TAROC: TCP-Aware RObust Header Compression Scheme

December 13, 2000

HongBin Liao, Qian Zhang, Wenwu Zhu, Ya-Qin Zhang

Microsoft Research China

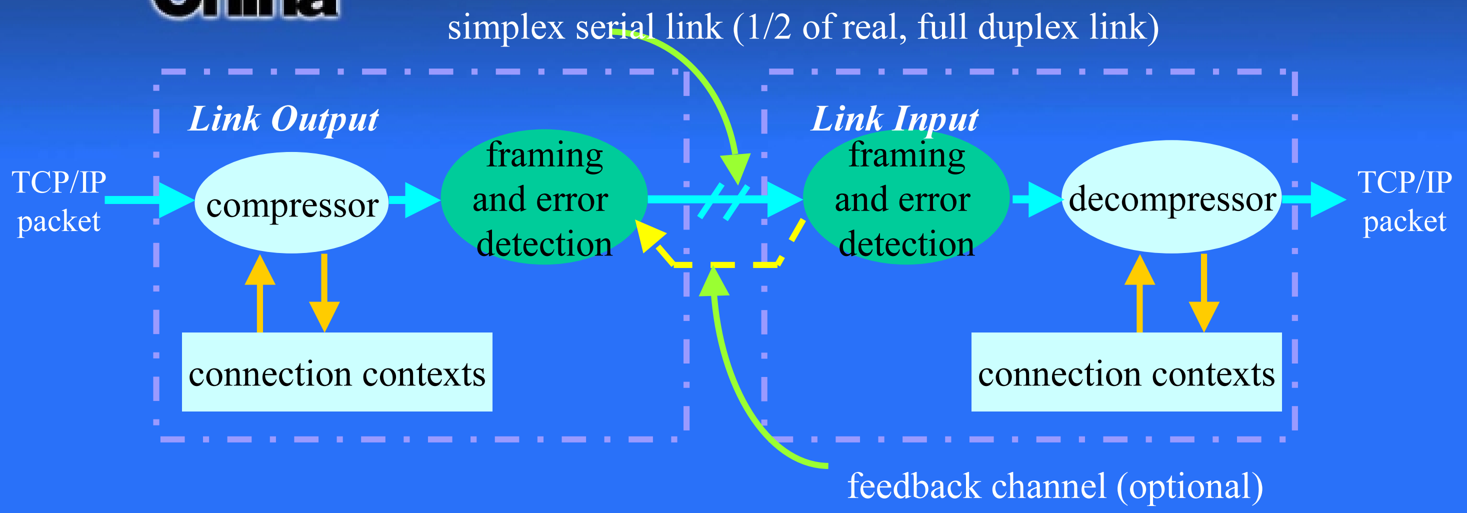
Outline

- Introduction
- Header compression/ decompression model
- Basic concepts of TAROC
- Details of TAROC
  - robustness of TAROC (window-based LSB encoding)
  - efficiency of TAROC (TCP congestion window tracking)
  - behaviors in compressor and decompressor
- Compression performance analysis

Introduction

- Related works
  - RFC 1144 (VJHC)
    - delta coding is not robust against error
    - needs TIME-OUT to get re-synchronization
  - RFC 2507 (IPHC)
    - twice: rely on the fixed delta
      - not suited with Timestamp and SACK options
      - not suited with burst loss that may happen before decompressor
    - header request: feedback channel is needed, not efficient with long RTT channel

Compression/decompression model of TAROC



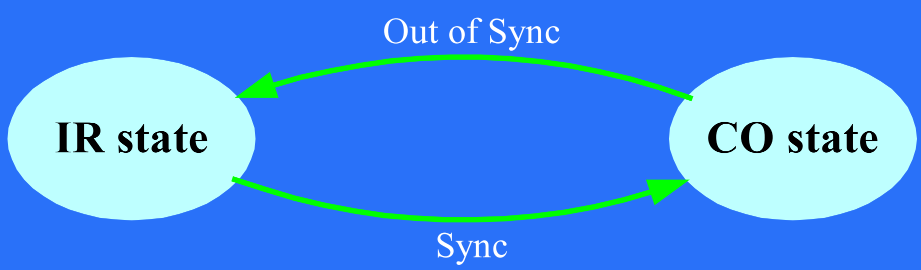
- Key ideas of TAROC
  - Window-based LSB encoding
  - TCP-congestion window tracking
    - Slow-start, Congestion-avoidance
    - Fast-retransmission, Fast-recovery

Basic concepts of TAROC (1)

- TAROC: TCP-Aware RObust header Compression
- Compressor starts from lower compression state and progressively transitions to higher compression state
- Compressor states:
  - IR (Initialization & Recovery):

    compressor sends full headers with certain pattern to ensure the consistency of the connection context on both side
  - CO (COmpression)

    higher compression state



Basic concepts of TAROC (2)

Microsoft Research China

- Higher compression ratio can be achieved only if the compressor has enough confidence that the decompressor has correctly received the info
- Confidence is not obtained by acknowledgments from the decompressor
- Rather, it is obtained from the TCP congestion control window

Window-based LSB encoding (1)

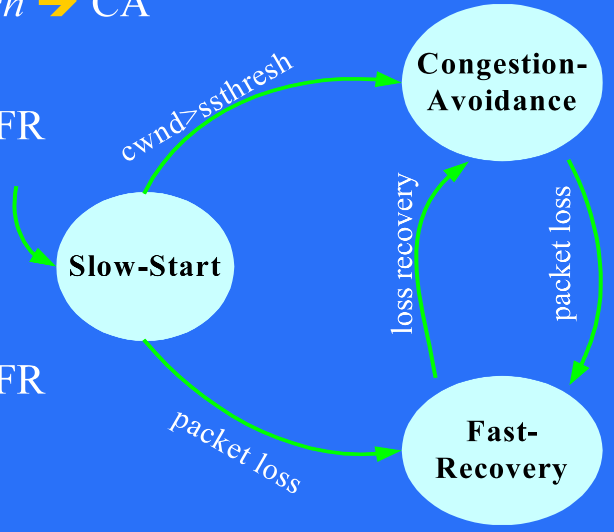
- Compressor maintains values of previous packets in window *VSW*
- Compressor sends *k* least significant bits of original value
  - *k* is smallest value to ensure correct decompression
  - *k* is determined by the length of *VSW*
- Robustness against errors
  - compressor chooses *k* so that no matter what value in *VSW* is chosen as reference by the decompressor, decompression is correct
- The length of *VSW* can be shrunk
  - once compressor knows that at least one packet in *VSW* was received
  - no feedback channel: the length of *VSW* can be controlled by TCP congestion window size

Window-based LSB encoding (2)

- Reference packet selection in decompressor
  - reference may be the one which is the last received non-retransmitted value, or
  - uncompressed value that passes the TCP checksum successfully
- Whether a compressed or uncompressed value is sent, this value should be added into *VSW*
- Applicable fields
  - TCP Sequence Number/ Acknowledge Number / Window
  - IP Packet ID
  - TCP Timestamp option

