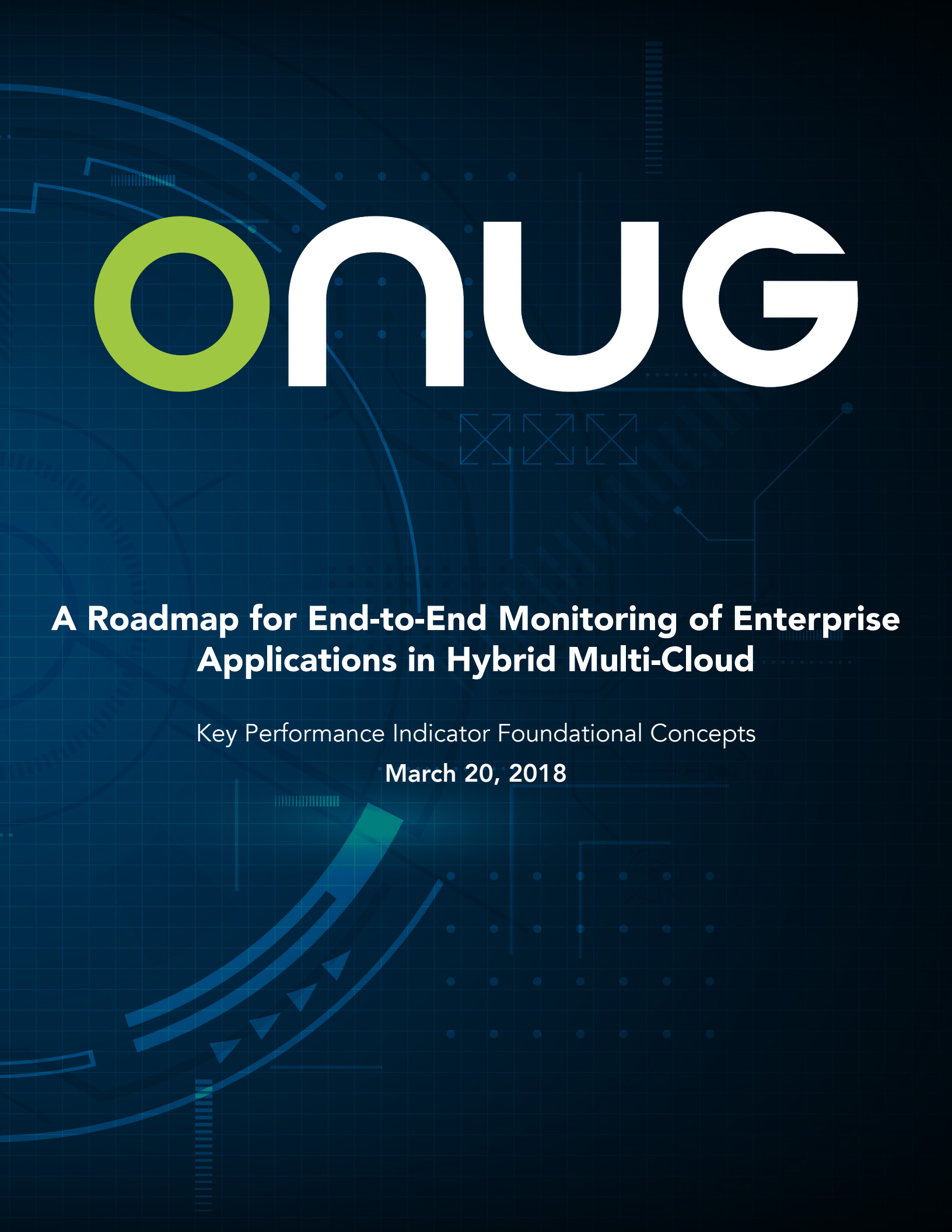

A Roadmap for End-to-End Monitoring of Enterprise Applications in Hybrid Multi-Cloud

Key Performance Indicator Foundational Concepts

March 20, 2018

Table of Contents

Introduction

ONUG provides a forum in which executives from IT organizations and vendors work together to develop guidance and recommendations for the IT industry. The ONUG Monitoring and Analytics (M&A) Working Group is focused on technologies and tools that support application and network assurance.

In 2016, the M&A Working Group published a whitepaper describing some of the most common use cases for monitoring. This document builds on that work and describes a framework for IT Operations Monitoring and Analytics. It introduces the three pillars of an Enterprise monitoring architecture (Application, Network, and Infrastructure assurance). This paper also describes various approaches to monitoring and analytics. Finally, it identifies the Key Performance Indicators (KPIs) across each of the above areas.

The contents of this paper largely reflect existing techniques and KPIs and their respective sources, which are still a fundamental prerequisite to bringing hybrid applications under management and preparing legacy applications for migration to the cloud. While many of these are applicable to multi-hybrid cloud infrastructure, such environments bring their own challenges. These include:

- managing ephemeral, containerized workloads as applications are instantiated, operated elastically, and then destroyed – often in a short space of time;
- consuming very large quantities of data as the rate of metric generation from infrastructure increases, and the quantity of network traffic increases with the number and activity of applications;
- gaining insight into the performance and impact of infrastructure owned and managed by a third party.

These challenges will be addressed in the next phase of the ONUG M&A Working Group's work plan. This paper does not delve into the architecture and building of monitoring and analytics solutions. It too remains the subject of another body of work. However, the ONUG M&A Working Group recommends that any monitoring and analytics solutions be designed and implemented with resiliency and disaster recovery in mind.

