Model Based Metrics

draft-ietf-ippm-model-based-metrics-00.txt

Matt Mathis
mattmathis@google.com
Al Morton
acmorton@att.com

IETF 87 IPPM
July 2013

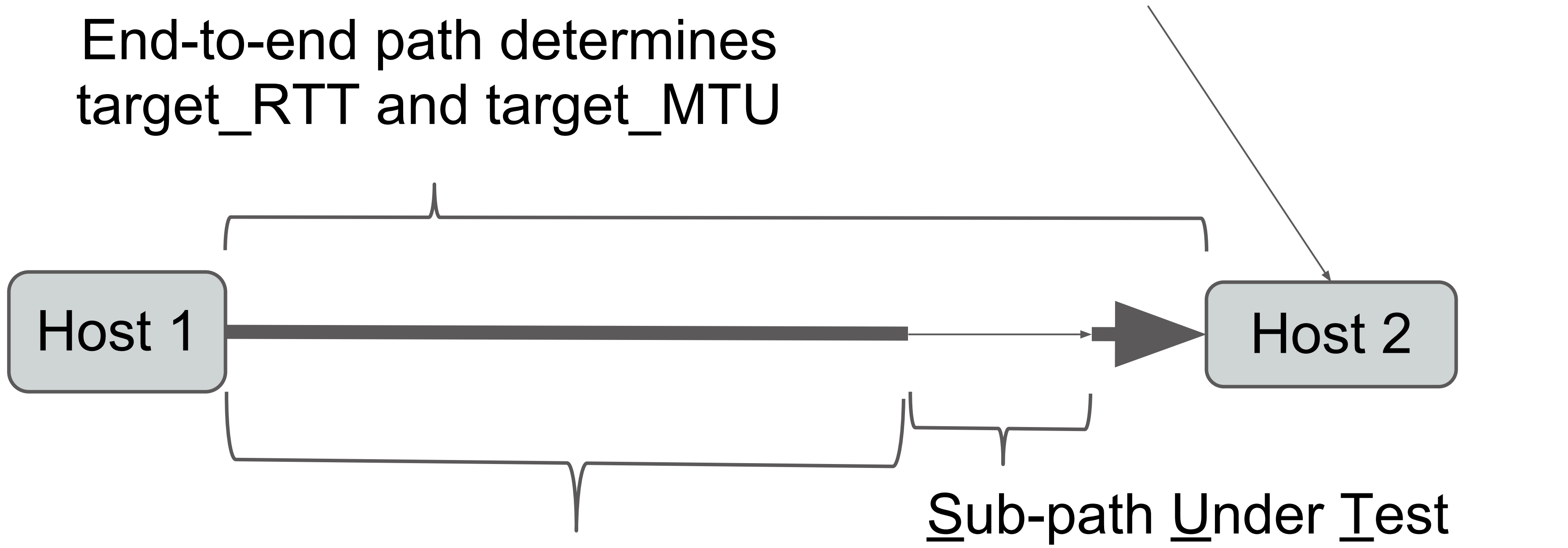
Bulk Transport Capacity testing is hard!

- TCP and all transports are complicated control systems
  - TCP causes self inflicted congestion
  - Governed by equilibrium behavior
  - Changes in one parameter are offset by others
- Every path component affects performance
  - even the end-hosts
- The Meta-Heisenberg problem
  - Difficult to assess cross-traffic effect on test traffic
  - Parallel TCP test connections have similar issue

Model Based Metrics: A better way to do BTC

- Open Loop TCP congestion control
  - Prevent self inflicted congestion
  - Prevent circular dependencies between parameters
    - Data rate, loss rate, RTT
- Independently control traffic patterns
  - Defeat congestion control (generally slow down)
  - Mimic all typical TCP traffic (bursts, etc)
- Measure path properties section by section
  - Mostly losses (defects)
  - Compare to properties required per models
  - E2E path passes only if all sections pass all tests

The pieces (simplified)