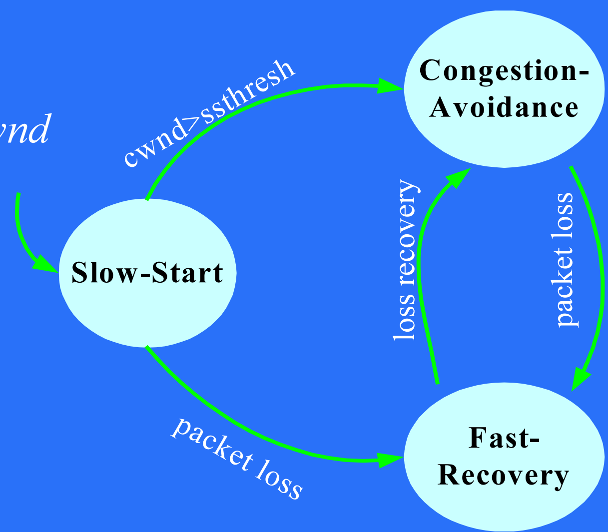
Tracking-based TCP congestion window estimation (1)

- Reconstruct the congestion control behavior of TCP sender at the compressor
- Adjust congestion window size in each state based on seqno
  - Slow-Start (SS)
    - – new ordered segment: *cwnd* += *d*, if *cwnd>ssthresh* ➔ CA
    - – out-of-order segment: ignore it
    - – retransmission segment: set *cwnd* and *ssthresh* ➔ FR
  - Congestion-Avoidance (CA)
    - – new ordered segment: *cwnd* += *d*MSS/cwnd*
    - – out-of-order segment: ignore it
    - – retransmission segment: set *cwnd* and *ssthresh* ➔ FR
  - Fast-Recovery (FR)
    - – new segment indicating recovery of loss ➔ CA



Tracking-based TCP congestion window estimation (2)

- Adjust congestion window size in each state based on ackno
  - Slow-Start (SS)
    - – new ordered ack: ndupacks = 0, *cwnd* += *d*, if *cwnd*>*ssthresh* ➔ CA
    - – duplicate ack: ndupacks ++,
      if ndupacks ≥ 3, set *cwnd* and *ssthresh* ➔ FR
    - – otherwise, ndupacks = 0
  - Congestion-Avoidance (CA)
    - – new ordered ack: ndupacks = 0, *cwnd* += *d*MSS/cwnd*
    - – duplicate ack: ndupacks ++,
      if ndupacks ≥ 3, set *cwnd* and *ssthresh* ➔ FR
    - – otherwise, ndupacks = 0
  - Fast-Recovery (FR)
    - – new segment indicates recovery of loss:
      ndupacks = 0 ➔ CA
    - – otherwise, ndupacks = 0



Compressor behavior

- Initialization & Recovery (IR) State
  - two types of packets in this state
    - full header packet + compressed packet
  - send full header packet if
    - it satisfies slow-start pattern or it is a retransmitted packet in *VSW*
  - add packet into *VSW*
  - tracking congestion window
  - adjust the length of *VSW* if possible

$$|VSW| \le K \times MAX(MAX(cwnd\_seq, 2 \times ssthresh\_seq), MAX(cwnd\_ack, 2 \times ssthresh\_ack))$$

F_PERIOD ($2^n \geqslant cwnd$)

$(1+2^0+1+2^1+1+2^2+\ldots+1+2^{n-1}=2^n+n-1)>cwnd$

Compressor behavior

Microsoft Research China

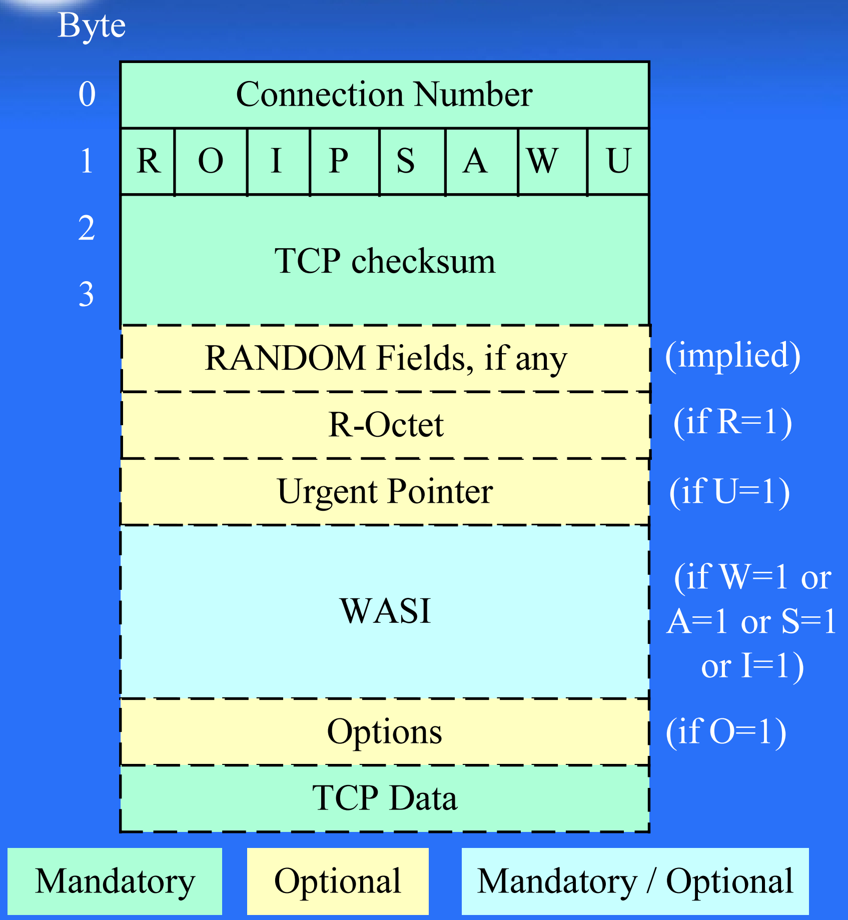
- COmpression (CO) state
  - only compressed packet in this state
  - compress packet
  - add packet into *VSW*
  - tracking congestion window
  - adjust the length of *VSW* if possible

$$|VSW| \le K \times MAX(MAX(cwnd\_seq, 2 \times ssthresh\_seq), MAX(cwnd\_ack, 2 \times ssthresh\_ack))$$

Decompressor behavior

- Reference packet selection in decompressor
  - reference may be the one that is the last received non-retransmitted value, or
  - uncompressed value that passes the TCP checksum successfully
- Decompressed value determination
  - de-compressor chooses the one, which is closest to reference and whose *k* LSBs equal the compressed value that has been received

