Gaining Application Visibility with VMware vRealize Network Insight

Joep Piscaer

CONTENTS
Application Discovery and Visibility ........................................... 2
Dependency Mapping ................................................................. 4
Plan for Micro-Segmentation ......................................................... 4
The Donut Deep Dive ................................................................. 6
Automatically Generated Firewall Rules ........................................ 7
In Search Of .............................................................................. 9
Entities and Projections .............................................................. 9
Audit Trails ............................................................................. 10

IN THIS PAPER
Applications are becoming more complex all the time, as are the environments in which they’re run. That means you need a powerful tool to track your application performance and efficiency, as well as spot potential trouble spots. The best way to do all that is with a single tool: VMware vRealize Network Insight.

Highlights include:
• Taking application inventory and discovery
• Application dependency mapping
• Security planning
It’s never been more true than it is today: it’s all about the application. The issue is that applications are infinitely more complex than they used to be. Apps can be anywhere now, and are often broken down into smaller pieces that can be scattered all over—the cloud, the network edge, and many more places.

That means more chances for things to go wrong, including bottlenecks, security breaches, and so on. It makes knowing where your apps are and what they’re doing more important than ever. Without this knowledge, you’re floundering in the dark and your network is more vulnerable and less efficient. Fortunately, VMware vRealize® Network Insight™ has a solution.

**Application Discovery and Visibility**

A key ability of vRealize Network Insight is putting the data it gathers to good use in gaining visibility into applications. It takes that networking data and, with the help of machine learning (ML), constructs meaningful insights into what networking components make up applications, how components are dependent on each other, which are shared, and where the different components run.

The process starts with vRealize Network Insight collectors taking inventory of the various physical components—switches, routers, firewalls, load balancers, and so on—as well as virtual components, including vCenter, NSX, and AWS inventories.

By turning on network flow collection, vRealize Network Insight helps admins understand the movement between application components, allowing network engineers to look at traffic data with an application perspective.

As mentioned, this method uses ML to discover application boundaries automatically. First, traffic flows are analyzed to figure out application boundaries, grouping VMs that mostly talk more among themselves, and determining tiers within the application based on similar traffic patterns (like VMs with the same network ports opened). The ML algorithms also detect shared services, identifying services like Active Directory or DNS.

Analyzing its traffic flows, vRealize Network Insight can help determine which workloads on the network communicate and over which protocols. Admins then group these components into applications, and optionally mark components as shared between applications.

The honeycomb widget is an ideal starting place to define and group discovered applications. Each hexagon in the widget is a collection of different infrastructure components that vRealize Network Insight has identified as part of an application definition.

---

**Figure 1:** The honeycomb widget visually represents discovered applications
A traffic flow is defined as a stream of packets, aggregated across individual sessions, with unique source IP address, destination IP address, IP protocol, and destination UDP/TCP port. A flow can represent unidirectional or bidirectional communication.

A flow allows traffic to be summarized for the purposes of security planning. vRealize Network Insight allows admins to dive into individual traffic sessions for more detailed information.

Filters can be applied to show different characteristics, like unprotected flows, talking to the internet, shared application services, or applications with issues. These filters will change the honeycomb visualization.

In addition to traffic flows, vRealize Network Insight has other ways of discovering the relationship between workloads on the network. vCenter tags and custom attributes; AWS tags; workload instance naming conventions; and pre-filled data from a CMDB are other often-used discovery methods. You can also group VMs using a regular expression.

Discovered application sets can be saved. This saves the grouping criteria like the regular expression, the definition and grouping of application tiers, and more. A saved application, like the one shown in Figure 2, allows admins to interact with all application components as a whole, show traffic flows between tiers, and see all networking components involved in traffic flows between application components.

Saved Applications can be viewed using the Application Dashboard, a per-application topology view with all of its VMs and networking devices. This scoping down helps admins to troubleshoot their environments, as it automatically limits where they should look to identify issues.