The "application" determines target_rate
End-to-end path determines target_RTT and target_MTU
Host 1
Host 2
Sub-path Under Test

"front" path is taken
to be effectively ideal,
alternatively
tighten the measurement
points (& targets) to the SUT

Must meet constraints determined by models based on target_rate, target_RTT and target_MTU

Measurement Algebra with an example

- Target parameters:
    - 1 MByte/s bulk data over a path that is
    - 10 Mb/s raw capacity (~1.2 MByte/s)
        - More than the target!
    - 20 ms RTT, 1500 Byte MTU, 64 byte headers
- Compute from TCP Macroscopic Model
    - target_pipe_size
        - target_rate*target_RTT / (target_MTU-header_overhead)
        - 14 packets
    - reference_target_run_length (= 1/p)
        - (3/2)(target_pipe_size^2)
        - 274 packets
        - Same as $p < 0.365\%$

Key Network Properties

A path can sustain a BTC TCP flow (Data rate, RTT, MTU) when:

Path Components:

- o The **raw link rate** is higher than the target data rate.
- o The **raw packet loss rate** is lower than required by a suitable TCP performance model

(in other words, all forms of loss < loss due to window probing at Equilibrium?)

Sufficient Buffering:

- o There is sufficient buffering at the dominant bottleneck to absorb a **slowstart rate** burst large enough to get the flow out of slowstart at a suitable window size.

  (ideally, "large enough" is ~ 2 * target pipe size - ACK'd segments in transit)
- o There is sufficient buffering in the front path to absorb and smooth **sender interface rate bursts** at all scales that are likely to be generated by the application, any channel arbitration in the ACK path or other mechanisms. (this last aspect is covered below)

  (how much "application detail" does the tester need? adds complexity...)

(contd.)

A path can sustain a BTC TCP flow (Data rate, RTT, MTU) when: (contd.)

...

Channel Access and Queue Management

- o When there is a standing queue at a bottleneck for a shared media subpath, there are **suitable bounds on how the data and ACKs interact**, for example due to the channel arbitration mechanism.

(for example, how does bi-directional transmission influence RTT variation?)

- o When there is a slowly rising standing queue at the bottleneck the **onset of packet loss has to be at an appropriate point** (time or queue depth) and progressive.

(in other words AQM -- first observed loss not too much longer than the target run-length)

The MBM tests

- Baseline CBR performance
- Slowstart style burst tests
- Server interface rate burst tests
- Reordering tests
- Standing queue test

Table of Measurement Methods

Single Property Tests (7.1, 7.2) and Applicable Test Procedures

Common Test Procedures	**Loss Rate at Full Data Rate**	**Loss Rate at Full Data Windowed Rate**	**Backgrnd Loss Rate Test**	**Std Queue, Congestion Avoidance**	Std Queue, Buffer Bloat	Std Queue, Duplex Self Interference
Paced Trans (Bursts) 6.1.1	Single Packet (true CBR)		ref to 6.1.1 Single Packet			
Pseudo CBR 6.1.2 (fixed window)		ref to 6.1.2, relies on self-clock	Alternatively 6.1.2			
Scanned Window Pseudo CBR 6.1.2.1				6.1.2.1. with additional inspection of loss (early,#)	6.1.2.1. with additional inspection of loss (late,#)	6.1.2.1. with additional inspection of RTT (monot)
Intermittent Testing						
Intermittent Scatter Testing						

Single Property Tests (7.3, 7.4) and Applicable Test Procedures

Common Test Procedures	**Full Window Slow-Start**	**Slow-Start AQM**	**Sender TCP Offload (TSO)**	**Sender Full Window Burst**		Section 8 canonical test
Paced Trans (Bursts) 6.1.1	SS_Burst = (target_pipe+ req_queue)* derate length	Lab or ITE, SS_rate until loss, collect RTT and Window size	Bursts at Server Rate length = MIN (target_pipe, 42)	Bursts at Server Rate length = target_pipe		target_pipe Bursts at Server Rate, every target RTT
Pseudo CBR (fixed window)						
Scanned Window Pseudo CBR						
Intermittent Testing						
Intermittent Scatter Testing						

