smerovač (telnet, privilege level 15)
* Na ploche máte odkaz na generátor D-ITG a jeho grafickú nadstavbu ITG-GUI (rada pre domáce použitie: sitahnite si D-ITG a ITG-GUI, v ceste k priečinku s D-ITG nesmie byť medzera)

Overenie konektivity:

* ping medzi PC1 a PC2 (vypnite Firewall vo Win10 – blokuje ICMP pakety)
* Telnet z PC1 na R2 ide

**Scenár a ciele**(Info čo sa ide diať, konkrétne zadanie je nižšie v Úloha 1, 2, 3, 4)

* Naučíme sa pracovať s D-ITG generátorom prevádzky
* Zahltíme linku tokom prevádzky generovaným v D-ITG:
  + UDP/TCP konšt. tok paketov, trvanie 3 min, pakety konšt. veľkosti 512 B
    - Spraviť 2 scenáre – pre každú úlohu:
      * Tok menší ako kapacita linky, 21 pak/s
      * Tok väčší ako kapacita linky, 42 pak/s
* Budeme sledovať správanie sa UDP a TCP protokolu pri zahltení a ich vplyv na iné aplikácie (TELNET)
* Odsledujeme fungovanie queueing mechanizmov FIFO a WFQ
* Toky odchytíme a zobrazíme vo Wiresharku (IO Graphs) a preskúmame
* Oneskorenie a straty si zobrazíme pomocou Loggera v ITG GUI

**Poznámky k meraniam, Wiresharku a DITG**

* Pre spustenie generovania tokov (Sender) a nastavenie príjemcu (Receiver) v D-ITG máte ukážku na konci tohto zadania
* Pre meranie oneskorenia je potrebná synchronizácia času – napr. NTP (alebo PTP – precise time protocol)
  + K dispozícii na Moodle: *Návod pre synchronizáciu času pomocou NTP.docx*
* V úlohe 4 je vhodné si vo Wiresharku zobraziť extra stĺpec do zobrazenia – DSCP hodnotu
  + K dispozícii na Moodle: *Návod pre pridanie stĺpca DSCP vo Wiresharku.docx*
* Pre efektívne zobrazovanie IO grafov vo Wiresharku odporúčame použiť instrukcie:
  + K dispozícii na Moodle: *Návod pre zobrazovanie IO grafov vo Wiresharku.docx*
* Prioritne používajte DITG a Wireshark v Linuxe, virtualbox nám merania skresľuje
* Pre zobrazovanie informácií o oneskorení a stratách v ITG GUI treba využiť Logger v ITG GUI, ako ho nastaviť nájdete tu:
  + K dispozícii na Moodle: *Návod pre nastavenie logovania v ITG GUI*.*docx*
* Poznámka ku generovaniu TCP segmentov:
  + pri zadaní konštantnej veľkosti TCP segmentov v DITG generátore, nemôžeme čakať, že aj vo Wiresharku uvidíme segment takýchto veľkostí. Je to dané tým, že TCP je tokovo orientovaný protocol, čiže nijako nám nezaručí že segmenty budú nejakej konštantnej veľkosti. TCP payload (telo bez hlavy) za sebou nasledujúcich paketov TCP spojí (uloží si do nejakého buffra) a potom si z toho vytvára TCP segmenty (čiže si ich následne rozkúskuje) veľkosti s ohľadom na hodnotu MSS (maximum segment size) – 1500B.

**Úloha 1:** Na sériovom rozhraní overte že je FIFO mechanizmus (show interfaces). Ak nie je FIFO, nastavte ho cez policy map – info je nižšie v úlohe 3. Zahlťte linku TCP prevádzkou, sledujte fungovanie pripojenia cez TELNET z PC1 na R2 a zobrazte si tok(y) v programe Wireshark. (malo by fungovať ok, TCP je ohľaduplné, a spomalí vysielanie keď je zahltený front). Pokúste sa pomocou D-ITG vyčítať aj straty a oneskorenie (na prijímacej strane, pozor pre meranie oneskorenia je potrebná synchronizácia času).