While this is a relatively new addition to vRealize Network Insight, it is one of the most helpful ways of structuring the networking data into a more logical, higher-level view of what’s going on over the network, allowing networking flow and configuration data to be contextualized in more human-readable formats.

In an application topology, admins can dive into physical network devices and interfaces. Figure 3 shows two physical switches as part of an application definition. In addition, this network overview allows a more detailed look at the relevant switch ports.

Naturally, the application landscape changes continuously. Running vRealize Network Insight’s application discovery process periodically will detail any changes to the landscape, as well as surface up new applications.

Figure 2: An illustration of an application’s topology

Figure 3: The Network Overview tab
Dependency Mapping

Relationships between different workloads are hard to keep track of. By analyzing traffic flows, however, these dependencies can be made visible automatically, as long as traffic is flowing between them.

Because of the high level of cardinality of security rules in a micro-segmentation environment, defining security rules is a daunting task—it’s not as easy as just turning micro-segmentation on.

vRealize Network Insight’s analytics engine analyzes these traffic flows to understand how components interact. An example of a dependency mapping is shown in Figure 4, which shows the relationship between application components based on traffic flow. This helps admins understand the logical relationship between application components.

Different colored lines indicate the direction of flow: outgoing, incoming, or bidirectional. Each of the slices in this “donut” can be clicked on to view dependencies between them, and each line (traffic) can be clicked on to show more details about the traffic flowing between the slices.

Plan for Micro-Segmentation

Even with visibility into applications and mutual dependencies, it’s not easy to plan for micro-segmentation, which is often the main use case customers start out with when using vRealize Network Insight.

Because of the high level of cardinality of security rules in a micro-segmentation environment, defining security rules is a daunting task—it’s not as easy as just turning micro-segmentation on. The effect of micro-segmentation with no firewall rules would be no traffic flows. This is because micro-segmentation equals zero trust, and zero trust means to deny all traffic unless explicitly allowed.

With this level of granularity, getting started is difficult. Imagine going through troves of documentation to find out the communication paths between application components with the correct protocols and ports, translating the different application components and roles to workloads installed in your environment, and manually creating the required firewall rules. Doing that without making errors would be nearly impossible, and errors like these tend to lead to application downtime.

vRealize Network Insight tackles this problem by analyzing the flow data it collects and using that data to generate recommended firewall rules for workloads. It simply looks at the traffic at a granular level, takes all it knows about the underlay network and the workload into account, and translates that information into ready-to-go firewall rules that can be easily imported into NSX.

These rules serve as a good baseline to continue to refine an application’s security posture, and can be tweaked or added to as new traffic patterns emerge. Of course, these rules are based on the traffic being observed, which may include undesired flows. Generated rule recommendations need an experienced eye and health check to filter out unwanted observations, like SSH or RDP access.
Technically, data collection works at the distributed virtual switch to capture these east-west data flows using IPFIX, which is forwarded to the local vRealize Network Insight data collector appliance. For other underlay networks, sFlow and NetFlow can be used as well. Note that NSX is not required for this entire security planning process, which makes it suitable for greenfield NSX deployments as well.

vRealize Network Insight correlates metadata from flows and configuration to determine which flows belong to which VM or container, and which underlying components are involved in delivering the traffic flow.

The Traffic Distribution overview, shown in Figure 5, helps admins understand how traffic distribution is built up. The percentage at the top left shows the amount of east-west traffic, expressed as a percentage of total traffic.

The switched and routed percentages are the amount of east-west traffic that are switched and routed, respectively, adding to 100%.
GAINING APPLICATION VISIBILITY WITH VMWARE VREALIZE NETWORK INSIGHT

Flow Key Metrics offer a timeline view of all flows in a given time period between two specific VMs over a port (Figure 9).

The Donut Deep Dive

The “donut” in vRealize Network Insight is a way to visualize traffic flows, grouped by a category of your choosing and sliced up per grouping. It allows quick filtering to find the proverbial “needle in the haystack.”