A look at Headway

Headway (as described in the IETF-86 slides for MBM)

- Target parameters:
  - 1 MByte/s bulk data = 8 Mbit/s
  - over a path that is 10 Mb/s raw capacity (~1.2 MByte/s), More than the target!
  - 20 ms, 1500 Byte MTU, 64 byte headers(1460 MSS)
- Compute two additional (new) parameters:
  - Headway at target rate (Bytes/pkt*bits/byte*sec/bits)
    - target_headway = target_MTU*8/target_rate
    - target_headway = 1.5 mS
  - Headway at bottleneck rate
    - bottleneck_headway = target_MTU*8/effective_rate
    - bottleneck_headway = 1.2 mS

Headway

- Headway at target rate
  - target_headway = target_MTU*8/target_rate
  - target_headway in units of
  - (Bytes/pkt*bits/byte*sec/bits), bits and bytes cancel
  - (1/pkt*sec/1) = seconds per pkt
  - target_headway = 1.5 mS (from the example)
  - (this is equivalent to pkt serialization time at target_rate)
- Headway for bursts
  - burst_headway (time from burst start to burst start)

      = burst_size*target_MTU*8/target_rate (or other rate)

      = burst_size*target_headway (seconds per burst)

    burst_size is in units of pkts per burst
-

Headway used in Pacing Methods

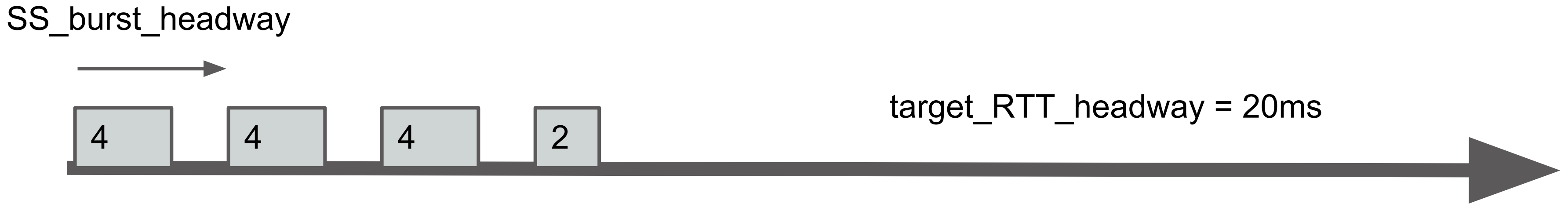
- Single packet pacing uses target_headway
  - packet to packet spacing
- Burst pacing uses burst_headway
  - burst start to burst start spacing
- Slow-Start:
  - 4 pkt bursts at server line rate
  - SS_burst_headway uses minimum of the 2 rates below

=burst_size*target_MTU*8/(2*effective_bottleneck_link_rate)

=burst_size*target_MTU*8/server_link_rate

Headway used in Repeated Slow-Start Pacing

- Slow-Start:
  - 4 pkt bursts at server line rate
  - SS_burst_headway as before
- Larger Pattern for Repeated Slow-Start:
  - Total of target_pipe_size packets
  - target_RTT_headway (pattern start to pattern start)
- target_pipe_size (from 1MByte/s, 20ms RTT example)
    - target_rate*target_RTT / (target_MTU-header_overhead)
    - =14 packets



Deciding if a test passes

- Recursive run length measurement
    - Progressive testing
    - Accumulate counts of losses and delivered packets
    - When to:
        - Declare success
        - Declare failure
        - Give up (declare inconclusive)
- Inconclusive also covers other non-results such as:
    - Tester failed to generate prescribed traffic patterns
    - Link was determined to be non-idle
    - etc
- Beware: inclusive tests can introduce sampling bias
    - Must strive to eliminate them

Sequential Probability Ratio Test (SPRT) **

Help Determining Sample Size & Pass/Fail/Indeterminate:

- In practice, can we compare the empirically estimated loss probabilities with the targets as the sample size grows?
- How large a sample is needed to say that the measurements of packet transfer/loss indicate a particular run-length is present (with desired error)?
- Lost packet or other impairment ~ Defect

We set two probabilities, one using target_run_length (H0) and another target_run_length/2 (H1), and the Type I and II errors (0.05).