* Wireshark – IO graphs,
* D-ITG – info o stratách a oneskorení cez Logger.

**Úloha 2:** Na sériovom rozhraní chceme stále FIFO. Zahlťte linku UDP prevádzkou, sledujte fungovanie pripojenia cez TELNET z PC1 na R2. (telnet nepôjde, UDP je agresívne a zahltí mi linku, cez ktorú už nepretlačím nič iné).

* Wireshark – IO graphs,
* D-ITG – info o stratách a oneskorení cez Logger.

**Úloha 3:** Pri každom variante sa snažte o postupný nástup tokov, t.j. generujte každý ďalší tok (z multiflow) pre každý variant (1-4 nižšie) s istým aspoň 10 sekundovým oneskorením, aby bolo vidieť efekt na výsledky. T.j. prvý tok z multiflow začnite generovať napr. v 10.sekunde (po spustení D-ITG), druhý v 20. sekunde, a pod. Výsledky sledujte tieto:

* Wireshark – IO graphs,
* D-ITG – info o stratách a oneskorení cez Logger.

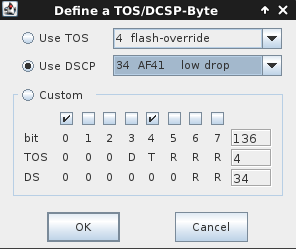
Zrealizuj tieto varianty:

1. Multiflow: 2xUDP, FIFO na rozhraní
   1. V súčte pod kapacitu linky
   2. V súčte nad kapacitu linky
2. Multiflow: 2xTCP, FIFO na rozhraní
   1. V súčte pod kapacitu linky
   2. V súčte nad kapacitu linky
3. Multiflow: 1xUDP, 1xTCP, v súčte aby presahovali kapacitu linky, FIFO na rozhraní
4. Multiflow: 1xUDP, 1xTCP, v súčte aby presahovali kapacitu linky, aktivujte WFQ na rozhraní:

* Pre smerovače s IOS ver. 15 použite nové MQC (modular QoS CLI) príkazy:
  + Router(config)# **policy-map** politika1  
    Router(config-pmap)# **class class-default**  
    Router(config-pmap-c)# **fair-queue** ! vypne FIFO a zapne WFQ  
    Router(config)# int s0  
    Router(config-if)# **service-policy** output politika1

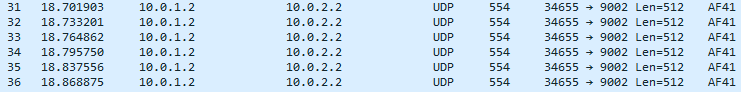
**Úloha 4:** Pokúste sa v D-ITG generovať označkované pakety v DSCP poli (EF a aspoň 3 rôzne AF), odchyťte ich na prijímajúcej strane a analyzujte cez Wireshark jednotlivé polia hlavičky.

* Nastavenie v D-ITG:



* Ukážka vo WireSharku:





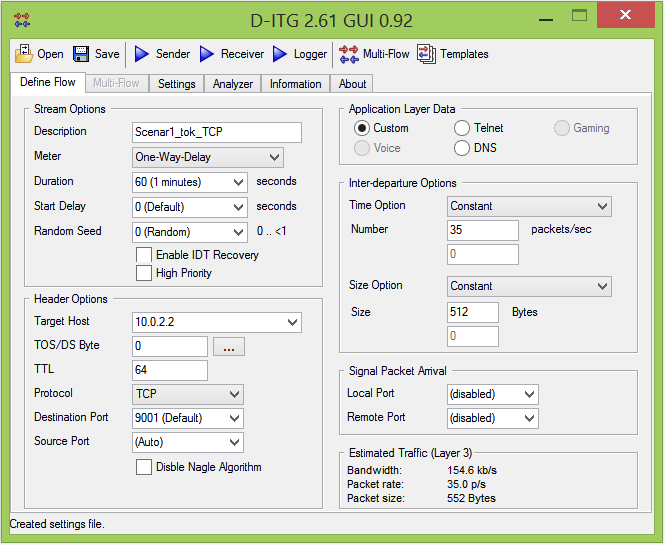
**Ukážka: Ako nastaviť D-ITG na strane odosielateľa (Sender) a príjemcu (Receiver)**

**1. Nastavenie prijímateľa dát (Receiver):**

* netreba nastavovať nič (Define Flow je len pre Sendera!), iba stlačiť “**>**Receiver”, dole v stavovom riadku musí byť, že Receiver started...

