Evaluation of Hash Functions for Passive Inter-Domain Measurements

Saverio Niccolini
(NEC Europe, Network Laboratories Heidelberg)

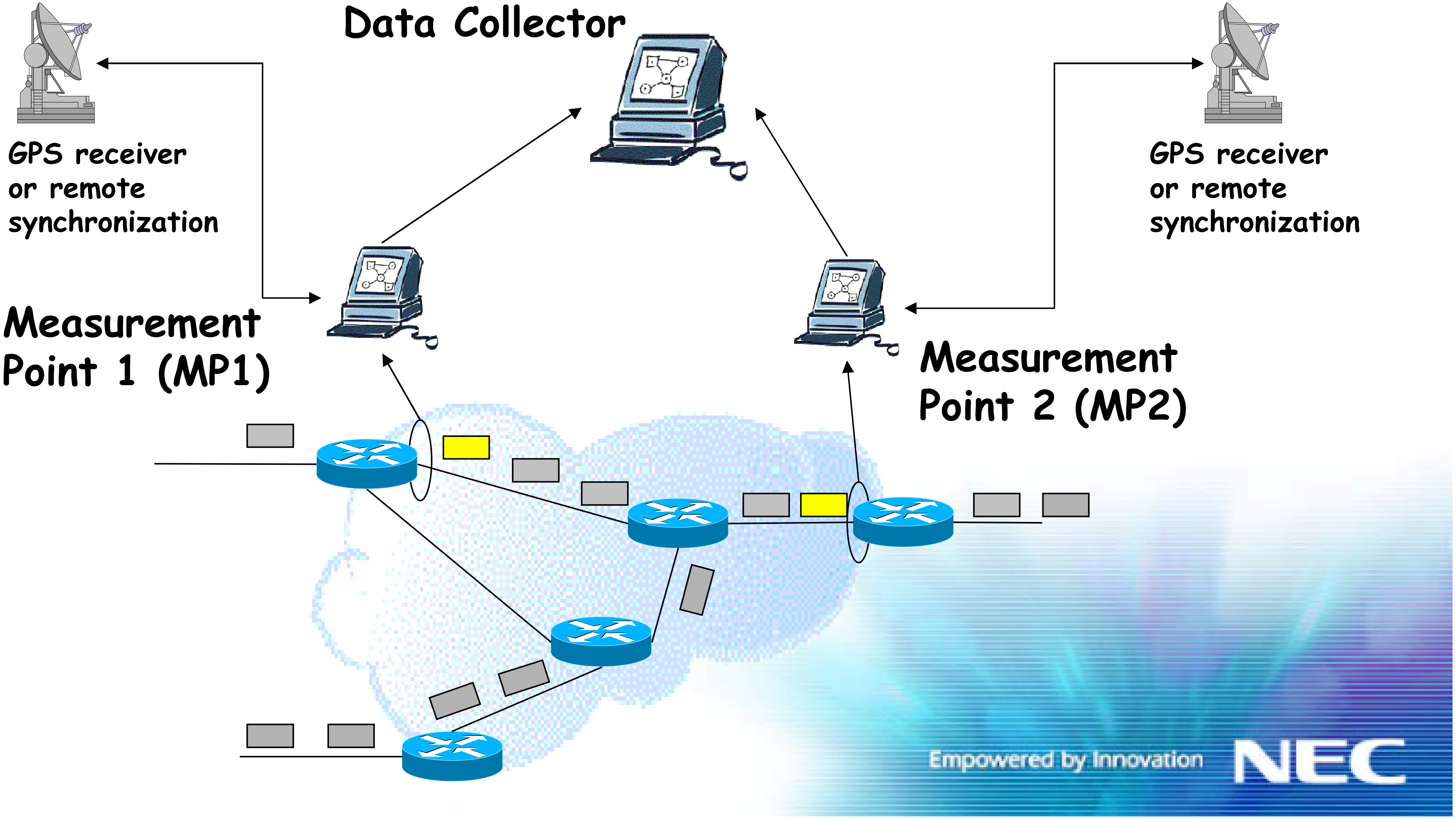
Maurizio Molina (DANTE)
Nick Duffield (AT&T Labs-Research)

Outline

- Sampling applied to Passive Measurements
- Hash-based packet selection and digesting for Inter-Domain applications
- Hash functions requirements
- Comparison results
- Conclusion and on-going work

Sampling applied to Passive Measurements

Capture and SAMPLE packets at every Measurement Point COHERENTLY, (timestamp them and) send a report to the collector