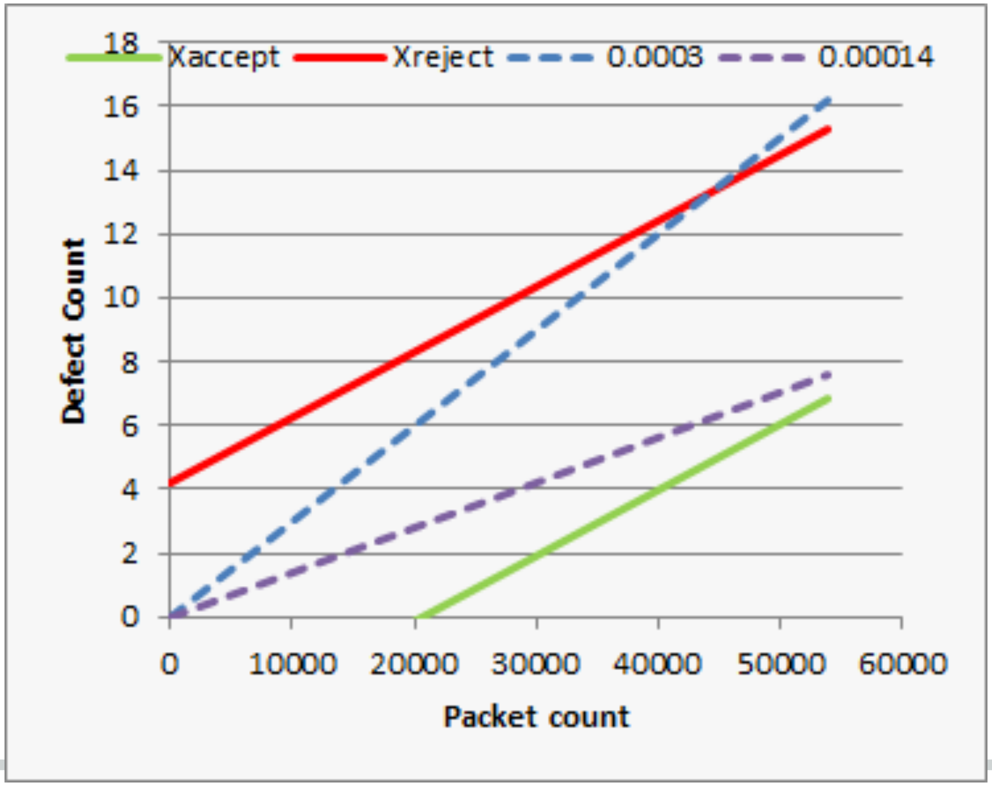
The SPRT test calculates cumulative limits to evaluate the defect ratio as the sample size grows.

** suggested by Ganga Maguluri, AT&T Labs

Different approaches to using SPRT in MBM

Using H0 (p0) at 80 Mbps Target p1 at 56.5 Mbps for "fail" (2p0 = p1)