The haystack, of course, is the sum of all traffic observed. The needle is the specific flow to be secured using vRealize Network Insight. Every slice or connection line can be clicked to reveal additional information, diving deeper into the data each time. The donut can be scoped down by parent object (like a vCenter inventory) to limit, for instance, micro-segmentation to a single cluster or data center.

Let’s walk through how to use the donut to visualize and isolate specific traffic flows.

In the lower row, VM-to-VM traffic and Within Host/AWS VPC indicate percentages of “locality,” i.e., how much traffic is VM-based and stays within a host or AWS VPC.

Finally, the Internet percentage indicates the remainder of traffic that is not east–west but north–south, or internet-facing traffic, as a percentage of all traffic, adding up to 100%, with the first percentage shown.

In similar fashion, the Top Ports by Bytes shows an overview of top ports.

For both the traffic as well as the ports overviews, additional details are available by clicking on a flow percentage or port.

Figure 7 shows additional flow details for port 3306. Clicking on any individual flow will show key flow properties (Figure 8), like the VM-to-VM path, which shows all underlay networking devices and constructs associated with traffic between the two VMs.

In similar fashion, the Top Ports by Bytes shows an overview of top ports.

Flow Key Metrics offer a timeline view of all flows in a given time period between two specific VMs over a port (Figure 9).
By then filtering a source VLAN, we can visualize all traffic between just these two VLANs and look at the recommended firewall rules for flows between them (Figure 11).

By clicking on any of the VLANs in the donut, admins can see an overview of services in this VLAN, services accessed from this VLAN and more (Figure 12). These overviews provide quick insight into what’s running in a specific network slice, along with the other services it’s consuming.

**Automatically Generated Firewall Rules**

NSX’s distributed firewall also sends IPFIX flow data to vRealize Network Insight, which can add metadata (like firewall rules IDs) and show flows that were blocked by a firewall rule (in addition to NSX, Microsoft Azure also logs dropped flows to vRealize Network Insight).

This allows vRealize Network Insight to show flows that are not protected by any rule. One thing to note with vRealize Network Insight is that it does not give certainty...
about firewall rules in the traffic path between two VMs with other firewall vendors, requiring admins to keep them straight. Use the visibility features of vRealize Network Insight to see which firewalls are in the path and manually determine which rules may be at play.

NSX isn’t the only source that records blocked flows, as Microsoft Azure also logs the flows that get blocked by Azure security policies. Selecting the “Dropped Flows” option causes Azure flows to turn up.

With all of this said, there are some best practices to properly do micro-segmentation that are hard to codify into vRealize Network Insight. In Martijn Smit’s vRealize Network Insight Cookbook, he described these best practices, which are summarized here:

• Start by segmenting traffic on the boundary of an application. This secures the application as a whole from the outside, and is akin to perimeter-based DMZ security.

• Scoping vRealize Network Insight to an application will secure any back-end tiers in a 3-tier application—web front-end, application server, and database in the back-end—so that any external user can only access the front-end, not the back-end. This will also codify application dependencies on external services into an actual firewall rule, explicitly allowing the traffic and making it harder to block by accident.

• Segment and secure traffic within an application based on application tier, which protects different tiers of the applications by allowing only specific types of traffic flows. In vRealize Network Insight, scope the donut to the specific application you’re protecting to get the right firewall rule recommendations.

It’s important to note that while vRealize Network Insight can easily export the rule recommendations to NSX, the rules can be used with other micro-segmentation technology as well.

vRealize Network Insight can export these recommended rules if you’ve grouped by application, application tier, or security group. Options for export formats include:

• CSV (generic format, human readable, easy to transform, combine and edit)

• YAML (when dealing with Kubernetes, and only available when grouping by Kubernetes Namespace or Service)

• XML (for NSX deployments)

Firewall Rule Recommendations

The Recommended Firewall Rules are context sensitive, based on the “group by” selection in the visualization “donut.” If you group by VLAN, the rules are on a per-network basis. If you group by VMs, the rules are on a per-VM basis.

vRealize Network Insight will always try to create the minimum number of rules in a recommendation to create the best manageable configuration, but that does depend on your inputs. Generally, you’ll want to segment based on “applications,” which balances granularity (too many rules) with specific control over security policies.