Data Collector
GPS receiver or remote synchronization
GPS receiver or remote synchronization
Measurement Point 1 (MP1)
Measurement Point 2 (MP2)
Empowered by Innovation
NEC

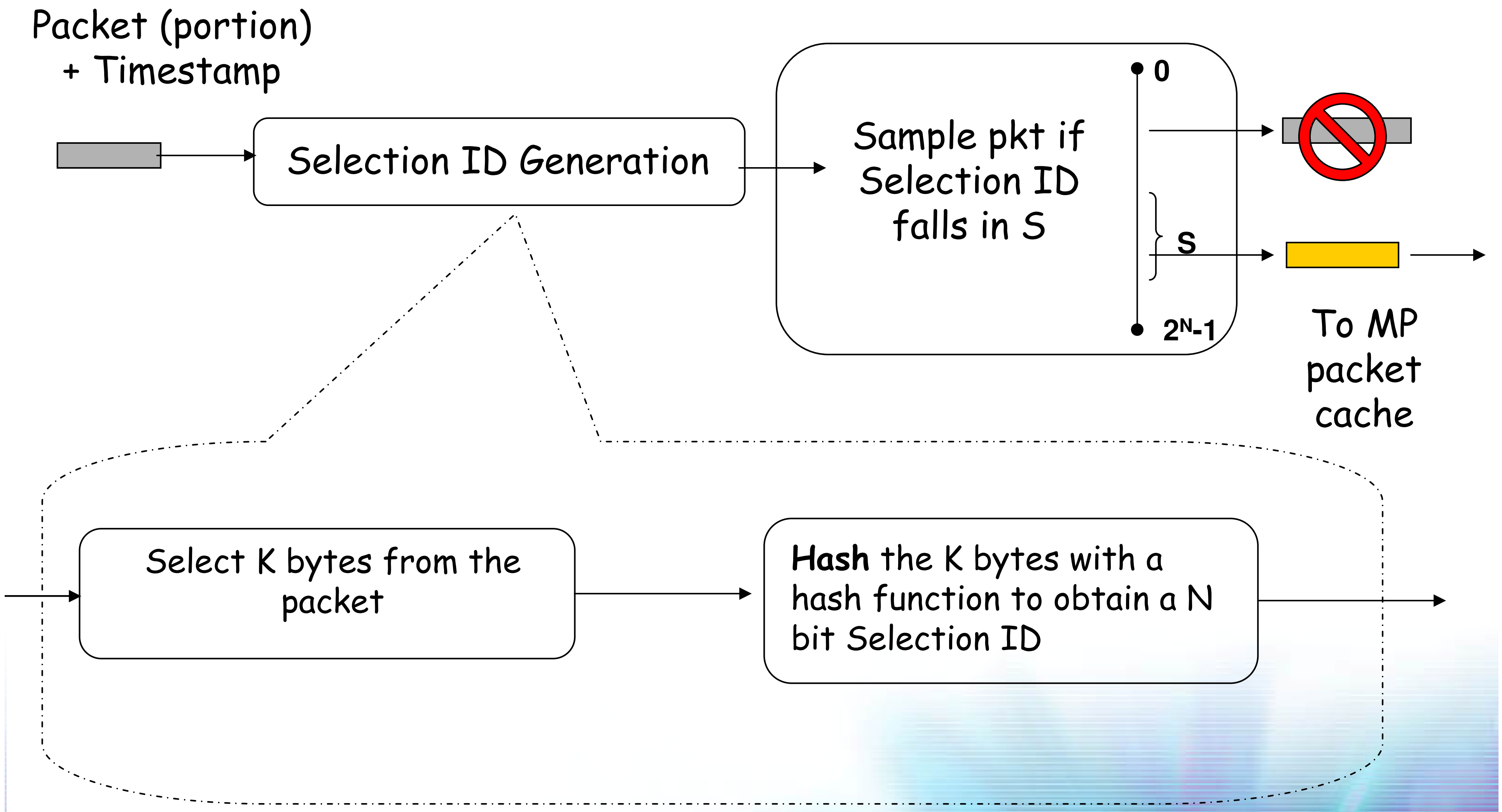
Application to inter-domain

- Need to relate packets coming from different Measurement Points
    - Need to have coherent selection of packets
- Coherent selection of packets achievable with:
    - Hash-based packet sampling
- Possible applications:
    - Trajectory sampling
    - One Way delay estimation
    - Etc.

