Šablona pro odevzdávání výstupů z distančních cvičení předmětu MPC-PKT určená k editaci a odevzdání po vytvoření PDF verze

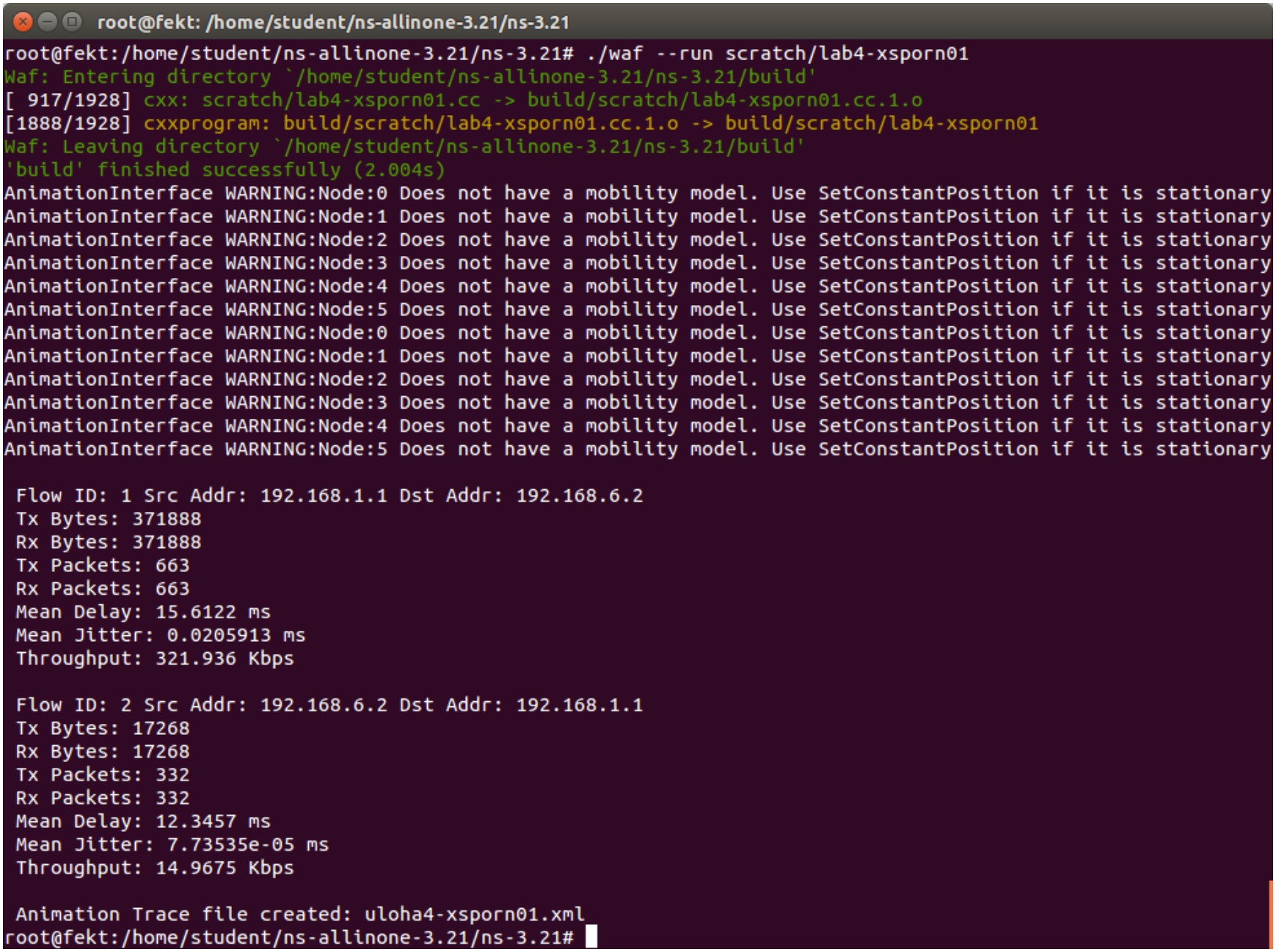
Vaše jméno	Alex Sporni
VUT ID	204633
Vypracovaný lab (označení)	**Lab4 (Unicast a statické směrování v NS3)**

---

1. číslovaný úkol z návodu

Zadání úkolu: **Pomocí printscreenu zdokumentujte a vložte do protokolu výstup z příkazového řádku po spuštění simulace. Výsledný výpis by měl odpovídat výstupu konzole terminálu viz. Obr. 2.**

Řešení:


root@fekt: /home/student/ns-allinone-3.21/ns-3.21
root@fekt:/home/student/ns-allinone-3.21/ns-3.21# ./waf --run scratch/lab4-xsporn01
Waf: Entering directory `/home/student/ns-allinone-3.21/ns-3.21/build'
[917/1928] cxx: scratch/lab4-xsporn01.cc -> build/scratch/lab4-xsporn01.cc.1.o
[1888/1928] cxxprogram: build/scratch/lab4-xsporn01.cc.1.o -> build/scratch/lab4-xsporn01
Waf: Leaving directory `/home/student/ns-allinone-3.21/ns-3.21/build'
'build' finished successfully (2.004s)
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
Flow ID: 1 Src Addr: 192.168.1.1 Dst Addr: 192.168.6.2
Tx Bytes: 371888
Rx Bytes: 371888
Tx Packets: 663
Rx Packets: 663
Mean Delay: 15.6122 ms
Mean Jitter: 0.0205913 ms
Throughput: 321.936 Kbps
Flow ID: 2 Src Addr: 192.168.6.2 Dst Addr: 192.168.1.1
Tx Bytes: 17268
Rx Bytes: 17268
Tx Packets: 332
Rx Packets: 332
Mean Delay: 12.3457 ms
Mean Jitter: 7.73535e-05 ms
Throughput: 14.9675 Kbps
Animation Trace file created: uloha4-xsporn01.xml
root@fekt:/home/student/ns-allinone-3.21/ns-3.21#

---

2. číslovaný úkol z návodu

Zadání úkolu: **Proč je hodnota „Throughput“ ve výpisu konzole vyšší ve směru z uzlu n0 na n5 než opačně? Co znamená hodnota „Jitter“ z výpisu a co ovlivňuje její velikost?**

