Analyse des protocoles TCP : Une étude comparative de New Reno et New Vegas dans les réseaux mobiles à l'aide du simulateur NS2

ETTAOUIL Oussama

Encadrant : Pr. Amine Berqia
École Nationale Supérieure d'Informatique et d'Analyse des Systèmes
Université Mohammed V de Rabat

Abstract

This project delves into the comparative analysis of two prominent Transmission Control Protocol (TCP) variants, namely New Reno and New Vegas, within the context of mobile networks. Leveraging the NS2 simulator, the project aims to simulate network scenarios and thoroughly evaluate the performance of these TCP variants under varying conditions. Through meticulous experimentation and analysis, the project seeks to discern the strengths and weaknesses of each protocol, ultimately determining which one exhibits superior performance in terms of throughput, latency, and overall efficiency in mobile network environments. By shedding light on the comparative merits of these TCP protocols, this study contributes valuable insights to the optimization and enhancement of network protocols, paving the way for more robust and efficient communication in mobile networks.

Introduction

Le domaine des réseaux mobiles est en constante évolution, avec une demande croissante pour des communications efficaces et fiables. Au cœur de ces réseaux se trouvent les protocoles de contrôle de transmission (TCP), des éléments essentiels assurant la gestion du flux de données entre les appareils. Parmi les nombreuses variantes de TCP, deux se démarquent particulièrement : New Reno et New Vegas. Cette étude s'attache à une analyse comparative approfondie de ces deux protocoles TCP dans le contexte spécifique des réseaux mobiles. En utilisant le simulateur NS2, nous plongeons dans des simulations de réseaux mobiles afin d'évaluer les performances de chaque protocole dans des conditions variables. Cette analyse vise à mettre en lumière les forces et les faiblesses de New Reno et Vegas, permettant ainsi de déterminer quelle variante offre une meilleure performance en termes de débit, de latence et d'efficacité globale dans les environnements de réseau mobile. Les résultats de cette étude contribueront à l'optimisation des protocoles réseau, ouvrant ainsi la voie à des communications mobiles plus robustes et plus efficientes.

Contexte de simulation et ses objectifs

Le contexte de simulation de ce projet repose sur une analyse comparative des protocoles TCP, plus précisément New Reno et Vegas, dans le contexte spécifique des réseaux mobiles. L'objectif principal est d'évaluer les performances de ces deux protocoles dans des environnements mobiles en utilisant le simulateur NS2. Dans les réseaux mobiles, la connectivité peut être altérée par des facteurs tels que la mobilité des appareils, les variations de la qualité du signal, et la disponibilité intermittente du réseau. Ces conditions posent des défis particuliers pour les protocoles de contrôle de congestion comme TCP, qui doivent s'adapter de manière efficace pour maintenir des performances acceptables malgré ces perturbations. Ainsi, cette étude vise à comparer les performances de New Reno et Vegas en termes de délai, de débit et de packet loss ratio dans des scénarios représentatifs des réseaux mobiles. En se concentrant spécifiquement sur le packet loss ratio comme mesure de performance, nous cherchons à déterminer lequel de ces protocoles est mieux adapté aux environnements mobiles, où la perte de paquets peut être plus fréquente et imprévisible.

Réalisation: Code tcl

Ce code implémente une simulation dans le simulateur NS2 pour étudier et comparer les performances des protocoles de transmission de contrôle TCP Vegas et New Reno dans des réseaux mobiles. Il initialise un réseau ad hoc avec des nœuds mobiles et configure les paramètres de simulation, y compris les caractéristiques du canal, les propriétés de propagation, les types de nœuds, et les agents TCP. En utilisant