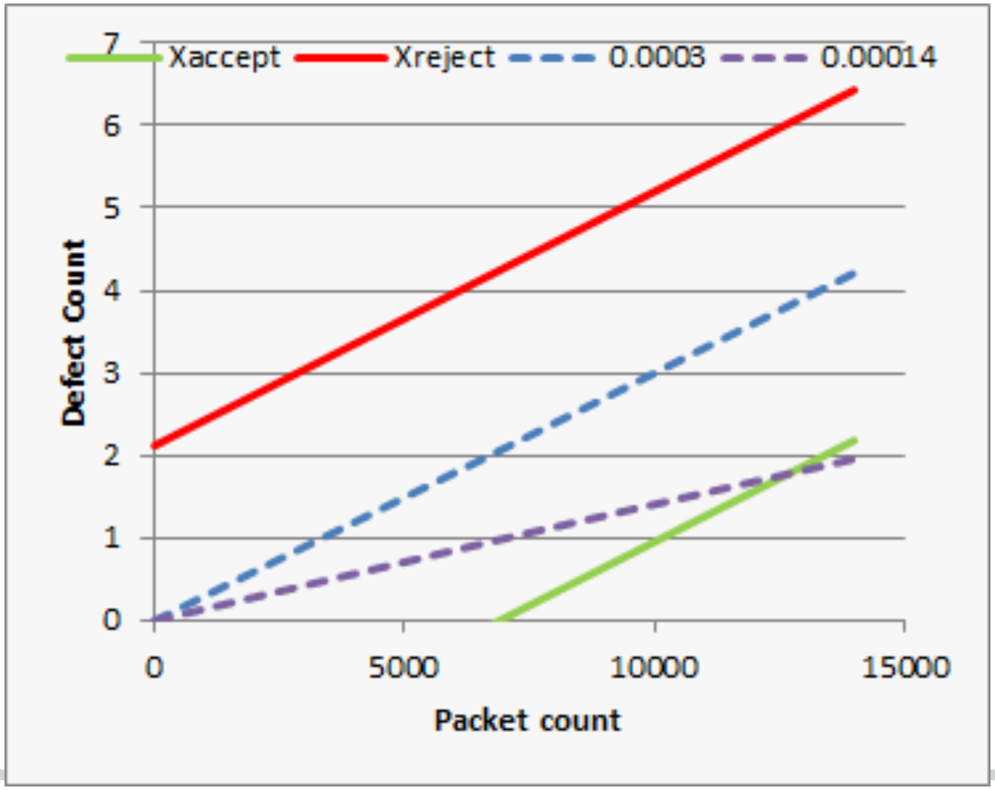
alpha beta	Target RTT, ms	Target Rate	Target Pipe	Target Run	Target p	
0.05	10	56500000	48.373288	3509.962	2.849E-04	p1
0.05	10	80000000	68.493151	7036.968	1.421E-04	p0



X axis crs	Y axis cross
20615.4	4.232249901

Using H0 (p0) at 80 Mbps Target p1 at 40 Mbps for "fail" (4p0 = p1)

alpha beta	Target RTT, ms	Target Rate	Target Pipe	Target Run	Target p	
0.05	10	40000000	34.246575	1759.242	5.684E-04	p1
0.05	10	80000000	68.493151	7036.968	1.421E-04	p0