N. Duffield, M. Grossglauser, “Trajectory Sampling for Direct
Traffic Observation”, IEEE/ACM Transactions on Networking, June 2001

S. Niccolini, M. Molina, F. Raspall, S. Tartarelli, “Design and
Implementation of a One Way Delay Passive Measurement System”,
in proceedings of NOMS 2004

Hash based coherent packet sampling in MP


Packet (portion)
+ Timestamp
Selection ID Generation
Sample pkt if Selection ID falls in S
0
S
2^N-1
To MP packet cache
Select K bytes from the packet
Hash the K bytes with a hash function to obtain a N bit Selection ID

Requirements for coherent sampling

- Input bytes for hash function must be:
  - the same for all MPs
  - invariant along the path
- Selection Range must be:
  - the same for all MPs
- Hash function must:
  - be the same for all MPs
  - be fast (works at line rate)
  - uniformly distribute output on $[0,2^N-1]$
    - thus sampling ratio is $S/2^N$,$\forall$ S

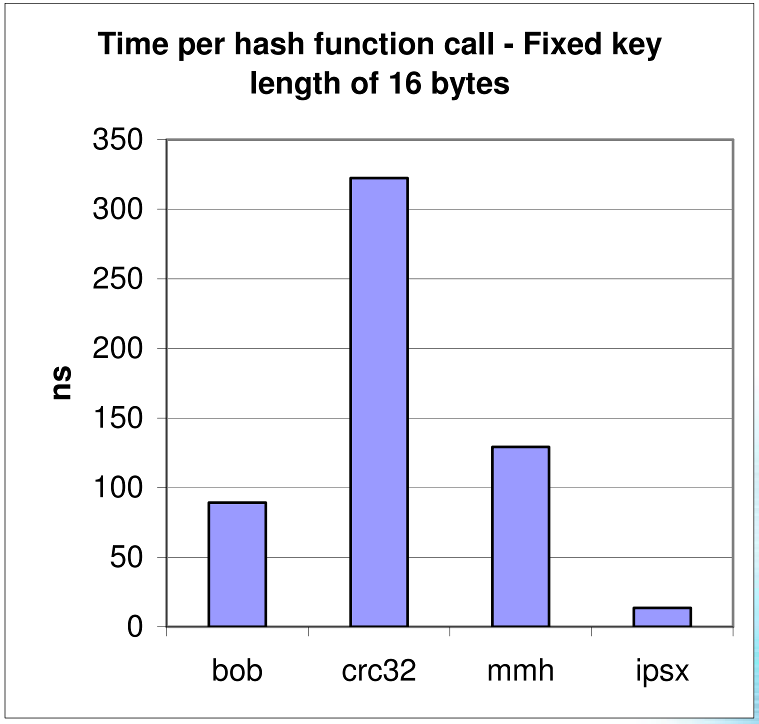
Evaluation methodology

- Two independent hash functions:
  - Selection ID (used to sample the packets)
    - Requirements (in order of importance):
      - speed, possibly line rate
      - uniformity of the output
  - Digest ID (used to assign an ID to each packet)
    - Requirements (in order of importance):
      - operate on keys of configurable length
      - low collisions over application-relevant timescales
      - speed

- Preliminary screening led us to:
  - CRC32 (classic CRC with 32 bit output)
  - IPSX (IP Shift and XOR)
  - Bob (collections of shift and XOR, like IPSX)
  - MMH (Multi-linear Modular Hashing)

Selection ID Hash Function - Speed

- Execution time of single hash computation
  - Absolute numbers may vary but relative values are invariant
  - Fair comparison (number of parameters, avoiding sub-calls, etc.)



