Analysis of TCP Bottleneck Footprints in Traces from live 3G Networks

Philipp Svoboda*, Fabio Ricciato†, Markus Rupp*
*INTHFT Department, Vienna University of Technology, Austria
Institut für Nachrichtentechnik und Hochfrequenztechnik
Technische Universität Wien, Austria
Gusshausstrasse 25/389, A-1040 Vienn a, Austria
Email: {psvoboda, mrupp}@nt.tuwien.ac.at
†ftw. Forschungszentrum Telekommunikation
Donau-City-Strasse 1, A-1220 Vienna, Austria
Email: ricciato@ftw.at

Abstract— **In this paper we evaluate four different metrics for non intrusive bottleneck detection based on TCP counters. Three using a reference and one being reference free. This work is based on the full TCP statistics recorded on five days over the last one and a half year within the live core network of a mobile network operator in Austria. We used two of the datasets, which had a known bottleneck in place as a reference for the metrics. We generated scatterplots, so called footprints, using the number of packets and the number of retransmissions for each user in the peak hours around 7–11 p.m. for all traces. Based on the bottleneck reference we tested the metrics for the detection probability of a bottleneck event. All four metrics detected the bottleneck with no problem, also the throughput figures between 2006 and 2007 had increased. The performance of a simple correlation performance was similar to other metrics based on more sophisticated functions.**

I. Introduction

In this paper we evaluate metrics for non intrusive bottleneck detection in a mobile core-network using TCP (Transmission Control Protocol) related counters. The protocol guarantees reliable exchange of data between sender and receiver. This is archived by the retransmission of data packets. The number of retransmission in the network is an indicator for losses. In wired networks the probability to lose a packet due to a link error is very small, therefore a high number of retransmissions indicate a congestion in the network. Due to errors on the radio link the BER (Bit Error Rate) in a mobile UMTS (Universal Mobile Telecommunications System) network is larger than in wired networks. Coupled with a higher delay many retransmissions are caused by the physical errors rather than by network congestion [1, 2]. Retransmissions are a part normal operation in a mobile environment. Therefore the detection of a bottleneck purely based on packet loss can be ambiguous. There are two main tracks to analyze available bandwidth, the passive and the active. Active tools generate traffic patterns to evaluate the available bandwidth via certain routes [3]. These tools are able to extract exact figures under all network conditions — passive tools can only work if there is user traffic — however the implementation puts extra load to all network components. Common passive tools record the actual load and compare it with the available bandwidth on the local link. Ref. [1] extended the idea applying statistical methods on the bandwidth values. The key idea is to extract the second and third order moments for different bandwidth values. A bottleneck is detected via a decreasing variance for higher numbers. The same authors show in Ref. [2] that this effect is due to the TCP mechanisms. To analyze the effect more deeply, we used an improved version of TCP-Trace presented in Ref. [4] to extract retransmission events on the TCP layer. This processing is useful if slowly approaching a bottleneck. However it does not state if or if not the link is already in saturation. Therefore we started to look for another indicator of bottlenecks.

A straight forward extension to the use of retransmitted packets, n_i, is a metric that is based on the normalization n_i/N_i, where N_i is the total number of packets for this user. Mobile data traffic is always related to a specific user, therefore it is possible to extract these numbers on a per user base. However the burstiness of the link error can rise false alarms, e.g. small number of N_i on bad radio conditions. To avoid this problem we decided to work with pairs of $[N_i, n_i]$ in scatter plots and applied a distance metric to compare the different scenarios. The goal was to find a metric that can be used as an indicator for hidden, e.g. not at the observed link, bottlenecks based on retransmissions. The anomaly is only visible as a part of the total traffic monitored.

The paper is structured in the following way. Section II presents the measurements setup, which is based on the METAWIN testbed developed at the ftw. Furthermore we use this section to describe the used traces. Section III we analyze the samples. In a first step only visually and then in a second step via a metric based approach. The last sections show related work and present the conclusions.

II. Measurement Setup

The reference network scenario is depicted in Fig. 1. As most access networks, the 3G mobile network has a hierarchical tree-like deployment. The mobile stations and base stations are geographically distributed. Going up in the hierarchy (first

BSC/RNC, then SGSN, ultimately GGSN) the level of concentration increases, involving a progressively smaller number of equipments and physical sites. In a typical network there are relatively few SGSNs and even fewer GGSNs. Therefore it is possible to capture the whole data traffic from home subscribers on a small number of Gn/Gi links. For further details on the structure of a 3G mobile network refer to [5].