des agents FTP pour générer du trafic, la simulation évalue les performances des protocoles TCP en termes de délai, de débit et de perte de paquets dans un environnement de réseau mobile simulé.

```

set val(chan) Channel/WirelessChannel;
set val(prop) Propagation/TwoRayGround;
set val(netif) Phy/WirelessPhy;
set val(mac) Mac/802_11 ;
set val(ifq) Queue/DropTail/PriQueue ;
set val(ll) LL;
set val(ant) Antenna/OmniAntenna ;
set val(ifqlen) 50 ;
set val(nn) 7 ;
set val(rp) DSDV ;
set val(x) 1000;
set val(y) 1000;

set ns [new Simulator]
$ns color 2 Rouge
$ns color 1 Bleu

set tracefd [open sanet3.tr w]
$ns trace-all $tracefd

set namtrace [open sanet3.nam w]
$ns namtrace-all-wireless $namtrace $val(x)
    $val(y)

set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)

create-god $val(nn)

nnset chan_1 [new $val(chan)]
set chan_2 [new $val(chan)]
$ns node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
                -ifqLen $val(ifqlen) \
                -antType $val(ant) \
                -propType $val(prop) \
                -phyType $val(netif) \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace ON \
                -movementTrace OFF \
                -channel $chan_1 \

# Cr ation et positionnement des noeuds
for {set i 0} {$i < 7} {incr i} {
    set ID_($i) $i;
    set node_($i) [$ns node];
    $node_($i) set id $ID_($i);
    $node_($i) random-motion 0;
    $ns initial_node_pos $node_($i) 20;
}

# R glage des seuils physiques
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 4.4613e-10    ;
Phy/WirelessPhy set RXThresh_ 4.4613e-10    ;

# Positionnement des noeuds sp cifiques
$node_(0) set X_ 460.0;
$node_(0) set Y_ 800.0;
$node_(0) set Z_ 0.0;

$node_(1) set X_ 700.0;
$node_(1) set Y_ 800.0;
$node_(1) set Z_ 0.0;

$node_(2) set X_ 575.0;
$node_(2) set Y_ 500.0;
$node_(2) set Z_ 0.0;

$node_(6) set X_ 700.0;
$node_(6) set Y_ 200.0;
$node_(6) set Z_ 0.0;

$node_(5) set X_ 460.0;
$node_(5) set Y_ 200.0;
$node_(5) set Z_ 0.0;

$node_(3) set X_ 575.0;
$node_(3) set Y_ 650.0;
$node_(3) set Z_ 0.0;

$node_(4) set X_ 575.0;
$node_(4) set Y_ 350.0;
$node_(4) set Z_ 0.0;

$ns at 0.0 "$node_(0) setdest 460.0 800.0 0.0
    ";
$ns at 0.0 "$node_(1) setdest 700.0 800.0 0.0
    ";
$ns at 0.0 "$node_(2) setdest 575.0 500.0 0.0
    ";
$ns at 0.0 "$node_(6) setdest 700.0 200.0 0.0
    ";
$ns at 0.0 "$node_(5) setdest 460.0 200.0 0.0
    ";
$ns at 0.0 "$node_(3) setdest 575.0 650.0 0.0
    ";
$ns at 0.0 "$node_(4) setdest 575.0 350.0 0.0
    ";

# D finition des noms des noeuds
$ns at 0.0 "$node_(0) label S1";
$ns at 0.0 "$node_(1) label S2";
$ns at 0.0 "$node_(2) label G1";
$ns at 0.0 "$node_(3) label G2";
$ns at 0.0 "$node_(4) label G3";
$ns at 0.0 "$node_(5) label D1";
$ns at 0.0 "$node_(6) label D2";

# Cr ation des agents
set tcp1 [new Agent/TCP/Newreno];
$tcp1 set class_ 2;
set sink1 [new Agent/TCPSink];
$ns attach-agent $node_(1) $tcp1;
$ns attach-agent $node_(6) $sink1;
$ns connect $tcp1 $sink1;
set ftp1 [new Application/FTP];
$ftp1 attach-agent $tcp1;
$ns at 3.0 "$ftp1 start";
$ns at 590.0 "$ftp1 stop";

set tcp2 [new Agent/TCP/Vegas];
$tcp2 set class_ 1;
set sink2 [new Agent/TCPSink];
$ns attach-agent $node_(0) $tcp2;
$ns attach-agent $node_(5) $sink2;
$ns connect $tcp2 $sink2;
set ftp2 [new Application/FTP];
$ftp2 attach-agent $tcp2;
$ns at 3.0 "$ftp2 start";
```

```
$ns at 590.0 "$ftp2 stop";

for {set i 0} {$i < $val(nn)} {incr i} {
    $ns at 600.0 "$node_($i) reset";
}
$ns at 600.0 "stop";
$ns at 600.01 "puts \"SIMULATION TERMIN E\"
    ;$ns halt"

proc stop {} {
    global ns tracefd namtrace
    $ns flush-trace
    close $tracefd
    close $namtrace
}

puts "D marrage de NS"
$ns run
```

Listing 1: *Tcl script for setting up NS2 simulation with New Reno and Vegas TCP variants*

Figure nam

NAM, ou Network Animator, est un outil de visualisation de réseaux informatiques utilisé principalement dans le domaine des réseaux de communication et de l'informatique distribuée. Il permet de visualiser et d'analyser le comportement des réseaux à l'aide de graphiques et de figures interactives.