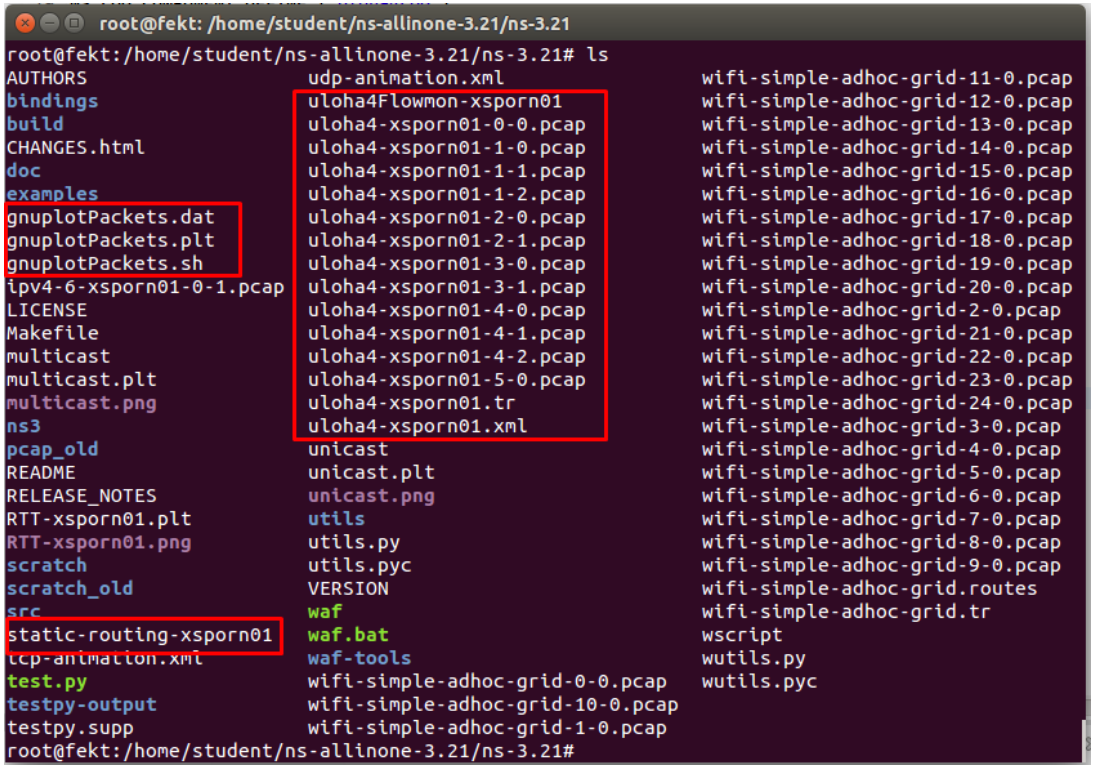
Řešení: Hodnota throughput je v prípade smeru z **n0 → n5** vyššia z dôvodu, že sa posielajú reálne dáta, v prípade opačného smeru **n5 → n0** sa posielajú iba ACK tj. potvrdenia o prijatí.

Jitter = V počítačových sieťach založených na IP protokole ako je Internet, jitter znamená kolísanie veľkosti paketov pri priechode sieťou. Vzniká napr. na smerovačoch ako dôsledok zmien smerovania, chovania interných front smerovača atď... Vyjadruje sa v sekundách po prípade v milisekundách ako v našom prípade.
zdroj: https://cs.wikipedia.org/wiki/Jitter

---

3. číslovaný úkol z návodu

Zadání úkolu: **Vytvořte printscreen složky ns3, kde budou vidět názvy výše zmíněných souborů z vaší simulace.**
Řešení:

