ARQ

Automatic Repeat reQuest

Κώστας Τσέας
Μάνος Κατσόμαλλος
Νικόλας Κατσιούλας

CS446 Προχωρημένα Θέματα Δικτύων

ARQ

Το ARQ είναι μια μέθοδος error-control για μετάδοση δεδομένων που χρησιμοποιεί acknowledgment (μηνύματα που αποστέλλονται από το δέκτη όταν έχει λάβει σωστά ένα πλαίσιο/πακέτο δεδομένων) και timeout (συγκεκριμένα χρονικά περιθώρια στα οποία αναμένεται το acknowledgment) για να επιτευχθεί αξιόπιστη μετάδοση δεδομένων πάνω από μια αναξιόπιστη υπηρεσία. Αν ο αποστολέας δεν λάβει acknowledgment πριν από το timeout, συνήθως αναμεταδίδει το πλαίσιο/πακέτο μέχρι ο αποστολέας να λάβει acknowledgment ή να υπερβεί ένα προκαθορισμένο αριθμό αναμεταδόσεων.

Πρωτόκολλα ARQ

Τα πρωτόκολλα ARQ λειτουργούν στο Data Link ή Transport επίπεδο του μοντέλου OSI (Open Systems Interconnection) και είναι τα εξής: Stop-and-wait ARQ, Go-Back-N ARQ και Selective Repeat ARQ. Συνήθως όλα αυτά τα πρωτόκολλα χρησιμοποιούν κάποια μορφή πρωτοκόλλου sliding window για να ενημερώσουν τον πομπό ώστε να αποφασίσει ποια πακέτα πρέπει να αναμεταδοθούν.

Stop-and-wait ARQ

Το Stop-and-wait ARQ αποτελεί την απλούστερη μέθοδο ARQ. Εξασφαλίζει ότι δεν υπάρχει απώλεια πληροφορίας εξαιτίας χαμένων πακέτων και ότι τα πακέτα λαμβάνονται με τη σωστή σειρά. Αρχικά, ο πομπός στέλνει ένα πλαίσιο τη φορά όντας μια ειδική περίπτωση του γενικού sliding window πρωτοκόλλου με μεγέθη window εκπομπής και λήψης ίσα με 1. Αφότου σταλεί το πλαίσιο, ο πομπός δε στέλνει περαιτέρω πλαίσια έως ότου λάβει ένα acknowledgement. Στη συνέχεια ο δέκτης αφού λάβει ένα σωστό πλαίσιο στέλνει acknowledgement. Αν το acknowledgement δε φθάσει στον πομπό πριν από το timeout ο πομπός ξαναστέλνει το ίδιο frame.

Go-Back-N ARQ

Το Go-Back-N ARQ αποτελεί μια ειδική περίπτωση του γενικού sliding window πρωτοκόλλου με μέγεθος window εκπομπής N και λήψης 1. Η διαδικασία εκπομπής συνεχίζει την αποστολή πλαισίων, ο αριθμός των οποίων εξαρτάται από το μέγεθος του window, λαμβάνοντας acknowledgement σειριακά για κάθε ένα πλαίσιο από το δέκτη. Έπειτα, ο δέκτης αφού λάβει ένα σωστό πλαίσιο και είναι στη σωστή σειρά στέλνει acknowledgement. Τέλος, ο πομπός ξαναστέλνει το πακέτο με όλα τα πλαίσια αν ακόμη και ένα από αυτά έχει ξεπεράσει το timeout και δεν έχει λάβει acknowledgement.

Selective Repeat ARQ

Στο Selective Repeat ARQ ο πομπός στέλνει έναν αριθμό από πλαίσια ο οποίος εξαρτάται από το μέγεθος του window χωρίς να χρειάζεται να περιμένει acknowledgement για κάθε πακέτο ξεχωριστά. Ακολούθως ο δέκτης στέλνει acknowledgement για κάθε πλαίσιο που έλαβε και τα πλαίσια (μη-ταξινομημένα) αποθηκεύονται στο buffer. Τέλος, ο πομπός ξαναστέλνει πλαίσια τα οποία έχουν ξεπεράσει το timeout και δεν έχουν λάβει acknowledgement.

Σχετικά με το project

Προβλήματα

Στόχος του project ήταν να πειραματιστούμε και με τα 3 πρωτόκολλα ARQ. Όμως, κατόπιν έρευνας, διαπιστώσαμε πως δε μπορούμε να αλλάξουμε το πρωτόκολλο αλλά μόνο να δουλέψουμε με τις παραμέτρους που περιγράφονται στο documentation του WiMAX. Έτσι αρκεστήκαμε στο να δουλέψουμε μόνο με το πρωτόκολλο που παρέχεται από το WiMAX 802.16e του base station και να παρατηρήσουμε τις μεταβολές στο RTT και στο πλήθος των χαμένων πακέτων αλλάζοντας το μέγεθος του block και window.