Compressed TCP/IP header structure



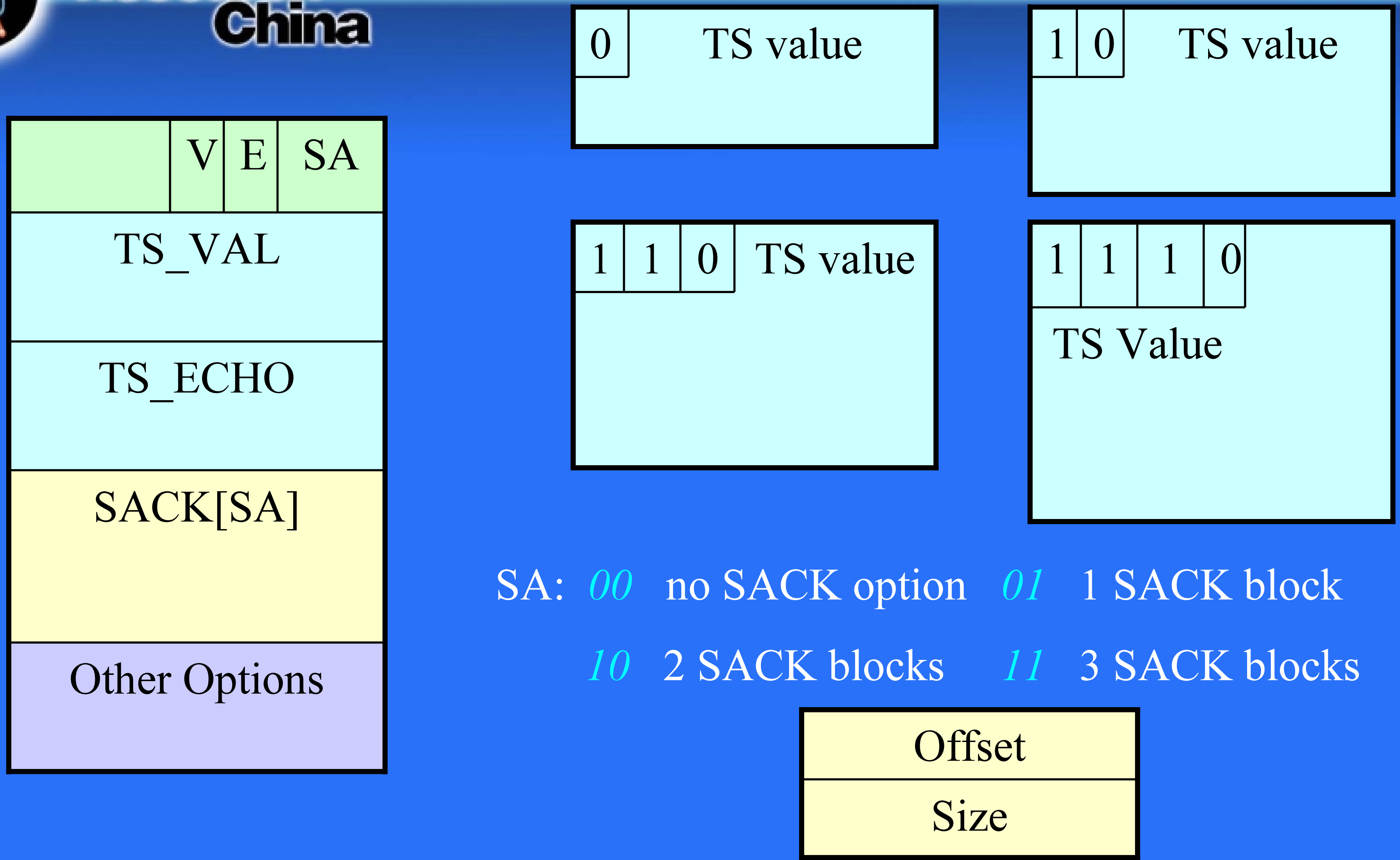
- Almost the same as the one in IPHC except for
  - Window, Acknowledgement Number, Sequence Number, IP-ID and Options fields
- W-LSB encoding applied to
  - Window, Acknowledgement Number, Sequence Number and IP-ID
  - TS value, TS echo reply in TCP Timestamp option
- Delta coding applied to
  - SACK option

WASI field

PT	C_WIN	C_ACK	C_SEQ	C_IP_ID	PADDING

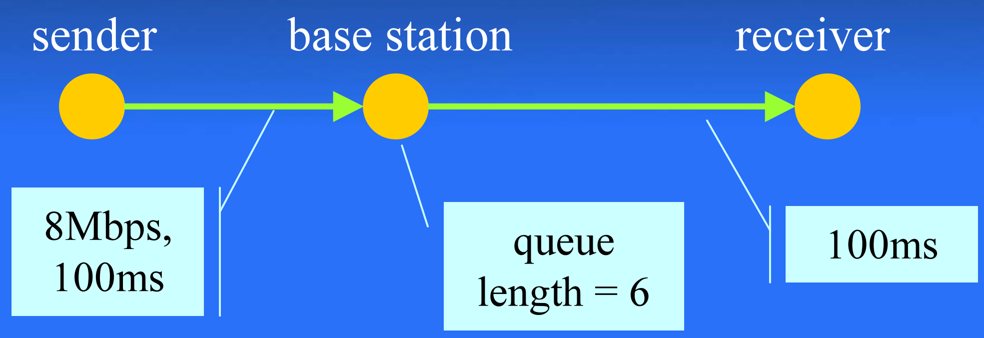
PT value	C_WIN length (bits)	C_ACK length (bits)	C_SEQ length (bits)	C_IP_ID length (bits)	length of WASI (bits)
00	10	11	11	3	2~5
01	14	15	15	7	3~7
10	16	19	19	11	4~9
11	16	23	23	15	5~11

Options field (Timestamp and SACK)



Performance analysis

- Simulation topology



- Tested header compression schemes
  - – no header compression (None)
  - – RFC 1144 (VJHC)
  - – RFC 2507 (IPHC)
  - – our scheme (TAROC)
  - – no error happens and has minimal header size (4) (Ideal)
- Bit error rate in wireless channel (10^{-9} to 10^{-3})
- Bandwidth in wireless channel (9.6Kbps to 384Kbps)
- TCP versions: Tahoe, Reno, Sack

Performance comparison: 9.6kb/s for Sack

BER	MTU (Byte)	Performance	None	VJHC	IPHC	TAROC	Ideal
10^{-7}	576	Throughput (Byte/s)	1115	1176	1175	1184	1190
		Improvement Percentage (%)	0	5.47	5.38	6.19	6.73
10^{-6}	576	Throughput (Byte/s)	1112	1131	1140	1179	1185
		Improvement Percentage (%)	0	1.71	2.52	6.1	6.73
10^{-5}	296	Throughput (Byte/s)	976	859	929	1100	1122
		Improvement Percentage (%)	0	-11.99	-4.82	12.7	14.96
10^{-4}	168	Throughput (Byte/s)	342	218	250	459	470
		Improvement Percentage (%)	0	-36.26	-26.9	34.21	37.43
$3*10^{-4}$	96	Throughput (Byte/s)	78	71	78	153	168
		Improvement Percentage (%)	0	-8.97	0	96.15	115.38
10^{-3}	96	Throughput (Byte/s)	1	1	2	7	9
		Improvement Percentage (%)	0	0	0	600	800

Performance comparison: *64kb/s for Reno*

BER	MTU (Byte)	Performance	None	VJHC	IPHC	TAROC	Ideal
10^{-7}	576	Throughput (Byte/s)	7274	7573	7483	7642	7704
		Improvement Percentage (%)	0	4.11	2.87	5.06	5.91
10^{-6}	576	Throughput (Byte/s)	6962	6577	6343	7316	7389
		Improvement Percentage (%)	0	-5.53	-8.89	5.08	6.13
10^{-5}	296	Throughput (Byte/s)	3399	2461	2297	3636	3665
		Improvement Percentage (%)	0	-27.6	-32.42	6.97	7.83
10^{-4}	168	Throughput (Byte/s)	455	300	322	595	607
		Improvement Percentage (%)	0	-34.07	-29.23	30.77	33.41
$3*10^{-4}$	96	Throughput (Byte/s)	91	84	85	176	186
		Improvement Percentage (%)	0	-7.69	-6.59	93.41	104.4
10^{-3}	96	Throughput (Byte/s)	1	1	2	9	10
		Improvement Percentage (%)	0	0	100	800	900