Monitoring vs. Observability

Observability is a term and practice made popular by Site Reliability Engineering (SRE), a new approach for releasing new features continuously into large-scale high-availability systems while maintaining high-quality end-user experience, developed by Google in 2003. Observability makes a distinction between monitoring, alerting/visualization, distributed systems tracing infrastructure, and log aggregation/analytics.

For all intents, per the definition of monitoring below, the ONUG M&A Working Group's definition is a comprehensive one inclusive of all data sources. In the working group's view, monitoring starts by understanding the KPIs that are required for managing one's applications, and collecting those KPIs in the most efficient manner available. We do not limit monitoring to any single data source or type. Monitoring is about the metrics that matter; a prerequisite step in a series of steps that may include data normalization, storage, visualization, alerting, analysis through applications of machine learning and artificial intelligence, which, in turn, may support a range of disciplines from operations management to capacity planning.

In order to develop a monitoring strategy for applications, the working group advocates first developing an application and infrastructure map, including all of an application's components and dependencies in software and hardware, then arrive at a full application service chain or stack. Finally, for each layer of the stack, identify the metrics that matter and their respective sources. Automated Application Discovery and Dependency Mapping (ADDM) is key to the success of a monitoring strategy, since it minimizes the manual upkeep required to maintain an accurate set of monitoring targets.

To demonstrate the point, let's consider the hypothetical case of a legacy application's capacity, which may be monitored through transaction rate vs. compute capacity. The former could be found in app server logs, and the latter could be CPU and memory utilization measured through polling of the Host Resources Management Information Base (MIB) via Simple Networking Management Protocol (SNMP). When the same application has been migrated to the cloud, CPU and memory monitoring may make no more sense. Compute is simply provided by the orchestration layer, which elastically expands and contracts between min and max instances to provide capacity. In this case, while the transaction rate is still collected from app server logs, the metric to watch for compute has changed to server restarts and rehydration, which may be found through orchestration layer logs.

This is a top-down approach that will assist in deploying comprehensive monitoring and improving the efficiency of monitoring solutions.

Definitions

This paper uses the following terms as defined below.

Analytics	The analysis of metrics and metadata
Application Assurance	The task of ensuring that an application is available, responsive, and functioning as expected; this does not include application optimization or debugging
Application	Software that runs on IT infrastructure in direct or indirect support of an organization's business
Big Data	In M&A context, the storage and analysis of very large data sets of data and metadata pertaining to Applications, Network, and/or Infrastructure
Instrumentation	Hardware or software that is installed in the IT infrastructure for the purpose of monitoring key performance indicators, events, and configuration
Infrastructure Assurance	The task of ensuring that application infrastructure including compute, storage, and utility services such as Name and Directory service are available, responsive and functioning as expected
IT Infrastructure	The hardware and software that supports IT applications, including network, compute and storage functions (see reference model)
Metadata	Data that describes and gives information about other data often containing insights not readily available in original data
Metric	Data that represents the Key Performance Indicators (KPIs) for activity and health of the IT infrastructure
Monitoring	The act of collecting metrics and/or metadata about an IT environment, either directly or indirectly from the applications, servers, operating systems, application logs, private and public cloud hypervisors, network devices, SDN and SD-WAN controllers, and any other IT infrastructure or application data source
Network Assurance	The task of ensuring that the network, as a fundamental block in application delivery service chain for internal and external customers, is available, being used for transport of legitimate traffic, is responsive to the needs of users and applications it is servicing, and is functioning as expected

IT Management

This document is concerned with the problem of managing large, heterogeneous IT deployments, including both applications and the end-to-end infrastructure that supports them. The infrastructure is composed of services or service chains that may be located on-premises, off-premises (for example outsourcing arrangements), or in the public cloud.

Cloud computing promises the ability to rapidly deploy application workloads through automation, scale these applications in response to demand, and then release resources when the application in question is no longer required. Computing environments today range from simple automation tools for servers running applications to robust, automated cloud infrastructure. Cloud computing environments offer features such as multi-tenancy, sophisticated orchestration, and chargeback accounting. These environments may be public or private. The goal of many organizations is to make the private and public clouds behave the same and offer the same capabilities so that they can choose to move workloads to where they are best suited and most economical. While many organizations aspire to move all of their workloads into a public cloud, private cloud or a combination of both, the reality is that many workloads are still running in simple virtualized environments and natively on bare metal servers. It is also common for organizations to be using multiple public clouds. Such hybrid cloud environments will be with us for a long time, and raise many management and monitoring challenges.

A successful monitoring strategy must address application assurance, network assurance and infrastructure assurance. These assurance areas are discussed in more detail in the following sections. The final section of this document describes the attributes that monitoring analytics systems should strive to provide.

How To Use This Framework

The ONUG M&A Working Group addressed the following requirements when embarking on writing on this document.

- To tactically fill the gaps in an organization's monitoring strategy
- To build and/or rebuild entire monitoring schemes using this framework as a guideline
- To evaluate existing monitoring schemes against the framework
- To be used as an educational resource for monitoring and analytics

Reference Model

While a three-tier application consisting of a web, app, and database servers is often used to illustrate a monitoring approach, it leaves out many other types of application architectures, and thus fails to demonstrate some monitoring challenges. In this section, we describe the reference model for the building blocks of applications and services we can expect to see in large enterprises.

Compute

While in today's enterprises there are still a sizable number of physical servers running a single operating system, according to IDC, in 2016, the number of virtualized servers surpassed them and will continue to grow. For this reason, in referring to compute going forward, we are referring to the whole spectrum of compute, from bare metal servers to virtual machines on a hypervisor to containers running in a host OS. Additionally, in reference to legacy servers, we consider everything from distributed to midrange to mainframe servers with their own networking and storage technologies. Large enterprises, in most cases, have the full range of servers described here in in conjunction with some level of compute in the form of Infrastructure as a Service (IaaS) in the public cloud.