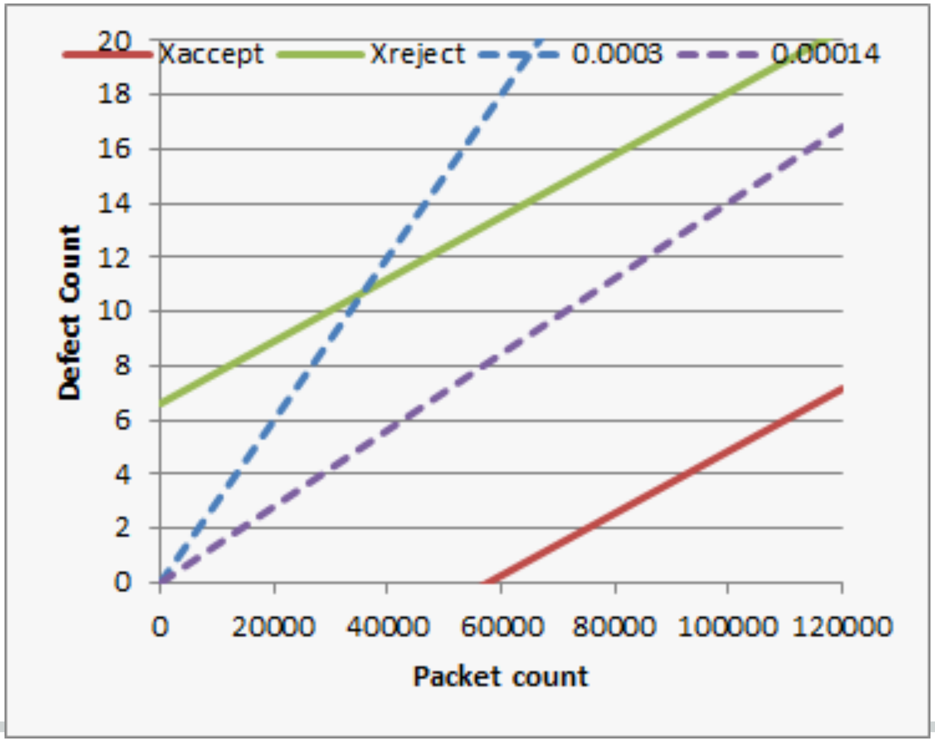
X axis crs	Y axis cross
6904.187	2.123310554

Using H1 (p1) at 80 Mbps Target

- Threshold for H1 preference is described as "fail" for rate less than target.
- Loss performance tending toward "pass" is relegated to "indeterminate" region
- Issue: measured loss performance would have to be much better than target for "pass" outcome with a reasonable sample size.
- Issue: if the target is a high percentage of a physical link capacity (say 80% of 100Mbps), then "pass" threshold is ~ target rate, leading to large sample size req.

Using H1 (p1) at 80 Mbps Target p0 at 100 Mbps for "pass"

alpha beta	Target RTT, ms	Target Rate	Target Pipe	Target Run	Target p	
0.05	10	80000000	68.493151	7036.968	1.421E-04	p1
0.05	10	100000000	85.616438	10995.26	9.095E-05	p0



X axis crs	Y axis cross
57548.63	6.596877743

Next Steps

- Incorporate some of the clarifications expressed here into the memo.
- Metric sub-section format

Backup...

An example

- Goal: 1 MByte/s BTC over a path that is
  - 10 Mb/s raw capacity (~1.2 MByte/s)
  - 20 ms RTT, 1500 Byte MTU, 64 byte headers
  - Invert TCP performance model [MSMO97]

$$Rate = \left(\frac{MSS}{RTT}\right)\frac{C}{\sqrt{p}}$$

  - Yields loss probability budget less than 0.3%
  - Test each short section at 1 MByte/s
- Fails if total loss probability is more than 0.3%
  - This is a pass/fail test, not a measurement
  - But passing this test alone is not sufficient
    - Because the link can still fail in other ways

Derating

- To some extent the models are subjective
  - ...and too conservative
  - What if TCP isn't standard Reno?
- Must permit some flexibility in the details
  - As TCP evolves
  - As the network evolves
  - The ID permits "derating"
- Actual test parameters must be documented
  - and justified relative to the targets
  - and proven to be sufficient
    - Meet the target goal over a derated network
- (ID will have) text about calibration and testing

All tests have valuable properties

- Tests do not depend on sub-path RTT
  - (Except one detail)
- Tests do not depend on measurement vantage
  - As long as rest of path is good enough
- Tests should not depend on implementation
  - Different parties should get the same results
- There is an algebra on test result
  - Summing (or pre allocating) losses
  - Any failed test on any sub-path fails the path

Keep these in mind

1) Baseline (CBR) performance test

- Measures basic data and loss rates
- Send one 1500 byte packet every 1.5 mS
  - 1 MByte/s target rate
  - Losses MUST be more than 274 packets apart
    - Otherwise "standard" Reno TCP can't fill the link
- Derated or Intermittent testing
  - e.g. reduced data rate for stealth mode testing
  - No derating on target_run_length
    - (Use a different model instead)