Performance comparison: 114kb/s for Sack

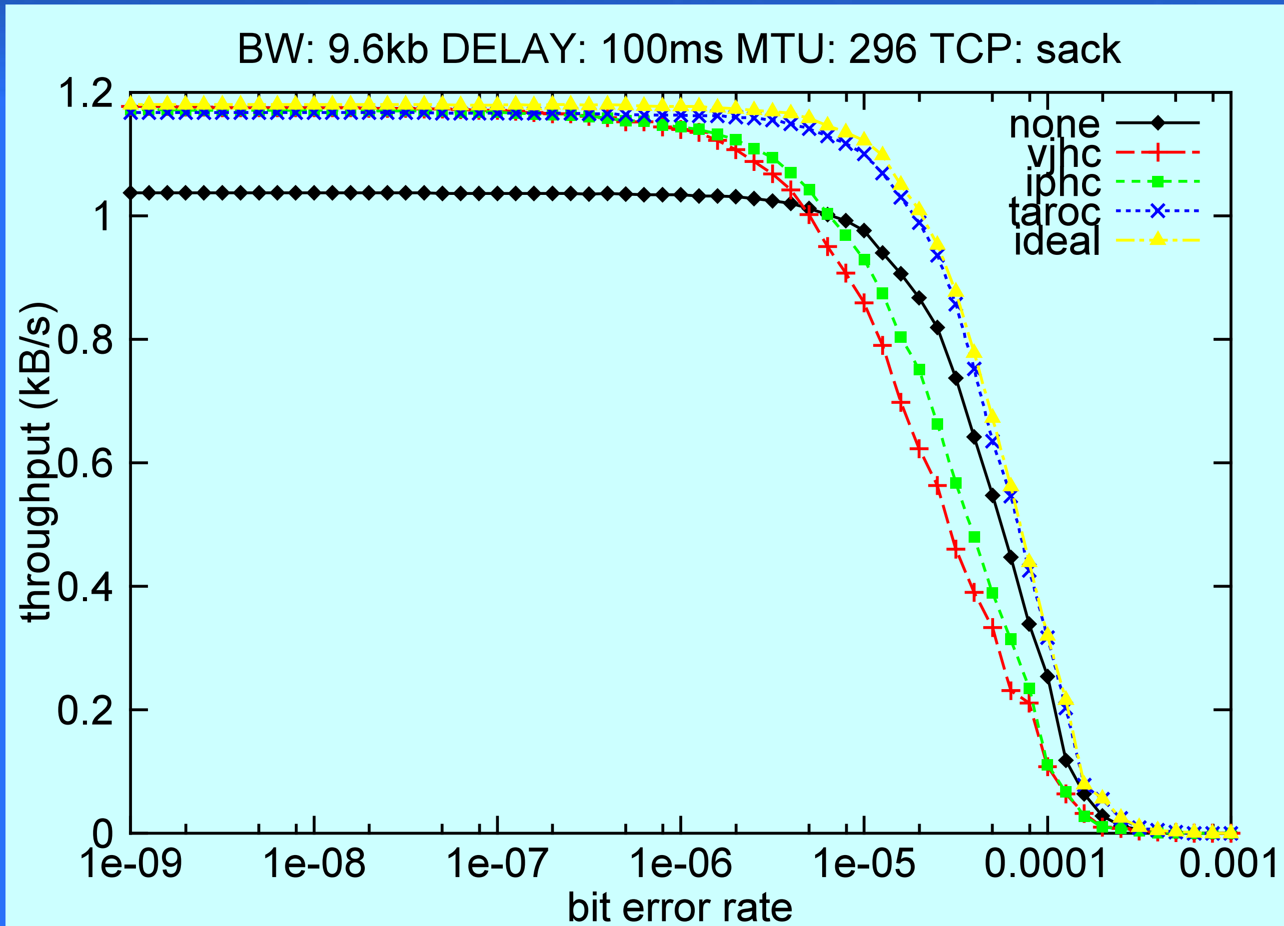
Microsoft Research China

BER	MTU (Byte)	Performance	None	VJHC	IPHC	TAROC	Ideal
10^{-7}	576	Throughput (Byte/s)	12004	12291	12127	12515	12557
		Improvement Percentage (%)	0	2.39	1.02	4.26	4.61
10^{-6}	576	Throughput (Byte/s)	11204	9806	9160	11724	11760
		Improvement Percentage (%)	0	-12.48	-18.24	4.64	4.96
10^{-5}	296	Throughput (Byte/s)	3936	2611	2599	4225	4239
		Improvement Percentage (%)	0	-33.66	-33.97	7.34	7.7
10^{-4}	168	Throughput (Byte/s)	492	308	339	645	657
		Improvement Percentage (%)	0	-37.4	-31.1	31.1	33.54
$3*10^{-4}$	96	Throughput (Byte/s)	93	84	89	183	196
		Improvement Percentage (%)	0	-9.68	-4.3	96.77	110.75
10^{-3}	96	Throughput (Byte/s)	1	1	2	9	10
		Improvement Percentage (%)	0	0	100	800	900