Considering the full range of compute is important because while the legacy compute management problem is solved to a large extent, the end-to-end management of a hybrid multi-cloud (HMC) presents new challenges and requires new approaches. This provides a good opportunity to revisit one's end-to-end monitoring strategy as the proliferation of point monitoring solutions that come with the adoption of HMC may not be sustainable economically and operationally. Tool standardization, automation, and use of big data solutions should be considered as key elements of a new strategy.

Application

Typically, discussions of managing newer and modern applications focus on an n-tier application consisting of a web server, an app server, and a database in its simplest form. However, a comprehensive discussion of enterprise management must consider other types of applications that present their own challenges and often require additional tools or approaches. This paper recommends that management solutions be designed with the complete set of requirements in mind. Besides the typical n-tier microservices applications, we note the following additional enterprise applications that are widely used within large enterprises and are also being offered as Platform as a Service (PaaS) cloud services:

- Unified communication and collaboration
- email
- Virtual Desktop Interface

Network

As applications evolve to smaller more modular components that may be distributed or mobile, the network is the glue that enables this transition. The complete perspective for the purposes of network management includes: Software-Defined Networking (SDN) and Software-Defined Wide Area Networking (SD-WAN) implementations, Cloud Service Provider (CSP) networks, which are restricted overlays, and legacy data center including network service provider offerings. In other words, everything that forms an application's communication service chain connecting its components' user communities is considered the network. The network category boundary is blurred with application infrastructure. Networking aspects appear in both the application infrastructure and networking sections.

Storage

We acknowledge that storage plays a crucial role in compute availability, data protection and resiliency, as well as application performance. However, this paper has chosen to leave an in-depth coverage of storage management for a later date.

Application Assurance

Definition:

For the purposes of this document, application assurance is the task of ensuring that an application is available, responsive and functioning as expected.

The definition of acceptable responsiveness will vary from application to application, and in many cases may be defined by a service level agreement (SLA).

Today's IT is responsible for applications and services ranging from custom applications to Commercial off the Shelf (COTS) applications to Software as a Service (SaaS) applications to Platform as a Service (PaaS) platforms.

The nature of applications has evolved over the past decades along with advancements in computing platforms and development methodologies from monolithic applications to multi-tier applications to microservice architecture applications. Similarly, the nature of delivery has changed to be more agile as characterized by new approaches such as DevOps, which was introduced at the Agile conference in Toronto in 2008. While this paper is focused on the "Ops" portion of the DevOps, it is worth noting that there are monitoring tools and techniques such as call stack traces, application and app container logs, and Real User Monitoring (RUM) that provide detailed insight into applications' performance but tend to be in the "Dev" toolbox due to cost and skillset often associated with their use.

In a microservice architecture, an application is decomposed into many dozens or even hundreds of microservices that can be maintained largely independently of each other and interact via Application Programming Interfaces (APIs). This model naturally maps to an agile software development model, which employs small teams, and allows applications to be updated much more frequently. The microservices architecture also naturally maps to a container computation environment. While microservices applications bring many advantages, they also introduce many dependencies, both within the application and with external components, such as external web APIs, and enabling services, such as Domain Name Server (DNS), Network Time Protocol (NTP), Dynamic Host Configuration Protocol (DHCP) and service registries.

Multi-tier applications often comprise a web front end, an application server and a database. However, in large legacy enterprises, they may still have dependencies on data or transactions handled by legacy mainframe and midrange platforms with their own databases. Additionally, all applications will have dependencies on common enabling services such as name services.

The function of applications can vary greatly. Websites, e-commerce, point of sale systems, Unified Communications (UC) & Collaboration (UCC) systems, Virtual Desktop Interfaces (VDI), and healthcare record management are just a few examples. Some, such as UC and VDI, even have specialized combinations of application code and compute platforms. Hence, a complete reference model is needed to ensure that the framework is complete.

Application Discovery and Dependency Mapping (ADDM) is a core requirement for application monitoring. It has also emerged as an important prerequisite for microsegmentation networks and security configurations. The reasons for this are that companies simply do not have good documentation about older applications, while newer ones seem to be in a state of lag between latest deployment and documentation. Another important emerging use case of ADDM is to track the application workloads and dependencies in an elastic container environment, where the ephemeral nature of workloads may have them in existence for only seconds at a time.

The consensus of the ONUG Community is that the tools that provide ADDM solutions must provide a programmatic means of exporting that data through modern and widely used API and data format technologies such as RESTful APIs and JSON (Javascript Object Notation). The uses of ADDM data are too important and varied. They include policy formation for microsegmentation networking, policy formation for a variety of security applications, Configuration Management Database (CMDB) population, and deployment validation.

Key Performance Indicators to collect

KPI	Source	Approach	Notes
Availability	Packets, synthetic transactions	Passive network monitoring, Synthetic Tx	
Response-time	Packets, synthetic transactions	Passive network monitoring, Synthetic Tx	
Errors	Packets, synthetic transactions	Passive network monitoring, Synthetic Tx	

Identified appraches to monitoring KPIs

Example Approach 1: Wire Data Monitoring

Most applications use a network to transfer information, both within the application and between the application and the broader enterprise. For example, a given application component may interact over the network with:

- Other tiers of microservices in the same application
- Clients in other applications and services
- End-users
- Web APIs and services
- Enabling services such as DNS, NTP and service registry

Using deep packet analysis to analyze this traffic can reveal information about the health, structure, and behavior of the application.

