

# Deep Observability with the Pensando Distributed Services Platform

SOFTWARE-DEFINED | EDGE-ACCELERATED | ALWAYS-SECURE | ANY-ENVIRONMENT

## Observability Challenges for Customers at the Edge

Gaining visibility into complex applications and systems has become one of the most pressing and complex problems for CIO's to solve in modern data centers.

Overall, customer challenges can be summarized in two main focus areas.

**Operational:** Gaining actionable insights from your data requires more than collecting and analyzing flows, metrics and logs. With the acceleration of customer and business demands, site reliability engineers and IT Ops analysts require operational visibility into their infrastructure and applications – something that traditional tools aren't fully equipped to provide. Current host-based tools don't scale, overconsume host CPU cycles, have visibility blind spots, and lack an understanding of network communication behavior that make troubleshooting difficult problems more complex. This increases mean time to resolution, which overall impacts operational cost and lengthens MTTR when problems occur.

- Lack of proactive health monitoring is resulting in outages that could have been easily prevented
- Problem isolation takes 10's of minutes to sometimes hours due to lack of end-to-end troubleshooting tools
- Resources are over-provisioned or under-provisioned, due to unavailability of accurate capacity and utilization information for the infrastructure
- Lack of integration with other ecosystem tooling, is resulting in challenges in correlating observed data with something that is more useful like the application (virtualized, bare-metal, container) information
- Most of the tools in use today are either not pervasive or uniform (on all workloads and always-on). Utilizing a comprehensive set of tools that cover all needs results in prohibitive cost

### Goals of This Paper

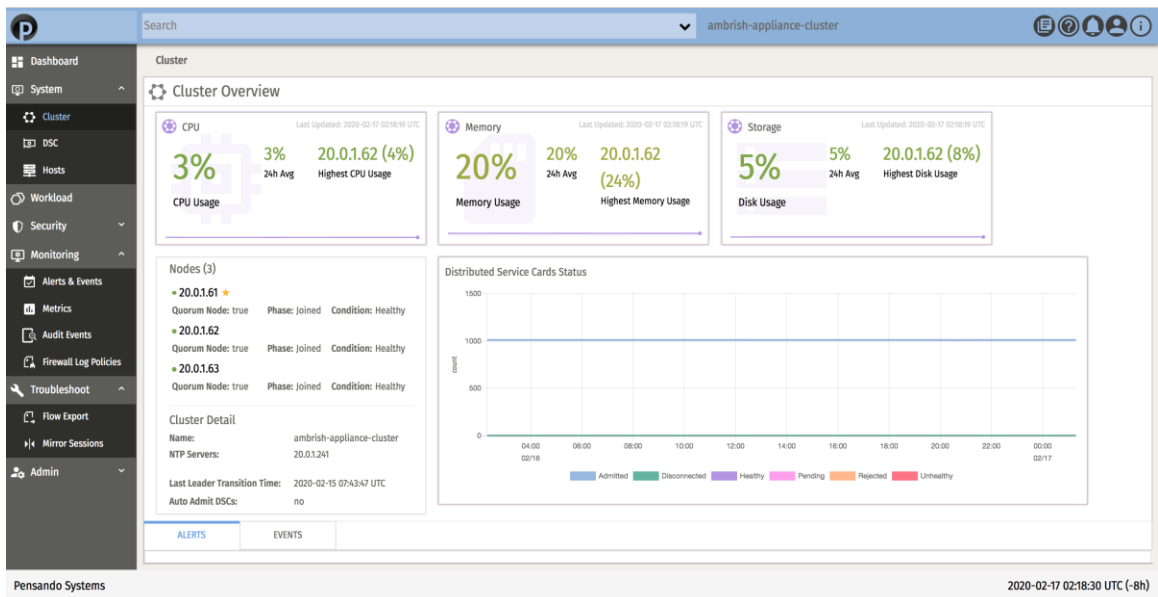
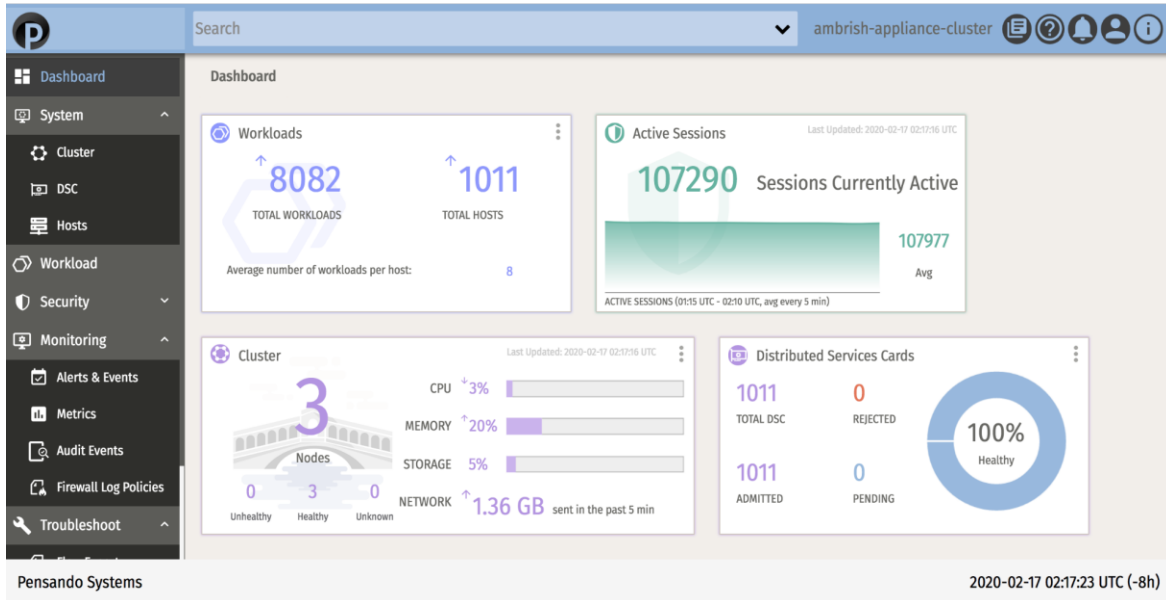
- Provide the reader with an overview of the operational visibility and security challenges facing IT organizations
- Describe the unique analytics foundation of the Pensando *Distributed Services Platform*
- Illustrate how Pensando's flexible, scalable, intelligent platform can accelerate mean time to resolution, enable a secure datacenter foundation and reduce operational costs

