AHEAD

Adopting Network Automation: *Practical Advice for Leaders*

A consistent theme among enterprise networks–much like the organizations that rely on them–is that most have a unique personality. Though many of the high-level goals of automation (e.g., more time for value added work, fewer outages, etc.) are similar across enterprises, one organization's strategy to adopt network automation will (and should) naturally differ from another. In this whitepaper, we've provided some 'thinking points' designed to help leaders decipher how to approach their automation challenges. They are by no means 'one size fits all' – there are many ways and varying depths to which an organization may adapt to automate their network infrastructure.

A reliable constant in this space is how much an IT organization's culture, people, and alignment to the business impact the way technical solutions ought to be designed, delivered, and operationalized. More often than not, the technology and technical skills required to master infrastructure as code (IaC) and automation aren't nearly as challenging as the organizational aspects.

Indeed, the term *network automation* brims with idealistic visions of engineers spending time on strategic, value-added projects while mundane tasks become the domain of robots, pipelines, and scripts. But while a technology-first viewpoint has its merits for developing a strategic vision, it typically doesn't itself result in what most organizations really need: the predictable, flexible, and stable networks without which digital businesses could not exist.

The reality is that most organizations encounter many of the same challenges to adopting network automation as others – some intrinsic, some complex, but many of them quite mundane and surmountable. And even then, not all of them need to be solved head-on. Let's take an honest inventory of those challenges, along with some guidance on the best approach to overcoming them in a practical manner:

01

LACK OF DEFINED ARCHITECTURE

Are standards set and not adhered to, or are there no standards at all?

02

UNDERSTANDING OF CURRENT PROCESSES

How are requests fulfilled and services delivered?

03

LACK OF GOAL DEFINITION AND LOOSE TIME TO VALUE

How will this add value, and when?

04

PEOPLE, CULTURE, AND LEADERSHIP

How will you keep it all glued together?

Lack of Defined Architecture

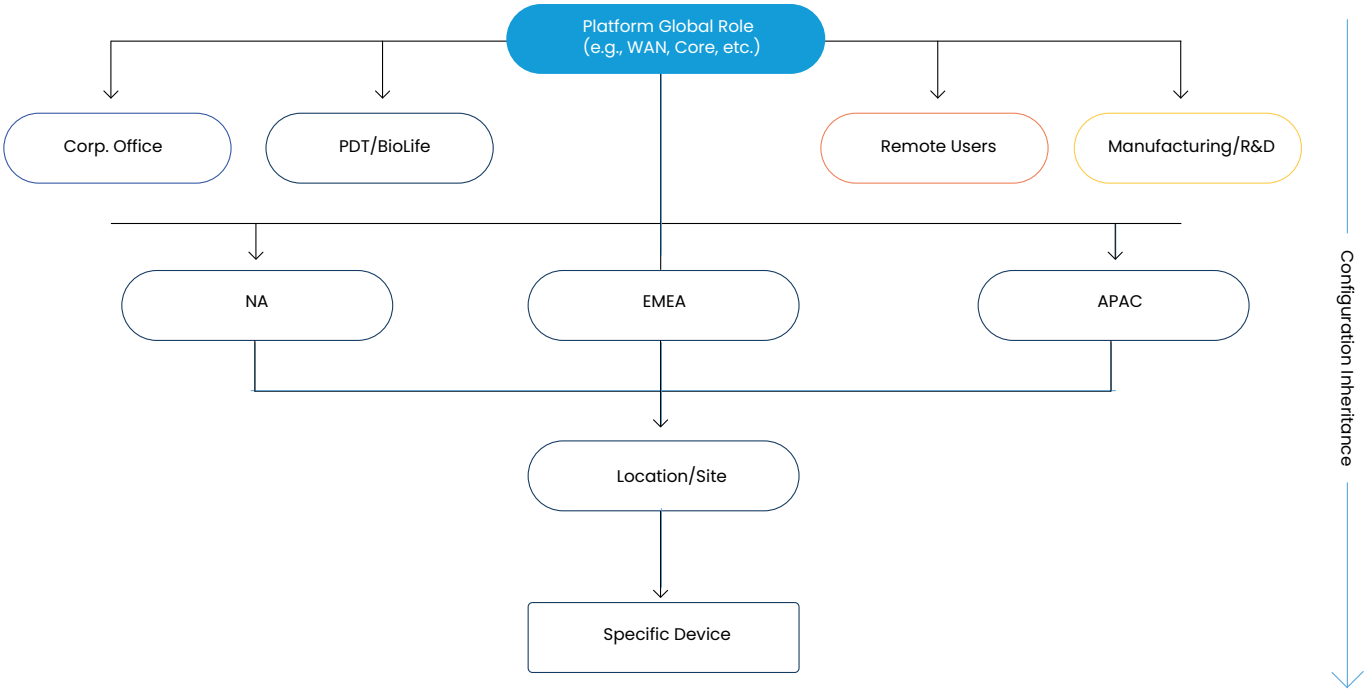
Historically, network automation has been challenging and sparsely adopted because of the nature of enterprise networks themselves. Simply put, they are made up of independent, multi-functional systems working together to form a larger system. With routing protocols, deliberate segmentation, and other complex interdependencies spanning these systems, they can be very difficult and/or cost prohibitive – if not impossible – to model completely in a test lab or accurately simulate in software.