It’s important to note that while vRealize Network Insight can easily export the rule recommendations to NSX, the rules can be used with other micro-segmentation technology as well.

vRealize Network Insight can export these recommended rules if you’ve grouped by application, application tier, or security group. Options for export formats include:

• CSV (generic format, human readable, easy to transform, combine and edit)

• YAML (when dealing with Kubernetes, and only available when grouping by Kubernetes Namespace or Service)

• XML (for NSX deployments)
In Search Of …

The search functionality is fundamental to vRealize Network Insight. Anything you do in the interface is actually a search command. The search is powered by Elastic Search, and the search language is their Regexp Query language, which is a natural language that’s easy to learn and translates well to the technical nature of vRealize Network Insight.

The search looks through all traffic flow data, as well inventory across vSphere, NSX, VMware Cloud on AWS, native AWS (EC2) and Azure, and events, as well as metrics across time.

Search is a very powerful feature. As vRealize Network Insight collects networking telemetry and configuration, tracks changes, and more, there’s a wealth of data available to the admin. By mastering the search language, admins can take full advantage of vRealize Network Insight’s data collection.

To help admins learn the language, VMware has created “search posters,” which serve as a quick reference to build more advanced search queries. Even though the search box in the UI has auto-completion, using the reference posters can help admins create those advanced queries, listing property and entity terms, function terms and other items to add to a query.

Searches and search results are reusable using pinboards, which are pages with dashboards based on the searches you saved.

Constructing searches can be a little bit daunting, but once you grasp the basic structure of a query, it’s a matter of understanding what entities, properties, filters and groupings are available (using the posters mentioned earlier or the autocomplete in the interface).

For instance, consider the following query:

`sum(bytes) of Flows where Application = ‘SavedApp’ group by Country`

`sum(bytes)` is the projection, `Flows` is the entity_type, `Application = ‘SavedApp’` is the filter, and `Country` is the grouping. The search is logically aware of various constructs; for instance, it understands if an IP address falls into a subnet, if a source IP address resides in a certain country (using GeoIP databases), and more.

There are many entity types, including VM, host, VLAN, flow, application. These are logically sorted into, for instance, VMware Cloud on AWS (VMC), AWS and Azure. Some of these types collate multiple specific types into a single category, called a meta entity. For instance: Azure, VMC, vSphere and AWS VMs are all VMs in the search.

Entities and Projections

Entities have properties. For instance, a VM has an IP address. Or a VM runs on a given host. Or the VM has a given amount of RAM. Using the where filter, specific properties can be searched for. Reference traversal queries allow search queries to reference a specific property, like a host’s memory usage, when looking for an unrelated entity, like a VM. Reference traversal in this example will find all VMs running on a host with high memory usage.

The filter weeds out unwanted results, like packet drops less than 1%, throughput higher than 1 Gbit, and so on. Filters can have operators: equal to, not equal to, in, more than, less than, and, or, like, and so on. Time is another way to filter results, allowing time ranges, rolling periods (the last 3 days, for instance) or specific time/dates.
In addition to built-in alerts, admins can create alerts for changes to any entity in vRealize Network Insight.

The *series* projection combines metrics from multiple objects into a single line graph, like the throughput of all VMs on a single host. These metrics can be any metric in vRealize Network Insight.

Specifying an order in the search query will sort the results by the parameter specified, while grouping bunches together results in a more human-readable, summarized form.

Audit Trails

The same search functionality can be used to track changes to firewall rules. This very is useful for troubleshoot and auditing.