Adaptive Signatures of Soft-Failures in End-User Devices Using Aggregated TCP Statistics

Chathuranga Widanapathirana, Jonathan C. Li, Milosh V. Ivanovich, Paul G. Fitzpatrick, and Y. Ahmet Şekercioğlu
Dept. of Electrical and Computer Systems Engineering, Monash University, Australia
{chathuranga.widanapathirana, jonathan.li, milosh.ivanovich, paul.fitzpatrick, ahmet.sekercioglu}@monash.edu

***Abstract*—We present a new approach for effective soft-failure characterization in end-user devices (EUDs) on networks that support the TCP/IP. Our method can be employed for creating fully automated, accurate and scalable fault diagnostic systems.**

First, we describe Normalized Statistical Signatures (NSSs), a technique for characterizing EUD soft-failures. We create the NSSs by using aggregated statistical features extracted from TCP packet streams collected on-demand upon user complaint.

We then introduce the Link Adaptive Signature Estimation (LASE) technique to minimize the number of NSSs needed to create diagnostic systems that have generalization capability for coping with communication link variations. To achieve this, we create Feature Estimator Functions (FEFs) using multivariate regression techniques and a minimal number of signatures of emulated EUD faults. We use these FEFs to generate synthetic NSSs which, can be used to train diagnostic systems with robust generalization capabilities. We expect that the combined use of NSSs and LASE technique will serve as the foundation of next-generation fault diagnosis systems.

I. Introduction

A fault in a networked device that causes loss of bandwidth or a degraded level of performance from an expected benchmark is commonly defined as a "soft-failure" [1]. The diagnosis of soft-failures in End-User Devices (EUDs) is critically important for maintaining Quality of Experience (QoE) of users [2].

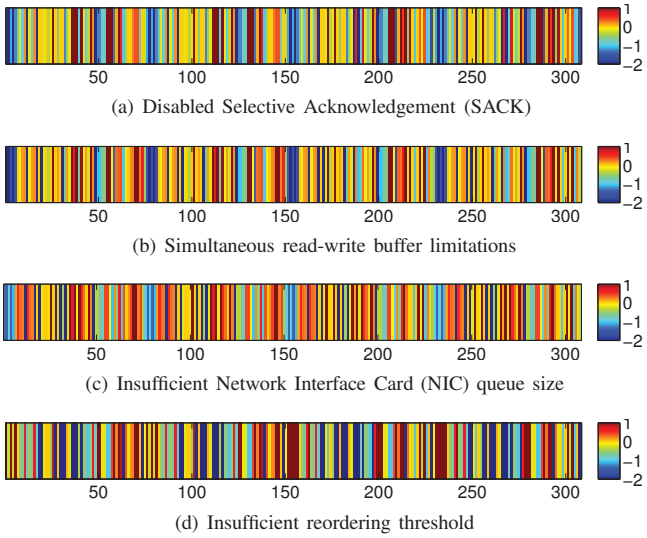
Recent studies have shown that Machine Learning (ML) based detection techniques can provide the automation desired in a diagnostic system [3], [4], [5]. ML-based techniques also offer better scalability compared to systems based on "expert-rules" such as pathdiag [6] and NDT [7].

These techniques rely upon the ability to detect and characterize the runtime behavior of networked devices to create a unique "signature" of a network or device failure. As remote access to a EUD is usually not possible for a network service provider, it is desirable to capture its runtime behavior by observing the device's interaction with the connected network.

Network signatures are often used in online traffic classification [8] and traffic anomaly detection [9]. Anomaly detection can be divided into Intrusion-Detection Systems (IDS) [10], and connectivity or soft-failure detection systems [9]. Traffic classifiers and IDS systems generate signatures by analyzing port numbers, application-specific payloads, and statistical properties of traffic volume to "fingerprint" a specific source [5], [8], [10], [9]. Often generated using live traffic, these signatures are specific to a particular detection task. In addition, these studies use evaluation data captured only from a single network location and the effects of network or link variations on signatures have been often overlooked. Attempts to use such signatures for soft-failure detection at the device-level result in unacceptably high false detection rates [11]. Source protocol variants [12], dependance on stable link properties, insufficient number of features, and sample sizes make the existing signatures published in literature unsuitable for soft-failure diagnosis in EUDs.

