Detection of State Transitions in Network elements: On-box demo

Parisa Foroughi[†‡], Wenqin Shao[‡], Frank Brockners[‡], Anil Kuriakose[‡] and Jean-Louis Rougier[†]
[†] Télécom Paris - Department of Networks and Computer Science
[‡] Cisco Systems
Email: parisa.foroughi,rougier@telecom-paris.fr; pforough, wenshao, fbrockne, akuriako@cisco.com

***Abstract*—Modern network devices like routers offer thousands of operational counters. All of them *could* be important for network monitoring, though their high number makes this process infeasible, often resulting in only a small subset of the counters to be considered for further interpretation and processing. This demo paper showcases the practical use of an unsupervised multivariate online detector, DESTIN [1], which could assist an operator in automatically monitoring all or at least a very large number of counters and exploring inter-dependencies between them to further the operator's understanding of the state of the network. DESTIN can detect changes in the network without any need for predefined KPIs on the router itself.**

***Index Terms*—Machine learning, Principal angles, Network management, Change detection.**

I. Introduction

Network operators has long been concerned about achieving a better understanding of their network devices. The common approach is to only monitor the key performance indicators (KPIs) picked based on experience [2] which facilitates the detection of already known events. A noticeable number of the counters which are constantly changing under the normal network dynamics [3] are usually disregarded when it comes to manual checkups and thus it requires more sophisticated models than mere thresholds. Moreover, the actual number of available counters on a device is beyond what a human can monitor manually. An unsupervised multivariate method can consume a large number of counters as input.

DESTIN [1], an online unsupervised multivariate change detection methodology, can infer the relationship between these counters and deduce the network/device state transitions. This paper showcases the practical usage of DESTIN on a router for exploring the traffic counters dependencies and detection of routing loop events.

II. Demo components and architecture

Fig. 1 depicts the four building blocks of an analytics system for a router's operational data structured using YANG [4] model. The system consists of collector, detector, explainer and exporter. The detector leverages DESTIN for detection of state transitions and the explainer block provides hints for the operators to define new rules using the methodology proposed in [3]. This demo shows the possibility of running this solution on the router itself. Therefore, eliminating the need for exporting all the counters out of box for processing. The code is mostly written in the python language.

978-3-903176-32-4 © 2021 IFIP

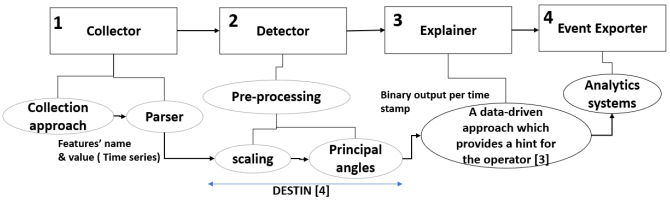

Fig. 1. Building blocks' details of the analytics system including DESTIN [1] as the detector for state assessment of network elements

A. collector

The counters are collected on the box through subscriptions to the streaming telemetry [5] with a regular cadence. The groups of counters to subscribe to are inferred from the configuration of the router (i.e. bgp, ipv6, icmp, etc.). The data collection cadence is automatically tuned according to the router's available resources. There also exists a parser code block which synchronizes all the counters from different subscriptions. The output of this block are equally spaced time-series ready to be fed to the detector block.

B. Detector

The detector block is responsible for data scaling and the detection of state transitions. The time-series data is first pre-processed according to its operational type. The processed data is fed to the detector (DESTIN) which generates an indicator representing its input. The indicator is a translation of the information about the device state from various counter into a uni-variate indicator. DESTIN consumes multivariate data and detects state transitions timely without any need for prior knowledge about the network. A detailed description on the mathematical basis of this indicator is presented in section IV of [1]. The stability of this indicator infers the stability of the relationship in its input data. As long as the indicator does not undergo an extreme divergence from its values, the device state is assumed to remain unchanged [1]. To automate the monitoring process for this indicator generated by the detector, a sigma detector is applied on this uni-variate output as detailed in section IV.B.5 of [1].

C. Explainer

This block is responsible for selecting the counters which can better explain the event occurred at the timestamp flagged by the detector as an event. This block looks into all the available counters and presents hints for the operators to justify the event. More details are available in [3].

D. Event exporter

The system only exports data out of the router when a change (event) is detected. To export the counters and the detector details, a YANG model [4] is defined. It is also subscribed to when running the analytics system.