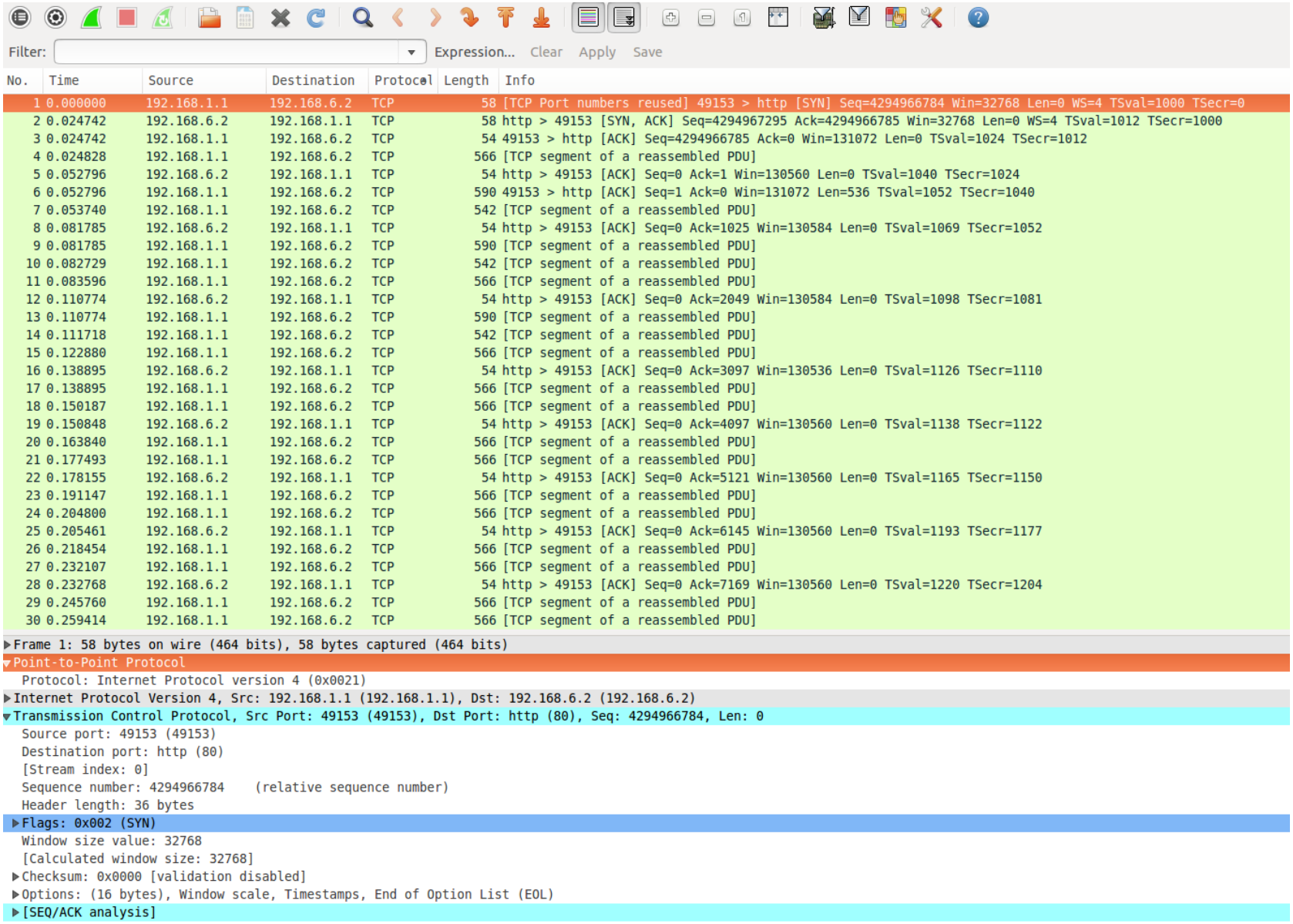


---

4. číslovaný úkol z návodu

Zadání úkolu: **Prostudujte si trasovací soubory pro jednotlivé uzly ve Wiresharku (uzly n0 a n5). Zaměřte se na IP adresy, typ protokolu, čísla portů a samotný průběh komunikace. Pomocí printscreenu zdokumentujte a vložte do protokolu výstup trasovacího souboru pro uzel n0 (viz ukázka na Obr. 3). Jak probíhá navazování a ukončovaní spojení u protokolu TCP?**

Řešení:



Nadväzovanie TCP spojenia: nadviazanie spojenia pomocou protokolu TCP sa zahajuje pomocou tzv. **3-way handshake** procesu (trojcestný handshake).

- Klient odošle na server datagram s nastaveným príznakom **SYN** a náhodne vygenerovaným poradovým číslom (x), potvrdzujúce číslo = 0
- Server odošle klientovi datagram s nastavenými príznakmi **SYN** a **ACK**, potvrdzujúce číslo = x + 1, poradové číslo je náhodne vygenerované (y)
- Klient odošle datagram s nastaveným príznakom **ACK**, poradové číslo = x + 1, číslo odpovede = y + 1

Obe strany si pamätajú svoje vlastné poradové číslo aj číslo protistrany.

Ukončenie TCP spojenia: ukončenie spojenia prebieha podobne ako jeho nadviazanie. Na ukončenie spojenia sa používa tzv. **4-way handshake** proces.

- Strana, ktorá nechce posielať ďalšie dáta, odošle datagram s nastaveným príznakom **FIN**
- Protistrana odpovie datagramom s nastaveným príznakom **ACK** s potvrdzovacím číslom o jednotku väčším, než bolo poradové číslo v datagrame s príznakom **FIN**
- Druhá strana, ktorá ukončuje posielanie dát, odošle datagram s nastavením príznakom **FIN**
- Protistrana odpovie datagramom s nastaveným príznakom **ACK**

Po prvých dvoch krokoch môže druhý strana pokračovať v posielaní dát. Pokiaľ žiadne dáta poslané nebudú, môže byť kroky 2 a 3 zlúčené. Až potom, po týchto 4 krokoch je spojenie ukončené.

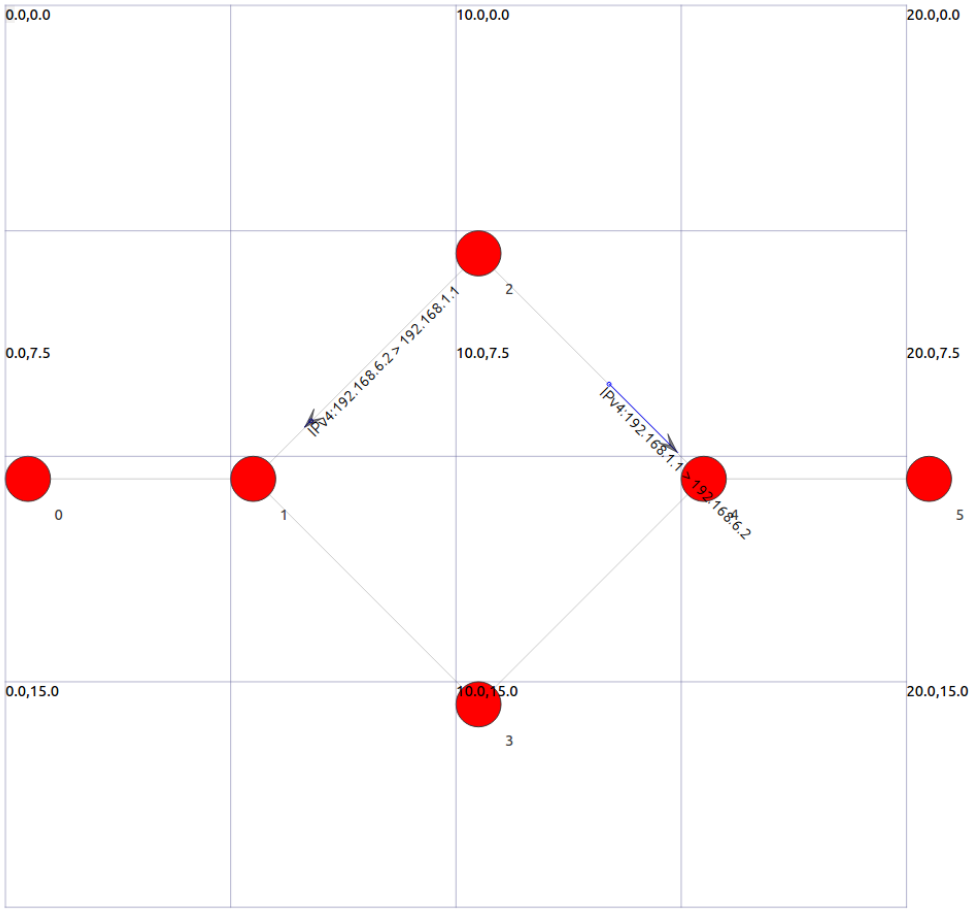
Zdroj: https://cs.wikipedia.org/wiki/Transmission_Control_Protocol

---

5. číslovaný úkol z návodu

Zadání úkolu: **Spusťte program NetAmin a otevřete soubor uloha4-VUTlogin.xml. Pomocí printscreenu zdokumentujte a vložte do protokolu výstup simulace z programu NetAmin. Zachyťte moment, kde bude zřetelně vidět, že komunikace probíhá v obou směrech přes uzel n2, jak je tomu ilustračně na Obr. 4.**

Řešení:



---

6. číslovaný úkol z návodu

Zadání úkolu: **Pomocí printscreenu zdokumentujte a vložte do protokolu výstup ze souboru static-routing-VUTlogin, který obsahuje směrovací tabulky. Směrovací tabulky by měly být shodné s těmi na Obr. 5. V printscreenu musí být vidět i název souboru s vaším VUT loginem, jako v ukázce.**