To meet privacy requirements traces are anonymized by hashing all fields related to user identity at the lower 3G layers (e.g. IMSI, MSISDN), while the user payload above the TCP/IP layer is removed. The TCP statistics were extracted using a modified version of `tcptrace` [6].

The input traces were captured on a live GPRS/UMTS network at the G_n interface by the METAWIN monitoring system [7]. It is a monitoring tool designed to record traffic in a mobile core network. Although the underlaying protocols and interfaces are similar to normal core networks, the presence of user mobility introduces intermediate protocols between the transport network and the user data (see [5]). Therefore METAWIN has to accomplish two main tasks: decoding the additional protocols and tracking the individual user sessions. In addition to this the system anonymises all the traces and strips off the payload in order to protect the privacy of the customers. This preprocessing allows to do research on live traces. The system is capable to monitor at three interfaces: $IuPs$, G_n and G_i. The G_i interface is a normal Ethernet interface between the mobile and the internet service provider. No further user specific information is transmitted. The $IuPs$ protocol is used between SGSN and RNC (Radio Network Controller) units. Although the user specific information is there, the extraction and reassembling is, due to the high number of links and protocol stacks, very complex at this interface. Due to this facts we decided to extract the G_n interface, because it provides both: few interfaces and the whole user specific information.

The extraction parser called MOTRA dumps the packet traces to a ring-buffer. We extended the TCP Trace[1] program within our project to be capable of reading such input files and processing session data.

This work is based on TCP statistics on the G_n interface refined per user. This is possible because in a mobile network, like GPRS or UMTS as well as their further evolutions brothers EDGE (Enhanced Datarates for Global Evolution) and HSDPA (High Speed Data Packet Access), each data packet is dedicated to a specific user. However, due to ciphering and security only session keys are stored in the packet header of the transport protocol visible on the interfaces (e.g. GPRS Transport Protocol (GTP) on G_n). The connection setup contains the user identifier (IMSI) and the session key. Therefore the monitor unit has to perform some kind of tracking.

As the traces were captures in a live network they include several error sources like misbehaving terminals (see [8]). These errors impact the tracking, and reduce the tracking rate to $\geq 98\%$. The remaining 2% of packets could not be addressed to any user and were excluded from the further processing. In addition to this we filtered out the TCP ports 135 and 445 because most of spurious packets in the network address these ports, e.g., from port scans and/or attacks see [9].

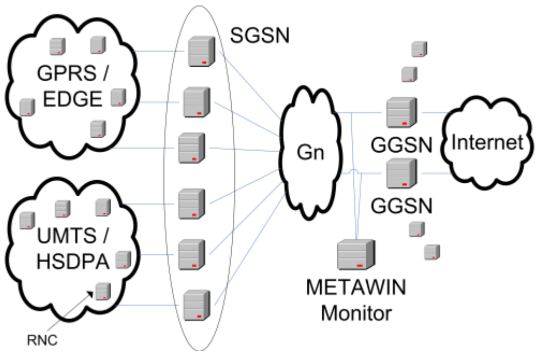


Fig. 1. Measurement Setup

The core dataset used in this work consists of the full TCP statistics for one UMTS SGSN over five different periods: one day in March 2006, one day in September 2006, one day in April 2007, four hours around the maximum load in September 2007 and finally four hours around the maximum load in October 2007. The two youngest traces focus only at a time frame from 7 p.m. to 11 p.m., including the peak hour around 8 p.m. in the evening.

Two samples are expected to be different from the rest: $\mathtt{s}_1$ and $\mathtt{s}_4$ where recorded with a bottleneck in place. The first sample was already used in the last publication and showed a clear difference when compared to footprints from normal operation. However in the following the second trace will be of more interest, as we monitored the same GGSN one week later without the bottleneck and recorded it to $\mathtt{s}_5$. Now we are able to draw a direct comparison between traces taken at the same weekday with and without bottleneck.

