White Paper

Communication Service Providers

NFV Troubleshooting for Communication Services Providers using HCL BluGenie Cognitive Analytics

HCL has developed the BluGenie Cognitive Analytics troubleshooting tool that employs algorithmic AI-driven approach to deciphering dynamic network behavior and for simplifying the troubleshooting of dynamic telco cloud environments.

BluGenie augments exploratory analysis to reduce the time needed to find service issues by automatically crunching multi-layer telemetry data from various network functions virtualization (NFV) layers through relevant machine learning/deep learning (ML/DL) algorithms.

This white paper provides a demonstration of how HCL BluGenie Cognitive Analytics derives insights in a cable network application using behavioral analysis of an Intel® vCMTS workload as target workload on top of a cloud native Kubernetes* infrastructure.

BluGenie delivers the following outcomes, capabilities and use cases:

For network function (NF) developers, this tool can help to realize higher performance by collecting less telemetry, which provides more value to CoSPs with very fast time to market (TTM) and meantime to repair (MTTR) on critical bugs in production.

Such assurance features will help network operators save money on CapEx and OpEx and increase customer satisfaction.

This white paper presents insights and correlations between key performance indicators (KPIs) and optimization parameters in workload and infrastructure layers that are identified through the triage and solving of complex performance problems quickly using BluGenie. BluGenie uses open industry standard interfaces to collect telemetry from the platform and leverages OPNFV/Anuket software with Kubernetes telemetry and CMTS application telemetry.

Table of Contents

1. MANO and BluGenie Cognitive Analytics

NFV management, orchestration and automation (MANO) systems pave the way for autonomous management of network functions deployed in the cloud. Fault vectors in the NFV stack are manifold because of multiple layers of telemetry data that needs to be investigated, including physical and virtual infrastructure layers, xNFs, workload characteristics and KPIs. Each layer comes with its own configuration options. With such large data, even subject matter expert (SME)-driven exploratory analysis / troubleshooting can be time consuming and is not scalable.

HCL BluGenie Cognitive Analytics troubleshooting employs an algorithmic AI-driven approach that simplifies troubleshooting of dynamic telco cloud environments. It augments exploratory analysis, reduces time taken to find service issues by automatically crunching multi-layer telemetry data through relevant ML/DL algorithms based on model arrived at.

BluGenie Cognitive Analytics is comprised of a data platform for handling streaming telemetry from various NFV layers and an AI algorithm (combination of ML/DL) layer to decipher dynamic network behavior – both design time and run time. Some of the functions include:

- Processing telemetry data for dimensionality reduction, finding anomalies, fault signatures of each layer being modeled / observed
- Compare and contrast NF behavior across different infrastructure environments / configurations / versions / types / vendor implementations
- Gain understanding of potential issues or possible improvements by understanding explicit relationships between two or more layers being modeled / observed, including identifying the bottleneck between infrastructure and service layers
- Capture experiential knowledge of workload characterization to identify the most probable system areas that provide opportunities for performance optimization
- Fault detection / isolation / root cause analysis; identify the attributes that impact service level agreements (SLA)
- Predict Service degradation and act on attributes identified by Root Cause Analysis via closed loop automation to prevent service level and overall system level degradation

Following are the algorithms that are being used in BluGenie Cognitive Analytics:

NAME	INPUT	STATISTICAL/ML TECHNIQUES	OUTPUT	ML TYPE
Telemetry Compaction	Original data from entity (tuple)	Feature engineering: redundancy analysis, zero variance filtration, data transformation	Compacted variables	U
Resource Sensitivity Analysis	Compacted data from entity	Data segregation and comparison	Image files with radar chart	U
Anomaly Detection	Compacted data from entity	DBSCAN clustering, active learning, random forest	Predicted data of anomaly/ normal	U,SS,S
Single-layer Fault Signature	Predicted anomaly data of entity	Percentage rate of change	Attributes showing fault signs	S
Cross-layer Fault Signature	Predicted anomaly data of all entities	Percentage rate of change	Attributes across entities showing fault signs	S
Cross Layer Correlation	Predicted Anomaly data of all entities	Data preprocessing, associate rule Mining, correlation analysis	Cross layer rules and root cause analysis	US, S
KPI Degradation	Predicted anomaly data of all entities	Cause and effect analysis, Granger Causality Wilcoxon test, correlation analysis	KPI degradation and attributes influencing it	S
Service degradation KPI Degradation	Fault Signature	Time series analysis, LSTM	Future pattern on fault parameters	U

VNF managers as well as NFV orchestrators can use the above AI ML techniques to detect anomalies in either NFs or the infrastructure and predict future behavior. This can help trigger suitable mitigation actions like life cycle management actions or fine-tuning configurations. AI ML models must be validated periodically to ensure they model the system accurately; otherwise replacing / retraining the model must be done. In summary, integration of AI ML intelligence is critical to enabling an adaptive system.

InfluxDB Relay, an open-source, scalable time-series database that stores metrics, events, and performance analytics is used for storing all the above metrics and analytics output, staging and presentation layer data.

Grafana is used to visualize time series data from InfluxDB and other analytics outcomes.

2. Intel® vCMTS Behavioral Analysis using HCL BluGenie

Intel has developed a media access control (MAC) data plane compliant with Data Over Cable Service Interface Specification (DOCSIS) 3.1 specifications and based on the data plane development kit (DPDK) packet-processing framework on industry standard high-volume servers. HCL BluGenie Cognitive Analytics undertakes behavioral analysis of vCMTS using a software traffic generator to identify optimization parameters and to understand leading indicators of service degradation using its AI ML/DL capabilities.