**Security:** Security architectures are changing dramatically in the datacenter based on the realization that perimeter-based security with appliances at the edge is not adequate in the modern datacenter. The old adage that only 'good guys' are within the perimeter is no longer valid.

Many enterprises have re-purposed edge appliances for use within east-west traffic flows and to protect against compromised workload lateral movement within the data center. Lateral traffic flows have become the dominant data center traffic flows and with this evolution comes the additional complexity of having poor ability to see these application traffic flows. Traffic within the datacenter must be segmented, and move towards "zero-trust" security, but to deliver an adequate segmentation policy requires a fundamentally deep understanding of application and user behavior, as **you cannot segment or secure what you cannot see.**

Deriving that segmentation policy can be challenging using solely software-based approaches as these suffer from performance challenges and in addition they put excessive burden on host CPUs.

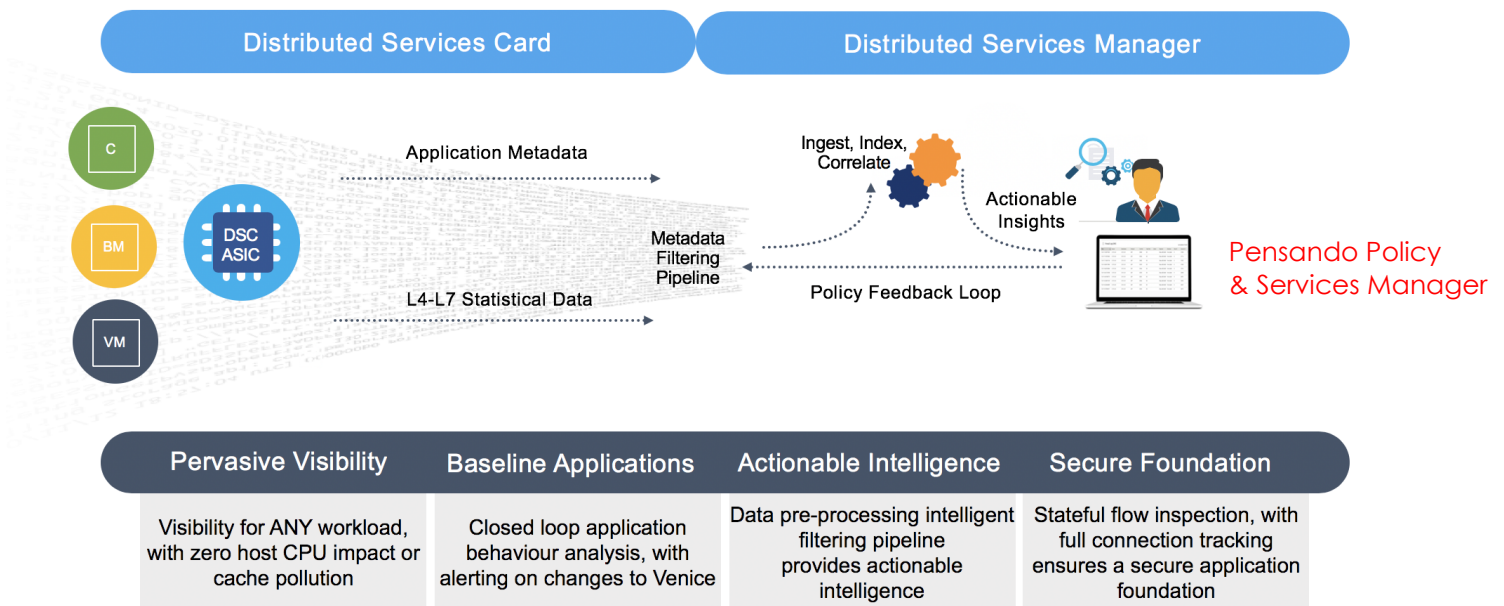
- Network based attacks within the perimeter are going undetected due to lack of visibility and improper infrastructure to detect anomalies
- In heterogeneous environments, it is becoming increasingly difficult to detect applications and determine their inter-dependencies. Consequently, implementing micro-segmentation is becoming increasingly challenging