Řešení:

```
root@fekt:/home/student/ns-allinone-3.21/ns-3.21# cat static-routing-xsporn01
Node: 0 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.1.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.6.0     192.168.1.2     255.255.255.252 UGS   0      -      -   1
192.168.4.0     192.168.1.2     255.255.255.252 UGS   0      -      -   1
192.168.2.0     192.168.1.2     255.255.255.252 UGS   0      -      -   1
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting

Node: 1 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.1.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.2.0     0.0.0.0         255.255.255.252 U     0      -      -   2
192.168.3.0     0.0.0.0         255.255.255.252 U     0      -      -   3
192.168.6.0     192.168.2.2     255.255.255.252 UGS   5      -      -   2
192.168.4.0     192.168.2.2     255.255.255.252 UGS   5      -      -   2
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting

Node: 2 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.2.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.4.0     0.0.0.0         255.255.255.252 U     0      -      -   2
192.168.6.0     192.168.4.2     255.255.255.252 UGS   0      -      -   2
192.168.1.0     192.168.2.1     255.255.255.252 UGS   0      -      -   1
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting

Node: 3 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.3.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.5.0     0.0.0.0         255.255.255.252 U     0      -      -   2
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting

Node: 4 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.4.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.5.0     0.0.0.0         255.255.255.252 U     0      -      -   2
192.168.6.0     0.0.0.0         255.255.255.252 U     0      -      -   3
192.168.1.0     192.168.4.1     255.255.255.252 UGS   5      -      -   1
192.168.2.0     192.168.4.1     255.255.255.252 UGS   0      -      -   1
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting

Node: 5 Time: 10s Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
192.168.6.0     0.0.0.0         255.255.255.252 U     0      -      -   1
192.168.1.0     192.168.6.1     255.255.255.252 UGS   0      -      -   1
192.168.2.0     192.168.6.1     255.255.255.252 UGS   0      -      -   1
192.168.4.0     192.168.6.1     255.255.255.252 UGS   0      -      -   1
  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
```

7. číslovaný úkol z návodu

Zadání úkolu: **Popište význam jednotlivých položek směrovací tabulky pro uzel n1 (Destination, Gateway, Genmask, Flags, Metric, Iface).**

Řešení:

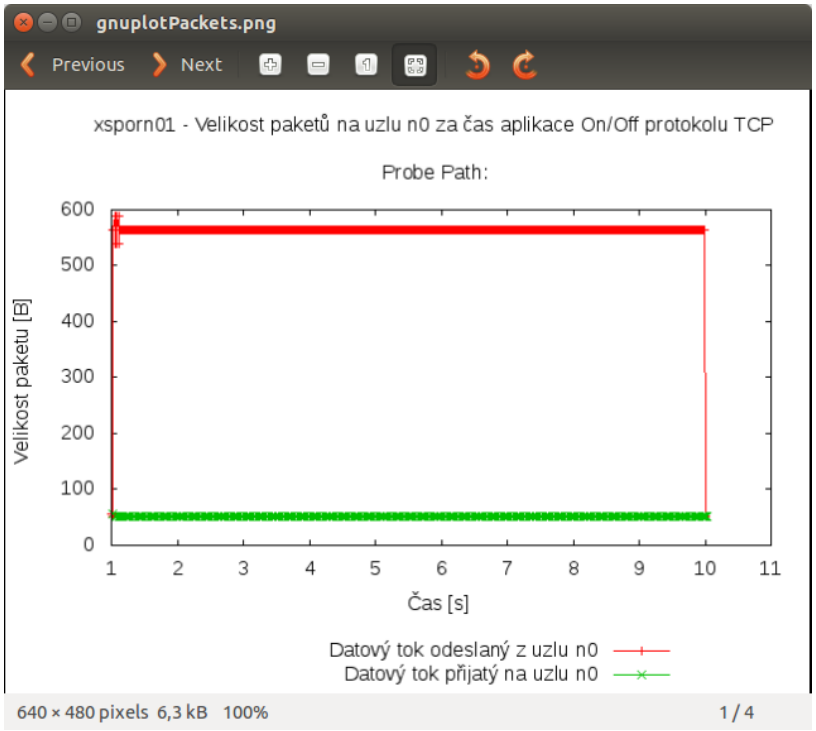
- **Destination** = destination označuje ciel siete. Za ciel sa považuje adresa podsiete, cielového počítača alebo implicitná (defaultná) trasa (tj. adresa smerovača, za ktorým je typicky zvyšok Internetu). V našom prípade na uzli **n0** prvý záznam tvorí adresa 127.0.0.0.
- **Gateway** = gateway alebo default gateway označuje bránu do Internetu. Hodnota brány udáva IP adresu najbližšieho smerovača, na ktorý majú byť datagramy predávané. Vo väčšine prípadov sa za IP adresu brány volí najnižšia v sieti. Tj. pre sieť 192.168.1.0/24 by to bola 192.168.1.1. V našom prípade sa jedná o adresu 0.0.0.0/0 je to tzv. quadra zero route alebo defaultna routa. Neoznačuje žiadnu špecifickú adresu next-hop smerovača práve naopak, špecifikuje všetky možné siete.
- **Genmask** = **subnet mask = sieťová maska** alebo **maska podsiete** slúži pre označenie rozsahu IP adries, pre ktoré záznam v tabuľke platí. Spravidla určuje tzv. **network portion a host portion.** Vymedzuje sieťovú a hosťovskú časť IP adresy. Napr. spomínaná adresa 192.168.1.0/24 má vyhradených 24 bitov pre sieťovú časť a zvyšných 8 bitov pre užívateľov → tj. 2^8 – 2 (network, broadcast) použiteľných adries pre zariadenia. V našom prípade sa často používa práve maska /30 tzv. Point to point maska, ktorá poskytuje práve len 2 voľné IP adresy z rozsahu.
- **Flags** = dané flagy (príznaky) hovoria o tom, že: **U** → sieť je živá – up, **G** → že sa jedná o hlavnú gateway pre danú sieť, **S** → že sa jedná o staticky zapísanú adresu
- **Metric** = je to tzv. metrika siete, vyjadruje relatívnu cenu pri použití danej trasy pre prenos dát k cieľu. Obvykle býva metrikou počet smerovaní (hop-count). V našom prípade máme len metriku **0**, ktorá označuje priamo pripojenú linku. Iné typy metrík viď. https://brunswyck.github.io/docu/_images/Administrative-Distance.png
- Iface = označuje rozhranie, na ktoré je pripojená daná linka do siete.

