Empirical Evaluation of Hash Functions for PacketID Generation in Sampled Multipoint Measurements

Christian Henke, Carsten Schmoll, and Tanja Zseby

Fraunhofer Institute Fokus, Berlin, Germany
{christian.henke,carsten.schmoll,tanja.zseby}@fokus.fraunhofer.de

Abstract. A broad spectrum of network measurement applications demand for multipoint measurements; e.g. one-way delay measurements or packets path tracing. A passive multipoint measurement technique is realized by generating a timestamp and a packet identifier (ID) for each packet traversing an observation point and sending this information to a common collector. The packet ID can be provided by using parts of the packet or generating a digest of the packet content. Multipoint measurements demand for high resource measurement infrastructure. Random packet selection techniques can reduce the resource consumption while still maintaining sufficient information about most metrics. Nevertheless random packet selection cannot be used for multipoint measurements, because the packets selection decisions on its path can differ. Hash-based selection is a deterministic passive multipoint measurement technique that emulates random selection and enables the correlation of a selected subset of packets at different measurement points. The selection decision is based on a hash value over invariant parts of the packet.

When hash-based selection is applied two hash values are generated - one on which the selection decision is based and a second one that is used as the packet ID. In a previous paper we already evaluated hash functions for hash-based selection. In this paper we analyze hash functions for packet ID generation. Other authors recommend the use of two different hash values for both operations - we show that in certain scenarios it is more efficient to use only one hash value.

1 Introduction

A variety of measurement applications demand for passive multipoint measurements; like delay and loss measurement or packet tracing. A passive multipoint measurement technique is realized by generating a timestamp and a unique packet ID for each packet traversing an observation point and sending this information to a common collector. The packet ID can be a combination of packet fields or a digest calculated over the packet content. In case the packet ID is unique in the measurement domain the collector can correlate the packet observations and trace the packet throughout the network. The one-way delay between two measurement points can be deduced by the timestamp difference.

S.B. Moon et al. (Eds.): PAM 2009, LNCS 5448, pp. 197–206, 2009.

Because a timestamp and packet ID needs to be generated and exported for each packet, the resource consumption for passive measurements can be immense. Packet selection methods provide a solution to reduce the resource consumption. However random selection techniques are not suitable for multipoint measurements, because a random and therefore different subset of packets is selected at each measurement node, which hinders packet tracing and delay calculation.

Hash-based selection [2] is a deterministic packet selection method that selects consistent packet subsets throughout the network. Hash-based selection is realized by the following technique: parts of the packet content that are invariant between measurement nodes are extracted and used as the hash input for a hash function. The hash function with a digest length of N bits maps the hash input to a value in the range R = $[0..2^N - 1]$. The packet itself is selected if the hash value - here called selection hash - falls into a predefined selection range $S \subset R$. The selection decision for each packet along its path is the same, provided that the selected packet content (hash input), hash function and selection range are the same at the different measurement points.

When hash-based selection is applied in multipoint measurements two hash values are generated. 1) The selection hash and 2) for selected packets a packet ID which is exported to the collector. The authors of [2] [9] [11] [7] recommend to use different hash values for packet ID and selection hash. This is reasonable because 1) good hash functions for hash based selection are not necessarily good packet ID generating hash functions and 2) the packet ID collision probability increases if only one hash value is used (see Sec. 4).

We already analyzed 25 hash functions for hash-based selection in [4]. In this paper we will analyze the same hash functions on their suitability for packet ID generation based on their hash value collision probability. Hash value collisions are critical for packet ID generation because packets with the same ID cannot be distinguished and properly traced.

In contrast to other authors we propose to use only one hash value for the hash-based selection decision and packet ID. The use of only one hash value will relieve the processing capacities of the observation points. Nevertheless the use of one hash function infers more packet ID collisions, which we propose to offset by some additional selected packets. We will use a mathematical model to calculate the implications of the decision and calculate the measurement traffic increase which can compensate for additional collisions.