Αρχικά εκτελέσαμε το πείραμα με τις default ρυθμίσεις του base station για 100sec. Παρατηρώντας όμως τις μετρήσεις, διαπιστώσαμε πως δεν υπήρχαν μεγάλες διαφορές τόσο στο RTT όσο και στον αριθμό των χαμένων πακέτων. Έτσι, για να δημιουργήσουμε διακύμανση στις μετρήσεις, αλλάξαμε την ισχύ του base station και το MCS. Δοκιμάσαμε όλους τους δυνατούς συνδιασμούς και καταλήξαμε σε:

- bstxpow = 21
- mcsProfile?dl = 20 (64 QAM 3/4)

Ήταν αρκετά δύσκολο να έχουμε όσο το δυνατόν πιο αξιόπιστα αποτελέσματα και να διατηρήσουμε τις συνθήκες εκτέλεσης του πειράματος σταθερές. Ο χρόνος εκτέλεσης και προετοιμασίας του πειράματος για κάθε περίπτωση ήταν αρκετά μεγάλος περίπου 10’. Οι παράμετροι (bstxpw και dl) που επιλέξαμε για την διασύνδεση των κόμβων στην ουσία είχαν ως στόχο να κάνουν πιο δύσκολη την επικοινωνία των κόμβων, οδηγούσαν σε επανειλημμένες αποτυχίες του πειράματος. Τέλος, οι συχνές διακοπές στη διαθεσιμότητα των κόμβων του testbed επιβάρυναν ακόμα περισσότερο την εκτέλεση.

Πείραμα

Χρησιμοποιήσαμε 2 διαφορετικές εφαρμογές για να μετρήσουμε τα χαρακτηριστικά QoS. Iperf για να μετρήσουμε το πλήθος των χαμένων πακέτων και ping για την παρατήρηση του RTT.

- iperf

```
// παραγωγή πακέτων UDP
app.setProperty('udp', true)
// παύση 1 sec μεταξύ των αναφορών για το bandwidth
app.setProperty('interval', 1)
// ορισμός buffer 600KB
app.setProperty('len', 600)
// ορισμός bandwidth 80Kbps
app.setProperty('bandwidth', 80000)
// ορισμός χρόνου εκπομπής iperf 50sec
app.setProperty('time', 50)
```

- ping

```
// ορισμός πλήθος ping 100
app.setProperty('count', 100)
```

Προετοιμασία

Για να εκτελέσουμε το πείραμα φορτώναμε την εικόνα δίσκου wmxlab-qos.ndz στους κόμβους 6 και 7. Θέταμε την ισχύ του base station σε 20 το MCS προφίλ σε 21.

Διαδικασία

Πειραματιστήκαμε για τιμές window 128, 256, 512 και 1024 και για τιμές block 16, 32, …, wsize/2.

- wsize = 128
  bloksize = {16, 32, 64}
- wsize = 256
  bloksize = {16, 32, 64, 128}
- wsize = 512
  bloksize = {16, 32, 64, 128, 256}
- wsize = 1024
  bloksize = {16, 32, 64, 128, 256, 512}

Εντολές

Αρχικά ορίσαμε την ισχύ του base station και το προφίλ MCS. Στη συνέχεια ορίζαμε το μέγεθος του window και για κάθε περίπτωση τις αντίστοιχες τιμές block. Σε κάθε περίπτωση κάναμε επανεκκίνηση στο base station για να αποθηκευτούν οι νέες τιμές.

- Ορισμός ισχύς base station:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/wireless/set?bstxpow=21"`
- Ορισμός προφίλ MCS:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/mcsProfile?dl=20"`
- Ενεργοποίηση/απενεργοποίηση ARQ:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?enable=true/false"`
- Ορισμός μεγέθους window:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?wsize={0-1024}"`
- Ορισμός μεγέθους block:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?bloksize={0-wsize/2}"`
- Επανεκκίνηση base station:
  `wget -qO- "http://wimaxrf:5052/wimaxrf/bs/restart"`

Αποτελέσματα

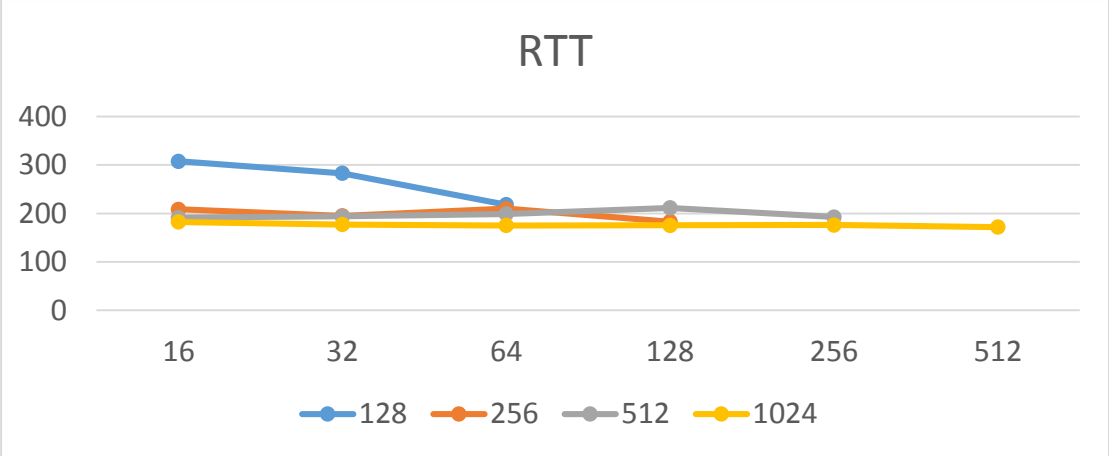
Για να έχουμε όσο το δυνατόν πιο ρεαλιστικά αποτελέσματα προσπαθήσαμε να τρέξουμε αρκετές επαναλήψεις για κάθε συνδυασμό μεγέθους window και block. Έτσι αφού παραλείψαμε κάποια αποτελέσματα στα οποία οι τιμές είχαν μεγάλη απόκλιση από τα υπόλοιπα προέκυψε ο παρακάτω πίνακας.