A steady shift is taking place toward virtual Converged Cable Access Platform (vCCAP) and Distributed Access Architecture (DAA) deployments by cable operators moving away from legacy central architecture. The combination of virtualized access and DAA gives operators the ability to more easily scale and also provides network design and service delivery flexibility

for the future of broadband meeting bandwidth and service demands. This necessitates a need for service assurance systems leveraging rich data and AI to simplify management and boost network performance while avoiding over / under utilization of resources through dynamic resource policies that meet service objectives. Machine learning-based model generation may improve reliability and efficiency of the overall system minimizing OpEx and improve service assurance.

HCL has conducted testing of Intel vCMTS using a DOCSIS traffic generator tool with telemetry support using BluGenie Cognitive Analytics.

In this whitepaper, we describe an approach to using HCL BluGenie Cognitive Analytics to provide insights by doing behavioral analysis of vCMTS and NFV infrastructure.

In design time, BluGenie helps to better dimension the overall system and identify bottleneck areas and the optimization parameters for scenarios which, in turn, can be tuned to deliver better system performance.

When such complex vCMTS systems are deployed in run time production environments, it is important to gather and analyze deep telemetry metrics to understand leading indicators of customer experience and act on them and make improvements proactively before customers experience service degradation.

BluGenie helps in both design and run time through cross layer correlation analysis between multiple layers of telemetry and arrives at fault signatures at each layer and then by identifying operational challenges in the field, finding the root cause attributes across layers that impact service KPIs and then developing solutions to mitigate the performance issues.

3. Applying BluGenie Cognitive Analytics on Intel vCMTS

BluGenie Cognitive Analytics insights can provide solutions to many activities like optimized initial placement, dynamic policy-based resource allocation and capacity planning. vCMTS with pinned resources deployment will be relatively static in terms of infrastructure requests. But differences in user activity, number of subscribers, channel configuration based on profile used, or encryption type can impact the required infrastructure resources. vCMTS instances may not be the only NFs running in the NFVI infrastructure. Other NFs may be sharing NFVI resources and such noisy neighbors make KPI optimization a necessity.

Our system under test (SUT) consists of three Intel workstations as described in the table below.

One server is used to deploy BluGenie Cognitive Troubleshooting platform. The Intel vCMTS data plane environment consists of a vCMTS data plane server and a traffic generation server node. Intel vCMTS nodes are connected back-to-back using four 25G Intel® Ethernet Network Adapter XXV710-DA2 dual-port network interface cards (NICs).

All three servers are also connected by a Gigabit switch. All tests in this paper are conducted on this testbed.

3.1 Server configuration

PHYSICAL	PURPOSE	CPU	CORES	RAM
Server	Kubernetes Cluster Master Node/Traffic Generator	Intel® Xeon® Gold 6139 CPU @ 2.30GHz	72	376 GB
Server	Kubernetes Cluster Worker Node/ vCMTS Data plane Server	Intel® Xeon® Gold 6230N CPU @ 2.30GHz	80	187GB
Server	HCL BluGenie Cognitive Analytics platform	Intel® Xeon® CPU E5-2699 v4 @ 2.20GHz	88	251GB

3.2 Software Configuration of vCMTS

NAME	DESCRIPTION
Host OS	Ubuntu 18.04 LTS
Kernel	4.15.0-106-generic
Ethernet Driver	version: 2.11.29
Firmware	7.30 0x80008387 1.2684.0
Kubernetes	Kubernetes v1.16.0
Docker	Docker version 19.03.11, build 42e35e61f3
Intel® vCMTS	v19-12-1

