The Self-Operating Data Center

A Guide to Intent-Based Networking

apstra®

To an outsider, a data center is the picture of order. Neat rows of racks, carefully labeled cables, and blinking lights create the impression of organization, reliability, and power. But anyone who actually works in a data center knows the truth: keeping all that traffic flowing correctly between thousands of switch ports is a messy and frequently chaotic undertaking.

Server automation is supposed to eradicate the chaos. But it hasn't turned out that way. At least, not yet. Server automation has made server management agile and repeatable – thousands of nodes behave exactly the same, so losing one is insignificant.

However, network automation is stuck with being fragile and unrepeatable. How do you automate thousands of disparate elements that behave differently, yet must behave well together?

When good intentions go bad

The first step toward network automation usually begins with homegrown scripts created by network engineers with some coding experience. An engineer sees a problem and proactively writes a script to fix it. A pressing problem has been solved, and now the data center is one step closer to becoming automated. Writing a script to check VLAN consistency is one example.

Somewhere else in the data center, another network engineer has spotted another problem and solved it with another script. This process compounds until the data center is running on an assortment of scripts that automate discrete parts of the network infrastructure. After a few months or years, pieces of the network are running on scripts written by five or ten different people. Nobody sees what's happening because nobody has a bird's-eye view, so nobody is aware that their network has entered a precarious state. The entire company is depending on a system that has:

- Ø Limited documentation
- Ø Limited quality assurance
- Ø Limited vulnerability testing
- Ø Limited forecasting capability
- Ø Limited business continuity
- Ø Limited disaster recovery

As long as the scripts are working, there is no need to move toward full automation. Shortages of budget and staff can result in the de-prioritization of automation, causing automation to become "next year's project" indefinitely. But when an outage occurs due to a problem with a rogue script, weeks can pass before the script is discovered to be a point of failure. Then, once a failed script is identified, its code will have to be reverse-engineered if the script wasn't documented. That can take even more weeks, and create a great deal of extra work for network engineers who are already working at their highest capacity.

Although modern data center networks are massively complex and use high availability design with many redundant paths, they can be easily brought down by a missing bit of syntax in an undocumented script. A misplaced semi-colon can cause a catastrophic failure of a data center, sending a lot of revenue down the drain.

A little automation can create a lot of risk

Data center network admins perform numerous and diverse tasks such as configuration changes, IP allocations, performance tuning, leaf or spine device replacements, and the addition or deletion of tenants, security groups, and virtual networks. Most of these tasks are still done manually on a box-by-box and interface-by-interface basis. Network admins have to conduct numerous tedious tests and validations in their heads or via an amalgamation of spreadsheets, scripts, and other resources, in order to support rapidly changing application needs.

These complex challenges are compounded by vendor-specific operating procedures. Due to company mergers and acquisitions over the years, the combined company's network infrastructure often became multi-vendor on a pod by pod basis. Now the networking teams have to cope with vendor-specific operating procedures on top of the already complex box-by-box management.

Companies like this tend to be in the next stage of network automation maturity. They use configuration management tools like Ansible, or Chef, Puppet, and SaltStack. These configuration tools are a better choice than home-brewed shell scripts because they have version control and templates, and are great for pushing common settings to devices, such as the same switch operating system (OS) to multiple switches and routers. These tools can't automate the dynamic interdependencies between network infrastructure elements, including logical IP addresses and routing tables, virtual networks, and physical elements, such as redundant links and transceivers.

If any one of these is automated incorrectly, a cascading effect causes multiple compute and storage nodes to suffer.