III. Analysis of TCP Footprints

We have found in [10], and which is quite straight forward, that the bottleneck will be visible around the daily peak throughput rate first. For the following section we reduced the older datasets to the same time frame from 7 p.m. to 11 p.m. and will only compare these figures. The measurements were taken from the live network of a major mobile provider in Austria, EU. Hereafter the datasets will be indicated by $\mathtt{s}$ with an index starting at one for the oldest set and ranging up to five (e.g., $\mathtt{s}_3$) represents the trace taken in April 2007.

The TCP statistics $\langle N_i, n_i \rangle$ were extracted for 30 minutes time bins. As the impact of a bottleneck is more evident during the peak hour we focus our analysis only on the period from 7 to 9 pm, for a total of four bins. We used scatterplots to visualize the process $\langle N_i, n_i \rangle$, i.e., to create "TCP footprints". As both variables span several orders of magnitude we introduced a logarithmic binning with 150 bins on each axis. The color of each pixel represents the number of occurrences within the bin.

[1] A diff package can be downloaded from the following homepage `http://userver.ftw.at/~vacirca/`.

We already described this procedure in [10]. There we compared one bottleneck trace with several different footprints without bottleneck. However with the new traces we are now able to see the long term evolution of the TCP footprints. This evolution is important to decide whether the approach we chose in [10] is applicable for anomaly detection or not.

A. TCP Footprints

In this paragraph we will present the unfiltered and not normalized footprints for the different traces recorded over the last two years. The following Fig. 2 depicts the scatterplots for $\mathtt{S}_{1-5}$, Fig. 2(a), 2(b), 2(c), 2(d), 2(e), and a GPRS footprint to extend the intra technology comparison Fig. 2(f).

At a first glance we can see that there is a difference between the leftmost figures in line one and two. In fact these two depict the bottleneck cases. Furthermore there is no significant difference in the shape of the footprints figures 2(b), 2(c), 2(e), which represent the good cases ($\mathtt{S}_{2,3,5}$). However, there is a small shift of the total number of transfered packets per user (N_i). This indicates an increased size of the traffic flows each user is downloading.

Comparing the two bottleneck traces, Fig. 2(e) and Fig. 2(a), we observe a shift in N_i. The shape itself is similar, however, the smaller footprint in March 2006 indicates a stronger congestion in place than the sample in September 2007. We will come back to this later when applying various metrics to the traces.

It is striking how similar the GPRS sample is compared to the UMTS samples. This is an interesting observation, since the different radio technologies offer different kind of capacity and therefore one could expect different shapes in footprints. We assume that the TCP footprint is caused by the services used, which are basically dominated by HTTP as we have shown in [11].

Comparing the figures in the first line, recorded in 2006, and the figures in the second line, recorded in 2007, we notice a visible increase in traffic. We will see later in the paper that although the traffic figures differ dramatically between the two years, after a normalization the metrics are capable do deal with this fact. This result indicates that the shape of the footprint is invariant against traffic changes. Please note that we are not allowed to publish full usage patterns, however, the color bar is in logarithmic scale!

Concluding this paragraph we can say that there is a clearly visible difference between a footprint affected by a bottleneck and a footprint recored under normal operation conditions. In addition we see that there is only slight difference in the shape after one year of network evolution and also between two fundamentally different technologies like UMTS and GPRS. The only visible difference between the figures is a shift in N_i toward higher values.

B. Matching the TCP Footprints

In this paragraph we apply four different metrics to analyze footprints for bottleneck situations. To make the benchmarking more robust to outliers we applied the following preprocessing steps In a first step we filtered the datasets using a median filter to remove outliers. In a second run we applied a mean filter with a window size of five to smooth the footprint. In a final step we normalized the value of the bins by the number of events in the traces. This step is also needed to eliminate the growth of the population between the different measurement periods. The datasets are now equal to an empirical binned bi-dimensional probability density function. This step compensated the varying amount of user traffic in the different measurement periods.

Δ_C	s_1	s_2	s_3	s_4	s_5
s_1	1.00	0.41	0.27	0.87	0.43
s_2	—	1.00	0.93	0.49	0.94
s_3	—	—	1.00	0.38	0.97
s_4	—	—	—	1.00	0.48
s_5	—	—	—	—	1.00

TABLE I

CORRELATION RESULTS

In order to coop with the increase in N_i we calculated a center of gravity for each plot. In metrics that need a reference, like the correlation coefficient, we aligned the values for N_i before applying the metric. Please note that this was not done in our last publication. Therefore, the values may differ, in fact some values improve while other values degrade.