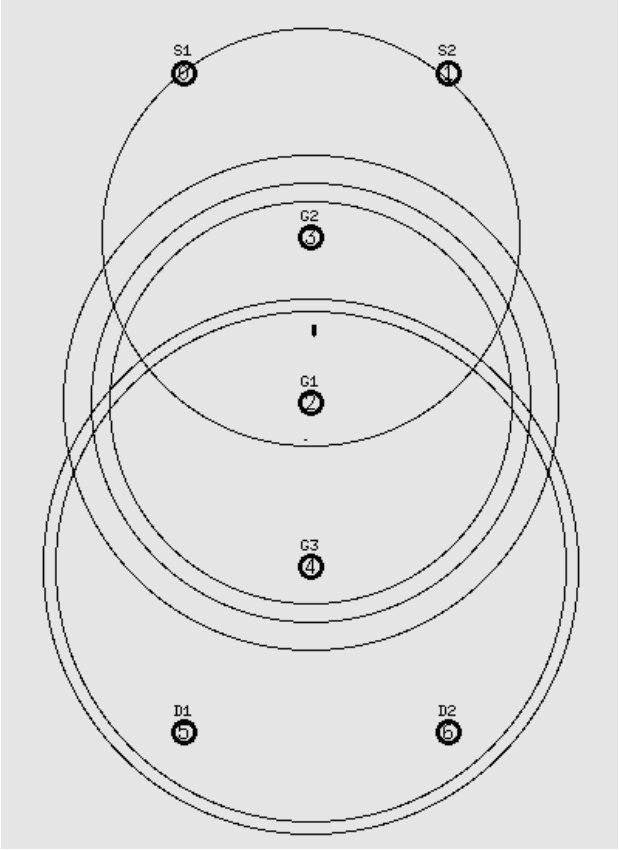

Figure 1: *Figure nam du reseau simulé*

Le script awk

Ce script AWK est conçu pour analyser les fichiers de traces issus de simulations de réseaux informatiques, notamment celles réalisées avec NS-2 (Network Simulator version 2). NS-2 est une plateforme de simulation largement adoptée pour modéliser et simuler divers scénarios de réseaux informatiques.

```
BEGIN {
    newreno_sent_pkt = 0
    newvegas_sent_pkt = 0
    newreno_rcv_pkt = 0
    newvegas_rcv_pkt = 0
    startTimeNewreno = 0
    isStartNewreno = 0
    startTimeNewvegas = 0
    isStartNewvegas = 0
    newreno_arrival_rate = 0
    newvegas_arrival_rate = 0
    packet_sent[7]=0
    packet_recv[7]=0
    packet_forw[7]=0

}
{
    event = $1
    time = $2
    node_id = $3
    level = $4
    pkt_type = $7

    if(node_id == "_0_" && pkt_type == "tcp")
     {
      if(event == "s" && level = "AGT") {
    packet_sent[0]=packet_sent[0]+1 }
      if(event == "r" && level = "AGT") {
    packet_recv[0]=packet_recv[0]+1 }
      if(event == "f") { packet_forw[0]=
    packet_forw[0]+1 }
    }
    if(node_id == "_1_" && pkt_type == "tcp")
     {
      if(event == "s" && level = "AGT") {
    packet_sent[1]=packet_sent[1]+1 }
      if(event == "r" && level = "AGT") {
    packet_recv[1]=packet_recv[1]+1 }
      if(event == "f") { packet_forw[1]=
    packet_forw[1]+1 }
    }
    if(node_id == "_2_" && pkt_type == "tcp")
     {
      if(event == "s" && level = "AGT") {
    packet_sent[2]=packet_sent[2]+1 }
      if(event == "r" && level = "AGT") {
    packet_recv[2]=packet_recv[2]+1 }
      if(event == "f") { packet_forw[2]=
    packet_forw[2]+1 }
    }
    if(node_id == "_3_" && pkt_type == "tcp")
     {
      if(event == "s" && level = "AGT") {
    packet_sent[3]=packet_sent[3]+1 }
      if(event == "r" && level = "AGT") {
    packet_recv[3]=packet_recv[3]+1 }
      if(event == "f") { packet_forw[3]=
    packet_forw[3]+1 }
    }
```

```
    if(node_id == "_4_" && pkt_type == "tcp")
     {
       if(event == "s" && level = "AGT") {
    packet_sent[4]=packet_sent[4]+1 }
       if(event == "r" && level = "AGT") {
    packet_recv[4]=packet_recv[4]+1 }
       if(event == "f") { packet_forw[4]=
    packet_forw[4]+1 }
    }
    if(node_id == "_5_" && pkt_type == "tcp")
     {
       if(event == "s" && level = "AGT") {
    packet_sent[5]=packet_sent[5]+1 }
       if(event == "r" && level = "AGT") {
    packet_recv[5]=packet_recv[5]+1 }
       if(event == "f") { packet_forw[5]=
    packet_forw[5]+1 }
    }
    if(node_id == "_6_" && pkt_type == "tcp")
     {
       if(event == "s" && level = "AGT") {
    packet_sent[6]=packet_sent[6]+1 }
       if(event == "r" && level = "AGT") {
    packet_recv[6]=packet_recv[6]+1 }