KPIs addressed by this approach

KPIs that can be obtained by wire data monitoring include:

- Protocols in use
- Transaction and data volume
- Response time of transactions
- Transaction error codes
- Metadata about transactions, e.g., transaction type, database, queue
- Values of transaction fields, e.g., a transaction ID or value
- Dependencies between different application components and on other services

Advantages and limitations

Packet flows provide a rich source of information for application assurance. They also provide direct evidence of the dependencies within and between applications.

If the packet analysis tool provides the ability to record packets, then the detailed contents of the packets can be used to perform advanced troubleshooting. For example, to identify a malformed Session Initiation Protocol (SIP) extension in the Voice over Internet Protocol (VoIP) signaling in a contact center environment or a field being incorrectly populated in a transactional protocol.

If network monitoring is performed using passive taps or mirror ports, it doesn't place any load on the application servers or virtual machines being monitored. If the network cannot be directly accessed, as may be the case in a public cloud environment, a process can be installed on the virtual machines that accesses the packets as they flow into and out of the virtual machine – either performing the analysis in place or using a packet encapsulation technique such as GRE (generic routing encapsulation) to forward some or all of the packet flows to a dedicated packet analysis process.

It is common for any encryption of external connections to a physical or virtual data center, e.g., over HTTPS, to be either terminated at the firewall or load balancer (often referred to as SSL offload) or at the first application tier, e.g., the web server. Encryption is not mandated in many types of enterprises, and not using it between application tiers and microservices provides excellent visibility for both application assurance purposes and for detection of cyber-attacks. However, in some verticals, notably banking and healthcare, SSL/TLS encryption is more likely to be used inside applications. In these situations, packets can either be decrypted by an authorized passive decryption device that has access to the SSL/TLS private keys or by a proxy that terminates the SSL/TLS connection on one side and establishes a new connection on the other. The packet analysis may then be performed in the decryption device or by a device connected directly to it.

Typical admin resource requirements

There are numerous tools that require access to packets, and it is common to install taps at key network locations, such as the firewalls at the edge of the data center, and use a packet broker to send copies of packets to multiple tools, including application assurance and security tools. In the case of software installed on virtual machines, the software can be simply added to the image or recipe for the virtual machine, so that the monitoring software is instantiated at the same time as the virtual machine.

Example Approach 2: Synthetic Transactions

Synthetic transactions generated by either hardware or software allow application and network connections to be tested at times when there is no production traffic and in places that are otherwise difficult to instrument. These solutions typically test targets every one to five minutes, but testing frequency varies based on the granularity and infrastructure coverage desired. This ongoing monitoring provides a baseline for application response time during off and on hours, from which deviations can be flagged. Additionally, strategic deployment of synthetic transaction agents in key user community locations, outside and inside the DMZ, as well as in the data center in front of application servers can greatly simplify the

task of fault isolation when issues arise.

Two of the most common examples of testing when there is no production traffic available are before a service has gone live, or during times of the day or week when there are no active users. In both of these cases, synthetic transactions allow testing whether the application can be reached and can perform a basic transaction, and its responsiveness prior to it being used by real users.

SaaS applications owned and operated by third parties, for example SalesForce.com or Office 365, are examples of applications that cannot be instrumented directly by an IT organization. Nonetheless, IT is responsible for the SLA to the end-users. Synthetic transactions are an ideal way to monitor such applications.

Synthetic transactions can be generated by either hardware-based or software-based solutions.

KPIs addressed by this approach

In principal, synthetic transactions can be written for almost any application, and can range from simple connectivity testing through to complex sequences of transactions. In practice, testing HTTP applications is one of the most common uses of synthetic transaction-based monitoring.

KPIs that can be generated from synthetic transactions of transactional protocols like HTTP include:

- Application response time (excluding network delay)
- Number and type of any errors returned by the application
- Site reachability
- Round trip network delay from the test appliance to the server

Note that the last two are actually network assurance KPIs that directly impact application response time.

The use of synthetic testing for measuring the ability of a network to carry real-time media services is covered in the Network Assurance section.

Advantages and limitations

As noted above, synthetic transactions enable applications to be tested in situations where it would be otherwise impractical. Webpage load time analysis includes the effect of any equipment such as cache servers (e.g. in a content delivery network). Testing can also show the impact of application design on production latency. For example, very "chatty" applications which employ fine-grained communications are typically more affected by network latency than applications that transfer data in larger blocks. For website optimization, modern browsers and browser plug-ins provide a native timing object that tracks the time taken during the various phases of loading a web page, and measurement tools and services enable automation of these tests.

One disadvantage of this approach is that typically, except for very simple and well-defined applications, e.g., a simple REST API, synthetic tests are unlikely to cover all possible sequences of events. This is especially true for applications like webpages that are directly exposed to human input. The solution that can deal with programming full sequence of transactions and human input tend to cost more but also often require special test accounts in the application that don't impact the ledger or inventory as well as requiring SSL/TLS keys for the application. While the cost and complexity may be justified to monitor an enterprise's most critical applications, a simpler test of an application entry point often acts as a good proxy for application availability in less critical cases.

Another disadvantage of synthetic transactions compared with analyzing live traffic is that the client software connecting to the application is typically generated from a fixed location and with a structured set of tests, whereas the real clients are likely come from different sources and span different versions, potentially uncovering edge cases in application delivery. The strategic deployment of synthetic tests at major user communities, VIP locations, outside and inside the DMZ, and in front of application servers provides an easy, quick, and powerful way to isolate the fault domain including that of a user's equipment by comparing results from different locations and paths for the same transaction.