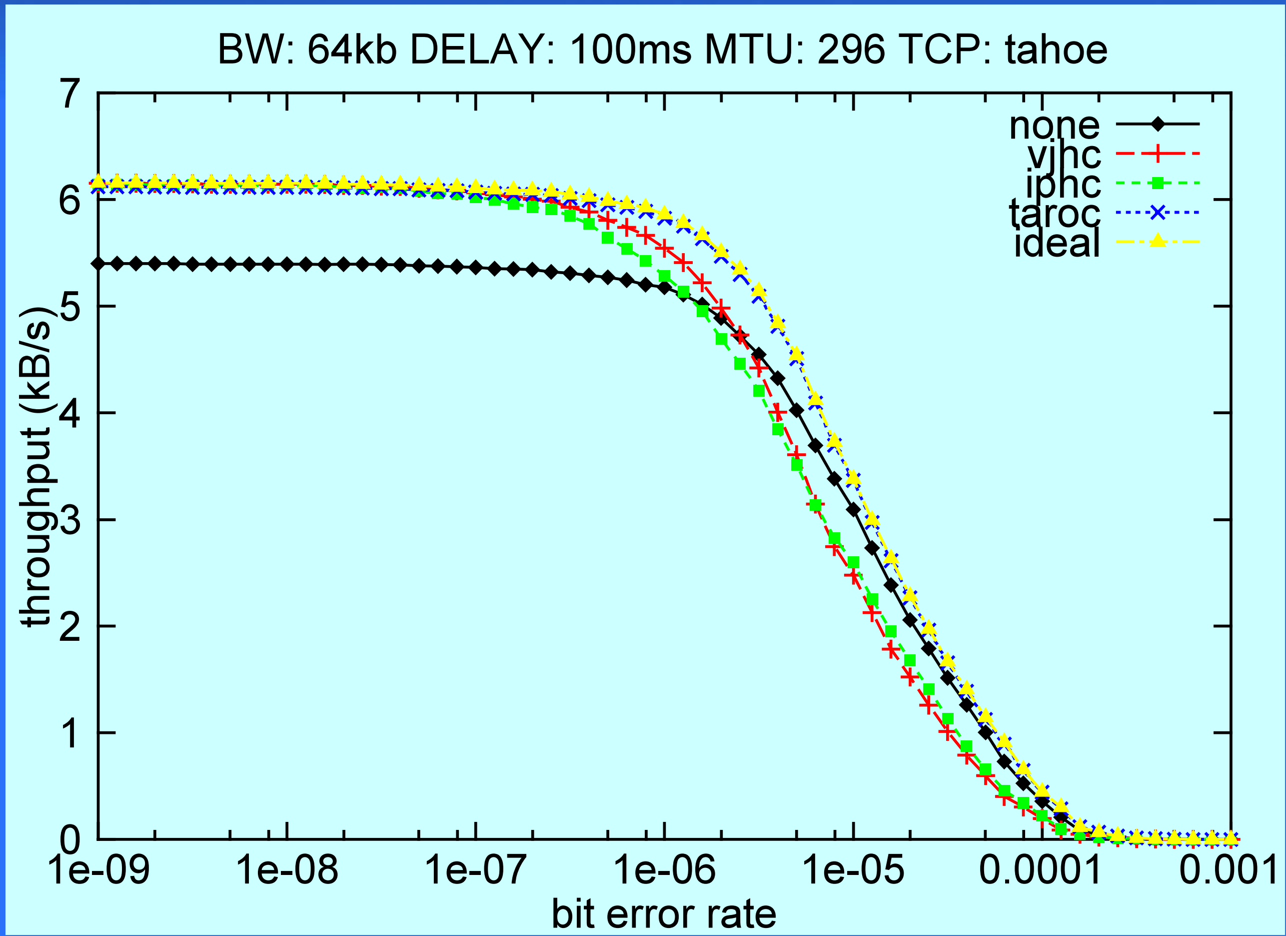
Performance comparison: *384kb/s for Tahoe*

BER	MTU (Byte)	Performance	None	VJHC	IPHC	TAROC	Ideal
10^{-7}	576	Throughput (Byte/s)	24650	24246	23702	24732	24753
		Improvement Percentage (%)	0	-1.64	-3.85	0.33	0.42
10^{-6}	576	Throughput (Byte/s)	16710	14725	13755	17148	17183
		Improvement Percentage (%)	0	-11.88	-17.68	2.62	2.83
10^{-5}	296	Throughput (Byte/s)	3548	2750	2927	3820	3835
		Improvement Percentage (%)	0	-22.49	-17.5	7.67	8.09
10^{-4}	168	Throughput (Byte/s)	493	324	363	625	636
		Improvement Percentage (%)	0	-24.28	-26.37	26.77	29.01
$3*10^{-4}$	96	Throughput (Byte/s)	92	85	89	184	195
		Improvement Percentage (%)	0	-7.61	-3.26	100	111.96
10^{-3}	96	Throughput (Byte/s)	1	1	2	9	11
		Improvement Percentage (%)	0	0	100	800	1000

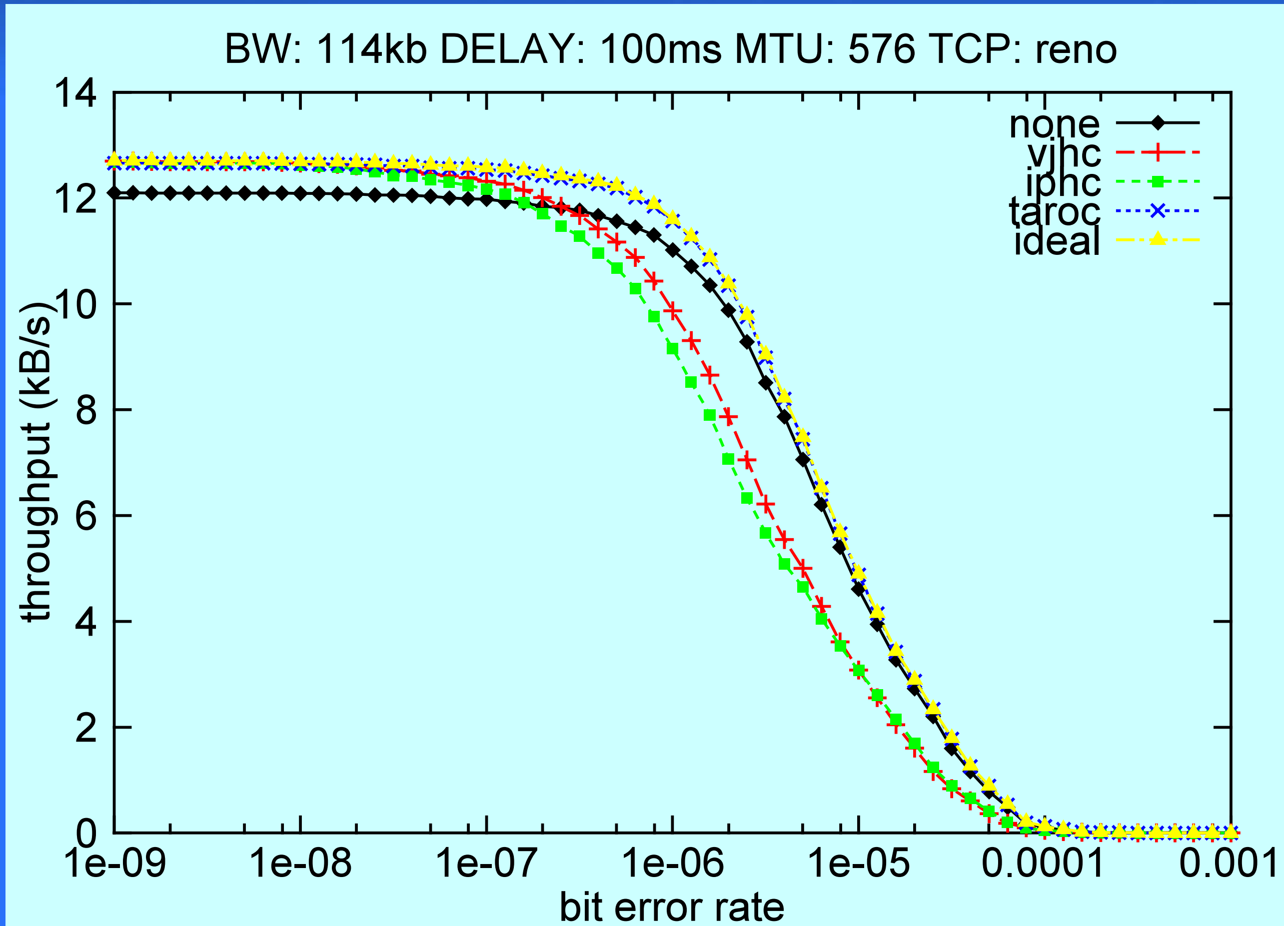
Performance comparison: *9.6kb/s for Sack*