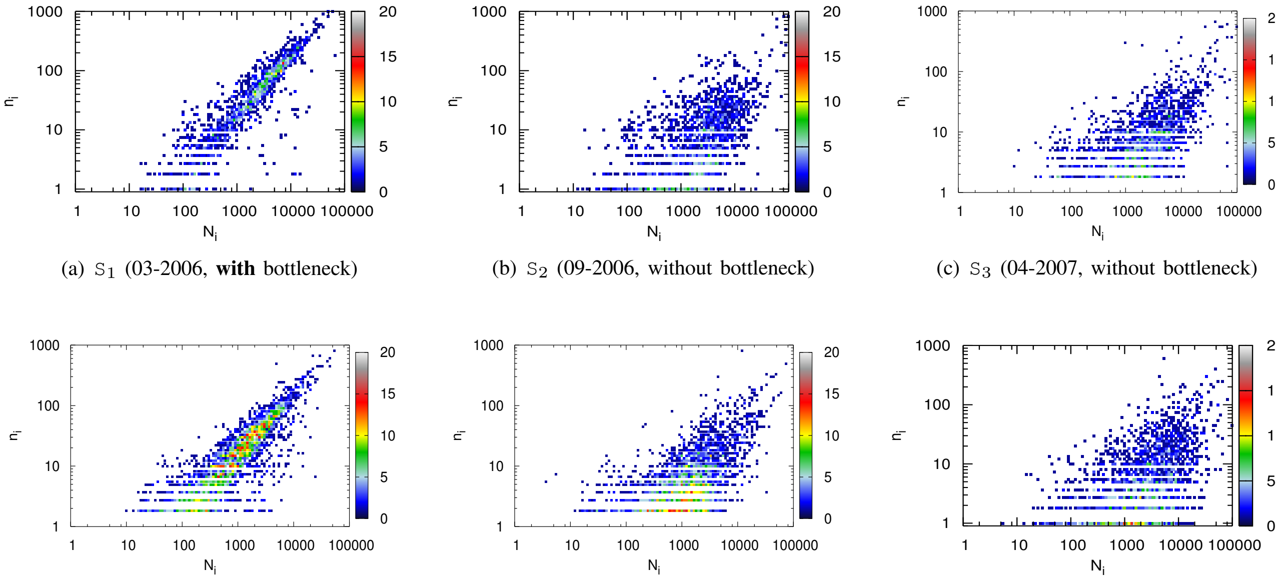
a) Correlation Coefficient: The first metric we tested was the 2-D correlation coefficient (Eq.(1)) for all permutations. Hence the parameter is relative simple to compute (see [12]) we decided to use it as a starting point. A correlation coefficient returns a dimensionless value in the interval of -1 to 1. A value of 1 indicates a perfect correlation while a value of 0 indicates that there is no dependency between the two parameters.

$$\Delta_C(A,B) = \frac{\sum_x \sum_y (A_{xy} - \overline{A}) \cdot (B_{xy} - \overline{B})}{\sqrt{\sum_x \sum_y (A_{xy} - \overline{A})^2 \cdot \sum_x \sum_y (B_{xy} - \overline{B})^2}} \tag{1}$$

The results of this first metric are presented in Table I. As the metric is symmetric we only write down the upper half of the resulting matrix for all five values.

The results for the different datasets are similar to what we have already observed visually. The coefficients between good data sets, e.g., $\mathtt{S}_{2,3,5}$, are well above 0.9. If we compare good with bad datasets the coefficient is around 0.45. It is interesting to see that the coefficient for good values compared with $\mathtt{S}_4$, which, as we have seen earlier, was a much less limiting bottleneck as in the $\mathtt{S}_1$ case, does not rise above 0.5 leaving a gap of 0.4 to detect such cases. In case we compare the bottleneck traces, $\mathtt{S}_1$ and $\mathtt{S}_4$, we also get a high correlation of 0.87. We conclude that this metric is well suited to detect such kind of events.

b) Kullback-Leibler distance: The second metric we evaluated was the Kullback-Leibler (KL) distance between the datasets. The KL-distance is a distance measure between a given probability distribution P and an arbitrary distribution



(a) $\mathtt{s}_1$ (03-2006, **with** bottleneck)