---

8. číslovaný úkol z návodu

Zadání úkolu: **Pomocí printscreenu zdokumentujte a vložte do protokolu výstup vytvořeného grafu gnuplotPackets.png. V názvu grafu pomocí vhodné změny kódu nahraďte část „VUTlogin“ svým VUT loginem. Váš výstup by měl být obdobný jako na Obr. 6. Proč se velikost paketů v jednotlivých směrech zásadně liší?**

Řešení: Veľkosť paketov sa môže líšiť z dôvodu veľkosti posielania dát. Ako som spomínal vyššie, tak zo smeru **n1 → n5** sa posielajú dáta o veľkosti 512b a z opačného smeru **n5 → n1** sú to len ACK pakety.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.1	192.168.6.2	TCP	58	[TCP Port numbers reused] 49153 > http [SYN] Seq=4294966784 Win=32768 Len=0 W
2	0.024742	192.168.6.2	192.168.1.1	TCP	58	http > 49153 [SYN, ACK] Seq=4294967295 Ack=4294966785 Win=32768 Len=0 WS=4 TS
3	0.024742	192.168.1.1	192.168.6.2	TCP	54	49153 > http [ACK] Seq=4294966785 Ack=0 Win=131072 Len=0 TSval=1024 TSecr=101
4	0.024828	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
5	0.052796	192.168.6.2	192.168.1.1	TCP	54	http > 49153 [ACK] Seq=0 Ack=1 Win=130560 Len=0 TSval=1040 TSecr=1024
6	0.052796	192.168.1.1	192.168.6.2	TCP	590	49153 > http [ACK] Seq=1 Ack=0 Win=131072 Len=536 TSval=1052 TSecr=1040
7	0.053740	192.168.1.1	192.168.6.2	TCP	542	[TCP segment of a reassembled PDU]
8	0.081785	192.168.6.2	192.168.1.1	TCP	54	http > 49153 [ACK] Seq=0 Ack=1025 Win=130584 Len=0 TSval=1069 TSecr=1052
9	0.081785	192.168.1.1	192.168.6.2	TCP	590	[TCP segment of a reassembled PDU]
10	0.082729	192.168.1.1	192.168.6.2	TCP	542	[TCP segment of a reassembled PDU]
11	0.083596	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
12	0.110774	192.168.6.2	192.168.1.1	TCP	54	http > 49153 [ACK] Seq=0 Ack=2049 Win=130584 Len=0 TSval=1098 TSecr=1081
13	0.110774	192.168.1.1	192.168.6.2	TCP	590	[TCP segment of a reassembled PDU]
14	0.111718	192.168.1.1	192.168.6.2	TCP	542	[TCP segment of a reassembled PDU]
15	0.122880	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
16	0.138895	192.168.6.2	192.168.1.1	TCP	54	http > 49153 [ACK] Seq=0 Ack=3097 Win=130536 Len=0 TSval=1126 TSecr=1110
17	0.138895	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
18	0.150187	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
19	0.150848	192.168.6.2	192.168.1.1	TCP	54	http > 49153 [ACK] Seq=0 Ack=4097 Win=130560 Len=0 TSval=1138 TSecr=1122
20	0.163840	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]
21	0.177493	192.168.1.1	192.168.6.2	TCP	566	[TCP segment of a reassembled PDU]

---

9. číslovaný úkol z návodu

Zadání úkolu: **Popište výhody a nevýhody statického směrování. Kde se toto směrování používá?**

Řešení: Statické smerovanie sa typicky používajú koncové stanice alebo smerovače v malých počítačových sieťach LAN, pretože záznamy nie je nutné v priebehu činnosti zariadenia meniť alebo preto, že sú záznamy jednoduché. V prípade väčších sietí by manuálne zadávanie administrátorom do smerovacích tabuliek bolo príliš časovo náročné a neefektívne. Aj z tohto dôvodu sa vo väčších sieťach uprednostňuje dynamické smerovanie pomocou smerovacích protokolov ako RIP, RIPv2, OSPF, EIGRP a BGP v prípade WAN.

Výhody: nižšia réžia (smerovače si nemusia posielať aktualizácie v prípade zmeny topológie), väčšia kontrola nad sieťou...
Nevýhody: Ako som spomínal vyššie tak: časovo náročnejšie, môže dôjsť k chybám so strany administrátora (chybný záznam), v prípade zmeny topológie (odstránenie smerovača alebo pád linky) nedôjde k automatického presmerovanie sieťového provozu ani k aktualizácii smerovacej tabuľky.

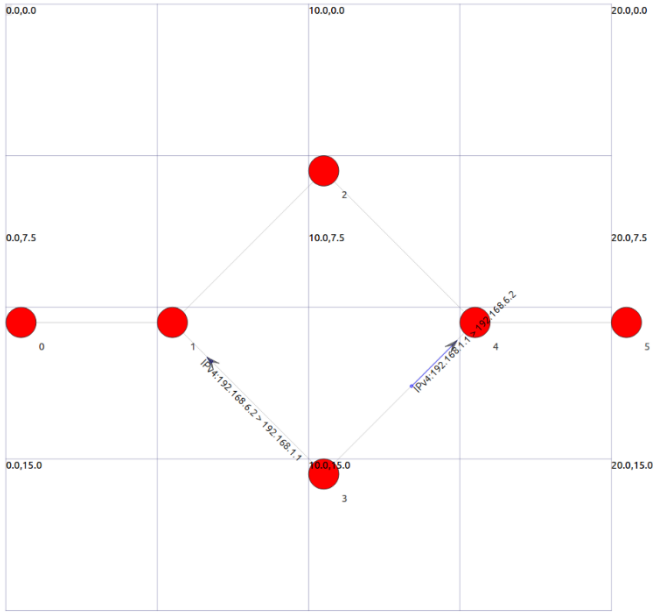
---

10. číslovaný úkol z návodu

Zadání úkolu: **Vhodně doplňte směrování (sekce Směrování ve zdrojovém kódu), aby byla vytvořena cesta vedoucí přes uzel n3 a to opět v obou směrech komunikace. Nastavení směrování musí být ve stejném duchu a rozsahu jako u směrování uvedeného v tomto návodu. Do protokolu uveďte část zdrojového kódu, který jste upravili a výstup z programu NetAmin, kde bude zřetelně vidět obousměrný provoz přes n3.**