3.3 Testing Topology

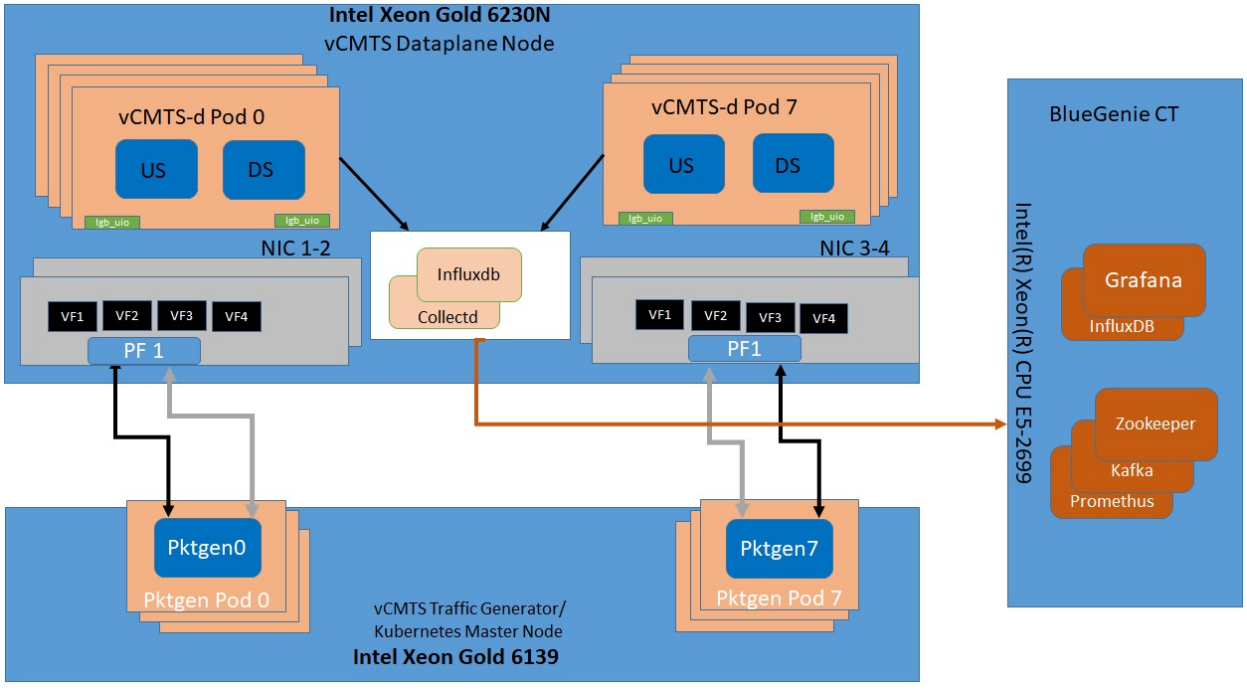


Figure 1. Blue Genie Analytic test bed configuration.

3.4 Testing Profiles

The following traffic profiles were developed to generate the following scenarios for HCL BluGenie CT Tool do the analytics using following commands (

Set upstream line rate: ***vcmts-pm traffic-rate -i 0-7 us -r <linerate>***
Set downstream line rate: v***cmts-pm traffic-rate -i 0-15 ds -r <linerate>***
Start the traffic on all pods: ***vcmts-pm traffic-start -i 0-7 usds***
)

TRAFFIC PROFILE	20-40% of line rate for all service groups **(normal behavior)**
	90% of line rate to disrupt system randomly between a certain time interval (configurable)
	For **testing** (predictions), an **RFC2544** script has been used

4. BluGenie Telemetry Data Platform for this SUT

The BluGenie Cognitive Analytics engine treats the following 3 as independent layers / tuples to be modeled / observed:

- CollectD attributes (infrastructure, middleware and VNF instances) – this includes Intel® Resource Director Technology (Intel® RDT) and Processor Counter Monitor (PCM) providing for detailed instrumentation and management of the platform
- Node exporter and pod metrics from Kubernetes
- Service groups specific KPIs

And all algorithms are run by correlating info from these three tuples. Each service group (SG)-related attributes are modeled independently.

4.1 Telemetry

Telemetry information serves as a vital resource for analyzing the platform's state and health, and for delivering on service assurance goals. Systems must be vigilantly monitored for utilization and malfunctions to mitigate disruptions, using both platform and container telemetry.

The following are the telemetry elements used to monitor infrastructure and containers; collection frequency is set to 10 seconds for all metrics listed below.

4.1.1 Infrastructure Metrics

- **OpNFV Barometer CollectD:** A statistics daemon that collects and reports on system and application performance metrics. Frequency for collecting CollectD metrics is 10 seconds.
  - Following CollectD plugins are enabled and are monitored:
    - CSV, CPU, cpufreq, ipmi, rdt, intel_pmu, unixsock, turbostat, network, python, write_http, disk, memory, numa, hugepages, RDT
- **Node Exporter:** The Kubernetes node has a finite CPU and memory capacity that can be leveraged by the running pods, so these two need to be monitored carefully. Other important metrics to monitor are disk-space usage and node-network traffic (receive and transmit). There are several node "conditions" defined that describe the status of the running nodes like Ready, MemoryPressure, DiskPressure, NetworkUnavailable, OutOfDisk.
  - Following Kubernetes node metrics are being collected and monitored:
    - node_load1, node_load5, node_load15, node_cpu_seconds_total, node_memory_MemAvailable_bytes, node_memory_MemTotal_bytes, node_memory_Buffers_bytes, node_memory_SwapCached_bytes, node_memory_Cached_bytes, node_memory_MemFree_bytes, node_memory_SwapFree_bytes, node_ipvs_incoming_bytes_total, node_ipvs_incoming_packets_total, node_ipvs_outgoing_bytes_total, node_ipvs_outgoing_packets_total, node_disk_reads_completed_total, node_disk_writes_completed_total, node_disk_read_bytes_total, node_disk_written_bytes_total, node_filesystem_avail_bytes, node_filesystem_free_bytes, node_filesystem_size_bytes
- **KPI Metrics**
  - vCMTS SG pod metrics are collected via an Intel-provided CollectD plugin. The following are the KPI-layer metrics being collected.
    - **downstream**
      - downstream_active_cms_per_second
      - downstream_average_frame_size_per_second_rx
      - downstream_average_frame_size_per_second_tx
      - downstream_average_frame_size_rx
      - downstream_average_frame_size_tx
      - downstream_bits_per_second_rx
      - downstream_bits_per_second_tx
      - downstream_dropped_per_second_rx
      - downstream_dropped_per_second_tx
      - downstream_ip_addr_per_cm
      - downstream_octets_per_second_rx
      - downstream_octets_per_second_tx
      - downstream_packet_latency_average
      - downstream_packet_latency_jitter
      - downstream_packet_latency_maximum
      - downstream_packet_latency_minimum
      - downstream_packets_per_second_rx
      - downstream_packets_per_second_tx
      - downstream_sched_dropped_per_second
      - downstream_sched_dropped
      - downstream_sched_red_dropped_per_second
      - ds_per_cm_ip_addr_0
    - **upstream**
      - upstream_active_cms_per_second
      - upstream_average_frame_size_per_second_rx
      - upstream_average_frame_size_per_second_tx
      - upstream_average_frame_size_rx
      - upstream_average_frame_size_tx
      - upstream_bits_per_second_rx
      - upstream_bits_per_second_tx
      - upstream_dropped_per_second_rx
      - upstream_dropped_per_second_tx
      - upstream_ip_addr_per_cm
      - upstream_octets_per_second_rx
      - upstream_octets_per_second_tx
      - upstream_packet_latency_average
      - upstream_packet_latency_jitter
      - upstream_packet_latency_maximum
      - upstream_packet_latency_minimum
      - upstream_packets_per_second_rx
      - upstream_packets_per_second_tx
    - **CAdvisor Metrics:**