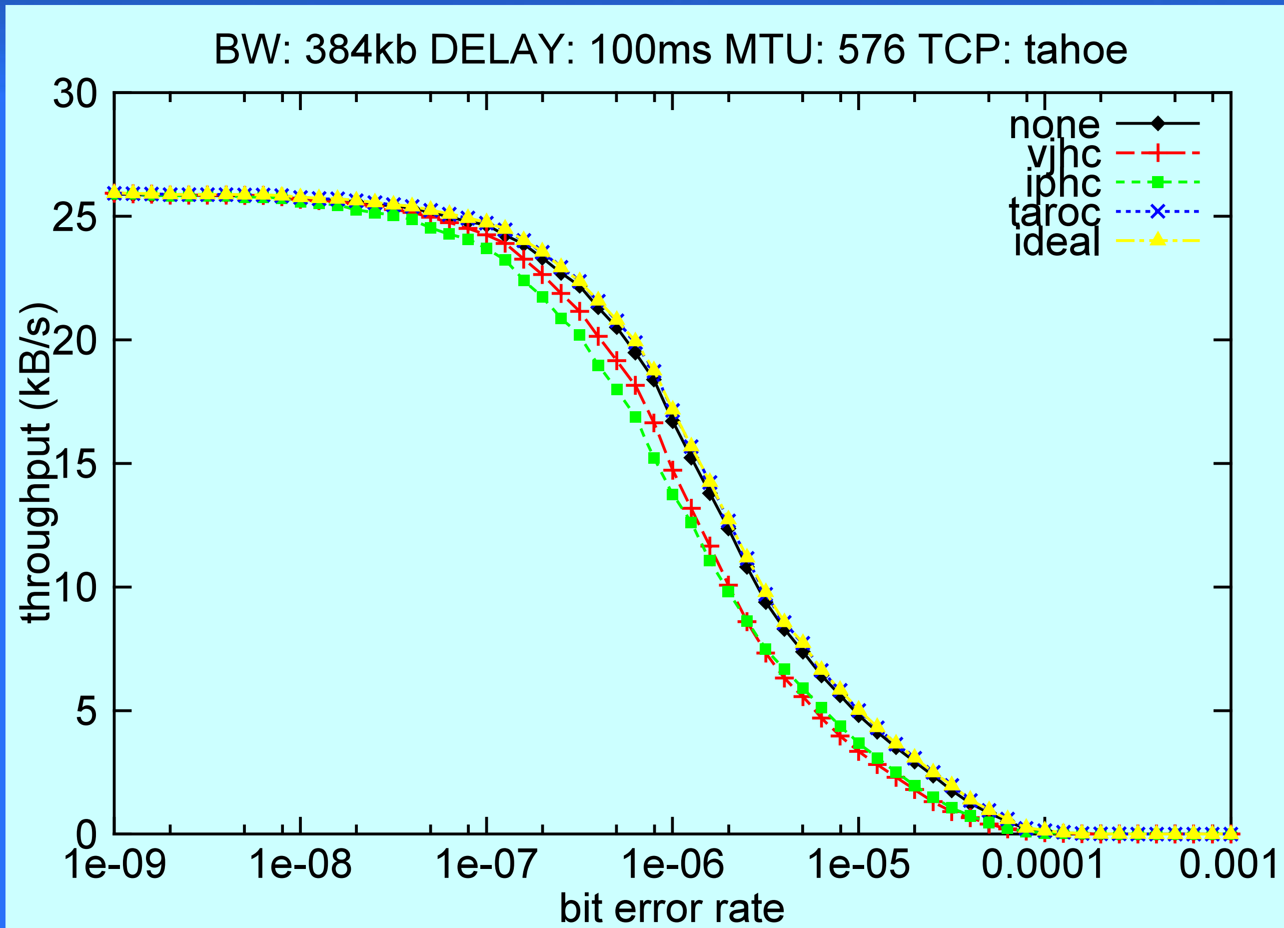
Performance comparison: *64kb/s for Tahoe*



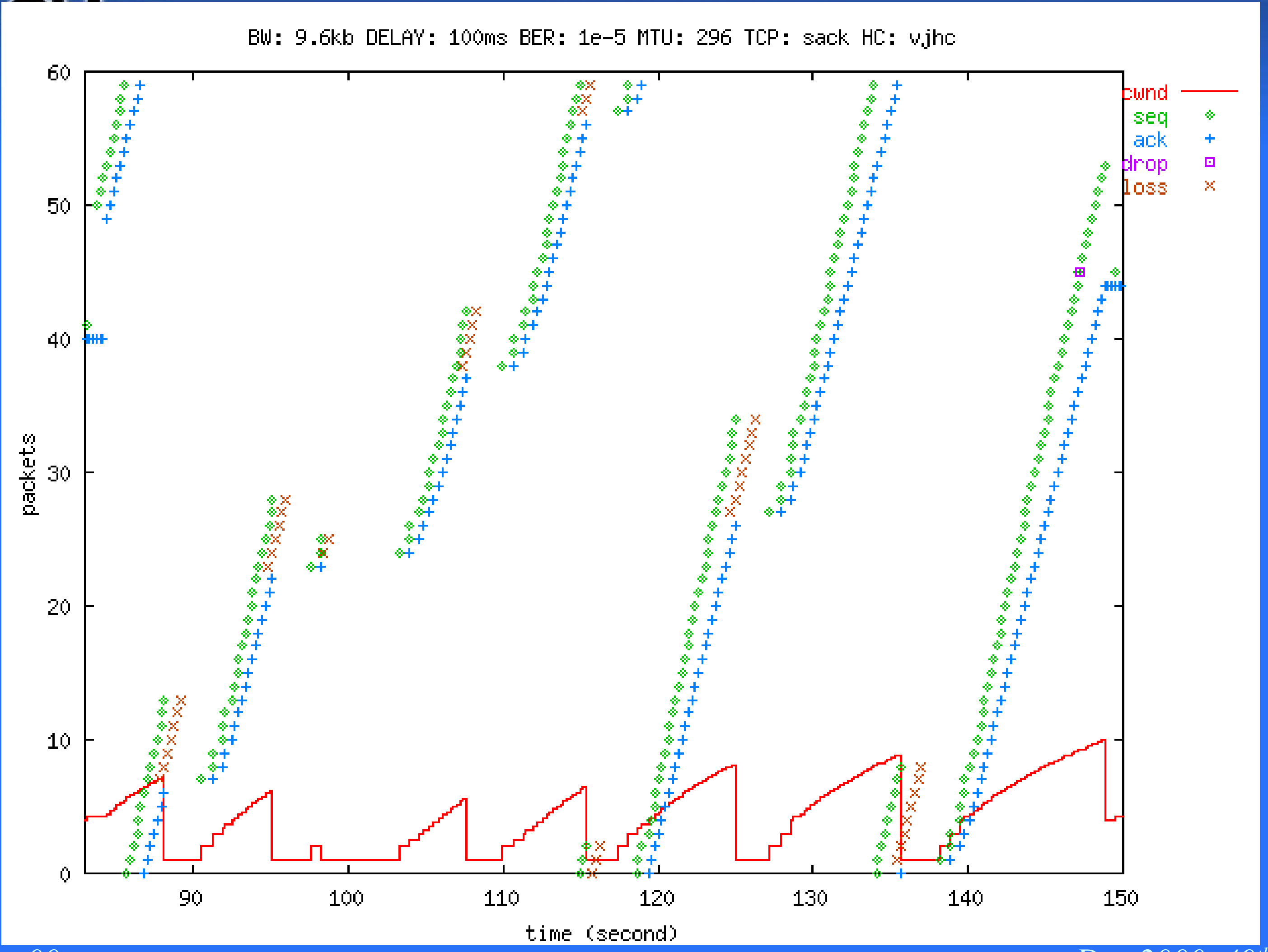
Performance comparison: *114kb/s for Reno*