(b) $\mathtt{s}_2$ (09-2006, without bottleneck)

(c) $\mathtt{s}_3$ (04-2007, without bottleneck)

(d) $\mathtt{s}_4$ (09-2007, **with** bottleneck)

(e) $\mathtt{s}_5$ (10-2007, without bottleneck)

(f) GPRS 09-2006 without bottleneck

Fig. 2. Scatterplot of N_i over n_i in the peak hours (log-binning, log scale)

Q. Often P represents some reference data obtained by measurements and Q is generated by a model approximating P. Eq. (2), shows the KL-distance metric.

$$D(P\|Q) = \sum_i P(i) \cdot \ln \frac{P(i)}{Q(i)} \tag{2}$$

The KL-distance is known to be very sensitive against changes, this is why we chose this metric. We applied the distance to our two dimensional domain. The logarithm in this formula was a problem for our datasets with empty bins. In this case it is not possible to use only the common domain, as it would neglect outliers outside this region. In case Q is zero the fraction will be undefined. To overcome this problem we modified the equation in such a way that we only calculated the sum over all bins where P and Q did both differ from zero. If we still encounter places where Q equals zero we added a constant offset $c = 0.01$ to these values of Q.

Note that the KL-distance is not, in general, symmetric (see [13]). We defined a symmetrized version of the metric as in Eq. (3), similar to [14].

$$M_{[p,q]} = \frac{1}{2} \cdot (D(P\|Q) + D(Q\|P)) \tag{3}$$

The value of M obtained by Eq. (3) is zero, if P equals Q and larger than zero in any other case. In addition to these steps each pair of values in Table II, e.g. $[s_1, s_2]$ and $[s_1, s_3]$, is divided by a factor of $M_{[s_1,s_2]}$. Equation (4) illustrates this normalization.

$$\Delta_{KL}(s1) = \frac{M_{[s_1,s_2]}}{M_{[s_2,s_1]}} \tag{4}$$

The second metric calculates the distance values given in Table II. Again the matrix is symmetric and we left out the lower part of the results. The KL-distance is equal to zero in case of equality, therefore every element on the main diagonal is equal to zero.

Δ_{KL}	s_1	s_2	s_3	s_4	s_5
s_1	0.00	1.74	3.65	0.32	1.59
s_2	—	0.00	0.40	1.20	0.34
s_3	—	—	0.00	1.75	0.35
s_4	—	—	—	0.00	1.23
s_5	—	—	—	—	0.00

TABLE II

RESULTS FROM A SYMMETRIC KL-DISTANCE

The basic results are identically to the first metric. The bottlenecks are clearly detected, note this time lower values indicate higher similarity. Again there is a clear gap between the two cases observed in the network.

The interesting result we obtain here is the fact that the manipulation of the center of gravity delivered a degraded performance in this case, compared to our previous publication. This problems come from the different distributions in case of bottleneck and non bottleneck scenarios. Therefore, the shift via the x-axes increases the KL distance. However, the new preprocessing leads to a much higher detection sensitivity.

c) Principle Component Analysis: The third metric is based on a PCA (Principle Component Analysis). In contrast to the first two metrics, this metric does not rely on a reference. This has several advantages, first we do not have to define a *reference* day in order to use our metric and second as we only consider the shape, we are independent from shifts seen in N_i over the last months.

The PCA is a mathematical transformation. It is based on the assumption that variance is a measure of information. We apply a linear orthogonal transformation which converts the

PCA	s_1	s_2	s_3	s_4	s_5
λ_1	33.3	26.8	27.7	31.1	25.8
λ_2	5.82	12.5	13.0	8.16	12.4
Fraction	5.72	2.14	2.13	3.81	2.08

TABLE III

RESULTS FROM A PRINCIPLE COMPONENT ANALYSIS

PSNR	s_1	s_2	s_3	s_4	s_5
s_1	∞	20.3	18.1	24.9	19.2
s_2	—	∞	25.0	18.9	27.7
s_3	—	—	∞	20.4	30.9
s_4	—	—	—	∞	21.6
s_5	—	—	—	—	∞

TABLE IV

PSNR VALUES IN dB

original coordinates to a new coordinate system such that the projection of the greatest variance is identical with the first coordinate. This first coordinate is also called first principle component.

In practice PCA is often used to reduce dimensionality. This is possible by omitting the higher principle components, depending on the information left on those variables.