However, predetermined architectural and operational standards can provide confidence that automating the same change or action across multiple systems will produce the intended result each time (i.e., 'idempotence'). Enterprise networks that lack standardization will render themselves unpredictable, and attempts to automate will follow suit.

Standardization, in this context, refers to rigid similarity across the entire enterprise network in terms of makes, models, operating code, and configurations.

If that sounds like a tall task, don't fret. While an ambitious few may be able to achieve that level of standardization, most will find it almost impossible—and for good reasons.

Standards can be developed and applied to critical functions while continuing to allow for flexibility in others. For example, standardizing management access, host-naming, and other relatively 'informational' details is uncontroversial and can provide a low-risk conduit to learn your automation tools and shake out what works and what doesn't. This enables teams to build confidence in their approach before moving on to more consequential standards, such as routing protocol operations across remote office networks of varying sizes.



Of course, standards are much easier to write down than they are to maintain due to the events of the real world that so many of us are familiar with. The short-notice acquisitions, late-night *I just want to get this working again* fixes, and other business pressures that don't afford time for a well-thought-out solution feed a process of entropy that is difficult to stop when such changes are performed through CLIs and "ClickOps."

Conveniently, one answer to maintaining standards over time lies in **automated configuration management**, which refers to the assurance that if a configuration differs from a defined standard, you will be alerted to the discrepancy so that it may be rectified. Using tools and processes to check configurations, operating code, and other on-board details against a 'gold standard' not only helps to keep standards in place, but it's also a great entry-level network automation use case.

LACK OF DEFINED ARCHITECTURE

Key Takeaway

Before diving into network automation, define the network infrastructure's architectural and operational standards and how they will be enforced. Work with your network architect to build enterprise-wide standards, then develop a roadmap to implement those standards in a progressive fashion. Adding automated configuration monitoring and management inline will help not only maintain those standards, but give your team a great place to start applying network automation and infrastructure as code to a real use case.

Understanding of Current Processes

It's more difficult to automate something without a mechanical understanding of how it works. This is as true for automating network infrastructure as it is for automating the processes that govern the network infrastructure. A lack of understanding of how work is processed and fulfilled within the organization will only be amplified when you sit down to try and automate that work.

After selecting a use case for your network automation program, such as automating data center port-level changes, make sure you completely understand the process of how such a change is accomplished *today*.

Organizations are frequently surprised by some of the details that emerge when digging into their internal processes; extraneous steps, over-reliance on single individuals, and ad-hoc communication are all important details to capture. Once known, the process can be re-defined to be more efficient in essence, and then translated into automated workflows, piece by piece.

UNDERSTANDING OF CURRENT PROCESSES

Key Takeaway

Fully map out any processes you're seeking to automate. Not only will you likely learn quite a bit about how your organization functions today, but you'll also find opportunities for optimization beyond the initial automation-related tasks.

Lack of Goal Definition & Unspecified Time to Value

Like any transformative effort, network automation needs clear and measurable goals that communicate how and when it will produce demonstrable value for the organization and business. Specifically, the strategy must clearly define the 'why' of network automation for your organization. In other words, how will an automation program benefit your customers (other IT teams, end users, etc.) in the short, medium, and longer term? Simply saying that "we want to free up time for strategic projects" is a fine enough reason in the abstract, but what does it really mean to your organization and its challenges?

Going a few layers deeper, if your team spends a lot of their time fulfilling firewall rule requests and receives frequent complaints from internal customers about long lead times, then a goal like "implement a new 1-day service level agreement (SLA) for firewall rule requests" communicates that your time and effort aims to improve operations in an appreciable way – not just to scratch a technical itch.

In addition to defining a strategy, construct a roadmap that lays out the steps needed to achieve defined goals and how long it will take to show value for the organization. Mapping out incremental gains in value are key here – a long, drawn out effort to achieve the goals is rarely a recipe for success. Rather, construct your roadmap in such a way that real value is added little by little until your full goal is achieved. A great way to kill momentum and support is spending six months learning network automation tools only to produce a few one-off port/VLAN change scripts.

Taking our firewall rule fulfillment example, a pragmatic (and simplistic) roadmap may look like something like this:

01 *Month One*

Rule requests are still processed manually, but executed consistently using Ansible playbooks with manual inputs.

Reduction of SLA by *x* days.