Typical admin resource requirements

Synthetic testing requires some initial planning and configuration. The amount will depend on how comprehensive the tests are; scripts designed to simulate complex sequences of events will inevitably take longer to prepare and may need to be updated when the application is updated. Simpler synthetic transactions (e.g., those intended to check reachability and basic responsiveness) will be quicker to produce and required less maintenance.

Additional approaches to characterize application performance include wire data analytics, deployment of agents designed for monitoring application and database containers, and generation and analysis of application logs.

Logs can be a valuable source of data for custom applications' performance and troubleshooting. Needless to say, that to be meaningful, application and infrastructure logs must be analyzed in an integrated fashion across an application's components along with any other time series performance data available. There is a wide range of open source and commercial log analysis tools that are suited for this task.

Infrastructure Assurance

Definition:
For the purposes of this document, infrastructure assurance is the task of ensuring that application infrastructure including compute, storage, and utility services such as Name and Directory service are available, responsive and functioning as expected. (Network is covered under Network Assurance.)

The infrastructure in a data center may broadly be divided into network, compute, storage, and associated management functions. In a physical data center, some information obtained for the purposes of monitoring and analytics will directly relate to hardware. However, in an environment with virtualized computation, the hypervisors can also be considered as part of the infrastructure, because the health of a virtual machine is just as dependent on the hypervisor as it is the physical hardware underneath. In a software-defined data center, there will also be virtualized network functions, including virtual switches, routers, firewalls, security functions and SD-WAN functionality. These too form part of the infrastructure.

Network infrastructure assurance is distinct from network service assurance in that it relates to the health of networking devices and software. This is distinct from the health of network traffic, although the two are closely related. For example, the dynamics of network traffic load will influence the consumption of network infrastructure resources, while misconfigured or overloaded network devices will impact the flow of the traffic.

Key Performance Indicators to collect

KPI	Source	Approach	Notes
CPU utilization	CLI, SNMP, WMI, REST API, system files	Device-based, orchestration and management, agents	By each layer of the compute stack
Memory utilization	CLI, SNMP, WMI, REST API, system files	Device-based, orchestration and management, agents	By each layer of the compute stack
Network interface utilization	CLI, SNMP, WMI, REST API, system files	Device-based, orchestration and management, agents	Virtual or physical
Temperature	CLI, SNMP, WMI, REST API, system files	Device-based, orchestration and management, agents	Specific metric may vary by vendor
Fans	CLI, SNMP, WMI, REST API, system files	Device-based, orchestration and management, agents	Specific metric may vary by vendor
Uptime	SNMP, synthetic transactions		
Virtual Instance State	API, Log		It is an aggregate metric

Identified approaches to monitoring KPIs

Example Approach 1: Built-in APIs

This approach exploits monitoring attributes that are directly built into the infrastructure being monitored.

One of the most common examples is device management. Historically, network device management has used the MIB although transition to YANG in networking and other JSON and XML based schemas in other devices is taking shape. Sticking with a historical view, MIBs are accessed via the SNMP, and allow a remote client to read and write configuration and state as well as setting traps in the MIB. While the use of MIBs is not restricted to network devices, and MIBs are also integrated into other parts of data center infrastructure including compute and storage components, more recently all devices are moving to open management schemas with RESTful interfaces. While these information databases can be used to modify the device being polled, for the purposes of monitoring and analytics, it is their ability to store health and usage metrics that is interesting, along with the ability to set traps on these values.

Another example is the Windows Management Instrumentation (WMI) framework, which can be accessed by DCOM (Distributed Component Object Model) or the Windows Remote Management (WinRM) SOAP (Simple Object Access Protocol) web service. Like the MIB, the WMI framework can be used to actively manage infrastructure, but it can also be used as a valuable source of metrics and status information for monitoring and analytics purposes.

KPIs addressed by this approach

Information sources like MIBs, schemas and WMI can provide a very rich set of data. Some of the metrics that are relevant to infrastructure assurance include:

- Network router and switch configuration and state (including forwarding tables)
- CPU usage information
- Memory usage information
- Storage capacity and utilization information
- Advantages and limitations

One of the advantages of this approach is that there no need to install dedicated monitoring agents on the device being monitored; instead, this approach exploits information that is already generated on the target device or software. However, as noted below, it may still be necessary to obtain credentials to access the information desired.

While there are many standardized MIBs and open schemas, many devices also provide private objects, which contain non-standard fields or information that is vendor specific. While this extends the amount of information that can be provided, it also means that any monitoring and analytics software consuming a private object has to be updated so the object can be parsed.

Another limitation is that built-in APIs are typically fixed and not extensible by the consumer.

Typical admin resource requirements

In some cases, it may be necessary to explicitly enable the generation of a particular metric or trap in a management object. This may require root or administrator privileges, and may also impact the performance of the device being monitored. Access to management APIs usually requires some form of authentication, which must be configured both on the device being monitored and the client receiving the information. It is good practice to use accounts with read-only privileges for monitoring purposes.

Example Approach 2: Orchestration and Management APIs

Controllers that provide services such as virtual machine orchestration or SDN management typically provide APIs to support data center automation. Like the built-in APIs discussed in the previous section, these APIs can be used both to change the controller's settings, but also to read any metrics that are maintained by the controller.

KPIs addressed by this approach

The infrastructure information provided by management APIs will be similar to that provided by the devices themselves. In addition, a controller is likely to have access to information that individual devices do not; for example, information about groups of devices.

Advantages and limitations

The first benefit of this approach is that the monitoring system only needs to connect with a small number of controllers, rather than a large number of individual devices. The second benefit, as noted above, is access to information that is not available to individual devices. However, as with built-in APIs, information will be limited to that provided by APIs, and may not be complete.