## PENSANDO'S DISTRIBUTED OBSERVABILITY ARCHITECTURE

As part of Pensando's goal to offer customers the most comprehensive, pervasive visibility for their application workloads and address CIO/CSO's top concerns, we have built a system for inline data collection with intelligent processing right at the source, without performance/latency loss within the Pensando Distributed Services Card (DSC). As part of the solution, Pensando has built a set of scale-out software services running within the Policy and Services Manager (PSM) to correlate, visualize and utilize the information.

# Pensando's Architectural Approach To Visibility



The collection of data happens naturally as application I/O is performed, not requiring the installation of any agents, root/kernel privileged software on the hypervisor or host operating system. It is also important to note that the DSC possesses enough capacity to collect every single packet/message flowing through the system with a guarantee of not losing any relevant information. Multiple Gigabytes of onboard memory on the card can perform significant stateful operations at line rate on the DSC.

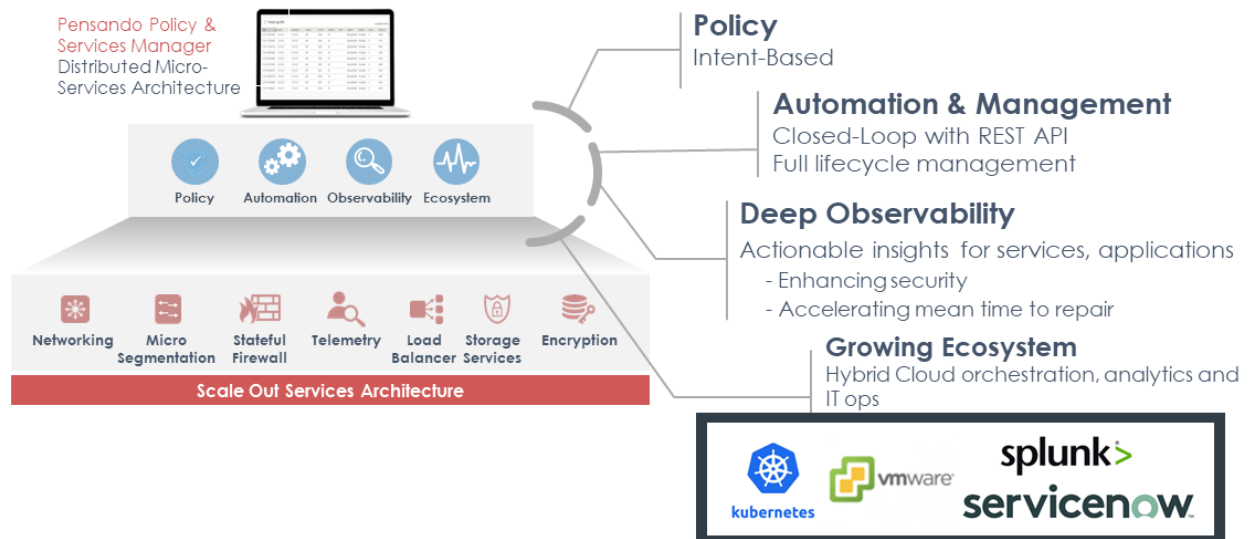
This architecture handles scale naturally because the Pensando DSC nodes perform up-front processing (per the policies from the central Policy and Services Manager (PSM) to avoid flooding the network with duplicate data as well as running into the risk of the traffic getting dropped. In order to satisfy requirements for packet capture – similar to network tap aggregation use cases – it is possible to utilize bi-directional dynamic flow mirroring at the source of every packet that is flowing through the system. This allows the reconstruction of the application communication flow offline on packet capture appliance infrastructure for network troubleshooting and security compliance.