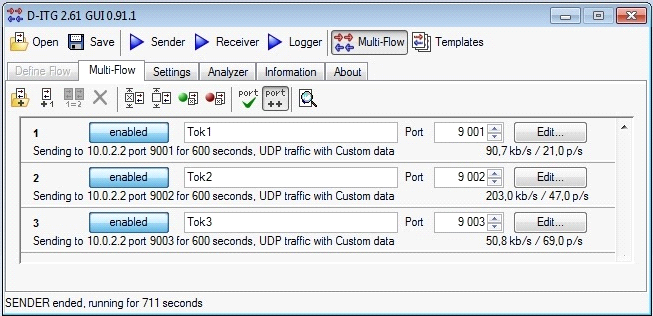
**2. Nastavenie odosielateľa dát (Sender):**

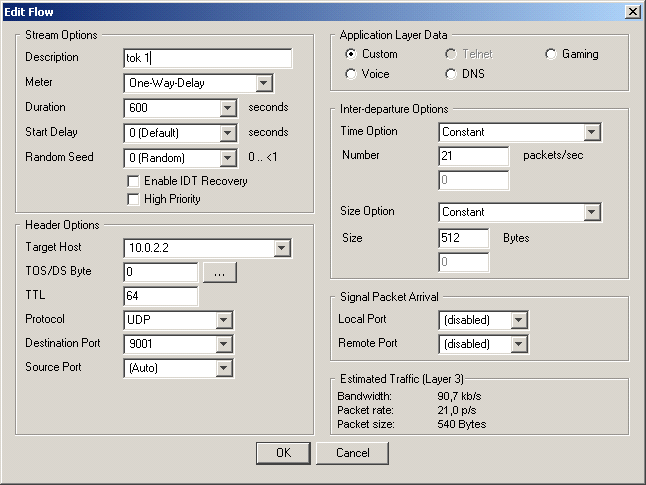
* Nastavte najprv parametre pre generovaný tok v “Define Flow” záložke:



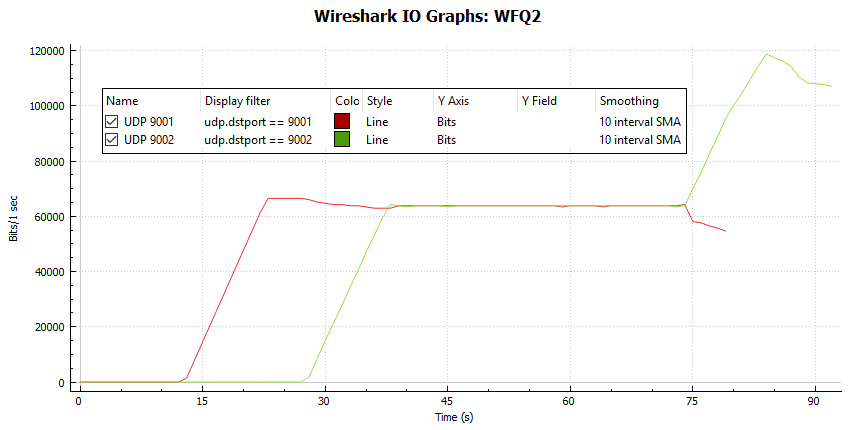
* Potom stlačiť “>Sender”

**Ako vygenerovať Multi-Flow:**

* na strane Sendera si vybrať záložku “Multi-Flow”, nastaviť parametre, a keď je pripravený Sender, stlačiť “>Sender”

"+1"  


* Ako vyzerá IO graf vo Wiresharku – sledovať treba na prijímajúcej stanici!