Why data centers keep getting harder to manage

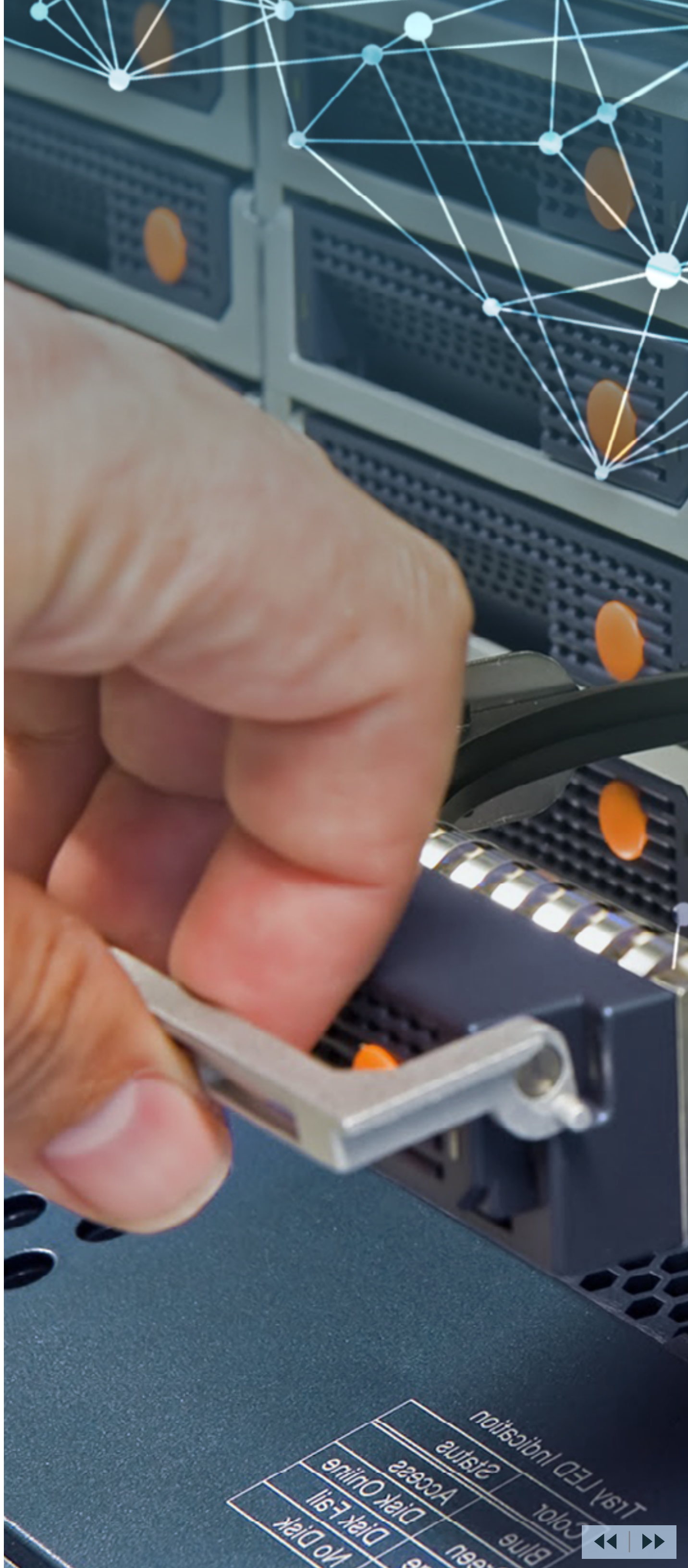
In the past, companies relied on a single vendor to provide their network technology because the use of multiple vendors was just too hard to manage, or was impossible based on how different vendors implement solutions. If a problem arose, who was responsible? The answer wasn't always clear.

Once in place, an incumbent vendor was difficult to remove. The business case to rip-and-replace would have to show a dazzling ROI and OpEx reduction to convince decision-makers that moving to a new provider was a better choice than sticking with the incumbent.

Many companies compromised by trying new vendors on a limited basis as new racks or pods were added. A year later, decision-makers would be acclimated to the new vendor and perhaps the cycle would start again, with yet another vendor debuting in the data center. The data center was becoming multi-vendor on a pod by pod basis.

As the motherboards and switches on the last pod from the original vendor reached their end of life, IT managers suddenly had options: instead of just placing an order with a sole incumbent like they used to do, they could choose a replacement from an array of vendors, all of whom had a history of success in the environment. Now the data center was multi-vendor within each pod.

Yet despite these advancements, over 70 percent of organizations continue to manage their networks manually with command-line interfaces. This approach to handling frequent or complex changes, like modifying virtual networks or segmentation, is simply not sustainable in modern data centers based on hyperscale leaf-spine switching architectures.
The economist Herman Stein said, "If something cannot go on forever, it will stop." The modern data center can't be managed with outdated manual processes or it will stop.



What is intent-based networking and why should you care?

Data center operators need to automate workflows, maximize uptime, and increase operational agility. And they need to do all that while reducing operating costs. Traditional management methods are not up to the task of powering today's business models.

Digital transformation, cloud, mobility, sophisticated applications, and end user demands are driving the transformation from a traditional data center to an Intent-Based Data Center (IBDC). An Intent-Based Data Center incorporates Intent-Based Networking, a distributed system architecture, self-operation, remediation and cloudification to increase application availability and reliability, simplify deployment and operations, and dramatically reduce costs. Data Center IT are faced with the threats, exposure, and vulnerabilities created by shadow IT – and the requirement to reduce operational and capex costs, as well as a desire to establish a cloud-like experience for their teams and their clients.

Data center operators need to automate workflows, maximize uptime, and increase operational agility. And they need to do all that while reducing operating costs. Traditional management methods are not up to the task of powering today's business models.