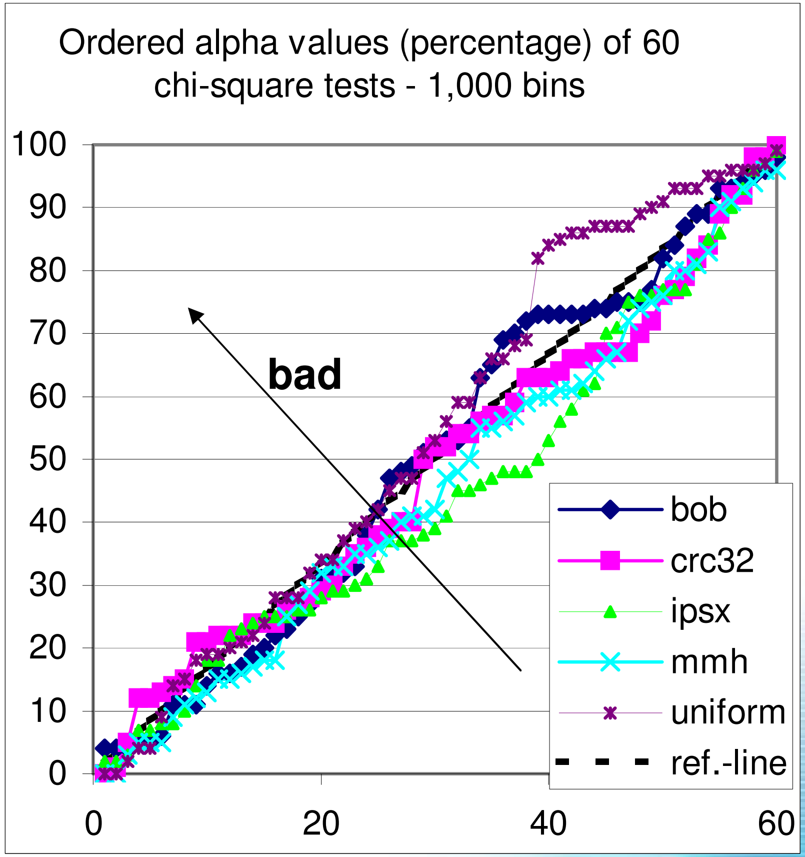
Selection ID Hash Function - Uniformity

- Significance test (emphasizing the non-uniformities)
  - Conformance to uniform distribution with Chi-square test
    - Dividing the hash range in N bins (N = 1.000 and N = 10.000)
    - Looking how many hits per bin (theoretically: packets/N)
    - 60 independent tests with 400.000 packets each
    - Plotting alpha values in increasing order
    - Tested with synthetic and real traces

- Variability metric
  - Measuring smaller non-uniformities
  - Smaller non-uniformities may not affect measurement application even if detected by previous test
  - calculation made on mean values and 95% confidence intervals

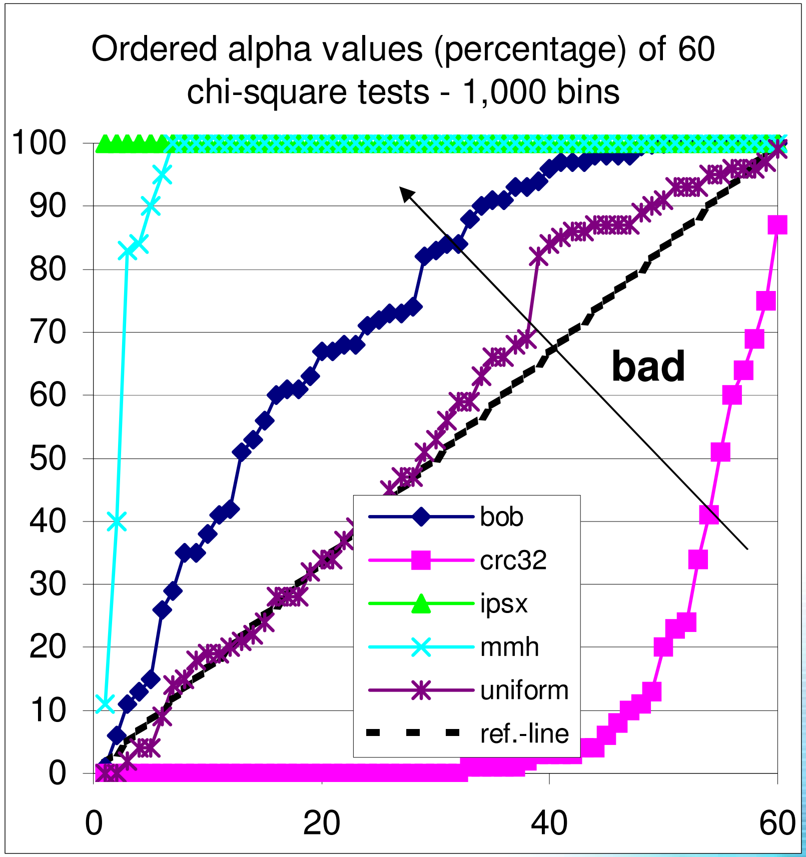
Selection ID Hash Function - Uniformity

- Significance test
  (testing uniformity with Chi-square test, synthetic trace)