Typical admin resource requirements

Since there will be far fewer controllers than the devices they manage, the set-up and maintenance of controller API connections will require less work than managing connections to many hundreds or thousands of individual devices.

Example Approach 3: Agents

As the name suggests, agents act on behalf of a client to gather and process information. For example, WMI is not available on Linux servers or virtual machines, but there are plenty of statistics stored in system directories that can be read by an agent and formatted as required. The agent can stream data or notifications to a client or the client can poll the agent at regular intervals.

KPIs addressed by this approach

The infrastructure information provided by agents is similar to that provided by the devices themselves, although, as noted below, it is also possible to generate new metrics derived from the raw device metrics.

Advantages and limitations

Agents allow access to information that is not exposed via built-in APIs such as WMI or a standardized MIB. They can also perform formatting, pre-processing and even data reduction or consolidation before passing data to the collector. However, it should be noted that agents must be installed on the infrastructure itself. This may not be possible with physical devices such as routers and switches. The installation may also require root or administrator privileges. In the case of servers, this may be relatively simple, since it can often be accomplished using automation techniques and server templates.

Example Approach 4: Logs

Many devices generate log files, which can contain information about the activity of virtualized network functions, applications, and the operating systems that host them. Log files are typically collected and published by an agent and sent back to a central repository in their original form where they can be stored and analyzed. To maximize value, logs should be analyzed in an integrated fashion along an application's service chain using open source or commercial log analysis tools.

KPIs addressed by this approach

There is no set format for log entries, and they are typically semi-structured text data. While log files can be used to generate KPIs, it is more usual to index them and then search them.

Advantages and limitations

Log files contain a lot of detailed information that can be helpful when trying to find the root cause of a problem. The downside is that extensive log collection can also create a lot of network traffic and require large amounts of storage.

Typical admin resource requirements

As noted, the format of log entries and will vary greatly from application to application and from vendor to vendor. This may require the administrator of the log collection system to write custom scripts or rules when new applications or devices are introduced into the data center.

Network Assurance

Definition:

For the purposes of this document, network assurance is the task of ensuring that the network, as a fundamental block in application delivery service chain for internal and external customers, is available, being used for transport of legitimate traffic, is responsive to the needs of users and applications it is servicing, and is functioning as expected.

Network availability is an intuitive concept. When the network is not available, no communication can occur between application components, including services such as DNS and Lightweight Directory Access Protocol (LDAP), and no communication can happen between applications and their users. Far more subtle are partial network failures or degradation.

Application performance over the network is directly affected by network roundtrip delay, jitter, and packet loss. Network roundtrip delay impacted by a number of factors, including network congestion, physical distance, and topology.

The requirements for networks vary significantly across applications and industries. Network response time is a function of bandwidth allocated, distance, and current utilization. Applications that use batch processing, for example, do not require very fast network response times. In contrast, transactional applications require adequate network responsiveness to ensure acceptable response time to users. There is a wide range of subjective criteria in this category. Consider the response times that you may tolerate in your interactions with a retail website, a healthcare portal, and the United State Internal Revenue Service, for example. On the other extreme of responsiveness and performance, there are market data feeds for financial trading or high-speed data buses that require networks with microsecond latency.

It follows then that network requirements should plan to satisfy the requirements for the most stringent applications that run on it. Understand the application responsiveness budget first, and then identify your application, compute, and storage latency, in order to identify a network latency budget. Independent of your application type, there are a few golden rules. First, users perceive lag on a display screen after 200ms, a VoIP channel requires 40 kbps of bandwidth to sound clear, and users tend to leave transactional websites, retail and otherwise, if their responses have not appeared within one second.

The network can be built private, outsourced to a service provider, or can use a combination of private and outsourced components due to public cloud and SD-WAN adoption or due to operating a mobile or remote workforce. The last mile may be also increasingly either WiFi or LTE. Therefore, it is vital to develop network Service Level Objectives (SLOs) and SLAs in order to manage your internal and external network service providers. It is recommended that these SLOs/SLAs be developed in terms of both availability and response time and not just availability.

Network infrastructure is constructed from physical communication devices, cabling, fiber, antennas, and so on. All of the components are susceptible to failure with time and misconfiguration with change. The net result is errors that impact network availability and responsiveness. Such examples include mixed duplex faults, improper local and global load-balancing, one-way routes or no routes, and data-link layer errors on the wire causing dropped packets. As a result, monitoring for and tracking such errors is a requirement of network assurance in addition to monitoring the availability, response time, and the use of the network (who, what, when, and how much).

Current IT infrastructure trends reflect the rapid adoption of new technologies. For example, SDN adds programmability and automation across the IT infrastructure. DevOps practices, rapid service deployment and microservices are enabled by SDN and virtualization technologies such as virtual machines and containers. These technologies allow expansion of the domain of control to larger networks, but they create infrastructure scaling challenges.

While virtualization has enabled software control, the scaling of IP services for both private and public networks requires a strategy for networking and security. Overlay network technologies such as IP virtual private networking (VPN) and ethernet VPN (EVPN) offer one popular way to manage and segregate the infrastructure. Overlay/underlay networks are also used to provide large Layer 2 or Layer 3 network domains in data centers and to support multi-tenant operation and backwards compatibility to legacy networks. In the majority of cases, the old legacy management solutions, while still required, will not extend to overlay networks, public cloud, and SaaS applications. A significant challenge in monitoring and analytics is to be able to correlate the logical overlay to the physical underlay.

Problems in both the underlay and the overlay networks can impact the health of services and applications running over the overlay network. Therefore, monitoring functions must provide service and application performance information (e.g., response times), must be able to identify data with respect to both the overlay network (e.g., 5-tuple, VNI/VSID, virtual port,

etc.) and the underlay network (e.g., 5-tuple, VLAN, port, node, etc.); analytics functions must be able to analyze metrics in terms of both the overlay and underlay network.

