ESN: Increasing TCP’s maximum window size

draft-bagnulo-tcpm-esn-00

M. Bagnulo & Y. Nishida

IETF100

Motivation

- TCP maximum window is determined by RCVWND + Window Scale Option
  - Max window is achieved with the maximum shift allowed by the WS option i.e. 14
    - **Max window is roughly 2^30 (1GByte)**
- Imposes an upper bound to TCP's maximum speed/bandwidth delay product
  - Example: with an RTT of 100 ms (frequent is intercontinental links), the max speed is 80Gbps, while 100Gbps technology is already available

Doubling Max window is easy!

- Window Scale maximum value of 14 is overly restrictive
  - Original motivation in RFC7323: distinguish “old” and “new” out of window segments
  - Not really necessary, only needed to determine out of window segments
- Simply allow WS value of 15 and obtain a maximum window of roughly 2^31 (2GBytes)
  - Few other tweaks to provide backward compatibility

Beyond 2^31 Max window

- If WS option is updated to allow values of 15 and higher, Max Window is limited by TCP’s sequence number
  - TCP seq number has 32 bits, so it limits the max window to 2^31.
- Max Window larger than 2^31 requires extending TCP seq number
  - Longer term solution
  - Requires more changes

Increasing TCP sequence number

- Overall approach: Carry a prefix for the sequence number in a TCP option.
- Which option:
  - Define a new option: see draft-looney-tcpm-64-bit-seqnos
  - Re-purpose the existent Time Scale option: draft-bagnulo-tcpm-esn

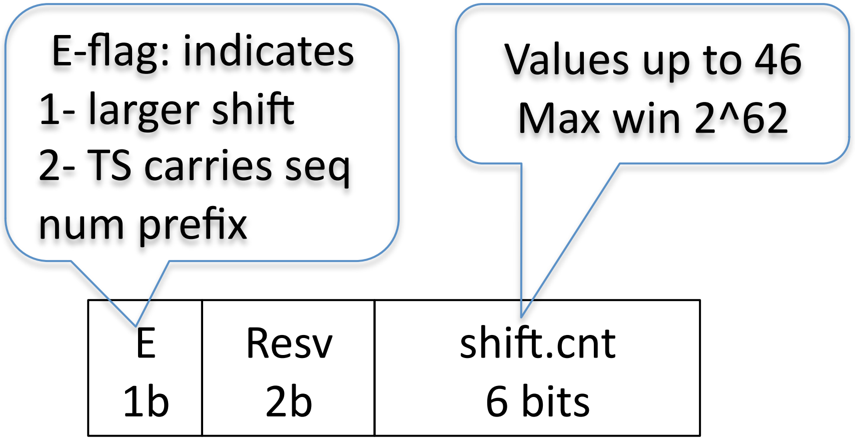
Motivation for using the TS option

- RFC7323 defines two uses for the TS option
  - RTT measurement
  - PAWS: this is unnecessary with a longer seq number
    - Re-purposing TS for Extended Sequence Number (ESN) would subsume PAWS
- Reduced option space consumption
  - TCP options limited to 40B
    - Critical in the SYN
    - Grace-full fallback: carrying TS in SYN allows use PAWS in case ESN is not supported
  - Deployability: unknown options are more likely to experience problems

Option	Bytes
MSS	2
SACK permit	2
WS	3
TS	10
TFO	6/18
Total	35

ESN mechanism

- Modify WS and TS
- WS modification



E-flag: indicates
1- larger shift
2- TS carries seq num prefix

Values up to 46
Max win 2^62

E 1b	Resv 2b	shift.cnt 6 bits

- TS modification
  - TS has 10B, 8 are used for TSval and Tsecr
  - Use 8 B to carry information about Sequence Number Prefix and ACK number prefix
    - Seq number composed as Prefix (carried in TS) + Seq Number (carried in TCP seq number field)
    - Same for ACKs.

RTTM

- Goal: preserve use of TS option for RTTM
- Option 1:

Kind=8	Length	F1	Tsval or Seq num Pref	F2	Tsecr or ACK Prefix

  - – Use two flags to indicate if the option carries Seq number/ACK prefixes or carries TS and TS echo values.
    - 63 bit sequence number
    - Sporadic use of RTTM
      - – Segments carrying RTTM info don't carry Prefix, so max flight-size is 2^32 when segments with RTTM info in flight!!!!
      - – In addition, potential problems with old dup packets

RTTM (2)

- Option 2: Always carry Seq num prefix, and either carry ACK prefix or Timestamp

Kind=8	Length	Res	Seq num Pref	F	Timestamp / ACK Pref

  - 62 bit seq number
  - 2-bit flag to indicate TSval, TSecr or ACK Pref
  - Some segments do not carry ACK Pref, but cumulative ACK should deal with this

RTTM (2)

- Option 3: Use 16 bit fields for Seq num prefix, ACK prefix, TSval and TSecr.

Kind=8	Length	Seq # Pref	ACK Pref	TSval	TSecr

  - – 48 bit seq number (140 TB max win, good enough?)
  - – 16 bits to encode timestamps
    - Variable precision encoding, see trammell-tcpm-timestamp-interval
    - Other option is to send either the TSval or the Tsecr with 32 bits, which may be ok.

Final thoughts

- Increasing RCVWND seems to be a current need in some scenarios. (Inter DC communications)
- Increasing seq number will take a while since it implies significant changes.
- Two step approach: first doubling (affects only WS) and then increasing seq number
  - May be worth figuring both of them now, as both can be used to increase the seq number

Thoughts?