      A daemon process running inside containers used to collect resource metrics.

      - container_cpu_usage_seconds_total
      - container_memory_cache_vcmts-pktgen-pod-0
      - container_memory_max_usage_bytes
      - container_memory_swap
      - container_memory_usage_bytes
      - container_memory_working_set_bytes
      - container_network_receive_bytes_total
      - container_network_transmit_bytes_total
      - container_network_transmit_packets_dropped_total
      - container_network_transmit_packets_total_
      - container_spec_cpu_shares
      - container_spec_memory_limit_bytes
      - container_spec_memory_reservation_limit_bytes

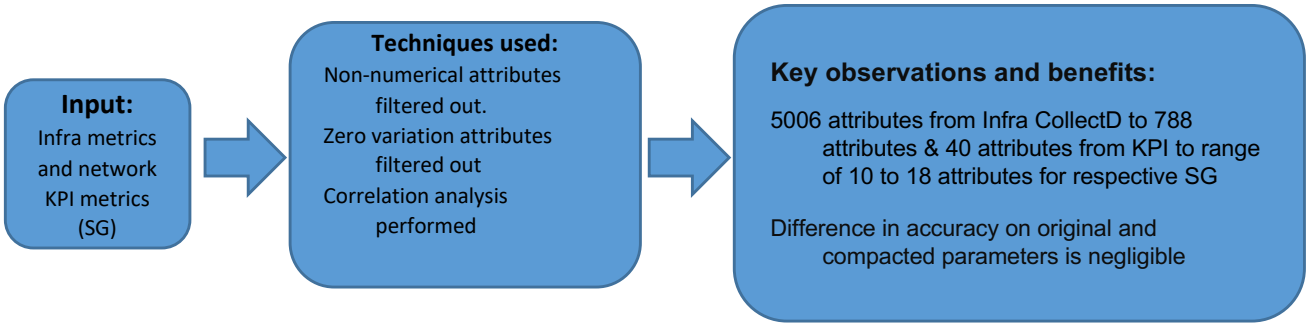
5. Analytics Engine and vCMTS modelling

5.1 Individual layer-based analytics outcomes:

In this section, we are going to focus on individual tuple / layer-based analytics outcomes namely, telemetry compaction, Resource intensity analysis, anomaly detection, and fault signature identification.

5.1.1 Telemetry Compaction:

Objective: Optimize time and resources by reducing the number of parameters while maintaining accuracy above the predetermined threshold.



We wanted to check how the behavior of the vCMTS system changes for various layers mentioned above by undertaking feature engineering and dimensionality reduction.

We have found that, we are getting different outcomes for compacted list of attributes as shown in Figure 3.

Intel vCMTS:

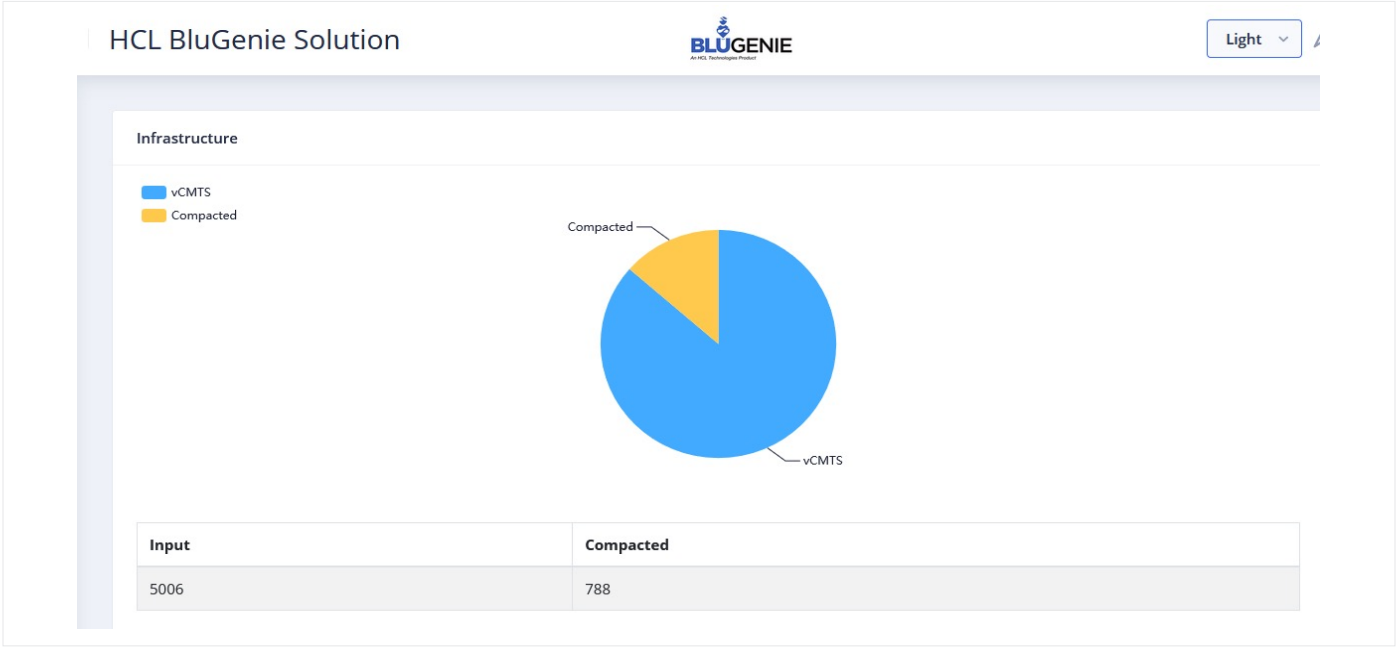


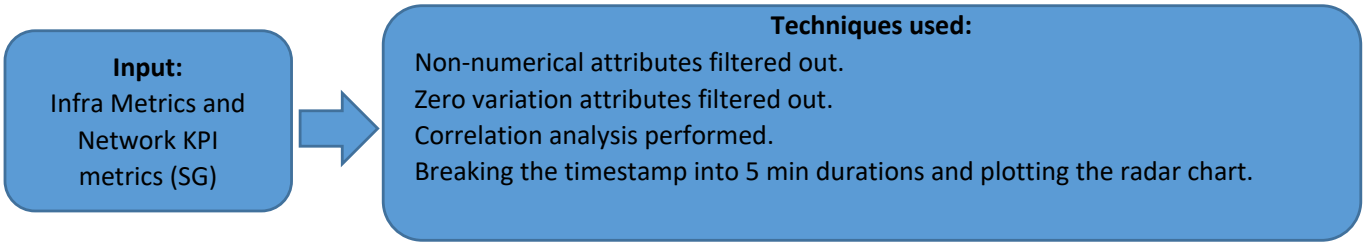
Figure 2. Original vs. Compacted Attributes

Service Groups: (KPI's)

Figure 3. KPI Parameters impacting infrastructure layer

5.1.2 Resource Analysis:

Objective: Radar Chart shows data comparison (Min./Avg./Max.) across key attributes



For a given network and for each service group:

- Plot multi series Radar chart for a given service group/ infra attributes that shows intensive changes occurred on average in the last 5 minutes (or configurable interval) and compare with minimum and maximum value of respective attribute.

5.1.3 Anomaly Detection:

Objective: To detect anomalies in respective tuple (Intel vCMTS / SG's)

This is done by detecting abnormal network behavior along with the timestamp using clustering technique DBSCAN, which is a density-based clustering technique that divides the data into multiple homogeneous clusters. DBSCAN is very useful in detecting outliers or noise. We also used deep learning algorithm AutoEncoder for detecting outliers / noise in data for comparison purpose.

Input: Compacted infra/network KPI metrics (per SG)

Algorithms used: DBSCAN and AutoEncoder algorithm used to find noise.

Key observations and benefits:

- The AutoEncoder model performs better on large datasets and DBScan shows good results on small datasets.
- If the system faces any degradation, anomaly detection is an important step to find out the reason behind it. Anomalies across all tuples have been identified to find root cause for system failure.

Figure 4 shows dashboard visualizations used to monitor attributes showing anomalous (red) / normal (green) behavior for KPIs, vCMTS:

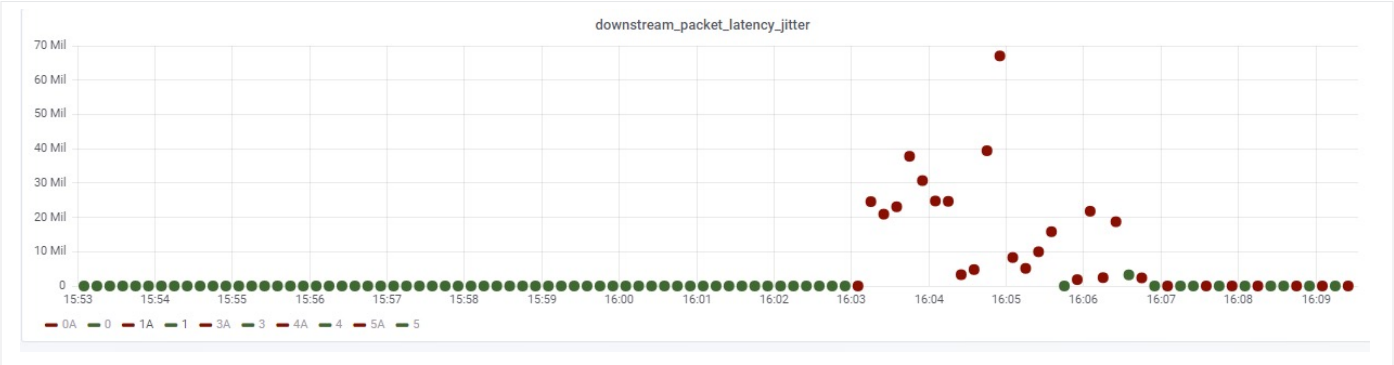


Figure 4. Anomaly Detection Illustration

5.1.4 Fault Signature Identification

A fault signature is defined as a combination of attributes in each layer that cause anomalous behavior in that layer.