TCP packet streams provide an ideal observation point to identify root causes of the network performance related issues in EUDs. Its position in the middle of the protocol stack and reliable transport functionality allows performance issues in other layers to embed unique anomalies on TCP packet streams. Supported by most network devices today, "controlled" TCP packet streams can be used to extract hundreds of statistically robust features to characterize anomalies and create signatures.

In this paper, we introduce; (i) aggregated TCP statistics-based normalized signatures (NSSs), a scalable and effective way of uniquely characterizing a EUD fault (see Figure 1) for a ML application, (ii) Link Adaptive Signature Estimation (LASE), a technique to dynamically generate NSSs for arbitrary link conditions from a limited set of collected NSSs.



(a) Disabled Selective Acknowledgement (SACK)

(b) Simultaneous read-write buffer limitations

(c) Insufficient Network Interface Card (NIC) queue size

(d) Insufficient reordering threshold

Fig. 1. Comparison of NSSs for four common soft-failures in EUDs. Here, the normalized value of 300 features have been scaled ($1 \leftrightarrow -2$) and then color coded for easy visualization and comparison. The combination of features uniquely represent each fault and can be used as a "fingerprint" for problem diagnosis.

978-3-901882-50-0 ©2013 IFIP

II. Normalized Statistical Signatures (NSS)

The scalability of a diagnostic system is highly dependent on its ability to generalize a "fault specific" signature for a wide variety of network conditions. This dependency poses many challenges, particularly at the device level; Observation point, protocol behaviors, packet capture mechanism, data extraction, and fault generation technique are some key aspects affecting the accuracy and effectiveness of the final signature.

Here, we limit our focus to the scenario where diagnosis is performed from the edge of the service provider domain using the access server as the observation point. This narrows down the path to an access link and eliminates complexities that can affect the uniformity of the features. The signature extraction process comprises of two components: 1) Capture module, and 2) Signature extraction module.

A. Capture module

Most signatures in the literature are generated using arbitrary traffic flows captured in live networks. However, our NSSs are generated using traffic generated by the user (healthy or faulty) on-demand through a web interface. Bi-directional TCP packet streams of an upload and a download of a fixed size file are captured at both the access server and the client device. We use standard packet capture libraries that do not require kernel manipulations. Constant transfer size, protocol, capturing parameters, and static server configuration provide a baseline performance characteristic, unavailable when using live traffic captures. The process also includes a fault emulator that manipulates the client to emulate common faults, in either the live network or a regulated test-bed. Collecting packet traces over regulated test-beds offers the additional advantage of link condition (bandwidth, delay, packet loss rate) control.

B. Signature extraction module

Collected TCP packet traces are analyzed using a tcptrace-based [13] sequential analysis tool and stored in a database $\Theta_{\text{cfd}} = \{(\mathbf{x}_1, y_1, \mathbf{z}_1), \ldots, (\mathbf{x}_n, y_n, \mathbf{z}_n)\}$. Here $\mathbf{x}_i \in \Re^m$ is the m-dimensional feature vector, $y_i \in \{\text{cf}_0, \ldots, \text{cf}_p\}$ is the fault label associated with the feature vector $\mathbf{x}_i$, p is the number of cases, $i = 1, \ldots, n$, and n is the number of signatures. $\mathbf{z}_i$ is the vector of link properties for the sample. The feature vector $\mathbf{x}_i$ combined with the class label y_i and link properties $\mathbf{z}_i$ is called the "signature" of the i^{th} instance.

For each case, we combine the upload trace with the download trace and extract approximately 320 total features, each an aggregated statistical attribute of the trace. The features can be broadly (but not exhaustively) categorize into following:

1. Cumulative totals of various packet types,
2. Cumulative payload characteristics,
3. Max, min, mean observation frequencies of events,
4. Max, min, mean information on variable parameters,
5. Initial state and final state parameters,
6. Round trip time, arrival time progression analysis,
7. Limited set of boolean parameters of TCP settings.