Řešení: Doplnil som červeno vyznačené riadky kódu, tentokrát ale bez metriky resp. s metrikou 0, ktorá sa do kódu neuvádza. Keďže v prípade metriky berieme vždy nižšiu hodnotu tak cesta medzi **n1 → n5** bude prioritne posielaná cez **n3.**

```
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1); // cesta z n0 do sítě 192.168.6.0 přes uzel n1 s metrikou 0
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1); // cesta z n0 do sítě 192.168.4.0 přes uzel n1 s metrikou 0
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1); // cesta z n0 do sítě 192.168.2.0 přes uzel n1 s metrikou 0
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.2"), 2, 5); // cesta z n1 do sítě 192.168.6.0 přes uzel n2 s metrikou 5
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.3.2"), 3);
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.2"), 2, 5); // cesta z n1 do sítě 192.168.4.0 přes uzel n2 s metrikou 5
staticRoutingN2->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.4.2"), 2); // cesta z n2 do sítě 192.168.6.0 přes uzel n4 s metrikou 0
staticRoutingN2->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.1"), 1); // cesta z n2 do sítě 192.168.1.0 přes uzel n1 s metrikou 0
staticRoutingN3->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.5.2"), 2);
staticRoutingN3->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.3.1"), 1);
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.4.1"), 1, 5); // cesta z n4 do sítě 192.168.1.0 přes uzel n2 s metrikou 5
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.5.1"), 2);
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.4.1"), 1); // cesta z n4 do sítě 192.168.2.0 přes uzel n2 s metrikou 0
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1); // cesta z n5 do sítě 192.168.1.0 přes uzel n4 s metrikou 0
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1); // cesta z n5 do sítě 192.168.2.0 přes uzel n4 s metrikou 0
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1); // cesta z n5 do sítě 192.168.4.0 přes uzel n4 s metrikou 0
```

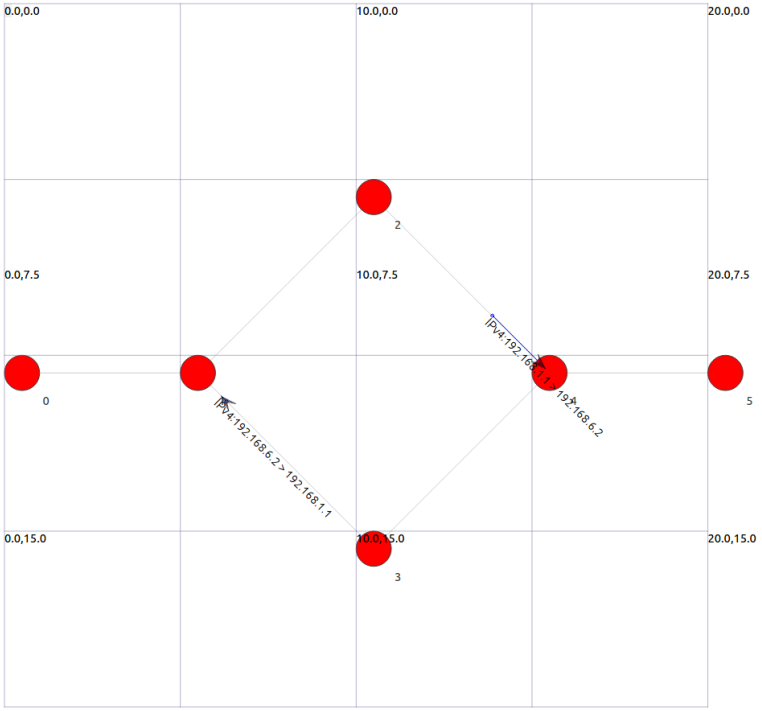


11. číslovaný úkol z návodu

Zadání úkolu: **Ovlivněte provoz za pomocí změny metriky na jednotlivých linkách tak, aby komunikace probíhala v jednom směru přes uzel n2 a v druhém (zpětném) směru přes uzel n3. Do protokolu uveďte část zdrojového kódu, který jste upravili a výstup z programu NetAmin, kde bude zřetelně vidět provoz dle tohoto zadání.**

Řešení:

```
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1);
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1);
staticRoutingN0->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.1.2"), 1);
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.2"), 2);
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.3.2"), 3, 5);
staticRoutingN1->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.2"), 2);
staticRoutingN2->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.4.2"), 2);
staticRoutingN2->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.2.1"), 1, 5);
staticRoutingN3->AddNetworkRouteTo (Ipv4Address ("192.168.6.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.5.2"), 2, 5);
staticRoutingN3->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.3.1"), 1);
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.4.1"), 1, 5);
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask ("255.255.255.252"), Ipv4Address ("192.168.5.1"), 2);
staticRoutingN4->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.4.1"), 1, 5);
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.1.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1);
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.2.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1, 5);
staticRoutingN5->AddNetworkRouteTo (Ipv4Address ("192.168.4.0"), Ipv4Mask("255.255.255.252"), Ipv4Address ("192.168.6.1"), 1, 5);
```



---

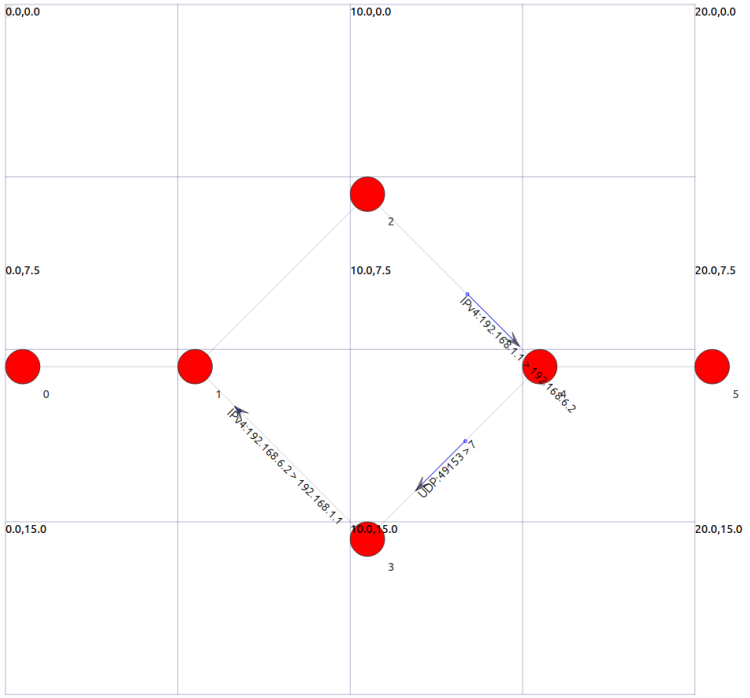
12. číslovaný úkol z návodu

Zadání úkolu: **Vytvořte další On/Off aplikaci, která bude využívat protokol UDP. Použijte port 7. Aplikaci nainstalujte na uzel n5 (cíl bude uzel n0). Aplikace bude spuštěna v 1.1 s a ukončena v 7 s. (Použijte třídu UdpSocketFactory). Tvorba aplikace je velmi podobná aplikaci využívající protokol TCP. Do protokolu uveďte část zdrojového kódu, který jste upravili a výstup z programu NetAmin, kde bude zřetelně vidět provoz obou aplikací.**