Given a set of N data vectors $\mathbf{x}_1 \ldots \mathbf{x}_N$, where each vector $\mathbf{x}_n$ is a single observation of the M variables, here M is de dimension of the underlaying dataset, which is two in our case. We first generate a Matrix **X** of the size MxN, with one row per variable and one column per observation. To apply a rotation we have to remove the mean, in our case the empirical mean of **X**. Now we need a orthonormal transformation matrix **P** of the form:

$$\mathbf{Y} = \mathbf{P}^T\mathbf{X} \tag{5}$$

such that **cov(Y)** is a diagonal matrix. After some matrix manipulations we get the final results,

$$\mathbf{Pcov(Y)} = \mathbf{cov(X)P} \tag{6}$$

which shows that the new matrix **P** can be found by calculating the eigenvectors of **cov(X)**. However, as we only have one observation, we have to calculate the empirical covariance matrix.

Table III shows the results for a the variance in the rotated coordinate systems for each sample. In case of a bottleneck, e.g., $s_{1,2}$, the variance in the second direction is smaller. This can also be interpreted with the fact that the footprint turns into a small ellipsoid like figure in the case of a bottleneck. Whether under normal operational conditions the footprint has a higher variance in the second component and a lower in the first, in other words the footprint is more round in this case. We then calculated the fraction for the variance in the first and the second component, e.g., the energy in the different components. Under normal operation the fraction stays around 2, while in a bottleneck scenario the fraction rises up to 4 in case s_4 or even up to 5 in case of s_1. There are now several advantages in this method: fist it does not need any preprocessing like the shift in N_i, second it can be used reference free and third it has the capability to detect the difference between the two bottlenecks. In fact s_1 was a quite heavy restriction in traffic, while s_3 was only an upcoming bottleneck, which is visible in the different energy distributions. However, the computational effort here is higher as in a simple correlation case.

d) Peak Signal to Noise Ratio: We used a peak mean square error as a forth metric, also called PSNR (Peak Signal to Noise Ratio). This is a term from engineering, which is used to compare the maximum possible power of a signal and the power of a corrupting noise. It is commonly expressed in logarithmic scale. The PSNR is often used in image processing as a benchmark for image quality.

In a first step the mean square error is calculated for every bin of P and Q. The result is then normalized to the maximum level of P and Q. As we have renormalized these values already before to minimize the effect of a population growth, we can directly compare any two scatterplots. This metric needs a reference, therefore we again would have to set a *reference* day in order to perform a benchmark. However, we will not consider this problem here. In case P equals Q the metric will go to infinity. A lower values indicates less similarity between P and Q.

$$mse(P,Q) = \frac{1}{m \cdot n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \|P(i,j) - Q(i,j)\| \tag{7}$$

$$PSNR = 10 \cdot \ln \frac{I_{max}^2}{mse} \tag{8}$$

The results itself gain a lot of improvement if we apply the shift of the center of gravity. This comes from the fact that this metric is extremely sensitive to small shifts, whereas it becomes less sensitive for larger deviations, e.g., a single pixel LSB (least significant bit) error results in a PSNR number of about 91dB, two pixel errors result in 2.3dB reduction. Therefore, this metric gains most from a good alignment between the traces.

The values for the PSNR metric are given in Table IV The numbers itself range around 20dB in case we compare a bottleneck with a non bottleneck case and around 25-30dB in case we compare good cases. With other words there is a clear gap of at least 5dB usable for detection. Therefore, this metric is very simple and on the other hand effective to detect this difference.

IV. PERFORMANCE ANALYSIS

In this paragraph we want to benchmark the performance of different metrics for an artificial generated dataset. The new datasets $s_{new[x]}$ was generated by randomly taking samples from the two dataset s_1 and s_5. The value x indicates the percentage of samples taken from s_5, e.g., $s_{new[80]}$ is generated from samples that originate to 20% from s_1, or in other words this should represent a relatively congested szenario.