For a given access link, traces captured from a "healthy" device with optimum system settings provide a base performance signature. We normalize the raw statistical data in "faulty" signatures (collected with emulated device faults) against the healthy baseline signature for that particular access link, resulting in the creation of NSSs for various EUD faults.

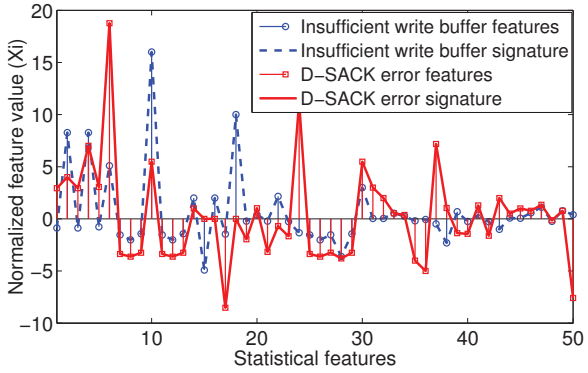


Fig. 2. Comparison of features in NSSs for devices with insufficient buffer and D-SACK error.

Figures 1 and 2 show two different representations of NSSs of common EUD problems. In Figure 1, the feature values have been color scaled ($1 \leftrightarrow -2$) for easy visualization and comparison. While the NSS extraction process appears to yield unique signatures, however, it is necessary to investigate the variation in NSS features between multiple samples.

The Figure 3 shows 50 normalized features from 30 samples of write buffer error NSSs. Taken over a stable link of a live network, we can clearly identify three types of features; 1) stable features (significant) that deviate from baseline, 2) features that do not deviate (insignificant) from baseline, and 3) unstable features. The figure shows that for every fault, only a subset of features provides consistent and significant information towards an identification. The impacted feature subset also varies with different types of faults. Having a large feature set therefore enables the NSS to characterize a wide range of faults through a single representation. A subsequent "feature selection" algorithm can be used to filter out insignificant/unstable features for individual fault types, for use in a machine learning fault diagnostic system.

The properties of the access link have a significant impact on the NSS. A comparison of NSSs collected in different link conditions showed that whilst a small subset of fea-

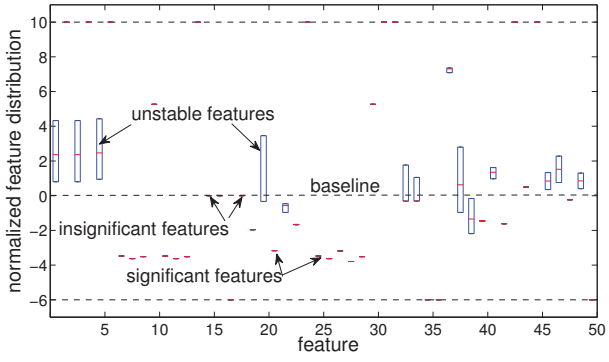


Fig. 3. Variance of write buffer error feature over 30 samples over a stable link

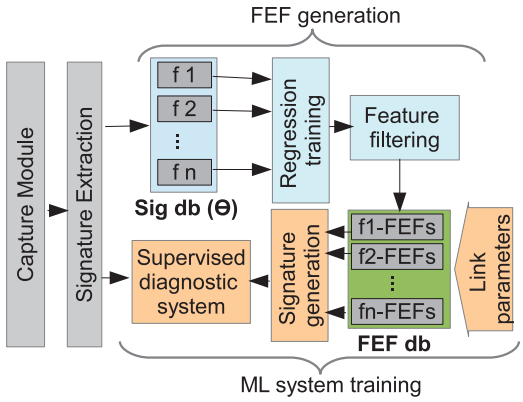


Fig. 5. Operational stages of LASE and diagnostic system training

tures remained link independent, the majority of the features are affected by link properties such as latency, supported bandwidth, packet loss, congestion, etc. The capabilities of supervised ML systems, however, are limited by the training samples. It would be impractical to collect enough samples to train a system for exhaustive combinations of link parameters. Consequently, there is a tradeoff between maximizing the ML diagnostic system performance and minimizing the number of NSS samples required for training. Analytically modeling the performance relationships between link properties and features has thus far proven unsuccessful.