Řešení:

```
OnOffHelper onoffUDP("ns3::UdpSocketFactory", Address(InetSocketAddress (interfaces1.GetAddress(0), 7)));
onoffUDP.SetAttribute("DataRate", StringValue ("0.3Mbps"));
onoffUDP.SetAttribute("PacketSize", UintegerValue (512));
onoffUDP.SetAttribute("OnTime", StringValue ("ns3::ConstantRandomVariable[Constant=1]"));
onoffUDP.SetAttribute("OffTime", StringValue ("ns3::ConstantRandomVariable[Constant=0]"));
ApplicationContainer appUDP = onoffUDP.Install(Node5);
appUDP.Start(Seconds (1.1));
appUDP.Stop(Seconds (7.0));

PacketSinkHelper sinkUDP("ns3::UdpSocketFactory", Address(InetSocketAddress(Ipv4Address::GetAny(), 7)));
appUDP = sinkUDP.Install(Node0);
appUDP.Start(Seconds(1.1));
appUDP.Stop(Seconds(7.0));
```

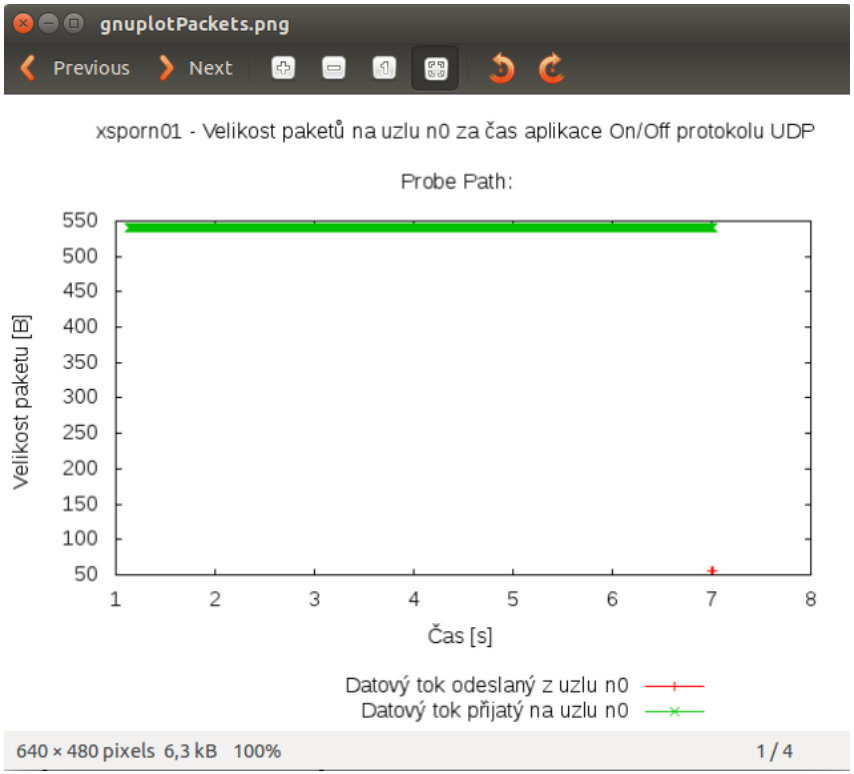


---

13. číslovaný úkol z návodu

Zadání úkolu: **V zdrojovém kódu zakomentujte celou TCP aplikaci a vygenerujte si znovu graf provozu, kde by měl být vidět jen UDP provoz. Graf vložte do protokolu. Proč je v grafu jen jedna křivka?**

Řešení: Jedna krivka je preto, pretože transportný protokol UDP je nespojitý protokol (neprebehne 3 way-handshake) a nevyžaduje ACK od príjemcu.