02 *Month Two*

Rule requests are automatically tested pre-deployment to check for overlap with another existing rule.

Reduction of SLA by *x* days.

03 *Month Three*

Rule requests are automatically tested post-deployment to verify working order, avoiding manual check with application owner.

Reduction of SLA by *x* days.

04 *Month Four*

Firewall rules service catalog item deployed in ServiceNow; inputs used in automation to deploy firewall rules following manual approval.

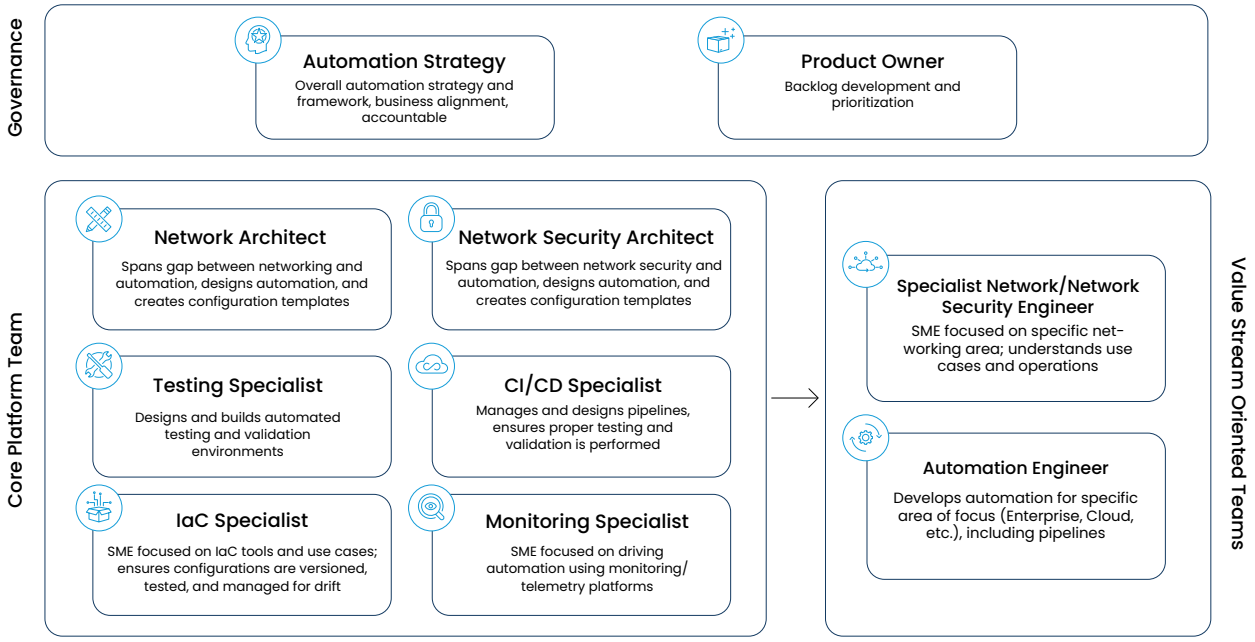
Reduction of SLA by *x* days.

LACK OF GOAL DEFINITION & UNSPECIFIED TIME TO VALUE

Key Takeaway

Roadmap the value that your network automation efforts will bring with real, measurable results that matter to your internal (and perhaps external) customers. Be specific and practical with your goals.

People, Culture & Leadership



Before forming the wider team, assigning roles, and building a skills plan, a strong technical leader must be identified and empowered. Such leaders make all the difference, as they bridge the gaps between the technical teams doing the automation and the business/internal customers that ultimately benefit from it. The following characteristics tend to be good predictors of success in such a role:

- Practical and not idealistic; understands that culture and process transformations take time and can break down the work into realistic, value-added pieces.
- Understands what the business needs and how network automation can help drive value on the front or back end of that business.
- Has established (or is likely to establish) good working relationships with other IT teams, leadership, and business stakeholders – this leader will need to lean on other IT disciplines and ultimately form a cross-functional team for the program to be successful.
- Has architectural skills that are informed by technical realities; architects must be able to inject real, lived experiences into their architectures to produce an actionable plan.

If a strong leader is not already in place for the prospective program, hiring such a person or seeking the assistance of a consulting partner to help get you started will be paramount to success.

The move towards automation and infrastructure as code necessitates a blend of networking and development skills. Programming/scripting skills are not commonly found within most networking teams and thus must be developed organically or sourced externally. As for the wider team, the need to develop talent organically in-house may in fact be an advantage; bringing up multiple resources at the same time often results in more durable teams and a better product, so long as the leadership (both technical and organizational) is solid.

Do not be discouraged – from a technical perspective, the underlying tools utilized in network automation are not as difficult to learn and adopt as they may seem. Learning 'low code' tools like Ansible and Terraform and even interpreted languages like Python are far easier than learning how to develop in C++, especially considering the number of established communities and sources of information freely available today.