**Poznámky ku konfiguračným príkazom**

<http://www.cisco.com/c/en/us/td/docs/ios/ios_xe/qos/configuration/guide/convert/qos_mqc_xe/legacy_qos_cli_deprecation_xe.html>  
<http://www.cisco.com/en/US/docs/ios/12_0t/12_0t5/feature/guide/cbwfq.html>  
[http://www.cisco.com/c/en/us/td/docs/ios/qos/command/reference/qos\_book/qos\_d1.html#wp1028176](http://www.cisco.com/c/en/us/td/docs/ios/qos/command/reference/qos_book/qos_d1.html) ... Effective with Cisco IOS XE Release 2.6  
Cisco IOS Release 15.0(1)S, and Cisco IOS Release 15.1(3)T, the fair-queue command is hidden in interface configuration mode. Although this command is still available in Cisco IOS software, the CLI interactive Help does not display it if you attempt to view it by entering a question mark at the command line. .. will be completely removed in a future ...

|  |  |
| --- | --- |
| **Hidden or Removed Commands** | **Replacement MQC Command Sequence** |
| Commands  •[fair-queue (WFQ)](http://www.cisco.com/en/US/docs/ios/qos/command/reference/qos_d1.html)  Command Usage  Router(config)# **interface** *type* *number*  Router(config-if)# **fair-queue** *[congestive-discard-threshold* [*dynamic-queue-count* [*reserved-queue-count*]]] | Command Usage  Router(config)# **policy-map** *policy-map-name*  Router(config-pmap)# class class-default  Router(config-pmap-c)# fair-queue  Router(config-pmap-c)# **fair-queue** *dynamic-queues*  Router(config-pmap-c)# **fair-queue queue-limit** *packets* |

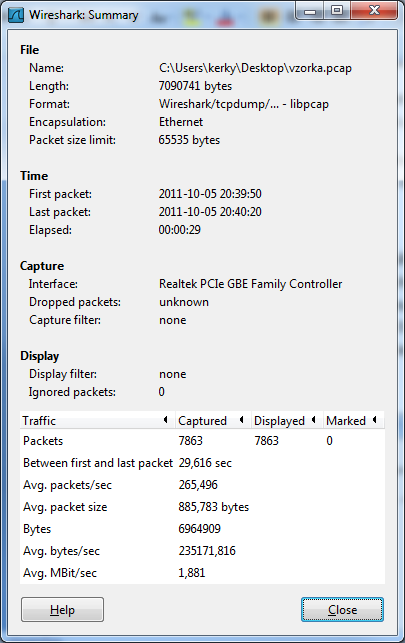
**Wireshark a sieťové štatistiky**

Wireshark ako jeden z najpoužívanejších protokolových analyzérov a paketových snifferov ponúka taktiež širokú škálu sieťových štatistík. Po odmeraní reálnej prevádzky v sieti je možné kliknúť v menu na 7. ponuku s názvom Statistics, ktorá poskytuje množstvo informácii o našom odmeranom toku. Kvôli lepšiemu znázorneniu týchto štatistík budú jednotlivé položky vysvetlené na ukážkovom sieťovom toku reálnej prevádzky o dĺžke 30 sekúnd. Pri testovaní sme použili verziu Wiresharku 1.6.2.