III. Link Adaptive Signature Estimation (LASE)

To avoid the need of complex analytical model or a large data set, we propose the concept of Link Adaptive Signature Estimation (LASE).

A. Overview

Based on multivariate regression techniques, LASE collects a smaller sample set for a EUD fault while varying the link parameters. This sample set is used to train a Feature Estimator Function (FEF) which can then create NSSs for specific fault, for any link within the parameter training range. These link specific NSSs can then be used to train the ML diagnostic system. Figure 5 shows the two main stages of LASE; 1) FEF generation, and 2) training of the supervised ML system. The "capture" and "signature extraction" modules collect controlled healthy and faulty NSSs over different links with delay, bandwidth, loss, etc combinations. These NSSs, together with the link parameters, are sent to regression training.

Regression is the process in which models (FEF) are fitted to observational data to identify the complex relationship between dependent variables (feature values) and independent variables (link parameters). FEFs are generated for all of the features in each type of fault NSS. However, as not all features are predictable, FEFs are tested and only the FEFs of predictable features for different faults are stored in a database.

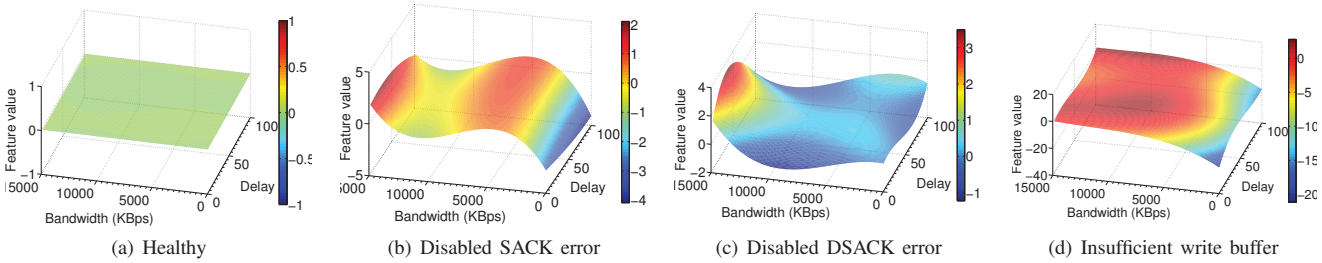
The stored FEFs are used to estimate NSS feature values (with a high degree of accuracy) for any given combination of link parameters (within the link parameter training range). When training a EUD diagnosis system for a specific link, FEF generated NSSs can be used either entirely, or in combination with those collected over the live network.

B. Experimental evaluation of LASE

As we do not have access to live ISP network environments, the performance of the LASE system was evaluated experimentally using network data collected in a laboratory test network consisting of an access server, dummynet-based link emulators, and end-user devices. The LASE was evaluated for common EUD bottlenecks and four cases discussed in [4] are presented here. We collect NSS samples whilst varying bandwidth between 100 KB/s-15 MB/s, and link delay between 0 ms-100 ms, which covers typical residential access network performance to high speed intranets. Although many link parameters affect the signature, at this stage, we only investigated bandwidth and delay variations as they are the most common in access links. We assume that links are free of congestion and do not suffer from any faults including intermediate devices.

Faults were emulated at the EUDs and TCP traces were collected using the capture module. Two data sets with varying link bandwidth and delay combinations were collected for the healthy and faulty cases. The first data set was used for training the FEFs and the second set to evaluate the performance of the FEFs. The training samples were limited to 5 samples per combination of 5 bandwidth and 5 delay levels (125 samples per fault). Samples were collected over links with parameter combinations that were not used for training, totaling 2200 per fault for statistically robust evaluation of FEF performance.