As the collector of this observability information from the DSC, the Policy and Services Manager organizes and provides queries into the data via APIs and enables visualization of the information across the cluster. The PSM offers all the features of a modern database, including data trend analysis, data aggregation based on attributes, while also providing data high availability at scale.

## Pensando Policy and Services Manager (PSM)

The PSM is a key component of the platform, providing Distributed Services Card (DSC) lifecycle management (discovery, admission, decommission, secure upgrade, device health), and distribution of policy for networking, security, and storage services. Additionally, a core role is to provide a detailed, coherent, unified view of the network and services health, utilizing network and application flow statistics and state, which enables operations teams from a centralized place to have a single source of truth, to quickly identify and troubleshoot problems and to rapidly remediate issues.

## Software Services, Centrally Managed



The distributed services manager also enables ecosystem integrations across compute orchestration, IT operations, and analytics applications used in the datacenter, with all state data query-able through a REST API interface.

## P4 Processing Pipeline

Core to Pensando's Distributed Services Card (DSC) is the implementation of an industry-leading multi-stage programmable P4 pipeline for filtering and processing application metadata at its source. Filtering at the source radically reduces data processing impact of the data collection process, freeing up compute cycles in the PSM for business applications, and irrelevant data can be automatically filtered out, reducing alerting noise while accelerating MTTR.

## Pensando Observability Functional Use-Cases

### Dynamic Flow Mirroring

Dynamic Flow Mirroring provides an ability to inspect packet content at line-rate, isolating and extracting application-specific traffic for delivery to appropriate tools for further processing. It uses traffic replication and spanning to a packet broker network that delivers packets to an analytics engine performing content inspection and correlation, for compliance or security. The Distributed Services Card (DSC) supports line-rate bidirectional (Tx/Rx) Dynamic Flow Mirroring to give a complete picture from the compute node's perspective when it comes to packet capture: ingress/egress from the tenant workload to the DSC, and ingress/egress from the DSC to the switch it is connected to, with all packets granularly timestamped.

Monitoring > Mirror Sessions

ACTIVE 3 EXPIRED 123

SPAN Name	Source	Destination	Ports/Apps	Time	Status	Packets	Actions
Completed ER Span	10.216.100.210	10.216.100.211	TCP/9898	04/15/2018 08:40 PST - 04/15/2018 08:45 PST	Completed	123,456	[Icons]
Canceled ER Span	12.316.104.211	12.316.104.215	TCP/8080	04/15/2018 08:30 PST - 04/15/2018 08:35 PST	Canceled	-	[Icons]
Canceled ER Span	12.316.104.211	12.316.104.215	TCP/8080	04/15/2018 08:20 PST - 04/15/2018 08:25 PST	Canceled	-	[Icons]
Completed ER Span	10.10.10.20, 10.10.20.30, 10...	10.10.10.20, 10.10.20.30, 10...	TCP/9898	04/15/2018 07:40 PST - 04/15/2018 07:45 PST	Completed	123,456	[Icons]
Canceled ER Span	10.10.10.20, 10.10.20.30, 10...	10.10.10.20, 10.10.20.30, 10...	TCP/8080, TCP/8081, TCP/808...	04/15/2018 08:30 PST - 04/15/2018 08:35 PST	Canceled	-	[Icons]
Canceled ER Span	12.316.104.211	12.316.104.215	TCP/8080	04/15/2018 08:20 PST - 04/15/2018 08:25 PST	Canceled	-	[Icons]
Completed ER Span	10.216.100.210	10.216.100.211	TCP/9898	04/14/2018 07:40 PST - 04/14/2018 07:45 PST	Completed	33,005	[Icons]