**2.1 Prehľad štatistík**

**2.1.1 Summary**

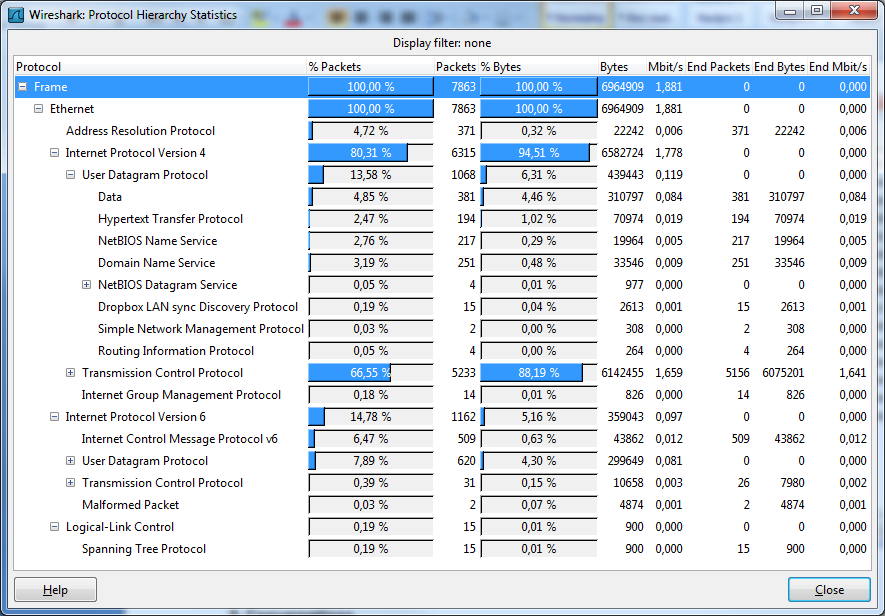
Hneď po rozbalení ponuky prvá položka s názvom Summary obsahuje informácie ako počet zachytených paketov, čas medzi príchodom prvého a posledného paketu, priemerný počet paketov za sekundu, priemernú veľkosť paketu v bajtoch ako aj celkovú veľkosť toku v bajtoch alebo napríklad priemerný počet dát za sekundu. V našom príklade máme zachytených 7863 paketov, ktorých priemerná veľkosť je 885,783 bajtov.



*Obrázok 21 Program Wireshark - Summary*

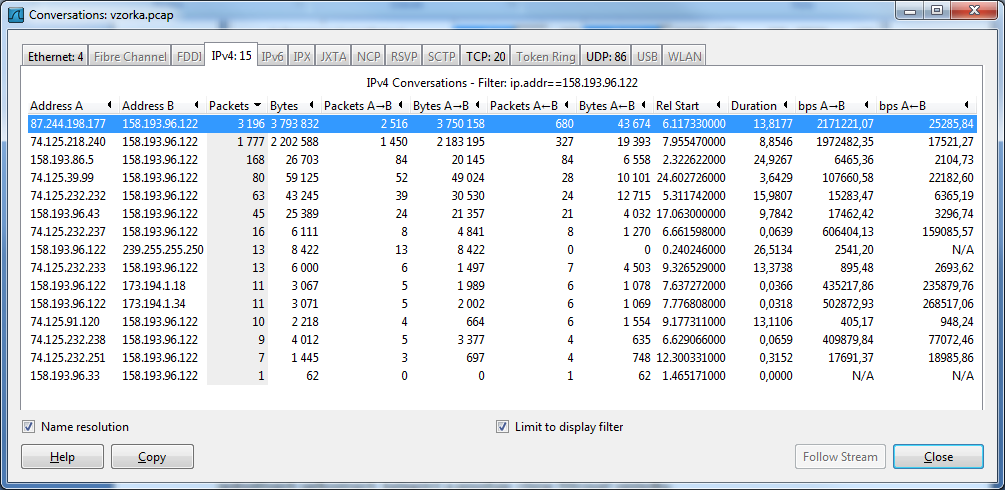
**2.1.2 Protocol Hierarchy**

Táto položka ponúka strom všetkých protokolov obsiahnutých v zachytenej prevádzke spolu s ich zastúpeniami v percentách či bajtoch. Napríklad pri protokole IPv4 bolo TCP používané viac než UDP.