       if(event == "f") { packet_forw[6]=
    packet_forw[6]+1 }
    }
}
END {
     newreno_sent_pkt = packet_sent[1] -
    packet_forw[1] ;
     newreno_rcv_pkt  = packet_recv[6] -
    packet_forw[6] ;
     newvegas_sent_pkt = packet_sent[0] -
    packet_forw[0] ;
     newvegas_rcv_pkt = packet_recv[5] -
    packet_forw[5] ;
     newreno_arrival_rate = (newreno_rcv_pkt
    / newreno_sent_pkt)*100 ;
     newvegas_arrival_rate   = (
    newvegas_rcv_pkt /  newvegas_sent_pkt )
    *100;
     newreno_loss_rate = (newreno_sent_pkt-
    newreno_rcv_pkt)/newreno_sent_pkt *100;
     newvegas_loss_rate = (newvegas_sent_pkt-
    newvegas_rcv_pkt)/newvegas_sent_pkt *100;

     printf("NewReno :\n Sent packets = %d \n
     Received packets = %d \n Arrival ratio =
     %f \n Packet loss ration = %f\n\n",
    newreno_sent_pkt, newreno_rcv_pkt,
    newreno_arrival_rate,newreno_loss_rate);
     printf("NewVegas :\n Sent packets = %d \
    n Received packets = %d \n Arrival ratio
    = %f \n Packet loss ration = %f\n",
    newvegas_sent_pkt, newvegas_rcv_pkt,
    newvegas_arrival_rate,newvegas_loss_rate)
    ;
     for(i=0;i<7;i++) {
      printf("node %d , packet recu = %d ,
    packet sent = %d , packet forwarded = %d\
    n",i,packet_recv[i],packet_sent[i],
    packet_forw[i])
     }
}
```

Listing 2: *Script AWK*

Résultat

lorsqu'on execute les scripts awk on obtient le résultat suivant :

```
NewReno :
 Sent packets = 20261
 Received packets = 13953
 Arrival ratio = 68.866295
 Packet loss ration = 31.133705

NewVegas :
 Sent packets = 11780
 Received packets = 8461
 Arrival ratio = 71.825127
 Packet loss ration = 28.174873
node 0 , packet recu = 3945 , packet sent = 11780 , packet forwarded = 0
node 1 , packet recu = 6800 , packet sent = 20261 , packet forwarded = 0
node 2 , packet recu = 21108 , packet sent = 10554 , packet forwarded = 10554
node 3 , packet recu = 21108 , packet sent = 10554 , packet forwarded = 10554
node 4 , packet recu = 21108 , packet sent = 10553 , packet forwarded = 10554
node 5 , packet recu = 9136 , packet sent = 671 , packet forwarded = 675
node 6 , packet recu = 14594 , packet sent = 641 , packet forwarded = 641
```

Figure 2: *Figure nam du reseau simulé*

Les données extraites des fichiers .tr ont été analysées et transformées en une courbe à l'aide de Python. Cette démarche nous a permis de visualiser graphiquement l'évolution d'une variable ou d'un phénomène spécifique au fil du temps. En utilisant les fonctionnalités de traçage de Python, nous avons pu représenter de manière claire et concise les tendances et les variations des données, offrant ainsi un moyen efficace d'analyser et d'interpréter les informations contenues dans les fichiers .tr.