ws	bs	rtt		packet lost	
		I	II	I	II
128	16	310	305	21	4
	32	309	257	6	11
	64	238	198	14	9
256	16	208	209	10	13
	32	210	180	12	8
	64	239	181	14	7
	128	190	176	13	6
512	16	194	189	17	4
	32	197	191	9	5
	64	197	201	14	8
	128	229	194	15	4
	256	196	189	8	3
1024	16	188	177	4	4
	32	182	172	5	2
	64	175	175	4	6
	128	181	170	6	7
	256	178	174	3	6
	512	170	174	4	2
disabled ARQ		165	162	236	227

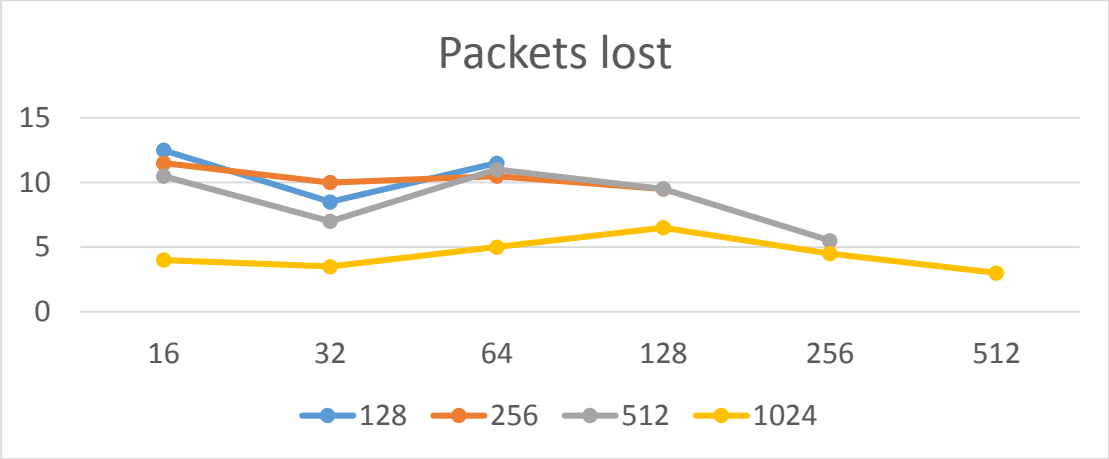
Χωρίς ARQ

- RTT: 163.5
- Σύνολο πακέτων: 1660
- Χαμένα πακέτα: 231.5

Με ARQ (RTT)



Με ARQ (Packets lost)



Συμπεράσματα

Στο RTT φαίνεται ότι οι χρόνοι αυξάνονται από τη στιγμή που ενεργοποιείται το ARQ σε σχέση με όταν είναι απενεργοποιημένο το οποίο είναι λογικό καθώς γίνεται προσπάθεια επανεκπομπής όταν ένα πακέτο χαθεί. Επίσης, όσο αυξάνεται το μέγεθος του window οι χρόνοι του RTT ανταποκρίνονται πιο ομαλά στην αλλαγή του μεγέθους του block.

Όσο αυξάνεται το μέγεθος του window το ARQ πρωτόκολλο είναι πιο αξιόπιστο και ο αριθμός των χαμένων πακέτων μειώνεται. Ακόμη, κάθε μέγεθος window επιτυγχάνει τη καλύτερη απόδοση σε διαφορετικό συνδυασμό σε σχέση με το μεγέθους του block. Συνήθως αυτό ο συνδυασμός είναι όταν το μέγεθος του block είναι ίσο με το μέγεθος του window/2.

Γενικά, το ARQ πρωτόκολλο είναι πιο αποτελεσματικό, δηλαδή πιο αξιόπιστο και πιο γρήγορο, με το μεγαλύτερο μέγεθος του window και το μεγαλύτερο μέγεθος του block αν και σε κάποια σημεία οι διαφορές είναι εξαιρετικά μικρές.

Αναφορές

- en.wikipedia.org/wiki/Automatic_repeat_request
- wimax.orbit-lab.org/wiki/WiMAX/17/00bwireless
- wimax.orbit-lab.org/wiki/WiMAX/17/01arq
- wimax.orbit-lab.org/attachment/wiki/WiMAX/30/07/iperf.rb
- dspace.mit.edu/bitstream/handle/1721.1/76816/825563806.pdf