The regression technique largely dictates the accuracy of the model and is an important design criterion. To build the FEFs, we used three different techniques: 1) polynomial regression with least-square optimization (Poly), 2) general regression neural networks with Levenberg-Marquardt training (NN), and 3) support vector regression with radial-basis kernel (SVR).



(a) Healthy (b) Disabled SACK error (c) Disabled DSACK error (d) Insufficient write buffer

Fig. 4. Behavior of "total-cumulative-acknowledgements" feature over links with different properties links for three EUD problems.

Fault	R^2	Example feature									
		tot.pkts	ack.pkts	avg.segm	avg.win.adv	stream	xmit.time	pure.acks	RTT.3wh	cum.ack	tri.dupacks
SACK error(f1)	Poly	**0.9939**	**0.9939**	0.8934	0.3947	0.5742	0.9875	0.1220	0.9793	0.9852	0.9846
	SVR	**0.9203**	**0.9203**	0.8936	0.0019	0.1111	0.8705	0.1302	0.9995	0.9221	0.9631
	NN	**0.9877**	**0.9877**	0.8925	0.0296	0.5035	0.9921	0.0356	1.0000	0.9821	0.9861
Limited write buffer(f2)	Poly	**0.9865**	**0.9865**	0.9948	0.2600	0.9935	0.9835	0.3608	0.9793	0.9913	0.9877
	SVR	**0.9559**	**0.9559**	0.9994	0.0041	0.9083	0.8143	0.3172	0.9902	0.8908	0.9783
	NN	**0.9890**	**0.9890**	0.9991	0.3947	0.9999	0.9970	0.2292	0.9885	0.9959	0.9998
Limited read buffer(f3)	Poly	**0.9825**	**0.9825**	0.9948	0.9944	0.9928	0.9829	0.3465	0.9793	0.9740	0.9877
	SVR	**0.9694**	**0.9694**	0.9994	0.9695	0.9021	0.8301	0.2536	0.8669	0.8804	0.9779
	NN	**0.9980**	**0.9980**	0.9999	0.9870	0.9999	0.9996	0.1177	0.9911	0.9949	0.9997
Dup-SACK error(f4)	Poly	**0.9939**	**0.9939**	0.8921	0.6254	0.8246	0.3653	0.1520	0.9793	0.9852	0.9846
	SVR	**0.9203**	**0.9203**	0.8918	0.4366	0.1111	0.4705	0.4502	0.9995	0.9221	0.9631
	NN	**0.9877**	**0.9877**	0.8978	0.7506	0.5035	0.5921	0.2456	1.0000	0.9821	0.9861

TABLE I. ESTIMATION PERFORMANCE LASE MEASURED WITH R^2.

The Figure 4 shows the estimated FEFs of the "total-cumulative-acknowledgement" feature for four types of EUDs. The healthy device (Figure 4(a)) shows a flat surface as features are always normalized against the healthy baseline. Figures 4(b), 4(c), 4(d) show the feature behavior with respect to the healthy baseline. The combination of these deviations become the unique NSSs used to train the diagnostic systems to identify faults.

The FEFs are evaluated by measuring how closely feature values generated by the FEFs match the real sample data. We use a common goodness of fit measure "coefficient of determination" (R^2), which is defined by $R^2 = 1 - SS_{\text{err}}/SS_{\text{tot}}$ where SS_{err} is the sum of squares of residuals, and SS_{tot} is the sum of the squared differences from the mean of the feature. R^2 can be between 0 and 1 with value closer to 1 indicating a better fit.

Table I summarizes the performance of LASE for 10 example features for four device faults. We can identify three main categories of features:

1. Features such as **tot.pkts**, **ack.pkts**, **RTT.3wh**, **cum.acks**, **tri.dupacks** can be predicted with extremely high accuracy using all regression models. ML systems such as SVMs are sufficiently generalized against the $< 2\%$ error of prediction.
2. The extent a feature is affected differ with the fault. Feature FEFs xmit.time,avg.segm,stream, and avg.win.adv are only predictable with certain faults. These FEFs are used only to train systems to detect that particular fault. Other faults severely affect the trace to the point that features become sporadic.
3. Non of the models succeeded in predicting pure.acks. These are considered unpredictable features and filtered out from NSSs.