III. Testbed and Demo Results

The test-bed in this demo (also available at [6] with better quality) consists of 4 spines and 8 leaves connected as shown in Fig. 2. The links connecting the routers use ECMP to distribute the traffic, as detailed in the figure. The fabric uses bgp as its routing protocol. The traffic is generated using ixia, using mainly two types of traffic: UDP traffic as well as a mix of media application traffic such as Webex, Facebook and Netflix summing up to a total of almost 340 Gbps of traffic in the network. The proposed analytics system is enabled on all routers. A routing loop is inserted by pushing static route configurations on dr02 and dr03, causing traffic to loop between dr03, dr02 and leaf8. This demo shows that DESTIN successfully detects the change without any need for KPI monitoring or prior training of an AI model.

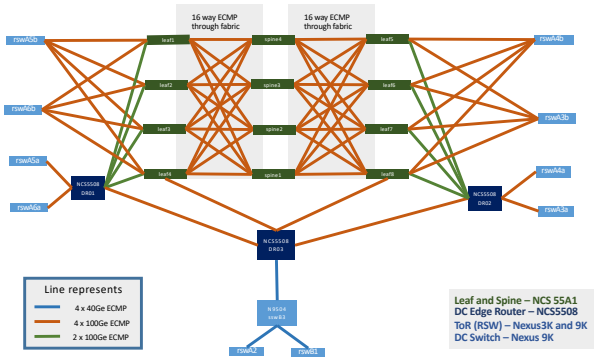

Fig. 2. Test-bed topology

The analytics system starts with a short lag. This is to give it enough time to infer the subscription groups and adjust the cadence automatically according to the configuration and the available memory and CPU. The router in this demo subscribes to $316,000$ counters. The proposed system consumes almost 1000 counters that their value often change by time. The whole system (exporter+detector+explainer) consumes 601 MB of memory and 8% of CPU usage on the router. The routing loop is created by changes in static routes, therefore, from a single device perspective, the routing protocols and forwarding plane are functioning as expected. However, due to the inserted static routes, a portion of the traffic will never make it to its destination. For instance, on leaf8, this traffic loop is about 5 Gbps which is almost 3% of the traffic destined towards specific prefix behind dr03. Since the control plane (the number of bgp sessions, session states, bgp routes from each neighbor etc.) is intact, the change cannot be simply detected by monitoring the control plane alone. The detector (DESTIN) consumes the counters and translates their dependencies into one single indicator updated per each data-point as described in detail in section IV of [1]. It then compares the values of the generated indicator using a simple sigma detector which fires an alarm if the delta between the past and the previous values cross a certain threshold. Fig. 3 depicts the divergence of values when the routing loop occurred. DESTIN's alarm is the trigger to activate the explainer to extract the important counters that best describe what happened. The information is made available by the YANG model output which were subscribed to when the service was enabled.

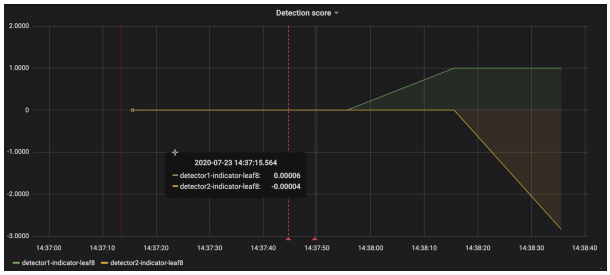

Fig. 3. The DESTIN values annotated as detector2 in this figure decrease when a routing loop is inserted. The vertical red dotted line marks the insertion of the event

IV. Conclusion

This demo paper showcases the potential of DESTIN in practice for detection of state transitions in network devices by showing an example detection of a routing loop event in a real test-bed. It is also shown that the methodology is lightweight enough to be run on the router thus eliminating the unnecessary data exports to external servers.

References

[1] P. Foroughi, W. Shao, F. Brockners, and J. l. Rougier, "Destin: Detecting state transitions in network elements," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM 2021)*. IEEE, 2021, pp. 1–10.

[2] G. Wang, L. Zhang, and W. Xu, "What can we learn from four years of data center hardware failures?" in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2017, pp. 25–36.

[3] T. Feltin, P. Foroughi, W. Shao, F. Brockners, and T. H. Clausen, "Semantic feature selection for network telemetry event description," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–6.

[4] E. M. Bjorklund, "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)," Internet Requests for Comments, RFC Editor, RFC 6020, October 2010. [Online]. Available: https://tools.ietf.org/html/rfc6020

[5] H. Song, F. Qin, P. Martinez-Julia, L. Ciavaglia, and A. Wang, "Network telemetry framework," Working Draft, IETF Secretariat, Internet-Draft draft-ietf-opsawg-ntf-05, October 2020.

[6] Cisco, "Cisco telemetry lab," 2020. [Online]. Available: https://github.com/cisco-ie/telemetry