Monitoring > Troubleshooting

Source IP and Port: 10.10.10.20 TCP/6000 Destination IP & Port: 10.10.10.10 TCP/8080 Time Selector: Now Tools: SPAN

SPAN Filter: frame.number >= 100 and frame.number < 500

Source				Destination			
Time	Protocol/Port	Length	Info	Time	Protocol/Port	Length	Info
44.774186	UDP/1020	210	47043 + palace-4(9995) [BAD UDP LENGTH 16]	44.774186	UDP/1020	210	47043 + palace-4(9995) [BAD UDP LENGTH 16]
44.774196	TCP/8080	98	47043 + palace-4(9995) [BAD UDP LENGTH 16]	44.774196	TCP/8080	98	47043 + palace-4(9995) [BAD UDP LENGTH 16]
44.774275	TCP/8080	88	46531 + palace-4(9995) [BAD UDP LENGTH 31]	44.774275	TCP/8080	88	46531 + palace-4(9995) [BAD UDP LENGTH 31]
44.774186	TCP/8080	150	47043 + palace-4(9995) [BAD UDP LENGTH 16]	44.774186	TCP/8080	150	47043 + palace-4(9995) [BAD UDP LENGTH 16]
45.774196	TCP/8080	234	47043 + palace-4(9995) [BAD UDP LENGTH 16]	45.774196	TCP/8080	234	47043 + palace-4(9995) [BAD UDP LENGTH 16]
45.774275	UDP/1020	210	46531 + palace-4(9995) [BAD UDP LENGTH 31]	45.774275	UDP/1020	210	46531 + palace-4(9995) [BAD UDP LENGTH 31]
45.774186	TCP/8080	98	47043 + palace-4(9995) [BAD UDP LENGTH 16]	45.774186	TCP/8080	98	47043 + palace-4(9995) [BAD UDP LENGTH 16]
45.774196	TCP/8080	78	47043 + palace-4(9995) [BAD UDP LENGTH 16]	45.774196	TCP/8080	78	47043 + palace-4(9995) [BAD UDP LENGTH 16]
45.774275	TCP/8080	150	46531 + palace-4(9995) [BAD UDP LENGTH 31]	45.774275	TCP/8080	150	46531 + palace-4(9995) [BAD UDP LENGTH 31]
46.774186	TCP/8080	234	47043 + palace-4(9995) [BAD UDP LENGTH 16]	46.774186	TCP/8080	234	47043 + palace-4(9995) [BAD UDP LENGTH 16]
46.774196	UDP/1020	184	47043 + palace-4(9995) [BAD UDP LENGTH 16]	46.774196	UDP/1020	184	47043 + palace-4(9995) [BAD UDP LENGTH 16]
46.774275	TCP/8080	98	46531 + palace-4(9995) [BAD UDP LENGTH 31]	46.774275	TCP/8080	98	46531 + palace-4(9995) [BAD UDP LENGTH 31]
46.774186	TCP/8080	78	47043 + palace-4(9995) [BAD UDP LENGTH 16]	46.774186	TCP/8080	78	47043 + palace-4(9995) [BAD UDP LENGTH 16]
47.774196	TCP/8080	150	47043 + palace-4(9995) [BAD UDP LENGTH 16]	47.774196	TCP/8080	150	47043 + palace-4(9995) [BAD UDP LENGTH 16]
47.774275	TCP/8080	234	46531 + palace-4(9995) [BAD UDP LENGTH 31]	47.774275	TCP/8080	234	46531 + palace-4(9995) [BAD UDP LENGTH 31]
47.774186	UDP/1020	184	47043 + palace-4(9995) [BAD UDP LENGTH 16]	47.774186	UDP/1020	184	47043 + palace-4(9995) [BAD UDP LENGTH 16]
47.774196	TCP/8080	88	47043 + palace-4(9995) [BAD UDP LENGTH 16]	47.774196	TCP/8080	88	47043 + palace-4(9995) [BAD UDP LENGTH 16]
47.774275	TCP/8080	173	46531 + palace-4(9995) [BAD UDP LENGTH 31]	47.774275	TCP/8080	173	46531 + palace-4(9995) [BAD UDP LENGTH 31]