The table shows that the models created using Poly performed slightly better than NN and SVR. However, NN and SVR are more suitable (than the simple Poly technique) to create complex models with higher dimensionality when more link parameters are considered. Also, SVR is more robust when training samples are contaminated with outliers specially when collected over live networks. Only with 125 samples per fault, LASE extended the NSSs (with filtered features) and consequently, the diagnostic capability over links with a bandwidth range of 100 KB/s-15 MB/s and delay range of 0 ms-100 ms. With LASE, the diagnostic systems can be easily adopted to new access links and links with time varying properties.

IV. CONCLUSION

We have proposed a technique to uniquely characterize network faults in end-user terminals towards creating an automated fault diagnostic system. Using aggregated TCP statistics, we created NSSs that are diverse and robust. To improve the application of NSS, we introduced LASE to estimate NSSs for a continuous range of link properties from a small data set. Our experimental results demonstrated the prediction capabilities of LASE and gave us insights on feature sub-types. During future work, the knowledge gained here will be extended to produce state-of-the-art automated end-user device diagnostic systems.

REFERENCES

[1] F. Feather, D. Siewiorek, and R. Maxion, "Fault Detection in an Ethernet Network Using Anomaly Signature Matching," *SIGCOMM Comput. Commun. Rev.*, vol. 23, no. 4, pp. 279–288, Oct. 1993.

[2] S. Lee and H. S. Kim, "End-user Perspectives of Internet Connectivity Problems," *Comput. Netw.*, vol. 56, no. 6, pp. 1710 – 1722, 2012.

[3] C. Widanapathirana, Y. Şekercioğlu, M. Ivanovich, P. Fitzpatrick, and J. Li, "Automated Inference System for End-To-End Diagnosis of Network Performance Issues in Client-Terminal Devices," *Int. Jour. of Comput. Netw. & Comm. (IJCNC)*, vol. 4, no. 3, pp. 37–56, 2012.

[4] C. Widanapathirana, J. Li, Y. Sekercioglu, M. Ivanovich, and P. Fitzpatrick, "Intelligent Automated Diagnosis of Client Device Bottlenecks in Private Clouds," in *Proc. IEEE UCC*, Melbourne, Australia, Dec. 2011, pp. 261 –266.

[5] B. Aggarwal, R. Bhagwan, and T. Das, "NetPrints: Diagnosing Home Network Misconfigurations Using Shared Knowledge," in *Proc. USENIX NSDI*, Boston, MA, USA, Apr. 2009, pp. 349–364.

[6] M. Mathis, J. Heffner, P. O'Neil, and P. Siemsen, "Pathdiag: Automated TCP diagnosis," in *Proc. PAM*, Cleveland, OH, USA, Apr. 2008, pp. 152–161.

[7] R. Carlson, "Developing the Web100 Based Network Diagnostic Tool (NDT)," in *Proc. PAM*, Apr. 2003.

[8] T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification Using Machine Learning," *IEEE Commun. Surv. Tutor.*, vol. 10, pp. 56 –76, 2008.

[9] M. Thottan and C. Ji, "Anomaly Detection in IP Networks," *IEEE T. Signal. Proces.*, vol. 51, no. 8, pp. 2191–2204, 2003.

[10] C. Manikopoulos and S. Papavassiliou, "Network Intrusion and Fault Detection: A Statistical Anomaly Approach," *IEEE Commun. Mag.*, vol. 40, no. 10, pp. 76 – 82, oct 2002.

[11] T. Kihara, N. Tateishi, and S. Seto, "Evaluation of Network Fault-Detection Method Based on Anomaly Detection With Matrix Eigenvector," in *Proc. APNOMS*, Taipei, Taiwan, 2011, pp. 1–7.

[12] C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Behavior Analysis of TCP Linux Variants," *Comput. Netw.*, vol. 56, no. 1, pp. 462–476, Jan. 2012.

[13] S. Ostermann. (2000) Tcptrace Official Homepage. [Online]. Available: http://www.tcptrace.org