One of the key promises of SDN is microsegmentation. This capability is used to provide zero trust networks for on-demand and elastic applications, and allows rapid rate of network overlay change through automation. The deployment of microservices and container technologies adds an ephemeral nature to workloads, and monitoring has to be able to track the orchestration and deployment of these services. Monitoring network state and checking for drift in network overlays is a relatively new but important requirement. Drift is defined as divergence between the desired and the actual state.

Key Performance Indicators to collect

Network infrastructure

KPI	Source	Approach	Notes
CPU utilization	Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	
Memory utilization	Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	
Network interface bandwidth	Wire data, Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	
Network interface error rate	Wire data, Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	
Temperature	Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	
Fans	Device with SNMP, API, streaming telemetry, or logs	Domain managers exporting raw and metadata or direct export to big data platform	

Network data plane assurance

KPI	Source	Approach	Notes
Network response time	Wire data, synthetic transactions		
Bandwidth utilization by protocol/ application and user	Wire data, xFlow, SNMP or streaming telemetry		Variants of NetFlow, noted xFlow, SNMP, streaming telemetry tend to have less granularity than wire data here
Packet capture and decode	Wire data solutions		Whether specialized or running as an app on compute
Link up/down events	Wire data, network device SNMP, streaming telemetry		
Profile of traffic on a link (volume in bytes and packets)	Wire data, network device SNMP, streaming telemetry, xFlow		
Packet loss and Jitter	Wire data		

Error frame statistics	Wire data, network device SNMP, streaming telemetry		
Network delay (propagation and queueing)	Wire data, network device SNMP, streaming telemetry		
Microbursts	Wire data		

Note: the data plane metrics can be advantageously broken down by Q0S marking (e.g., DSCP), Virtual network identifiers, e.g. VLAN, VRF ID, VNI, VSID, unicast and multicast, and transmit and receive

Network control plane assurance

KPI	Source	Approach	Notes
Running configuration	CLI		What the user explicitly controls
Forwarding state	CLI & SNMP		What results from user configuration and protocol-based learning: FIB, ARP, MAC, ACL, NAT table
Link up/down events	SNMP trap, Synthetic tests		
Profile of control plane traffic	Wire data, network device SNMP, streaming telemetry, xFlow		
Error frame statistics	Wire data, network device SNMP, streaming telemetry		e.g., Malformed packet
Routing convergence time	CLI & SNMP		
Compute health	See network infrastructure section, employ synthetic transactions		See Network Infrastructure Metrics
Device responsiveness	See network infrastructure section, employ synthetic transactions		
Health checks	See network infrastructure section, employ synthetic transactions		

Identified approaches to monitoring KPIs

Example Approach 1: Wire Data Monitoring

In addition to providing insight into the health of the applications running on a network, wire data monitoring can also be used to gain visibility of the health of the network itself.

KPIs addressed by this approach

Wire data monitoring can provide a rich set of network assurance KPIs.

- Network delay (propagation and queueing)
- Virtual network identifiers (VLAN, VXLAN VNI, nvGRE VSID)
- Class of service (IP DSCP, MPLS class of service)

- TCP problems, e.g. zero window size, resets
- Bandwidth utilization by protocol/application and user
- Link up/down events
- Profile of traffic on a link (volume in bytes and packets)
- Packet loss and jitter
- Impact of network packet loss and jitter on voice and video mean opinion score
- Frame error statistics
- Microbursts

Advantages and limitations

Analysis of wire data provides direct insight into the operation and health of the network. None of the KPIs listed above are affected by SSL/TLS encryption or SRTP (Secure Real-time Transport Protocol) encryption of voice and video, which does not encrypt the RTP header, only the payload.

Typical admin resource requirements

See the section on wire data monitoring for Application Assurance above.

Example Approach 2: Flow-based Monitoring

Flow metadata collection mechanisms, like NetFlow, IPFIX, J-Flow, Netstream, AppFlow, and CacheFlow, collectively referred to as xFlow in this paper, are typically generated by the network infrastructure and provide a summary of the traffic flowing on the network.

KPIs addressed by this approach

Basic flow information provides the following KPIs:

- Source IP address
- Destination IP address
- Protocol
- Service (reported as the server-side TCP or UDP port)
- Client TCP or UDP port
- Number of packets transferred in each direction in the reporting interval
- Number of bytes transferred in each direction in the reporting interval

Some of the more recent flow engines may also attempt to identify protocols through heuristic rules or using information provided in templates.

Advantages and limitations

Flow information can be a useful tool in situations where it may not be possible to deploy a full deep packet analysis probe, for example, at a small, remote site. It can also be used to understand application dependencies, although it does not provide the application assurance KPIs provided by wire monitoring. Enabling flow generation places additional load on a router or switch, which may reduce its performance.

Typical admin resource requirements

Flow is typically enabled via the configuration settings of the network or device. Collecting basic flow information does not require much configuration; however, more advanced use cases, such as providing templates, is more involved.

Example Approach 3: Synthetic Transactions from Network Infrastructure (e.g., IPSLA or RPM)

Some routers support synthetic transactions either to a real server or between two routers. For example, many routers can establish test connections to DNS, HTTP and DHCP servers. It can also initiate ICMP ping tests and TCP and UDP socket connections. They may even send VoIP RTP packets between routers. The results are typically retrieved using SNMP.