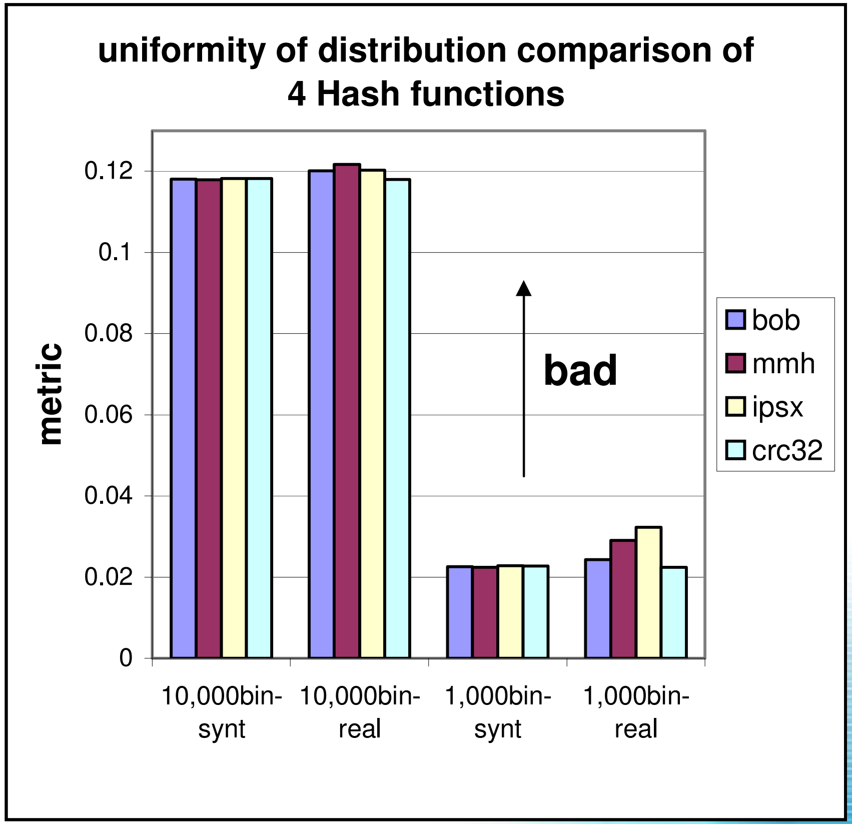
Selection ID Hash Function - Uniformity

- Significance test
  (testing uniformity with Chi-square test, real trace)



Selection ID Hash Function - Uniformity

- Variability metric
  (the lower the value, the more uniform the behavior)



Empowered by Innovation NEC

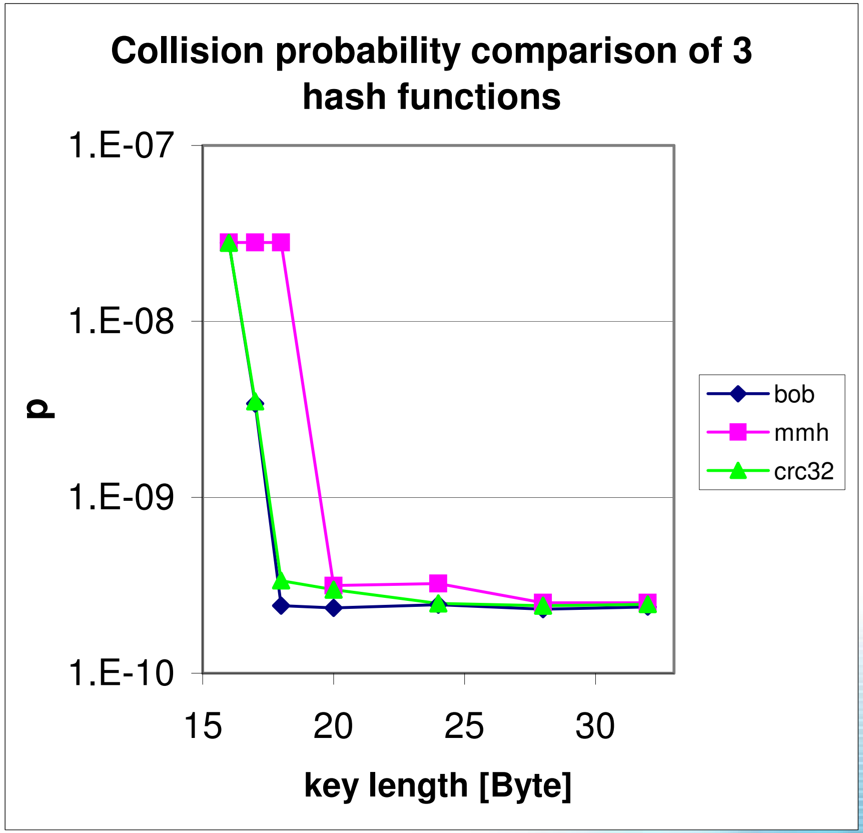
Selection ID Hash Function – Results

- IPSX has a big advantage in computational speed
- IPSX and MMH conform the worst to the uniform distribution
- Performance of Bob worsens but not severely
- CRC32 is the slowest hash function
- CRC32 results seems to improve when hashing real trace (but results are not stable)
- Bob, MMH and IPSX had stable results
- Variability is almost the same for all functions