2 State of Art

In [2] Duffield and Grossglauser introduced the hash-based selection technique for the purpose of packet tracing. They evaluate a simple modulus hash function for hash-based selection using four different traces. Molina [7] analyzes four hash functions (CRC32, MMH, IPSX, BOB) for hash-based selection and packet ID generation. With regards to packet ID he analyzes the uniformity of hash values and the collision probability depending on the length of the hash input. In [4] we evaluated a set of 25 hash functions for hash-based selection in terms of

1) performance 2) non-linearity 3) unbiasedness and 4) representativeness. The analysis has shown that the BOB and OAAT hash function have the best overall results from this hash function collection. Nevertheless an evaluation for packet ID generation is missing. [14] compares 6 hash functions for packet ID generation (40 bytes of unprocessed fields, CRC-16, a 16-bit hash function, CRC-32, a folded 32bit MD5 and MD5-128bit) on the hash value collision probability and processing time. The unprocessed fields do not require any processing and no collisions, but imply maximum measurement traffic. The CRC-32, folded 32bit MD5 and 128 bit MD5 hash functions provide no collision in the analyzed short traces of 100,000 packets.

3 Hash Function for Packet-ID Generation

Hash value collisions are crucial for packet ID generation, because packets with colliding hash values cannot be distinguished and properly traced. In order to resolve ambiguous traffic traces the packets are discarded at the multipoint collector. The lower the collision probability of the packet ID generation hash function the fewer packets have to be discarded. The PSAMP [13] working group recommends non-cryptographical hash functions with a short digest length of 32 bits because these can be supported by low resource PSAMP devices. Cryptographical hash functions (like SHA or MD5) that are proven to have low collision probability may not be suitable for packet ID generation, because of 2 reasons: 1) the digest length is at least 128 bits long which would mean four times the measurement traffic for each packet and 2) the processing time is higher than for non-cryptographical hash functions. We will analyze a collection of 25 hash functions each with a 32 bit digest length on two criteria which are important for packet ID generation 1) uniformity of hash value distribution and 2) hash value collisions. The hash function collection is available at [1] with references to their original sources. The cryptographical SHA1 and MD5 hash function are included in the collection as well but were trimmed to 32 bits by adding each 32 bit subblock of their hash value.

3.1 Uniformity of Hash Value Distribution

Non-uniform distributed hash values pile up at specific intervals or values and hence have a higher collision probability. The chi-square goodness-of-fit test is used to analyze if the hash value distribution is significantly different from a uniform distribution. Because the amount of possible hash values R is large and only few occurrences of the same hash value are expected, the hash values are categorized in b bins that consist of $\frac{R}{b}$ values. The test statistic T is obtained by

$$T = \sum_{i=1}^{b} \frac{(n_i - \overline{n})^2}{\overline{n}} \tag{1}$$

where n_i are the observed frequencies in bin i. By dividing the number of total observed packets N by the amount of bins b, one obtains the number of expected

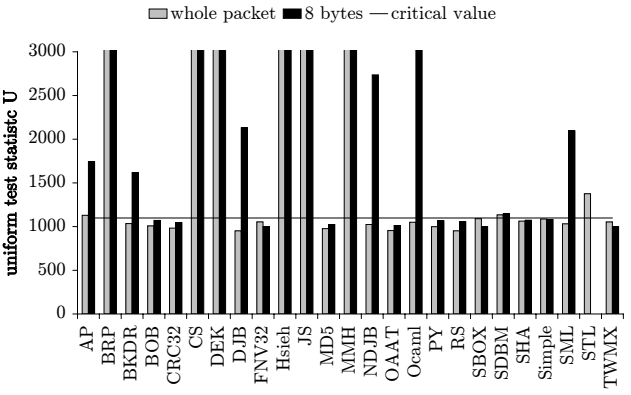


Fig. 1. Chi Square Uniformity Tests - For the test we used 5 traces, 1 million unique packets each. Hash functions whose test statistic T is higher than the critical X^2 value have a hash value distribution statistically different from the uniform distribution and presumably generate more collisions than a random generator

observations $\overline{n}$ in each bin. The hypotheses H0 that the hash value distribution is uniform is rejected if T is above the critical X^2 value with b-1 degrees of freedom with an error of α.