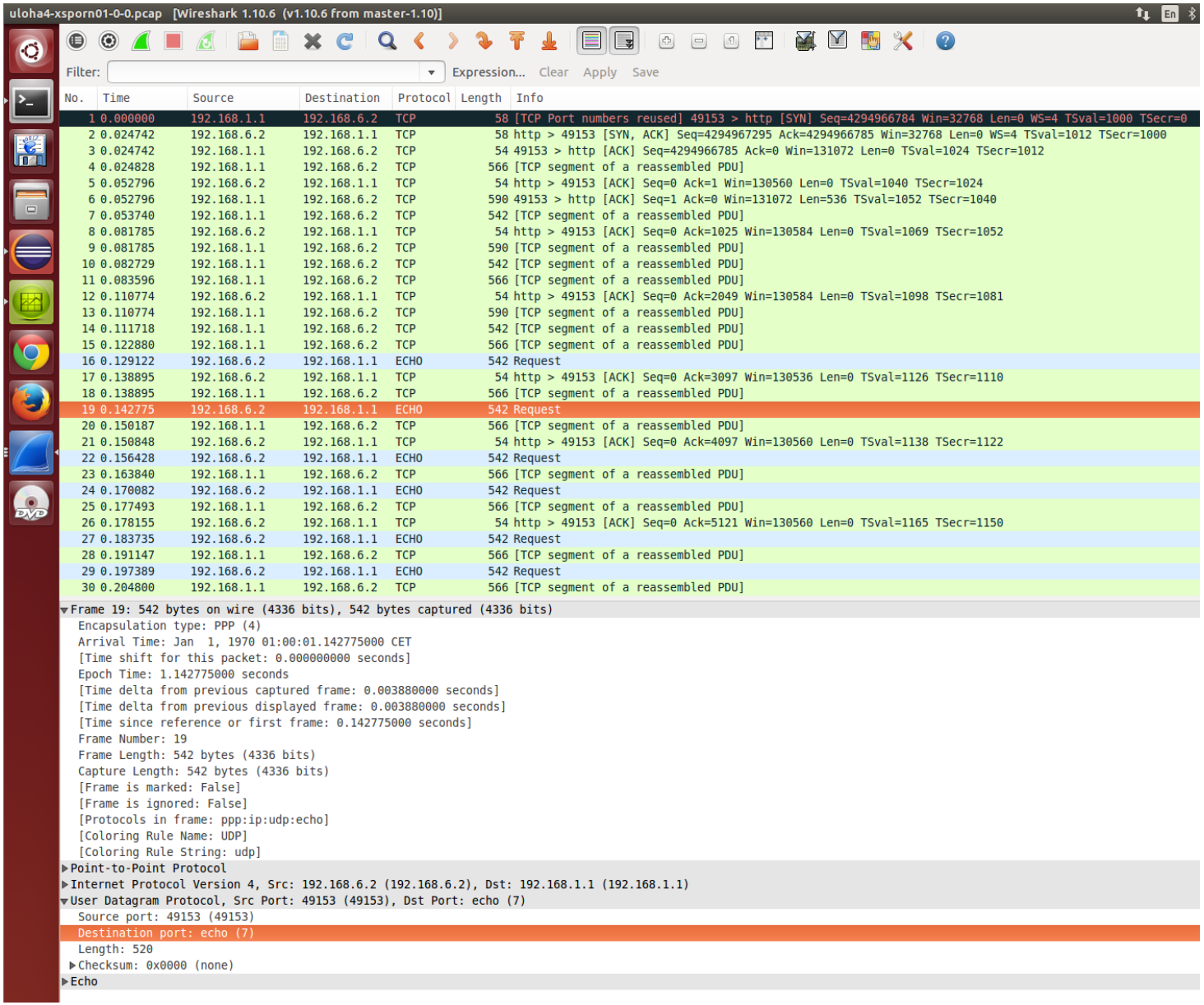
---

14. číslovaný úkol z návodu

Zadání úkolu: **V programu Wireshark prostudujte komunikaci mezi jednotlivými uzly. Všimněte si, jak se zobrazuje číslo portu 80 a 7. Do protokolu uveďte, jak probíhala komunikace u TCP protokolu a jak u protokolu UDP, zejména na začátku a na konci běhu dané aplikace.**

Řešení:

- **TCP** = vidíme že v prípade TCP na začiatku komunikácie prebehne **3 way-handshake**. Na porte 80 (http) sa posielajú správy medzi uzlami **n0** a **n5**, **n5** posiela ACK správy a potvrdzuje komunikáciu, spojenie sa ukončí **4 way-handshakom**.
- **UDP** = komunikuje cez port 7, kde sú posielané iba ICMP ECHO pakety z uzla **n5** na **n1**. Neprebieha žiadne potvrdzovanie.


uloha4-xsporn01-0-0.pcap [Wireshark 1.10.6 (v1.10.6 from master-1.10)]
Filter:
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
1 0.000000 192.168.1.1 192.168.6.2 TCP 58 [TCP Port numbers reused] 49153 > http [SYN] Seq=4294966784 Win=32768 Len=0 WS=4 TSval=1000 TSecr=0
2 0.024742 192.168.6.2 192.168.1.1 TCP 58 http > 49153 [SYN, ACK] Seq=4294967295 Ack=4294966785 Win=32768 Len=0 WS=4 TSval=1012 TSecr=1000
3 0.024742 192.168.1.1 192.168.6.2 TCP 54 49153 > http [ACK] Seq=4294966785 Ack=0 Win=131072 Len=0 TSval=1024 TSecr=1012
4 0.024828 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
5 0.052796 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=1 Win=130560 Len=0 TSval=1040 TSecr=1024
6 0.052796 192.168.1.1 192.168.6.2 TCP 590 49153 > http [ACK] Seq=1 Ack=0 Win=131072 Len=536 TSval=1052 TSecr=1040
7 0.053740 192.168.1.1 192.168.6.2 TCP 542 [TCP segment of a reassembled PDU]
8 0.081785 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=1025 Win=130584 Len=0 TSval=1069 TSecr=1052
9 0.081785 192.168.1.1 192.168.6.2 TCP 590 [TCP segment of a reassembled PDU]
10 0.082729 192.168.1.1 192.168.6.2 TCP 542 [TCP segment of a reassembled PDU]
11 0.083596 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
12 0.110774 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=2049 Win=130584 Len=0 TSval=1098 TSecr=1081
13 0.110774 192.168.1.1 192.168.6.2 TCP 590 [TCP segment of a reassembled PDU]
14 0.111718 192.168.1.1 192.168.6.2 TCP 542 [TCP segment of a reassembled PDU]
15 0.122880 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
16 0.129122 192.168.6.2 192.168.1.1 ECHO 542 Request
17 0.138895 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=3097 Win=130536 Len=0 TSval=1126 TSecr=1110
18 0.138895 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
19 0.142775 192.168.6.2 192.168.1.1 ECHO 542 Request
20 0.150187 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
21 0.150848 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=4097 Win=130560 Len=0 TSval=1138 TSecr=1122
22 0.156428 192.168.6.2 192.168.1.1 ECHO 542 Request
23 0.163840 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
24 0.170082 192.168.6.2 192.168.1.1 ECHO 542 Request
25 0.177493 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
26 0.178155 192.168.6.2 192.168.1.1 TCP 54 http > 49153 [ACK] Seq=0 Ack=5121 Win=130560 Len=0 TSval=1165 TSecr=1150
27 0.183735 192.168.6.2 192.168.1.1 ECHO 542 Request
28 0.191147 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
29 0.197389 192.168.6.2 192.168.1.1 ECHO 542 Request
30 0.204800 192.168.1.1 192.168.6.2 TCP 566 [TCP segment of a reassembled PDU]
Frame 19: 542 bytes on wire (4336 bits), 542 bytes captured (4336 bits)
Encapsulation type: PPP (4)
Arrival Time: Jan 1, 1970 01:00:01.142775000 CET
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1.142775000 seconds
[Time delta from previous captured frame: 0.003880000 seconds]
[Time delta from previous displayed frame: 0.003880000 seconds]
[Time since reference or first frame: 0.142775000 seconds]
Frame Number: 19
Frame Length: 542 bytes (4336 bits)
Capture Length: 542 bytes (4336 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: ppp:ip:udp:echo]
[Coloring Rule Name: UDP]
[Coloring Rule String: udp]
Point-to-Point Protocol
Internet Protocol Version 4, Src: 192.168.6.2 (192.168.6.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 49153 (49153), Dst Port: echo (7)
Source port: 49153 (49153)
Destination port: echo (7)
Length: 520
Checksum: 0x0000 (none)
Echo