Intent-based networking (IBN) is the next stage of network management. IBN is software that treats data center networks as a single system composed of precise and dynamically coordinated individual elements. IBN automates and validates best practice leaf-spine topology and its respective underlay and overlay configuration in minutes. IBN allows network engineers and operators to easily declare high-level intent for the system with a few mouse clicks or API calls, such as the number of compute and IP storage endpoints, bandwidth and latency requirements, and ideal placement in the fabric.

When you change your high-level intent, IBN software automatically renders all low-level dependencies. It continuously validates intent in the presence of change—in real time. When you need to assess the health of the network or troubleshoot an issue, IBN software allows you to dynamically query the real-time state of the leaf-spine network across the underlay and overlay. As you change your query to drill down to a specific area of the network, IBN dynamically calculates the most relevant pieces of telemetry to give you actionable insights, along with the context you need.

A STATEMENT OF INTENT WILL LOOK SOMETHING LIKE THIS:

> Provide connectivity to 1000 servers, using Layer2 and/or Layer3 access at the edge, with no oversubscription, with endpoints such as hosts, VMs or containers grouped into isolation domains (including both traffic and address space isolation).

The statement of intent is based on business policies that are already in place to govern the intended state of the network. For instance, in the example above, policies would already be in place to define the specific parameters associated with "traffic isolation."

To ensure the desired state of the network is maintained, IBN software streams data continuously from network devices to feed contextual analytics – and validates the current state of the network continuously against its intended state.

IBN decouples the What from the How

Automation scripts written with NETCONF or YANG still require the network administrator to provide step-by-step instructions on how to achieve a desired state. But networks are dynamic, so scripts that prescribe actions are fragile. If one aspect of the network infrastructure is modified, the automation may crash.

With IBN, a statement of intent tells the network what to do–but not how to do it.
For example, an administrator can tell IBN to allocate this block of IP addresses to these devices and set up BGP neighbor relationships — the what, not the how. IBN figures out how to apply vendor device-specific CLI commands, allocate the IP addresses correctly, properly set up BGP peering and routing tables, and test and validate that BGP routes are flowing as intended.

The how is defined in a blueprint called a reference design. The reference design defines roles and elements. This is a radical departure from the traditional approach, which would define individual devices and functions. IBN doesn't care if a specific box is online and running a specific type of software. IBA only cares that a resource is available to fulfill a certain role.
If a task that only requires a small set of capabilities comes up, the IBN system will direct it to any device that is capable of performing that set of capabilities. A bargain-priced barebones switch is the same as a costly best-of-breed product to the IBN system, if each one can perform the necessary role.

Because individual components don't matter, modifying a device or taking it offline does not impact the state of the network. An IBN solution will find another device to take over the role of the missing machine. IBN allows you to implement reference designs where individual components don't matter.

Analytics turn raw telemetry into insights rapidly

Intent Based Analytics (IBA) is an important feature of Apstra Operating System (AOS) and a main enabler for closed loop telemetry.

In the context of IBA, analytics are defined as the extraction of knowledge out of raw data. Intent-based implies that intent as a single source of truth and context is the main enabler for making this extraction process efficient and powerful.

The extraction process is implemented as a configurable data processing pipeline, where each stage of the pipeline is in constant sync with the intent. Since the intent is constantly changing, keeping the pipeline stages "in sync" is the most important and challenging aspect to implement.

Total network failures may be what the boardroom fears, but network administrators know that gray failures are a more common problem, and are often the first sign that a total failure is about to occur.

Network administrators understand the criticality of gray failures and strive to deal with them proactively, but troubleshooting a gray failure can take days of their time – or even months, if parts of the network are running on undocumented scripts. Network administrators need to specify which raw telemetry to acquire, how to acquire them, and how to extract actionable insights from them. Whenever there's a change to the network, data acquisition and analytics pipelines need to be updated. Businesses can find themselves enmeshed in a never-ending game of whack-a-mole that runs up costs but never roots out the causes of their slowdowns, disconnections, and outages.

IBA eliminates those inefficient and uncertain processes by the use of semantic reasoning. The system runs queries that capture dependencies in a massively complex environment and pinpoints changes that cause problems. But analytics are useful for more than preventing network outages and gray failures; they are useful in any stage of the service lifecycle, whether that stage is design, build, deploy, or validate.

IBA starts with an administrator's intent, then builds and stores all interdependencies to understand how to build a fabric. For example, if an app running on a compute node (virtual or physical) is slow, and it is connected to a database node (physical or virtual) through a network fabric, these two endpoints talk to each other via multiple possible fabric links.

Troubleshooting without IBA

1. Find their MAC addresses and IP addresses
2. Find the switch ports and ethernet interfaces they are connected to
3. Troubleshoot switch by switch, hop by hop, and link by link.

Administrators can do this because they built the fabric and they know the interdependencies in their heads. However, this approach is very time-consuming.