## NetFlow Export

NetFlow is a standard network protocol for monitoring and collecting network traffic. A picture of network traffic flows can be built by analyzing NetFlow data exported by a device. Flow exporters on the Pensando DSC can be enabled to export flow information and statistics to flow monitors on remote NetFlow collectors in IPFIX (Netflow.v10) format. Centralized configuration allows tuning of export timeouts, export intervals, export format as well as specifying output to more than one collector. In addition to exposing the standard set of NetFlow fields, the Distributed Services Card (DSC) also exposes flow start and last seen time, maximum segment size, and state offering a wealth of information for troubleshooting and security use-cases.



## Flow Logging & Tracking

The Pensando DSC supports full TCP connection state tracking, providing deep information inspection and telemetry with an application viewpoint. This function is normally offered only by high-end stateful firewalls and provides a way to evaluate whether the packets within a given flow are legitimate and are going to be actually received by an application by inspecting and evaluating them against the current state of their TCP connection. Connection tracking at wire-rate is performed in the data plane, and if the distributed firewall is enabled, flows are marked as "allow" or "deny", with actions performed accordingly in hardware. Connection tracking also helps in cleaning up state information related to closed sessions instead of waiting for a full inactivity/idle time period.

Once connection tracking is enabled, the flow/session entries for the following protocols are validated in the pipeline:

- **TCP state and connection tracking**
  - Perform TCP SYN validation, evaluating security policy prior to pipeline flow programming.
  - Validate TCP sequence and ack numbers are within the expected TCP window, for all packets.
  - Perform session closing state tracking, FIN/RST, and adjusting TCP state to closing.
- **ICMP request and response tracking**
  - Using ICMP ID and sequence numbers, invalid ICMP responses can be filtered, and requests and responses can be correlated.
  - ICMP sessions can be aggressively aged out instead of waiting for inactivity period to expire.

The Policy & Services Manager collects flow logs that indicate source and destination IP's, ports, action, rule id, direction, making it easy for users to correlate and search.

Firewall Logs (2,103) Past one day Last Updated: 2019-10-14 21:34:58 UTC  

Time ↓	Source	Destination	Protocol	Src Port	Dest Port	Action	Reporter	Direction	Session ID	Session Action	Policy Name
2019-10-14 20:46:310.833387234 L	10.0.0.50	10.0.0.10	TCP	40624	22	Allow	ae-s7.pensando.	From Host	158	flow_delete	Test1
2019-10-14 20:46:310.752208637 L	10.0.0.50	10.0.0.10	TCP	40624	22	Allow	ae-s8.pensando.	From Uplink	239	flow_delete	Test1
2019-10-14 20:44:030.611049558 L	10.0.0.50	10.0.0.10	TCP	57078	1020	Implicit-deny	ae-s7.pensando.	From Host	1153	flow_delete	
2019-10-14 20:44:030.610389643 L	10.0.0.50	10.0.0.10	TCP	58612	1023	Implicit-deny	ae-s7.pensando.	From Host	1162	flow_delete	
2019-10-14 20:44:030.609716289 L	10.0.0.50	10.0.0.10	TCP	36246	1001	Implicit-deny	ae-s7.pensando.	From Host	1156	flow_delete	
2019-10-14 20:44:030.609073865 L	10.0.0.50	10.0.0.10	TCP	50948	1017	Implicit-deny	ae-s7.pensando.	From Host	1154	flow_delete	
2019-10-14 20:44:030.608432215 L	10.0.0.50	10.0.0.10	TCP	39910	1019	Implicit-deny	ae-s7.pensando.	From Host	1150	flow_delete	
2019-10-14 20:44:030.607787881 L	10.0.0.50	10.0.0.10	TCP	42178	1003	Implicit-deny	ae-s7.pensando.	From Host	1159	flow_delete	
2019-10-14 20:44:030.607131096 L	10.0.0.50	10.0.0.10	TCP	59566	1002	Implicit-deny	ae-s7.pensando.	From Host	1157	flow_delete	

## Application Connectivity Graphing

To truly understand and baseline application behavior, requires building an architecture that enables stateful analysis of every application messages (packets, transactions and flows), fully tracking connection state and application messages. Application connectivity information, along with bandwidth usage creates a foundation for determining application posture and inter-dependencies, and the data analysis engine built on this information in the PSM provides alerting (deviations/anomalies), charting, and enables base-lining the security posture of the system.