Moreover, learning to use a version control system, an orchestration tool like Jenkins, or how to leverage a CMDB's data programmatically is not as difficult as transitioning a complex network from ten different OSPF and EIGRP processes to a unified border gateway protocol (BGP) design while causing minimal disruptions.

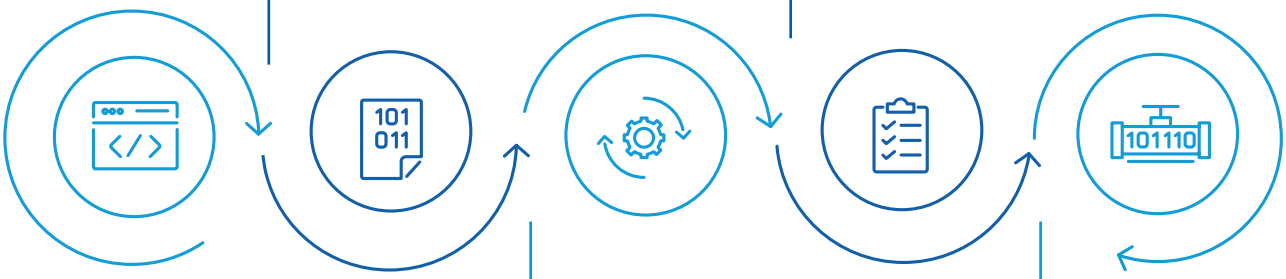
Encouraging teams to participate in trainings, attend conferences, and dedicate time for participation in communities is incredibly valuable and should be prioritized when possible. However, putting learned skills into play with real use cases often serves as the best teacher and should be arranged as soon as feasible through some form of pilot. In other words, after a team feels comfortable using a basic set of tools in a lab (e.g., Python and a VCS), they should begin work on simple and informational use cases like gathering data from the network. This data could then be used to create new and more accurate network diagrams, for example, or executed during certain types of incidents to shortcut some of the initial information gathering tasks that often occur. Leaders may be surprised at how much teams can learn just by attempting simple use cases, not to mention how much breaking through that initial barrier does for a team's confidence.

LEARN VERSION CONTROL/ REPOSITORY

Learn the repository of choice for the Client (GitHub, etc.) and perform initial configuration, create a repository for the required scripts, templates, etc. Commit scripts to the repository and continue using the repository for all subsequent changes.

IDENTIFY PRACTICAL USE CASES

Isolate use cases to networking and read-only operations first; often, this is simply doing something read-only in code that normally would happen manually. This is best done in the cloud first to limit risks.



LEARN TOOL(S)

AHEAD highly recommends HashiCorp Terraform given multi-CSP and potential on-prem use cases, paired with Ansible, which is suited to long-term configuration management. Excellent community support combined with the ability to learn within a test CSP private environment makes upskilling straightforward. Ansible is another common tool and is better suited for configuration management tasks.

LEARN PIPELINE & ORCHESTRATION TOOLS

After learning the scripting/ templating tools and how to manage version control, move on to learning the chosen CI/CD tools (GitLab, etc.) to prepare for integration with other teams' pipelines.

DEVELOP & UTILIZE A SIMPLE AND PRACTICAL PIPELINE

Ultimately, the goal is to integrate specialized experience in overall cross-functional pipelines; start with a harmless activity like building a test environment and validating operations.

PEOPLE, CULTURE & LEADERSHIP

Key Takeaway

A good leader that can walk both the technical and business halls is critical. Building lasting and effective skills will pave the way for tremendous benefit by quickly moving to addressing real use cases, and doing so can follow a gradual progression that minimizes risk to operations.

Parting Thoughts: *Sources of Truth*

As organizations begin their network automation journey, one thing that becomes abundantly clear is the criticality of centralizing and managing the data that forms the intent of the network infrastructure. There are many, many tools and methods to address this vital piece of the puzzle. One such example, Network to Code's Nautobot, frequently serves as the "missing piece" in network automation engagements. The community and support provided around the platform are also exceptional and can make a big difference in your own efforts.

To learn more about how AHEAD helps accelerate the path towards network automation, contact our team today.

Contributing Author:

Ryan Alt, Managing Principal

AHEAD

Combining cloud-native capabilities in software and data engineering with an unparalleled track record of modernizing infrastructure, we're uniquely positioned to help accelerate the promise of digital transformation.

National Hubs

CHICAGO
401 Michigan Ave.
#3400
Chicago, IL 60611

NEW YORK
500 Fifth Avenue
Suite 1500
New York, NY 10110

ATLANTA
1117 Perimeter Center
W406
Atlanta, GA 30338

SAN FRANCISCO
2000 Crow Canyon Place
Suite 250
San Ramon, CA 94583