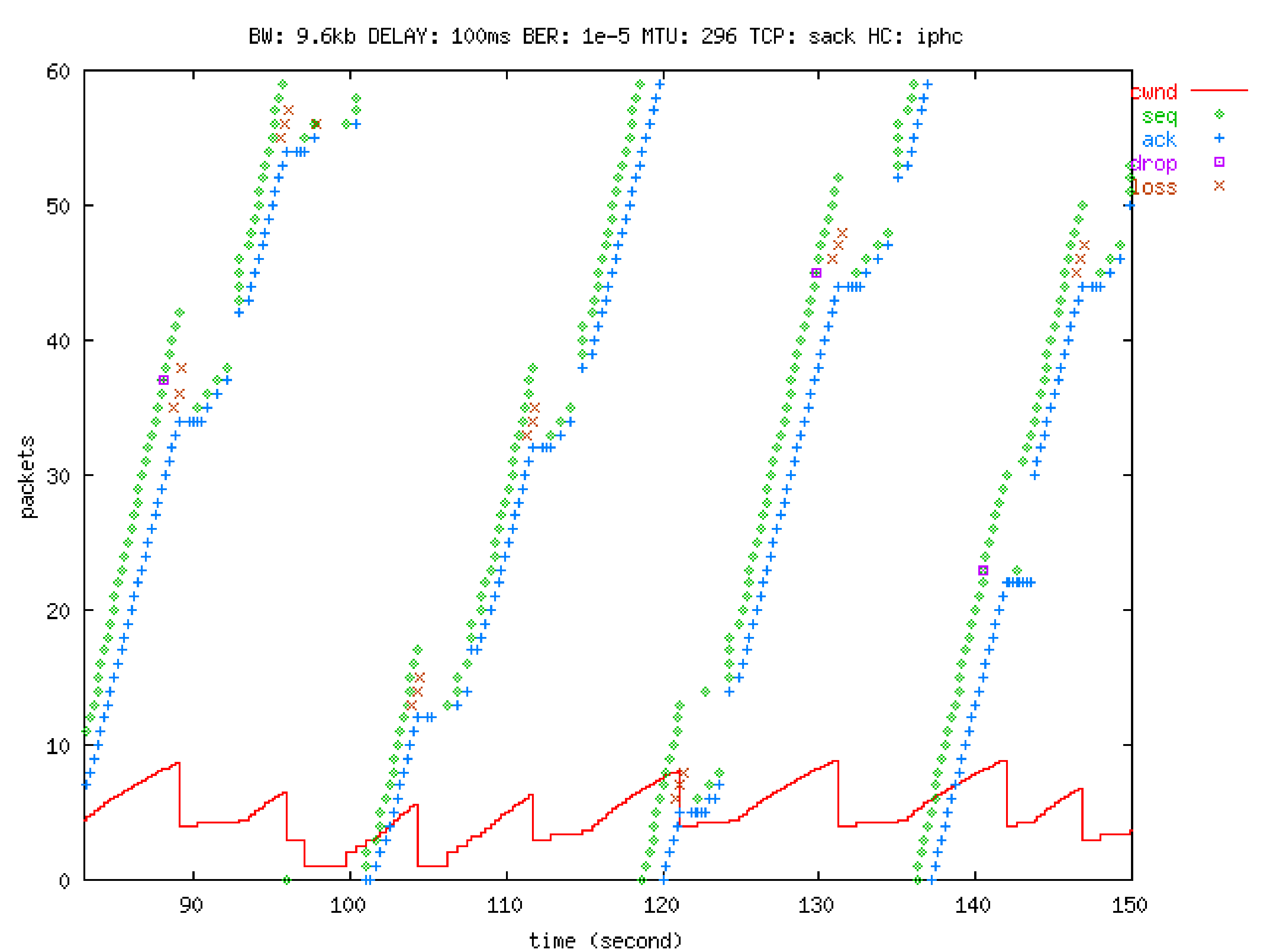
Performance comparison: 384kb/s for Tahoe



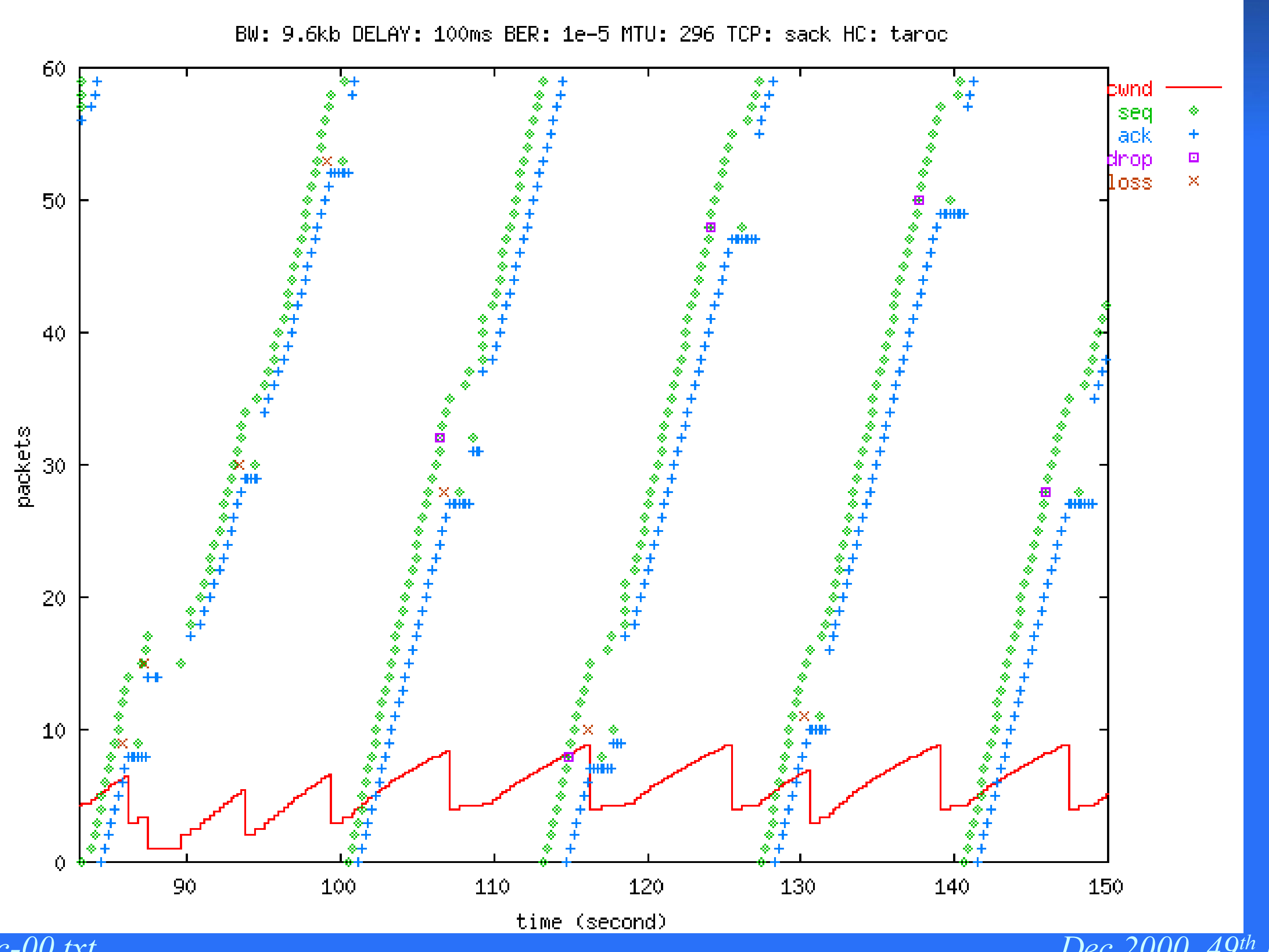
TCP behavior using RFC1144(VJHC)