Experimental Setup. The uniform distribution evaluation is based on 5 traces each including N=1,000,000 unique packets, i.e. duplicate packets are removed from the trace. We removed these packets, because they distort the measurement results, packets that are identical will always have the same hash value and will be discarded for delay measurements anyways. The traces are taken from different trace groups - NZIX, Twente, FH Salzburg (accessible via [8]) and two trace groups from a european telecom operator (LEO1 and LEO2) which we already used in [5] and [4]. For the evaluation the hash input consist of the IP and Transport Header except Version/IHL, TOS, TTL, IP Checksum, Flags/Fragment Offset and IP Options. This guarantees that the hash input is unique and constant between measurement points. The hash values are categorized in $b = 1024$ bins. We are using a global error rate $\alpha = 5\%$.

Uniformity Results. The measurement results for all 25 hash functions are depicted in Fig. 1. For every hash function 5 tests are conducted based on each trace group. The problem with multiple Chi square tests is that every test inherits an error. We are not interested in individual tests that falsely reject H0, but in tests that show a truly significant deviation. A common approach when multiple statistical tests are performed is the Dunn-Sidak correction [12] which we already used in [4]. This adjusted critical X^2 value is used in Fig. 1. The hash functions that pass the uniformity test are BOB, CRC32, FNV32, MD5, OAAT, PY, RS, SBOX, SHA, SIMPLE, TWMX. These hash functions are very likely to possess low collision probability and will be further evaluated.

3.2 Mathematical Model for Collision Probability

Before we evaluate the hash function collection on their collision probability we will describe a model to calculate the amount of expected collisions when a random number generator is used for packet ID generation. Hash functions that generate statistically significant more collision than a random number generator should not be applied for packet ID generation.

Parameters of the Collision Model. Besides of the type of traffic included in the traces the hash value collision probability depends on 1) the number of packets N that are observed at all ingress nodes during the critical time interval $[t_0..t_0 + t_{max}]$ and 2) the hash range size R. For accurate delay measurements duplicate packet IDs are identified at the ingress nodes of the measurement domain (see [2]). Ingress nodes are incoming links from external routers or traffic sources within the measured network. It is not necessary to discard all duplicate packet IDs in the measurement domain, but only packets with equal IDs occurring close together in a time interval $[t_0..t_0 + t_{max}]$, where t_0 denotes the first occurrence of the packet ID at any ingress node and t_{max} is the maximum delay inside the measurement domain. If two equal packet IDs occur within the critical time interval at equal or different ingress nodes than all packet ID observations of both packets in this time frame are discarded, because one cannot ensure to form correct packet ID pairs. Assessing the maximum delay t_{max} is tedious and influences the delay measurement. If the maximum delay is chosen too large the collector 1) has to store immense amounts of packet IDs before it can purge old reports and 2) discards more packets than are necessary. If the maximum delay t_{max} is assessed too low, it is possible that wrong packet ID pairs are formed and the delay measurement becomes inaccurate.

The average amount of packets N that are generated during the critical time interval at all k ingress nodes each with a data rate B_i can be calculated using the incoming data rate $B = \sum_{i=1}^{k} B_i$, the average packet length $\bar{l}$ and t_{max} by $N = \frac{B \cdot t_{max}}{\bar{l}}$. As an example we assume a measurement domain with 4 backbone ingress links each $B_i = 2500 Mbit/s$ and $\bar{l}$=500 bytes and with a maximum domain delay of $t_{max} = 1s$. In a fully loaded case the critical interval has a size of 2.5 million packets.

Collision Model. The amount of collision can be obtained by using a two step model. First the probability is calculated that one specific hash value occurs m times when N hash values are randomly generated. Second, these probabilities for one hash value are transferred to all hash values to obtain the distribution of the amount of unique hash values.

First Step: A random variable X is defined:

X = amount of draws of the specific hash value i.

Because the hash value generation is random and independent the probability distribution of X is given by the binomial distribution $B(m|p, N)$ which can be approximated by the poisson distribution for $N > 50$ and $p < 0.05$ and $\lambda < 9$ [10]

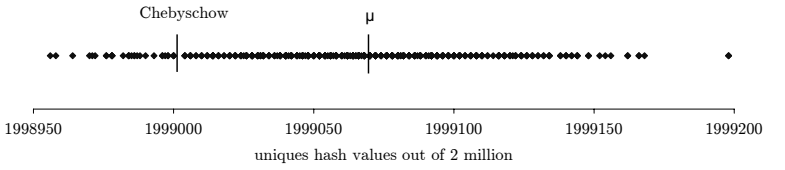


Fig. 2. In each of the 500 tests we generated 2 million random 32bit values and counted the amount of unique values. Each dot represents one test. The amount of expected unique values μ and the variance represented by the chebychev bound are shown as well. The empirical results are consistent with the mathematical model.

