ARQ

Automatic Repeat reQuest

Kostas Tseas

Manos Katsomallos

Nikolas Katsioulas

ARQ

Automatic Repeat Query

ARQ

Automatic Repeat reQuest (ARQ), also known as Automatic Repeat Query, is an error-control method for data transmission that uses acknowledgements (messages sent by the receiver indicating that it has correctly received a data frame or packet) and timeouts (specified periods of time allowed to elapse before an acknowledgment is to be received) to achieve reliable data transmission over an unreliable service. If the sender does not receive an acknowledgment before the timeout, it usually re-transmits the frame/packet until the sender receives an acknowledgment or exceeds a predefined number of re-transmissions.

ARQ Protocols

ARQ protocols reside in the Data Link or Transport layer of the OSI model (Open Systems Interconnection).

- Stop-and-wait ARQ
- Go-Back-N ARQ
- Selective Repeat ARQ

All three protocols usually use some form of sliding window protocol to tell the transmitter to determine which (if any) packets need to be retransmitted.

Stop-and-wait ARQ

Stop-and-wait ARQ is a method used in telecommunications to send information between two connected devices. It ensures that information is not lost due to dropped packets and that packets are received in the correct order. It is the simplest kind of automatic repeat-request (ARQ) method.

- A stop-and-wait ARQ sender sends one frame at a time, it is a special case of the general sliding window protocol with both transmit and receive window sizes equal to 1.
- After sending each frame, the sender doesn't send any further frames until it receives an acknowledgement (ACK) signal.
- After receiving a good frame, the receiver sends an ACK.
- If the ACK does not reach the sender before a certain time, known as the timeout, the sender sends the same frame again.

Go-Back-N ARQ

Go-Back-N ARQ is a specific instance of the ARQ protocol, in which the sending process continues to send a number of frames specified by a window size even without receiving an acknowledgement (ACK) packet from the receiver. It is a special case of the general sliding window protocol with the transmit window size of N and receive window size of 1.

- The receiver process keeps track of the sequence number of the next frame it expects to receive, and sends that number with every ACK it sends.
- The receiver will discard any frame that does not have the exact sequence number it expects and will resend an ACK for the last correct in-order frame.
- Once the sender has sent all of the frames in its window, it will detect that all of the frames since the first lost frame are outstanding, and will go back to sequence number of the last ACK it received from the receiver process and fill its window starting with that frame and continue the process over again.

Selective Repeat ARQ

Selective Repeat ARQ is a specific instance of the ARQ protocol used to solve sequence number dilemma in communications. Selective Repeat is one of the automatic repeat-request (ARQ) techniques.

- With selective repeat, the sender sends a number of frames specified by a window size even without the need to wait for individual ACK from the receiver as in Go-back N ARQ.
- However, the receiver sends ACK for each frame individually, which is not like cumulative ACK as used with go-back-n.
- The receiver accepts out-of-order frames and buffers them. The sender individually retransmits frames that have timed out.

About the project

Issues (1)

Our initial target was to experiment with all 3 aforementioned ARQ protocols.

- We can not change the ARQ protocol of the base station.
- We worked only with the protocol provided by the WiMAX 802.16e base station and observed the alternations in the RTT and the number of lost packets by changing the block and window sizes.

Issues (2)

- We executed the experiment with the default settings but we did not observe any differences in the RTT and lost packets.
- In order to cause some variation in our measurements, we tweaked the power of the base station and the MCS. We tried and concluded to the following values:
  - `bstxpow = 21`
  - `mcsProfile?dl = 20 (64 QAM 3/4)`

Issues (3)

It was quite difficult to get reliable results and maintain the execution conditions of the experiment constant.

- Preparation and execution time ~ 10 minutes.
- bstxpw and dl parameter tweaks leaded to connection failures.
- Often interruption to node and testbed availability.