- Result: IPSX performed slightly worse in uniformity but its speed make it be the best candidate as Selection ID hash function (better trade-off)

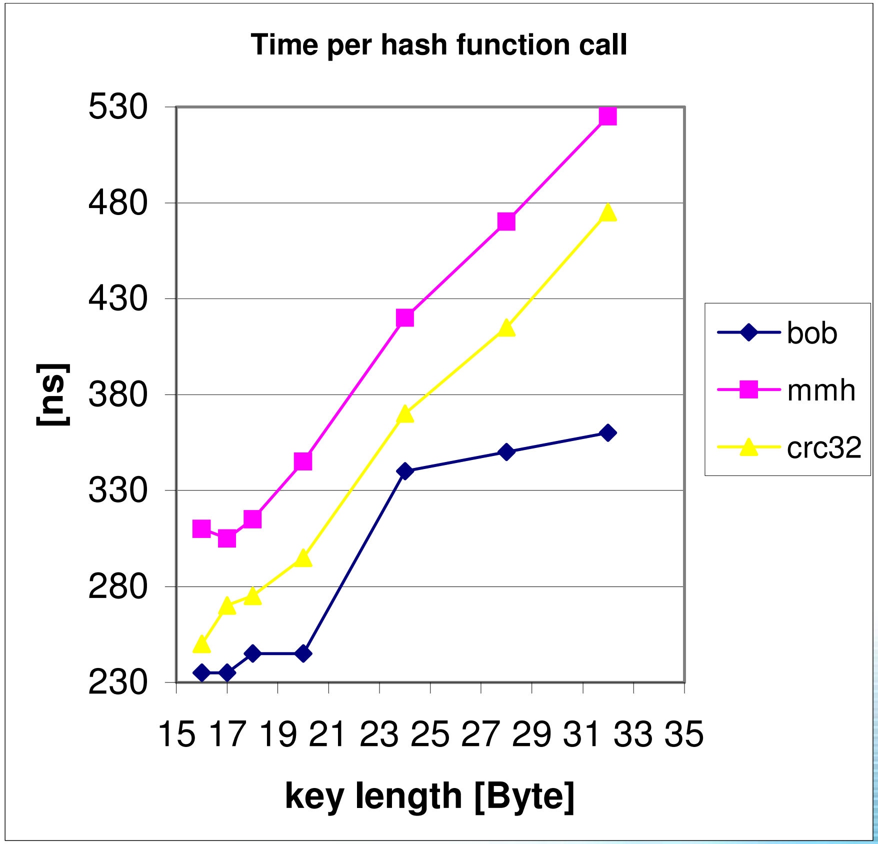
Digest ID Hash Function - Collision

- Collision probability



Digest ID Hash Function - Speed

- Time per hash function call



Digest ID Hash Function – Results

- IPSX was not eligible because it does not accept a variable string as an input (fixed to 16 bytes)
- Bob performed slightly better than MMH and CRC32 in collision probability
- Bob is the fastest one among the three

- Result: Bob is the best candidate as Digest ID hash function

Conclusions

- We presented a methodology for testing hash functions for packet sampling
- We performed tests on synthetic and real traces
- Results
  - Selection ID hash function:
    - IPSX
  - Digest ID hash function:
    - Bob
- Results and hash functions description where contributed to the IETF in 2 PSAMP drafts:
  - draft-ietf-psamp-sample-tech-05.txt
  - draft-niccolini-hash-descr-00.txt (expired)

On-going and future work

- On-going:
  - Extend the software to read from tcpdump and .tsh file (done)
  - Extend the tests to a more complete set of real traces
    - MAWI traces
    - NLANR traces
  - Results are already there (raw files at least)
    - Organizing and visualizing them
    - Comparing to what we already have
- Future:
  - Further tests on raw bin occupancy (more detailed)
  - Packet sampling applied to IPv6
  - Extend the tests to IPv6 traces

Thank you!

Questions?