(which is given for our configuration) - where $p = \frac{1}{R}$ is the success probability that the specific value i is drawn in one try and $E(X) = \lambda = Np$:

$$B(m|p,N) = P(X = m) = \binom{N}{m} p^m (1-p)^{N-m} \approx \frac{\lambda^m}{m!} e^{-\lambda} \tag{2}$$

Second Step: We define a new random variable Y:

Y= amount of unique hash values in the critical interval.

One approach to gain a distribution of non-colliding hash values is the following. First it is assumed that already $N-1$ hash values are generated. All hash values in the hash range R are then binomial $B(m|N-1, \frac{1}{R})$ distributed, i.e. the chance to draw a hash value that already occurred m times is approximately P(X=m), because N is large. The Nth hash value is generated and it is observed how many times this hash value has been previously drawn. If the hash value has not been already drawn than a unique hash value has been generated. The probability that the hash value has not been generated before is approximately $P(X = 0)$. Moreover this conclusion can be applied for all other N-1 hash values. The success probability to draw a unique hash value is $p_y = P(X = 0)$. Because $N \ll R$ all experiments X_i can be assumed to be independent and $Y \sim B(k|N, p_y = P(X = 0))$. The expected amount of unique hash values μ and the variance σ^2 is:

$$\mu = Np = Ne^{-\frac{N}{R}} \qquad \sigma^2 = Np(1-p) = N(e^{-\frac{N}{R}} - e^{-\frac{2N}{R}}) \tag{3}$$

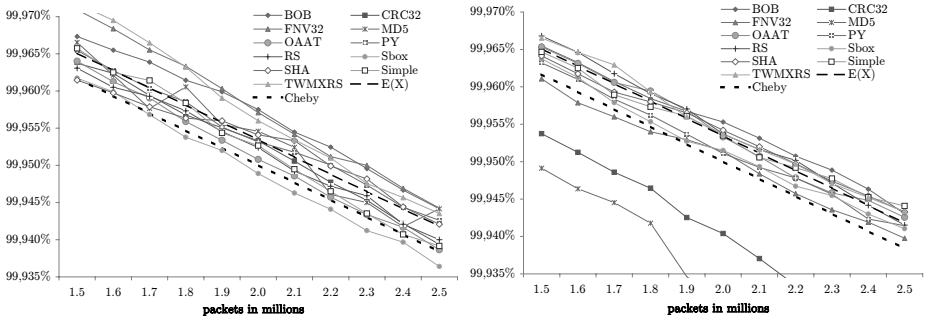
3.3 Empirical Collision Probability

First, we will use a random number generator to verify the mathematical model. Afterwards, hash functions that performed well in the uniformity tests in Sec.3.1 will be empirically analyzed on their packet ID collision probability.

Evaluation of Model using Random Generator. We used the mt19937 random number generator from the GNU scientific library [3] to obtain 32bit random values. In 500 tests the amount of unique hash values from 2 million random values are counted. For 2 million generated 32bit values the expected amount of unique hash values is (see Eq. 3) $\mu = 1999069$. The mean amount

of empirical measured unique hash values in the 500 runs is $\bar{U} = 1999066$. The measurement results are depicted in Fig. 2. This figure also shows the lower bound L=1999001 for the expected amount of unique hash values in b=90% of the cases using chebyschev's inequality. 32 tests (=6.4%) show less unique hash values than the chebyschev 90% bound. The lower bound using chernoffs inequality [6] is $L_c = 1996034$ which is significantly less strict than chebychev in our setup and is therefore not further considered. The results are consistent with the mathematical model.