Experiment

- iperf:

```
// produce UDP packets
app.setProperty('udp', true)
// pause for 1 second between reports
app.setProperty('interval', 1)
// set buffer to 600KB
app.setProperty('len', 600)
// set bandwidth to 80Kbps
app.setProperty('bandwidth', 80000)
// set iperf broadcast time to 50sec
app.setProperty('time', 50)
```

- ping:

```
// set ping count to 100
app.setProperty('count', 100)
```

Preparation

- Load disk image: wmxlab-qos.ndz
- Nodes: 6, 7
- Set bs power: 20
- Set MCS profile: 21

Procedure

- 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}

Commands

- Base station power:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/wireless/set?bstxpow=21"
  ```
- MCS profile:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/mcsProfile?dl=20"
  ```
- ARQ enable/disable:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?enable=true/false"
  ```
- Window size:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?wsize={0-1024}"
  ```
- Block size:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/arq/set?bloksize={0-
  wsize/2}"
  ```
- Base station reboot:
  ```
  wget -qO- "http://wimaxrf:5052/wimaxrf/bs/restart"
  ```

Results

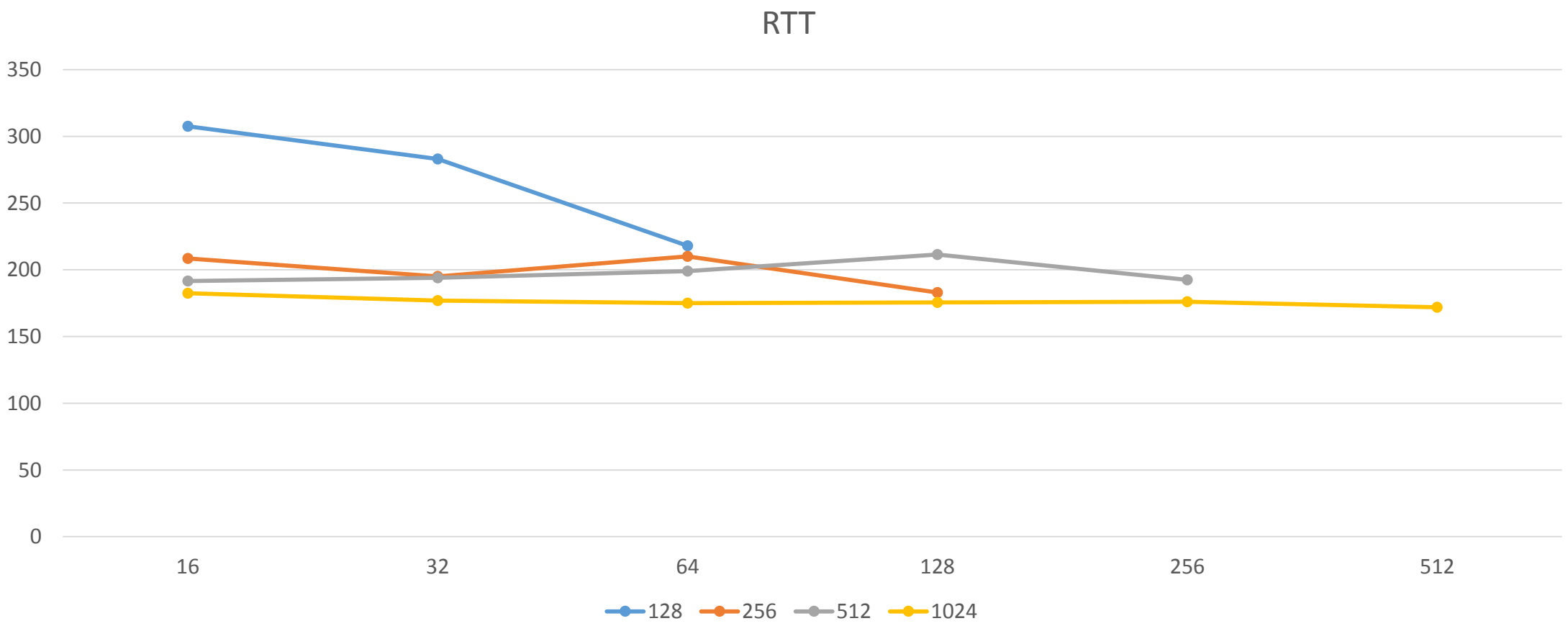
Results

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

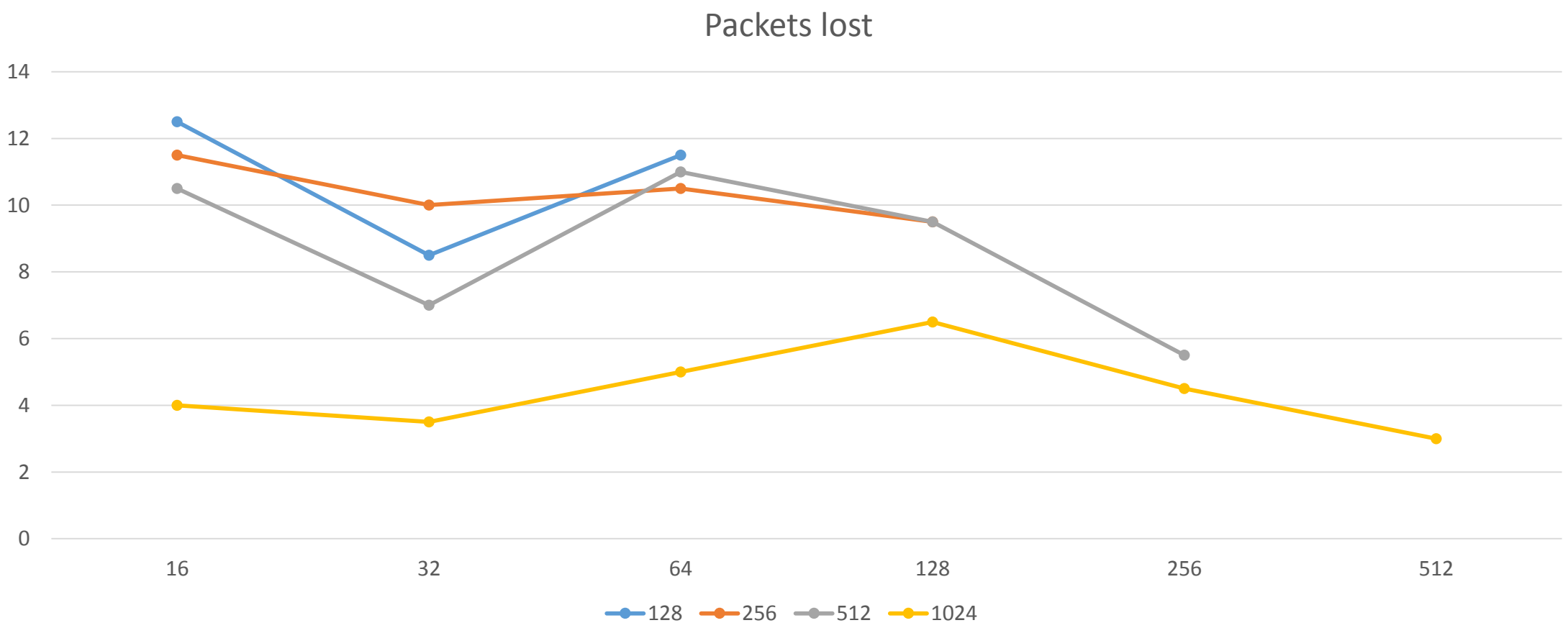
ARQ disabled

- RTT: 163.5
- Total packets: 1660
- Packet lost: 231.5

RTT (ARQ enabled)



Packets lost (ARQ enabled)



Conclusions

- RTT increases when ARQ is enabled.
- While the window size is increased, the RTT responds smoothly to the change of block size.
- While the window size is increased, ARQ is more reliable.
- Every window size has different combination of block size for better efficiency of ARQ.
- Generally, ARQ protocol is more effective for larger window block and sizes.

References

- 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

Questions

?