*Obrázok 22 Program Wireshark - Protocol Hierarchy*

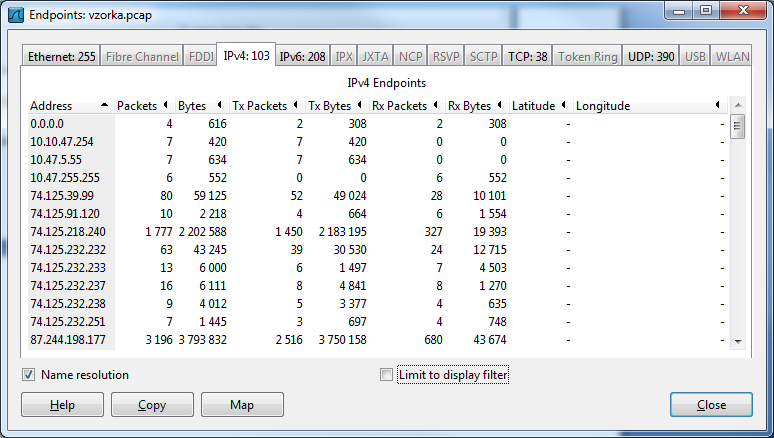
**2.1.3 Conversations**

****

*Obrázok 23 Program Wireshark - Conversations*

Okno poskytujúce prehľad konverzácií medzi koncovými bodmi v sieti. Na obrázku č.23 vidíme filtrovanú IP konverzáciu medzi adresou 158.193.96.122 a ľubovoľnou inou adresou z našej vzorky.

**2.1.4 Endpoints**

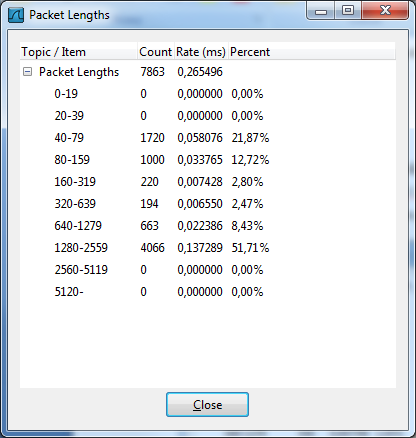


*Obrázok 24 Program Wireshark - Endpoints*

Položka Endpoints ponúka štatistiky o koncových bodoch v zachytenom toku. Číslo za názvom protokolu udáva počet zachytených koncových bodov. Napríklad v našom prípade máme zachytených 103 IPv4 koncových bodov.

**2.1.5 Packet Lengths**

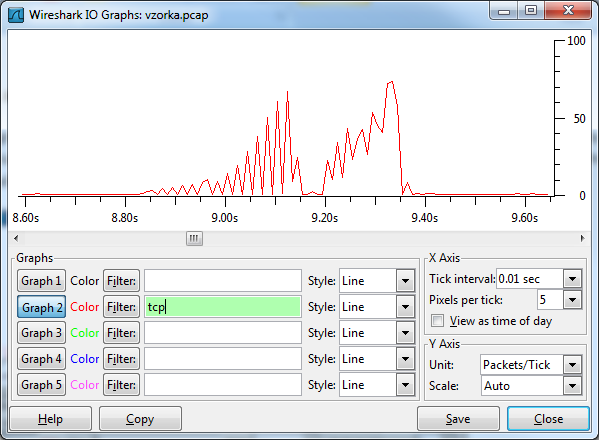
Stručný prehľad dĺžok paketov zatriedených do veľkostných kategórií. Ukazuje taktiež percentuálne zastúpenie jednotlivých veľkostných kategórií.



*Obrázok 25 Program Wireshark - Packet Lengths*

**2.1.6 IO Graphs**

Z grafových zobrazení nám položka IO Graphs ponúka pohľad na počty príchodov paketov z hľadiska časového t.j. za určitú jednotku času, ktorú si môžme navoliť. Prípadne počet paketov na osi y je možné nahradiť za počet bajtov alebo bitov. Podľa obrázka môžeme usúdiť, že približne od 9. sekundy začali TCP pakety prichádzať vo väčšom množstve.



*Obrázok 26 Program Wireshark - IO Graphs*

**2.1.7 Conversation List**

Položka poskytuje rovnaké informácie ako "2.1.3 Conversations".

**2.1.8 Endpoint List**

Položka poskytuje rovnaké informácie ako "2.1.4 Endpoints".

**2.1.9 Service Responce Time**

Ako sám názov hovorí, táto sekcia nám bude hovoriť o čase medzi žiadosťou (request) a odpoveďou (response). Máme na výber z množstva protokolov.

Ďalšie položky na karte Statistics sa zaoberajú špecifickými protokolmi.