Empirical Evaluation of Hash Functions for Packet ID Generation on Real Traces. The empirical hash function evaluation is based on 5 traces from which all duplicate packets are removed. The traces are taken from trace group NZIX, FH Salzburg, Twente, LEO1 and LEO2 in order to ensure a variety of traffic patterns. Eleven critical measurement intervals N ranging from 1.5 to 2.5 million packets are used. As the hash input we applied the same configuration as in Sec. 3.1. The expected percentage of usable packets $\mu = E(U)$ is shown by the straight black linear line in Fig. 3. In order to represent the variance, the expected least amount of non-colliding packets in 90% of the cases is calculated using chebyshev's inequality, shown as the black dashed line. Most hash functions are very close to the expected amount of collision for all 5 traces, except the LEO2 trace. For this trace the MD5 and CRC32 hash generate 0.01 to 0.03% more colliding packet IDs than expected. Almost all other deviation can be explained by statistical variance, because they are within the 90% chebyschev bound. The BOB and RS Hash function never generated fewer unique packetIDs than predicted by the chebyschev bound.



(a) NZIX trace (b) LEO2

Fig. 3. We evaluated the hash function collection on their collision probability based on 5 different traces - only 2 depicted here. The percentage of unique hash value were compared to the mathematical model as derived with the random number generator. Most hash functions (even not cryptographically strong) are very close to the expected amount of non-colliding packet IDs. Almost all deviation are within a 90% chebyschev bound of statistical variance.

4 Packet ID and Selection Hash

When applying hash-based selection, it may or it may not be reasonable to use the same hash value which is intended for the sampling decision to use as packet ID. Using the same hash value as packet ID will relieve the measurement points processing capacities, because one does not need to calculate another hash value for the selected packets. On the other hand it will increase the collision probability of the packet IDs which can be shown by using Eq. 3. The amount of unique hash values from $s \cdot N$ packets when two hash functions U_2 or one hash function U_1 are used is:

$$U_2 = sNe^{-\frac{sN}{R}} \quad > \quad U_1 = sNe^{-\frac{sN}{sR}} \tag{4}$$

The increase of collision probability is the reason why [2] [7] recommend to use a different hash function for packet ID generation and packet selection. Contrary, we propose to use only one hash value for most scenarios. For example, in a scenario with a sampling fraction s close to 1, a new hash value calculation for almost all packets is not justified because there is almost no collision improvement. Instead of generating packet IDs for the sampled packets one can select additional packets in order to compensate for the additional collisions. Of course the additionally selected packets increase the measurement traffic. An example will show the trade-off. Assuming an amount of 2.5 millions packets within the critical measurement interval and a selection fraction of s=10% the amount of additional collisions $U_2 - U_1$ is 131. This means that instead of calculating an extra hash value on 250000 packets one can just select≈131 packets additionally which is an increase of measurement traffic of $\frac{C}{sN} = 0.052\%$. This value only gives an approximate figure of measurement increase because the additional selected packets have a chance to collide as well. For a more accurate value one has to look at a more practical question: How does the selection range need to be configured to gain a certain target selection fraction f_t of unique packets?

Selection Range Adjustment for One Hash Function. In the case that only one hash function is used for hash-based selection and packet ID generation, the selection range fraction of the hash range s_1 has to be configured according to Eq. 5 in order to gain a target sampling fraction f_t of usable packets.

$$f_t N = s_1 N e^{-\frac{s_1 N}{s_1 R}} \quad \rightarrow \quad s_1 = f_t e^{\frac{N}{R}} \tag{5}$$

Selection Range Adjustment for Two Hash Functions. In the case that two different hash functions are used for hash-based selection and packet ID generation there are still packet ID collisions. Therefore it is still required to adjust the selection range s_2 in order to gain a target sampling fraction t of usable packets.

$$f_t = s_2 e^{-\frac{s_2 N}{R}} \tag{6}$$

We solved Eq. 6 numerically in order to enable a comparision between both approaches.

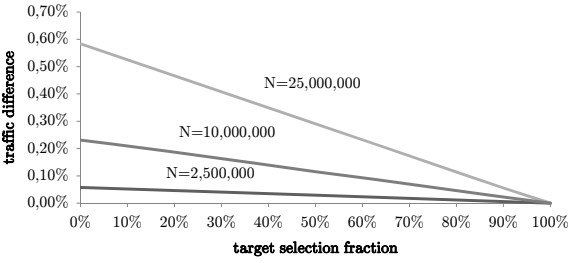


Fig. 4. Measurement Traffic Increase using 1 instead of 2 Hash Functions. When only one hash value is used for packet selection and packet ID we can relieve the measurement point, nevertheless the packet ID collision probability increases. We propose to compensate this increase by selecting some more packets, which will increase the measurement traffic. The measurement traffic increase is neglectable for large sampling fractions or small amount of packets within the critical time interval.