KPIs addressed by this approach

Network assurance KPIs that can be obtained by synthetic transactions from the network infrastructure include:

- Round-trip delay
- Response time (for server connection)
- TCP statistics
- Packet loss and jitter

Advantages and limitations

These types of tests are relatively easy to configure and can be useful in proactively detecting a loss of connectivity or sudden change in the delay of a route. As with any proactive testing, the benefit is early detection of problems.

Typical admin resource requirements

Synthetic tests originating from a network device are typically set up using the network device's configuration interface. Tests originating from an external system use a wide variety of configuration methods.

Other measurement methods

Additional approaches to characterize network performance include device KPI through SNMP, API, Streaming Telemetry, or Log and configuration management information.

Additional Concepts

In the light of the industry-wide application of big data analytics to enterprise infrastructure management, the ONUG Community feels the following topics are worthy of mention.

Metric tagging

Full end-to-end management requires KPI collection from everything in the application service delivery chain. In this picture, a compute instance running an application server or a container running a piece of microservices code for an application has a very clear and singular role. Its context is very clear. If we draw a map of the service delivery chain for that application, it will only appear in one place.

In contrast, a data center WAN router supports external communication for everything in that data center. It has a much broader context and will, in turn, appear in every application's service chain map. In spite of that, legacy management schemes relegate the monitoring of that router to a network management tool within a network management silo.

Metric tagging refers to the ability to tag metrics, if appropriate, with a multiplicity of contexts. such as line of business, application, geolocation, VLAN, and VNID. so they can be properly identified and processed by big data solutions.

Customer/Service provider feedback loops

This item refers to the shortcomings of current CSP offerings. In the current model, enterprises lease infrastructure on which to run their applications. They almost always have means of monitoring the availability and performance of their critical applications. However, migration to the cloud has introduced a compute and network combination without traditional levels visibility. The idea is that if the CSP manages to its SLA, everything will be fine. However, the experience of such offerings over the past three decades has shown that these SLAs are often deficient and problems eventually arise.

A feedback loop, in a form to be determined, would ensure that the CSP is also aware of an issue with applications running on its infrastructure and could quickly verify their operational integrity instead of relying on the customer to exhaust all possibilities and use valuable time.

Trigger-based collection of expanded network metrics

Customers have requested the ability to collect more detailed information when traffic patterns deviate from norms or are interrupted. Supporting the ability to collect more frequent network traffic KPIs or entire network traffic traces due to event triggers is a requirement. This requirement acknowledges the need for a preexisting tap and packet recording appliance. The data and traces generated by the trigger could be saved to a local packet recorder appliance or tunneled to a remote appliance for later analysis.

Summary

This paper has described various approaches to monitoring and analytics from three different, but related, perspectives:

- Application assurance
- Infrastructure assurance
- Network assurance

The most suitable KPIs and approaches will vary from situation to situation, and many of the examples provided complement each other.

This paper has sought to describe existing KPIs and methods. The ONUG M&A Working Group is now turning its attention to the challenges presented by highly agile, containerized application environments operated in hybrid-multi cloud environments, including on-premises software-defined data centers, public cloud and hybrid cloud. As noted in the introduction section, such environments bring their own challenges, which are likely to require the development of new KPIs and approaches, the requirements for which will be described in future M&A material.

ONUG Monitoring & Analytics Working Group

Editor
Babak Roushanaee, NETSCOUT

Co-Chairs
Aryo Kresnadi, FedEx
Ted Turner, Intuit
Paul Barrett, NETSCOUT
Richard Whitehead, Moogsoft (2018)
Nabil Bitar, Nuage Networks (2017)

M&A Working Group Members
Stephen Collins, ACG Research
Shane Jenkins, Adobe Systems
Venkatarao Mokkapati, American Express
Yan Filyurin, Bloomberg
Chris Cheu, Cigna
Kunal Mahajan, Columbia University
Vesko Pehlivanov, Credit Suisse
Yong Xue, DISA/DoD
Jeff Kopko, DTCC
Jamie Jacobs, eBay
Luyuan Fang, eBay
Mark Kisner, Fidelity
Michael Wynston, First Data
Regis Rogers, General Electric
Snehal Patel, Gap
Jem Pagan, JNK Securities
Russ White, LinkedIn
Rhett Dillingham, Moor Insights & Strategy
Ian Flint, Oath
Harinder Singh, Sainsburys
Niranjan Nandakumar, Susquehanna International Group
Brian Anderson, Tegna
Arpit Rana, University at Buffalo
Yufeng Xin, UNC at Chapel Hill
Sean Wang, University of British Columbia
Victor Liu, Visa
Kaya Westling, Wells Fargo
Abhinav Modi, Avi
Kerry Takenaka, cPacket
Josh Joiner, cPacket
Nikhil Handigol, Forward Networks
Michael Haugh, Gluware
Don Fedyk, Hewlett Packard Enterprises
Srihari Venkiteswaran, Hewlett Packard Enterprises
Linda Dunbar, Huawei
George Zhao, Huawei
Wei Wei, Huawei
Huiyang Yang, Huawei
Margaret Chiosi, Huawei
Konstantinos Kanonakis, Huawei
Jacopo Pianigiani, Juniper
Travis Newhouse, Juniper
Deepti Chandra, Juniper
Eric Graham, Kentik
Justin Ryburn, Kentik
Totti Murakami, NetOne Systems
Yusuke Fujita, NetOne Systems
Daisuke Morita, NTTi3
Jeff Tantsura, Nuage
Mostafa Mansour, Nuage
Gopi Gopalakrishnan, ThousandEyes
Sanjay Mehta, ThousandEyes
Brighten Godfrey, Veriflow
Bryan Larish, Verizon
Cheng Liu, Verizon
Dogu Narin, Versa Networks