Troubleshooting with IBA

1. Tell IBA to provide all known paths between endpoints, as well as all interface counters, etc.
2. Instantly know where the traffic congestion or imbalance is happening between the two endpoints.

With IBA, administrators can state an intent – what the two endpoints entail, such as their hostnames, MAC, or IP addresses. IBA does the rest.

A DATA CENTER RIDDLE ?

John pulls two cables and mistakenly swaps ports on the same switch. Your network will look like this:

- **The Ethernet interfaces will electronically show as "up" via a NMS**
- **BGP routes will be missing**
- **Some apps may stop responding**
- **Some BGP sessions will be down**
- **And some routing tables on some devices will be incorrect.**

In this example, the interdependencies go from bottom of the networking stack (physical interface) to the routing layer.

John knows how to troubleshoot. However, without IBN, John will have to piece together the app, the server MAC address and IP address, BGP sessions, routing tables, routes, and interface status in his head. John spends a lot of time working through these variables while his other work is stacking up.

STAGE 1

1

Where are you on the network automation journey?

ASSESS YOUR DATA CENTER NETWORK AUTOMATION MATURITY

TECHNICAL

- We automate small parts of our data center with DIY scripts as needed, but we don't have a plan to move toward automation.
- We still perform a lot of processes manually and we don't have a clear idea of the true state of our network at any given moment.
- We have massive amounts of data that we can't make actionable because we lack a single source of truth.
- We have a lot of gray failures but we aren't able to find their root causes to fix them.
- We have trouble attracting qualified network engineers and the ones we have are too busy to learn the coding skills we need to implement configuration management software.

SOCIAL

Our leadership isn't comfortable embarking on a journey to automate.

Cost is an issue, but they are also concerned that new technology might not provide enough of an ROI to make the risk worth it. We are not incorporating digital transformation, IoT, or 5G into our corporate strategy, so the data center is not perceived as critical to revenue generation or competitiveness.

STAGE 2

Where are you on the network automation journey?

ASSESS YOUR DATA CENTER NETWORK AUTOMATION MATURITY

TECHNICAL

- We automate some repetitive tasks in our data center using configuration management software, but we still do a lot manually.
- We still perform a lot of processes manually and we don’t have a clear idea of the true state of our network at any given moment.
- We have massive amounts of data that we can’t make actionable because we lack a single source of truth.
- We have a lot of gray failures but we aren’t able to find their root causes to fix them.
- We have trouble retaining qualified network engineers because of the long hours we still spend troubleshooting

SOCIAL

We understand the value of full network automation, but we don’t have a strategy to get us there.

Our leadership thinks that the configuration management tools we use provide all the automation we need. We’re too busy meeting immediate needs to build out proof-of-concept and we haven’t been able to make a business case for automation that resonates with leadership.

However, our company is engaged in digital business and is incorporating IoT and 5G into our strategic plans, so we know we need to make a move soon. We just aren’t sure where to start.

STAGE 3

Where are you on the network automation journey?

ASSESS YOUR DATA CENTER NETWORK AUTOMATION MATURITY

TECHNICAL

○ Our data centers are mostly automated. When something goes wrong, we know about it, but most problems have already been solved by the time we've finished reading the alert.

○ We do not perform any processes manually, so our engineers focus on strategic issues that support business growth and agility. We always have a clear understanding of the state of our network.

○ We have massive amounts of data that is reliable and actionable because we have a single source of truth.

○ We don't have gray failures because the network repairs them before they materialize.

○ We attract and retain network engineers because we offer predictable hours and the opportunity to innovate. Our networking engineers are encouraged to develop skills in coding, security, and business, so they can continue to grow their careers.

SOCIAL

Our leadership is fully committed to digital business, IoT, and 5G.

We are focused on streamlining and automating many aspects of our operations, so the investment in data center technology is seen as a competitive advantage.

Apstra Operating System (AOS) transforms your data center into a business operating system

The Apstra AOS platform does something no other IBN system can do: it automates the entire lifecycle of data center operations across the network. AOS dramatically simplifies operations by treating data center fabric as a single system with a single source of truth, regardless of vendor mix.

AOS provides full visibility into the network through advanced analytics and continuous validation, as well as auditing and reporting. The result is faster times to network service delivery, the elimination of outages, and the reduction of both CapEx and OpEx.

Roll out a single pod or roll out an entire data center. AOS delivers the benefits of automation at any scale.

CAN YOUR NETWORK KEEP UP WITH YOUR BUSINESS STRATEGY?

If digital transformation, IoT, or 5G are in your future, now is the time to ask questions. We can help you find out if IBN is right for your business.

CALL APSTRA AT 1-844-927-7872 X2 or
EMAIL: SALES@APSTRA.COM

apstra®