The Pensando Distributed Services Platform enables customers to:

- Fully decode protocols, including stateful ones (such as TCP and SSL/TLS), from network packets;
- Extract application data from the packets using application inspection, and enriching the decoded data with contextual information specific to the application – well known applications can be decoded and tagged in an application dependency map;
- Identify error conditions based on observed protocol information within the application header (HTTPv1/HTTPv2/SQL, etc.) and alert on events of interest for the application, along with providing statistical data per application;
- While in many situations analyzing layer 4 flow information is deemed sufficient, it may mislead the observer about the nature of data a particular flow is exchanging within the flow. Providing this visibility not only helps with determining security aberrations, but also helps with identifying and organize applications into appropriate buckets for micro-segmentation.

## Security Policy Generation & Simulation

The pervasive application visibility enabled on the Distributed Services Card, allows dynamic always-on learning of application behavior, enabling customers to move gradually towards a distributed policy-based security model. This way security policies can be generated accurately based on learnt application behavior from the application node, enabling a segmentation policy that can restrict access to critical systems to only authorized entities, for bare metal, virtualized or containerized applications, while avoiding unintended isolation. The Pensando Distributed Services Platform has the capability of automatically generating rules that implement such security policies. All of this results in a platform for policy based secure segmentation, that is proactive, automated and closed loop.

## Application Performance Monitoring

Root causing sources of application related latency, are one of the most difficult problems for network operators to solve. Accelerating “mean time to innocence” is a key concern of stakeholders when problems are identified by application owners where the network is referenced as the source of the issue. Having the necessary intelligence and observability at the edge is critical to troubleshooting such problems.

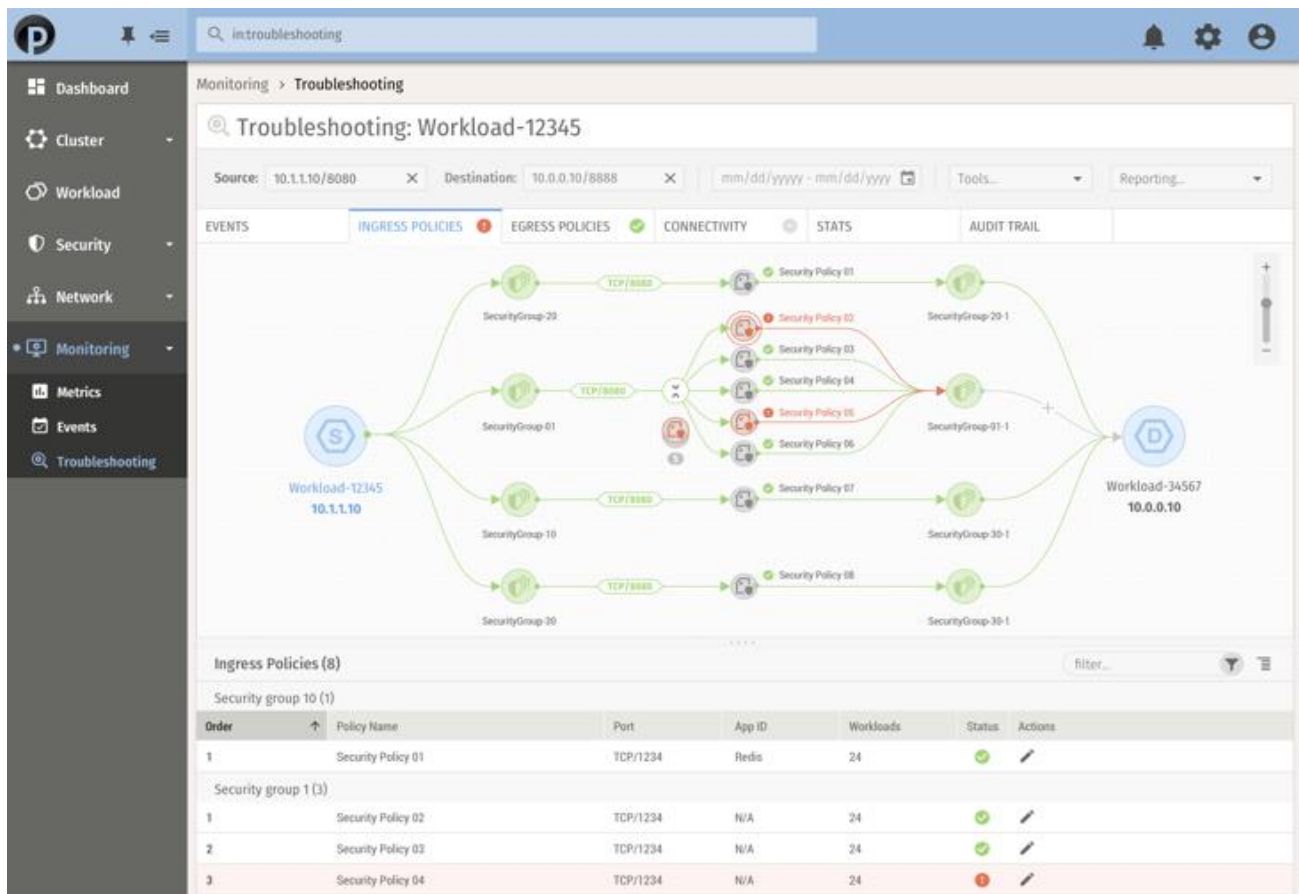
Pensando's Application Congestion Monitoring functionality leverages a processing pipeline to generate alerts by examining detailed TCP state and response timeouts, which helps to quickly isolate application performance issues such as:

- Identify when an application process or server cannot process traffic fast enough;
- Network congestion is causing TCP windowing collapse;
- Enhanced per flow TCP metrics are captured exposing sessions with zero window size, reset or terminated;
- Operating system receive-side buffer exhaustion/drops (RXBuffer), while identifying impacted applications;
- End to End round trip time, with time-stamped flows;
- Filter on any TCP flow state change, resets, first or last packet in a flow, Maximum Segment Size (MSS) changed;
- Identify TCP congestion problems related to zero windowing;
- Monitor TCP handshake related issues, including long handshakes;
- Bandwidth based alerting on thresholds, which can be used to identify abnormal bandwidth spikes from applications.

## Fabric Packet Capture and Health Probing

Packet data plane probes can be instantiated by the Distributed Services Cards (DSCs) at the server edge, centrally initiated from the centralized Policy & Services Manager, to understand and debug sources of application degradation in the network and/or where a packet got dropped in the network. Source UDP/TCP port can be varied, to ensure ECMP paths are exercised for application flows, allowing customers to understand latency end to end, deduce if drops occurred end-to-end across the network for specific flows (Sent SYN, didn't receive a SYN-ACK).

Coordinated packet captures with export from the DSCs to the Policy & Services Manager (PSM) can be initiated, ensuring that when difficult to troubleshoot issues occur, the customer has the complete picture and logs necessary to troubleshoot the problem.



## Summary

The unprecedented scale and diverse connectivity of applications in Enterprise IT has exposed the scale and performance inadequacies of existing software and network visibility and observability tools. The place to understand application behavior and identify sources of application outages is at the server edge, closer to the workloads and directly in-line with the network traffic. To deal with the volume of data and scale of applications, and provide the necessary observability to accelerate troubleshooting, enhance security posture with a move to zero-trust policy, the Pensando Distributed Services Platform delivers a scalable distributed analytics architecture for the next-generation edge accelerated datacenter.



## About Pensando

Founded in 2017, Pensando Systems is the company pioneering distributed computing designed for the New Edge, powering software-defined cloud, compute, networking, storage and security services to transform existing architectures into the secure, ultra-fast environments demanded by next generation applications. The Pensando platform, a first of its kind, was developed in collaboration with the world's largest cloud, enterprise, storage, and telecommunications leaders and is supported by partnerships with Hewlett Packard Enterprise, NetApp, Oracle, IBM, Equinix, and multiple Fortune 500 customers. Pensando is led by Silicon Valley's legendary "MPLS" team – Mario Mazzola, Prem Jain, Luca Cafiero, Soni Jiandani and Randy Pond – who have an unmatched track record of disruptive innovation having already built eight \$Bn/Year businesses across storage, switching, routing, wireless, voice/video/data, & software-defined networking. The company is backed by investors that include Lightspeed Venture Partners, Goldman Sachs and JC2 Ventures. For more information, please visit [www.pensando.io](http://www.pensando.io)