TCP behavior using RFC2507(IPHC)

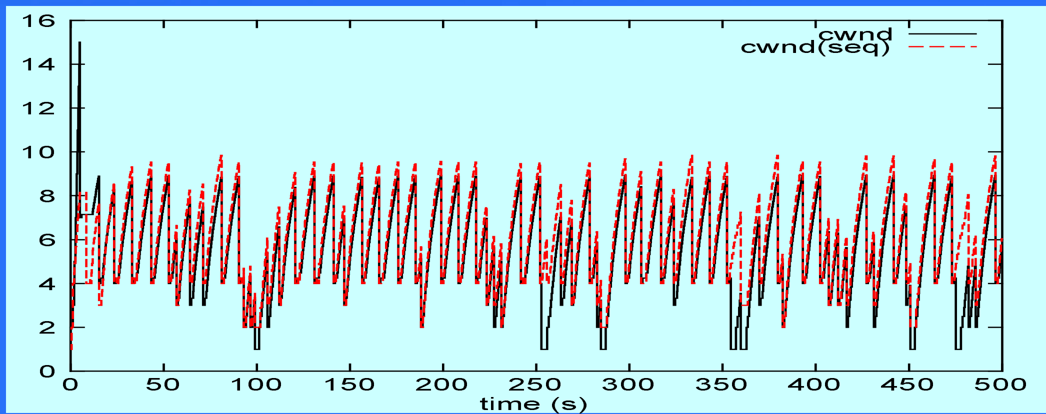


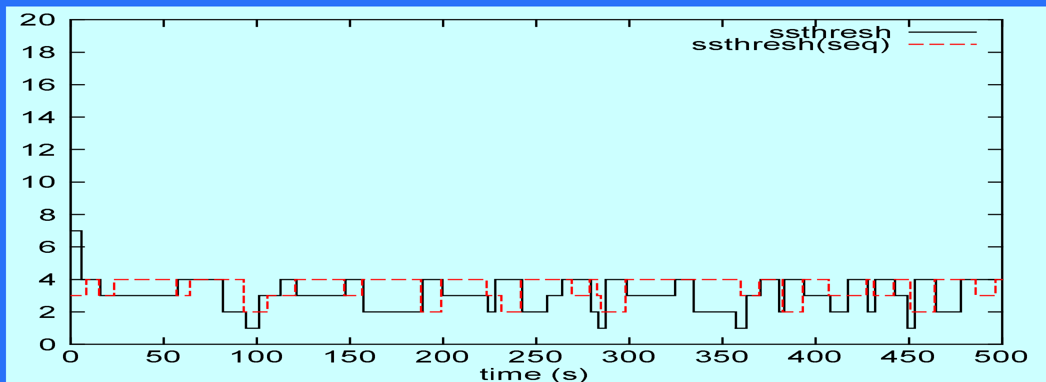
TCP behavior using TAROC



Accuracy of congestion window tracking

- Tracking *cwnd* and *ssthresh* for seqno

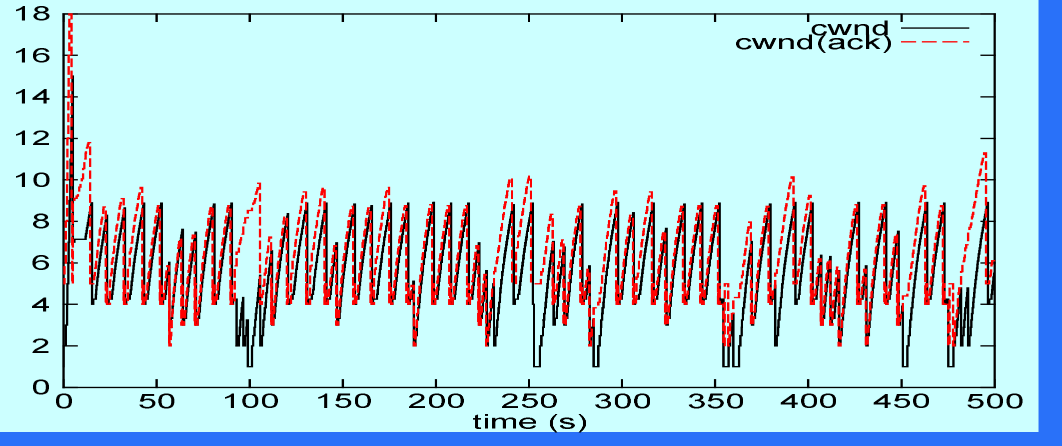


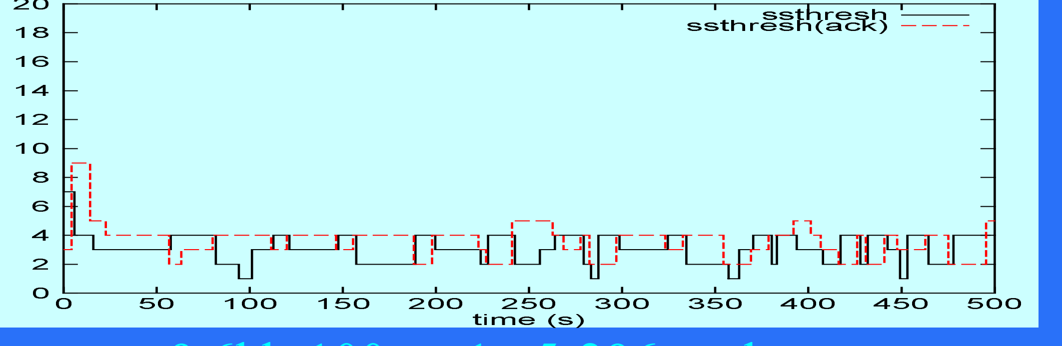


9.6kb-100ms-1e-5-296-sack

Accuracy of congestion window tracking

- Tracking *cwnd* and *ssthresh* for ackno





9.6kb-100ms-1e-5-296-sack

Conclusions

- Robustness to
  - packet loss and misordering before the compressor
  - packet loss between compressor and decompressor
- Resilient to handover
  - seamlessly run across compressor/decompressor entity relocation
- Accurate TCP congestion window tracking
- High compression efficiency
  - average overhead at or slightly above 6 byte per packet, across a wide range of error rates
- Sufficient simulations demonstrate effective of TAROC

Thanks for your attention!