Analyse de la courbe

Au départ, les données révèlent que NewVegas présente un taux de perte nettement supérieur à celui de NewReno, qui reste toutefois constant. Cependant, au fil du temps, le taux de perte de NewVegas diminue rapidement, le plaçant finalement en dessous de celui de NewReno. Cette évolution suggère que NewVegas possède une capacité supérieure à s'adapter aux fluctuations du réseau, grâce à une mise en œuvre efficace de stratégies de gestion de congestion et de mécanismes de retransmission performants. Par conséquent, NewVegas se distingue par sa capacité à réduire rapidement les pertes, ce qui entraîne une amélioration globale des performances du réseau par rapport à NewReno. En outre, NewReno pourrait également bénéficier d'autres avantages tels qu'une couverture réseau plus étendue, des vitesses de transmission plus rapides ou des tarifs plus compétitifs pour ses utilisateurs.

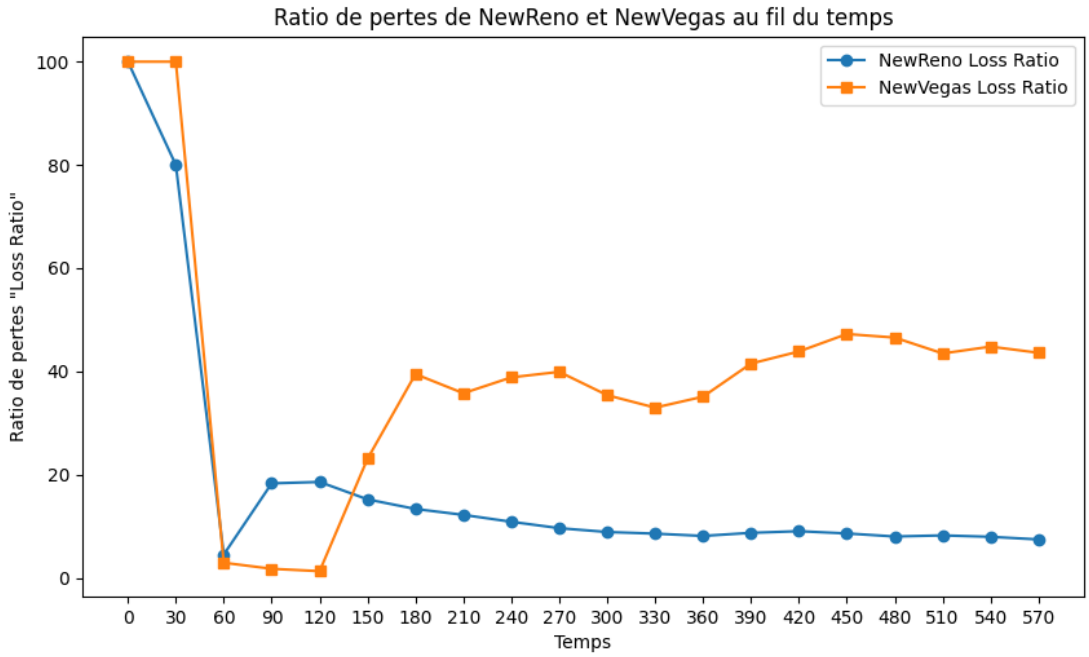

Figure 3: *Ratio de pertes de NewReno et NewVegas au fil du temps*

References

[1] Mohammad Jahangir Alam and Tanjia Chowdhury, Performance Evaluation of TCP Vegas over TCP Reno and TCP NewReno over TCP Reno

[2] Amélioration de la performance de TCP dans les réseaux mobiles ad hoc., Yassine DOUGA ,https://www.memoireonline.com/03/19/10650/m_Amelioration-de-la-performance-de-TCP-dans-les-reseaux-mobiles-ad-hoc20.html

[3] Ayman El-Sayed, Enhanced TCP Westwood congestion avoidance mechanism (TCP Westwood-New)

[4] VLAN implementation using NS2 https://www.nsnam.com/

[5] https://www.geeksforgeeks.org/what-is-tcp-new-reno/

[6] Jim Kurose, Modeling TCP through-put: A simple model and its empirical validation

[7] Tcl/Tk Documentation https://www.tcl.tk/doc/

[8] Matplotlib 3.8.2 documentation https://matplotlib.org/stable/index.html

[9] Henderson, T. R. (2002). Routing in Wireless Mesh Networks. *IEEE Communications Magazine*, 40(2), 145-149.

[10] Allman, M., Paxson, V., & Stevens, W. R. (1999). TCP Congestion Control. *RFC 2581*.