Objective: Identify the attributes that are contributing to anomalous behavior in a system.

Input: Anomaly detection output

Technique / approach: Once the anomaly is predicted, we consider the points where there are consecutive number of anomalous records occur (this is configurable; it is currently set to 3 consecutive). Next, we find out the rate of change of all the attribute values with their corresponding previous values and observe the significantly changed attributes (based on statistical techniques). These often-occurring combinations of attributes evolve into a fault signature.

Following are the attributes that show significant changes at anomalous events in their respective tuple:

Intel vCMTS:

rdt_bytes_0_llc	rdt_memory_bandwidth_total_2_local
rdt_bytes_1_llc	rdt_memory_bandwidth_total_2_remote
rdt_bytes_2_llc	rdt_memory_bandwidth_total_3_local
rdt_bytes_3_llc	rdt_memory_bandwidth_total_3_remote
rdt_bytes_4_llc	rdt_memory_bandwidth_total_4_remote
rdt_bytes_5_llc	rdt_memory_bandwidth_total_5_remote
rdt_bytes_6_llc	rdt_memory_bandwidth_total_6_remote
...	...
rdt_bytes_76_llc	rdt_memory_bandwidth_total_69_local
rdt_bytes_77_llc	rdt_memory_bandwidth_total_69_remote
rdt_bytes_78_llc	rdt_memory_bandwidth_total_70_local
	rdt_memory_bandwidth_total_70_remote
	rdt_memory_bandwidth_total_71_remote
	rdt_memory_bandwidth_total_73_local
	rdt_memory_bandwidth_total_73_remote
	rdt_memory_bandwidth_total_74_remote
	rdt_memory_bandwidth_total_76_local
	rdt_memory_bandwidth_total_76_remote
	rdt_memory_bandwidth_total_77_local
	rdt_memory_bandwidth_total_77_remote
	rdt_memory_bandwidth_total_78_local
	rdt_memory_bandwidth_total_78_remote
cpufreq_20	pmu_gauge_23_LLC-load-misses-rate
cpufreq_23	pmu_gauge_24_LLC-loads-rate
cpufreq_24	pmu_gauge_26_L1-dcache-load-misses-rate
cpufreq_25	pmu_gauge_26_LLC-load-misses-rate
cpufreq_26	pmu_gauge_27_LLC-load-misses-rate
cpufreq_27	pmu_gauge_28_L1-dcache-load-misses-rate
cpufreq_35	pmu_gauge_28_LLC-load-misses-rate
cpufreq_62	pmu_gauge_39_instructions-rate
cpufreq_63	
cpufreq_65	
cpufreq_74	

KPIs (SGs):

downstream_active_cms_per_second_total	upstream_active_cms_per_second_total
downstream_average_frame_size_per_second_rx	upstream_average_frame_size_per_second_rx
downstream_bits_per_second_rx	upstream_bits_per_second_rx
downstream_dropped_per_second_rx	upstream_dropped_per_second_tx
downstream_octets_per_second_tx_total	upstream_octets_per_second_rx_total
downstream_packet_latency_average	upstream_packet_latency_average
downstream_packet_latency_jitter	upstream_packet_latency_jitter
downstream_packet_latency_maximum	upstream_packet_latency_maximum
downstream_packet_latency_minimum	upstream_packet_latency_minimum
	upstream_packets_per_cm_lookup
	upstream_packets_per_second_rx

Key observations and benefits:

- Fault signatures can be monitored to increase SME efficiency.
- When fault signatures of each layer are observed, they can be given additional focus to avoid future anomalies. Also, SMEs can decipher bottleneck patterns between layers.

5.2 System based analytics outcomes:

This section is focused on system level analytics outcomes as follows: Cross layer correlation, root cause analysis, and service degradation prediction.

5.2.1 Cognitive Behavioral Analysis:

1. **Objective: Correlation Analysis leading to identification of vCMTS pod metrics that mimic infra metrics.**

Technique/Approach: We have identified infra-attributes that are highly correlated (>95%) to pods metrics in a respective SG.

5.2.2 Root cause analysis (RCA)

Objective: Find root cause if system shows degradation

When a KPI is impacting patterns to other layers, it helps to use RCA, which is used for modeling system-level behavior to predict service degradation.

Input: Data with anomalous events only.

Technique/Approach:

1. **Wilcoxon test**

   To compare two anomalous events to determine if they are similar and causing degradation. Filter all such events for further processes.

2. **Finding Rules- Associate Rule Mining:**

   To find co-occurring events requires the unsupervised machine learning technique called Associate Rule Mining (ARM). ARM requires that input data to the model is transactional data. Each row of that input data must correspond to the list of variables that have been changed significantly in the tuple during the anomaly. ARM is a technique used to find the associations between variables. ARM is basically used for "if-then" analysis, that is if this is happening then it will also happen. There are three ways to measure the association: support, confidence, and lift.