$s_{new[x]}$	Corr	KL-dist	PSNR	PCA
$x =$	30%	25%	26%	20%

TABLE V
SHARE OF CONGESTED TRAFFIC TO RAISE AN ALARM

Table V present the detection result for the different metrics. In the correlation based metric we took the lowest score for bottleneck free minus a 10% margin to trigger an alarm. The alarm is triggered in case that approximately 30% of traffic come from the congested trace. For the KL-distance we took the largest distance from a bottleneck free operation and added again 10% as an detection threshold. In this case the alarm was triggered for 25% of congested traffic. Applying the same rules for PSNR lead to a detection limit of 26%. The PCA was able to lower the bound to even 20% of congested traffic.

Concluding this benchmark we can say that PCA wins, however, at a higher cost of complexity than a simple correlation. We think that PSNR is the best trade off between complexity and detection threshold reached.

V. SUMMARY AND CONCLUSIONS

In this paper we present four different methods for non intrusive bottleneck detection in a cellular mobile core network based on counters for TCP related events. The datasets we used were recorded in a live 3G network at an operator in Austria. We worked with five different datasets spanning nearly one and a half year of network evolution, of which two represent known bottleneck problems in the network. A first interesting result was that there is now significant difference in shape between the non bottleneck cases although there was a huge increase in throughput between 2006 and 2007. We observed only a slight shift in the size of a TCP stream, called N_i for the number of packets in a stream.

We used four different metrics in this paper, namely: the correlation coefficient, a modified KL-distance, a PSNR method and a principle component analysis. The first three metrics give a distance between the actual footprint and a reference. Therefore, one needs two references, e.g., one with and one without bottleneck. In contrast to these the PCA method needs no reference and can directly tell if there is a bottleneck or not. This last metric identifies the shape of the given footprint.

All four metrics allowed a clear detection of the bottleneck. It was interesting to see that also the second bottleneck, which had a much smaller impact on the footprint, is detected flawless. By applying a shift in N_i, e.g., aligning the center of gravity, the detection event even works for traces that where taken more than one year apart.

We concluded that the correlation coefficient is reliable enough to detect a bottleneck footprint. However, in case one does not have a reference bottleneck, and this maybe true for most of the operators, the PCA is the better choice because it does not need a reference footprint.

In the following we want to focus on the fact that the footprints in GPRS do look similar to extend the detection to the total traffic in the network.

ACKNOWLEDGEMENTS

This work is supported by the K-Plus initiative of the Austrian government. We thank mobilkom austria AG and the Kapsch Karrier Com for technical and financial support of this work. The views expressed in this paper are those of the authors and do not necessarily reflect the views within mobilkom austria AG.

REFERENCES

[1] Fabio Ricciato, Francesco Vacirca, and Martin Karner. Bottleneck detection in umts via tcp passive monitoring: a real case. In *CoNEXT'05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, pages 211–219, New York, NY, USA, 2005. ACM Press.

[2] F. Ricciato, F. Vacirca, and P. Svoboda. Diagnosis of capacity bottlenecks via passive monitoring in 3G networks: an empirical analysis. *Computer Networks*, 57:1205–1231, March 2007.

[3] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *Network, IEEE*, 17(6):27–35, 2003.

[4] Francesco Vacirca Fabio. Large-scale rtt measurements from an operational umts/gprs network.

[5] H. Holma and A. Toskala. *WCDMA for UMTS, Radio Access For Third Generation Mobile Communications, Third Edition*. Wiley, 2004.

[6] Modified tcptrace. `http://userver.ftw.at/~vacirca`. FTW.

[7] METAWIN Project. `http://www.ftw.at/ftw/research/pro` FTW.

[8] Fabio Ricciato Forschungszentrum. Unwanted traffic in 3G networks.

[9] R. Pang et al. Characteristics of Internet Background Radiation. *Proc. of the International Measurements Conference (IMC'04), Taormina, Sicily, Italy.*, October 2004.

[10] P. Svoboda, F. Ricciato, and M. Rupp. Bottleneck footprints in TCP over mobile internet accesses. *Communication Letters*, 11(11):839–841, 2007.

[11] P. Svoboda and F. Ricciato. "Composition of GPRS and UMTS traffic: snapshots from a live network". *IPS MoMe 2006, Salzburg*, pages 42–44, Feb 2006.

[12] T.C Moon and W.C. Stirling. *Mathematical Methods and Algorithms for Signal Processing*. Prentice-Hall, 2000.

[13] S. Kullback and R.A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22:79–86, 1951.

[14] D. Johnson and S. Sinanovic. Symmetrizing the Kullback-Leibler distance. *Technical Report*, 2001.