Difference Between One and Two Hash Function Approach. The selection range configuration differs if either one or two hash functions are used for packet ID generation and hash-based selection. The ratio $\frac{s_1 - s_2}{s_2}$ denotes the proportional difference of measurement traffic between both approaches. The increase is negligible for high target sampling ratios and small amounts of packets within the critical measurement interval N. As it is obvious from Fig. 4 the traffic increase is a linear function of the target sampling fraction. The measurement traffic increase ΔMTI can be formulated as:

$$\Delta MTI = (1 - e^{-N/R})(1 - t) \tag{7}$$

In our example with 4x2.5GB fully loaded ingress nodes and $t_{max} = 1s$ the additional measurement traffic is less than 0,06%. For scenarios with 10 / 100 as many nodes the additional measurement traffic is about 0,6% / 6% which can still justify to use only one hash function. Only in larger domains the measurement operator should use two hash functions or a longer packetIDs.

5 Conclusion

In this paper we analyzed a set of 25 hash functions on their suitability for packet ID generation. The evaluation is based on 5 different real traces. We showed that a set of 11 hash functions have comparable collision probabilities as expected from a random number generator. We recommend the BOB and RS hash function for packet ID generation because they never generated more collision than predicted by the chebychev bound and therefore they provide low and predictable collision probability.

Contrary to current approaches we analyzed the usage of only one instead of two hash functions for packet ID generation and hash-based selection. Using the mathematical model derived in Sec. 3.2 we showed for both approaches how the selection range has to be adjusted to gain a certain target sampling fraction of non-colliding and usable packet IDs. With these results we calculated the

additional traffic that is required when only one hash function is used for packet ID generation and hash-based selection. For small to medium sized measurement domains it is reasonable to use only one hash function. Because the BOB hash function proofed to be under the best two functions for hash-based selection and packet ID generation, we recommend the BOB hash function in case only one hash function is used.

References

1. Hash functions, `http://net.fokus.fraunhofer.de/research/hashfunctions.html`
2. Duffield, N., Grossglauser, M.: Trajectory sampling for direct traffic observation. IEEE/ACM Trans. Netw. 9(3), 280–292 (2001)
3. Gsl - gnu scientific library, `http://www.gnu.org/software/gsl/`
4. Henke, C., Schmoll, C., Zseby, T.: Empirical evaluation of hash functions for multipoint measurements. SIGCOMM Comput. Commun. Rev. 38(3), 39–50 (2008)
5. Henke, C., Schmoll, C., Zseby, T.: Evaluation of header field entropy for hash-based packet selection. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 82–91. Springer, Heidelberg (2008)
6. Jukna, S.: Crashkurs Mathematik für Informatiker. Teubner (2007)
7. Molina, M., Niccolini, S., Duffield, N.: Comparative experimental study of hash functions applied to packet sampling. In: ITC-19, Beijing (August 2005)
8. Traffic measurement database, `http://www.ist-mome.org/`
9. Niccolini, S., Molina, M., Raspall, F., Tartarelli, S.: Design and implementation of a one way delay passive measurement system. In: IEEE NOMS 2004 (2004)
10. Papoulis, A., Pillai, U.: Probability, Random Variables, and Stochastic Processes, 4th edn. McGraw-Hill, New York (2002)
11. Raspall, F., Quittek, J., Brunner, M.: Path-coupled configuration of passive measurements. In: IPS 2004, Budapest (2004)
12. Miller Jr., R.G.: Simultaneous Statistical Interference, pp. 5–8, 15–16. Springer, Heidelberg (1981)
13. Zseby, T., Molina, M., Duffield, N., Niccolini, S., Raspall, F.: Sampling and filtering techniques for ip packet selection. IETF Internet Draft
14. Zseby, T., Zander, S., Carle, G.: Evaluation of building blocks for passive one-way-delay measurements. In: PAM 2001 (2001)