3. **System level degrading attributes:**

   This step involves selecting the attributes / VNF that are strongly impacting KPIs. From the rules are selected the unique attributes involved in the rules set having support greater than 0.5. These are used to determine the attribute's values so as to find the top 5 attributes/ VNF that are strongly correlated with KPI parameters.

   Key observations and benefits: Attributes that are root cause for degradation in system and corresponding tuple / pods are identified so corrective action can be taken.

RootCause

Tuple	Variable
vCMTS	vCMTS_intel_rdt_bytes_21_llc
vCMTS	vCMTS_intel_pmu_gauge_23_LLC-load-misses-rate
vCMTS	vCMTS_intel_pmu_gauge_39_instructions-rate
vCMTS	vCMTS_intel_rdt_memory_bandwidth_total_27_local
vCMTS	vCMTS_intel_rdt_memory_bandwidth_total_15_remote

Figure 5. Root Cause Analysis for Service Degradation

This finding highlights the optimization possibilities that exist with last level cache, and RDT memory bandwidth-related configuration and fine tuning.

5.2.3 Service Degradation Prediction

Objective: Forecast future patterns on current fault signatures so as to take proactive measures and prevent any degradation from happening.

Input: Fault Attributes across system

Technique/Approach:

Long short-term memory (LSTM) algorithm is used for predicting patterns on fault signatures that is variables that are causing anomaly at system level. This algorithm is also used for taking proactive measures on upcoming anomalous events that may cause alarming situation in the system.

Input for this step are the fault signature parameters, which are first transformed to a dataset that takes 180 samples as predictor and next sample as target. Training an LSTM network is done using one-week of data and patterns are predicted for next one hour (configurable).

Infra fault attributes forecast and anomaly detection:

Figure 6. Infra Layer Attributes - Anomaly Detection

KPI fault attributes forecast and anomaly detection:

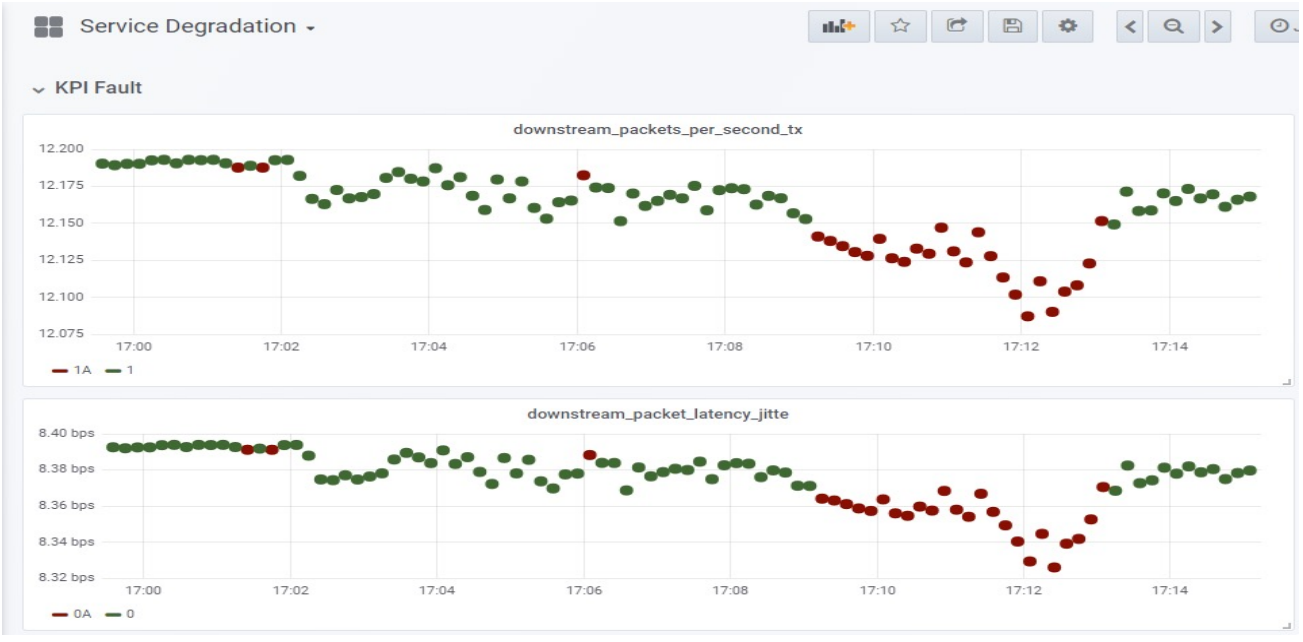


Figure 7. KPI Layer Attributes - Anomaly Detection

Key observations and benefits: Once the future data for fault signatures across tuples is available, all steps explained above (anomaly detection, root cause analysis, etc.) can be applied to take proactive measures to prevent any system failure event from happening.

6. Conclusion: BluGenie Cognitive Analytics on Intel vCMTS Summary

Configuring NF instances on a shared NFVI is a must to improve the use of available system capacity. It is important to understand the behavior of NF instances so their SLAs are not impacted. Resource analysis of individual pods and comparative analysis of different configurations (pinning of 2 separate cores versus 2 instances sharing a single core) – allows for performance behavior analysis of throughput and to check if the system is under or over provisioned. Such placement decisions will help utilize a minimal number of cores and machines required by cable service providers to help deliver service improvements to customers and reduce TCO.

It is also possible to do root cause analysis for service degradation prediction through cross layer correlation that may help identify the fault signature parameters giving KPI-impacting parameters across infrastructure and NF attributes. This data can be acted upon in a proactive fashion to help avoid service degradation.

These analytics may, save cable operators from downtime and operational expenditure to troubleshoot and fix critical network performance, service degradation, customer experience issues.

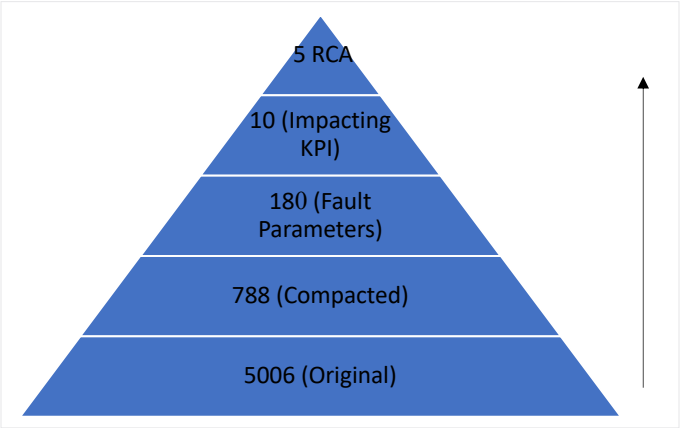


Figure 8. Summary of parameters

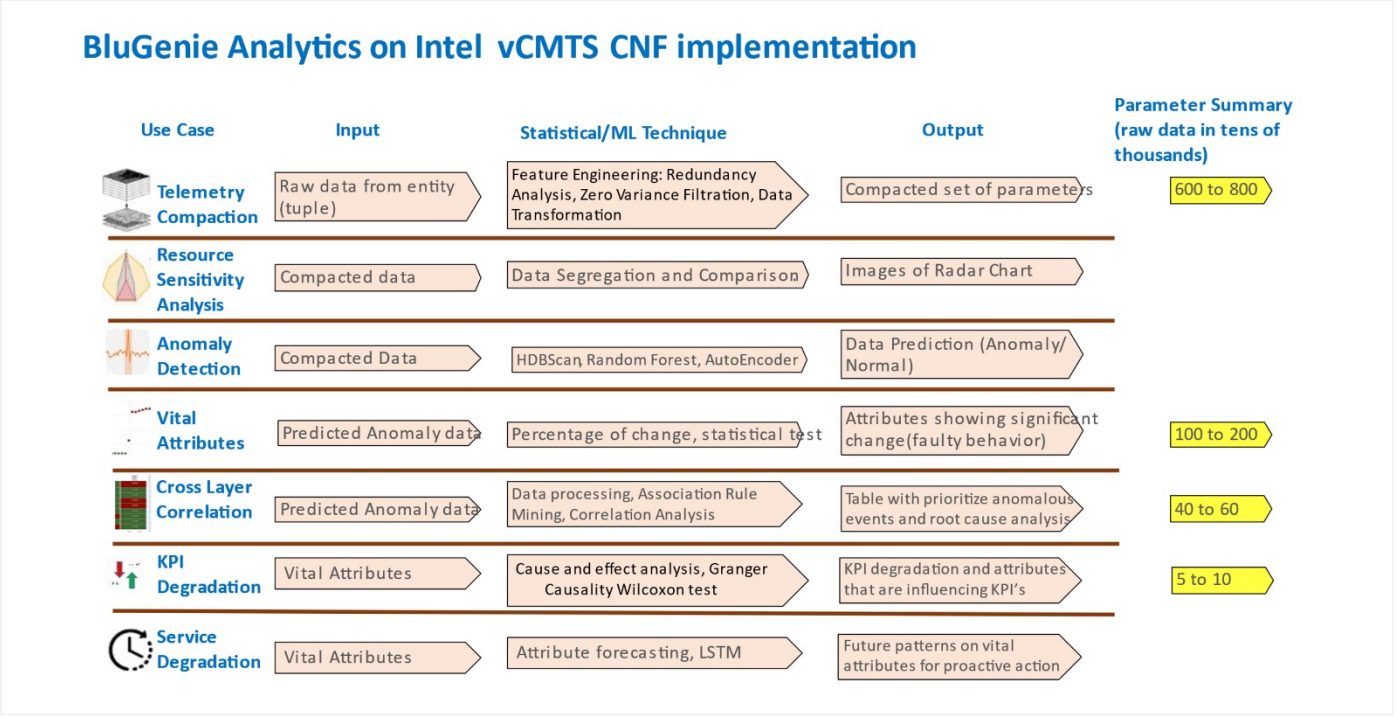


Figure 9. BluGenie Analysis Summary on Intel vCMTS CNF

As illustrated MANO solution using BluGenie capabilities can help with:

- Implementing an adaptive system that can automatically do resource allocation-based placement policy for addition of new NF types or user behaviors.
- Improving quality of service with multiple layers of anomaly detection and fault correlation to the service KPIs.
- Save OpEx by narrowing down the root cause and identifying suitable mitigation option(s) to avoid service degradation.

To sum it all up, what is measured gets managed, what is analyzed gets improved; the analytics available from BluGenie help transform systems to be more efficient.

Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

Intel technologies may require enabled hardware, software or service activation.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

0522/TM/HO9/PDF ♻Please Recycle 351010-001US