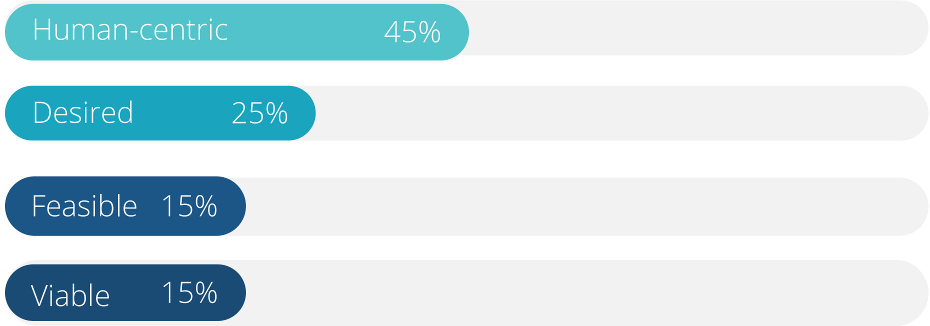
netways

Your Innovation Partner and Catalyst for

Your Intelligent Digital Transformation Journey



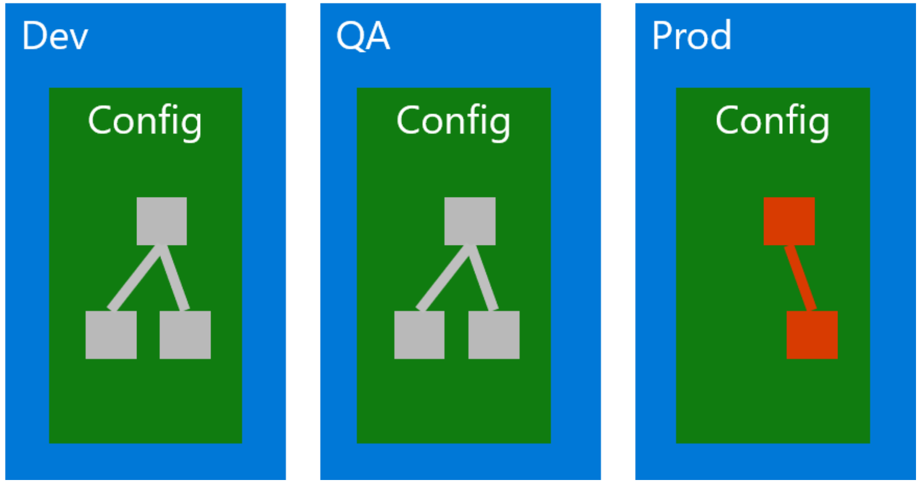
Azure Infrastructure as Code (IaC)

netways

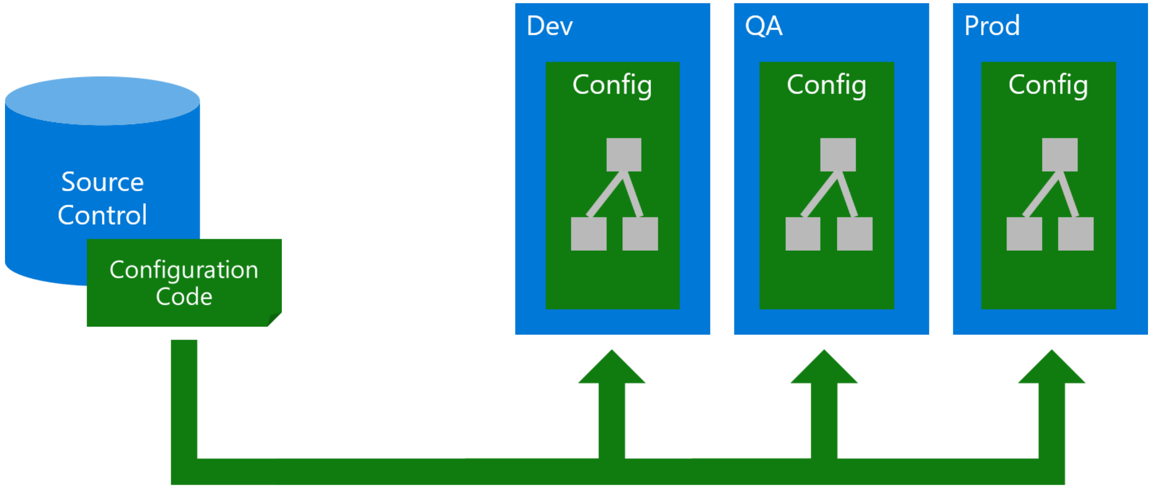
Azure IaC

Introduction

One of the challenges we encounter when transitioning applications to production is the **presence of slight configuration difference** between production environment and the environments used for testing.



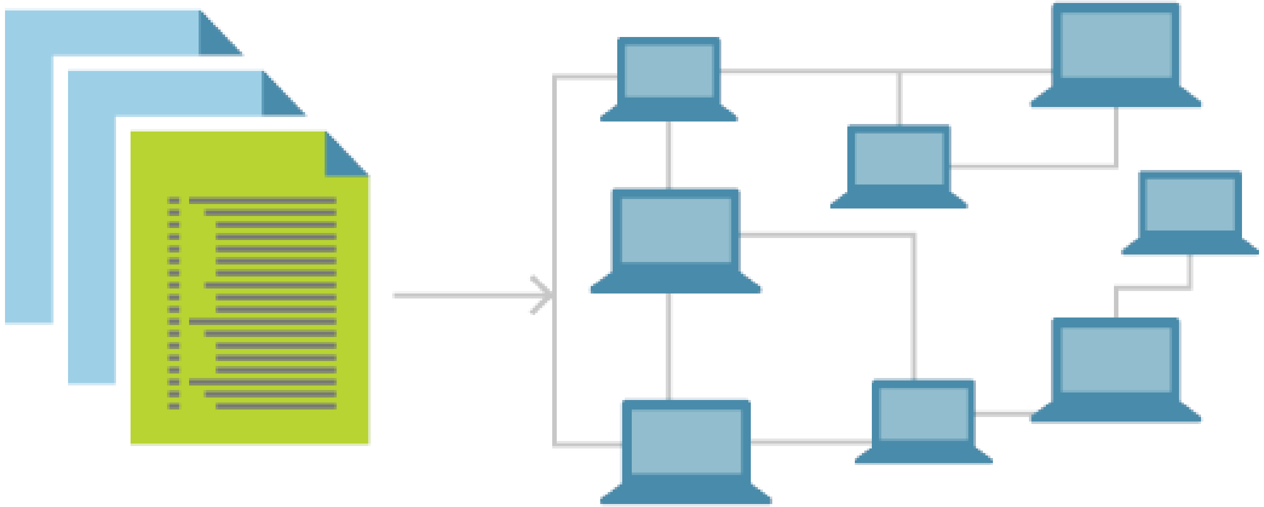
In Azure IaC approach, modifications to an environment are carried out using version-controlled scripts and templates, with minimal human intervention. This ensures that **environments deviate from each other only when essential**, such as when they require distinct database connection strings.



Azure Infrastructure as Code

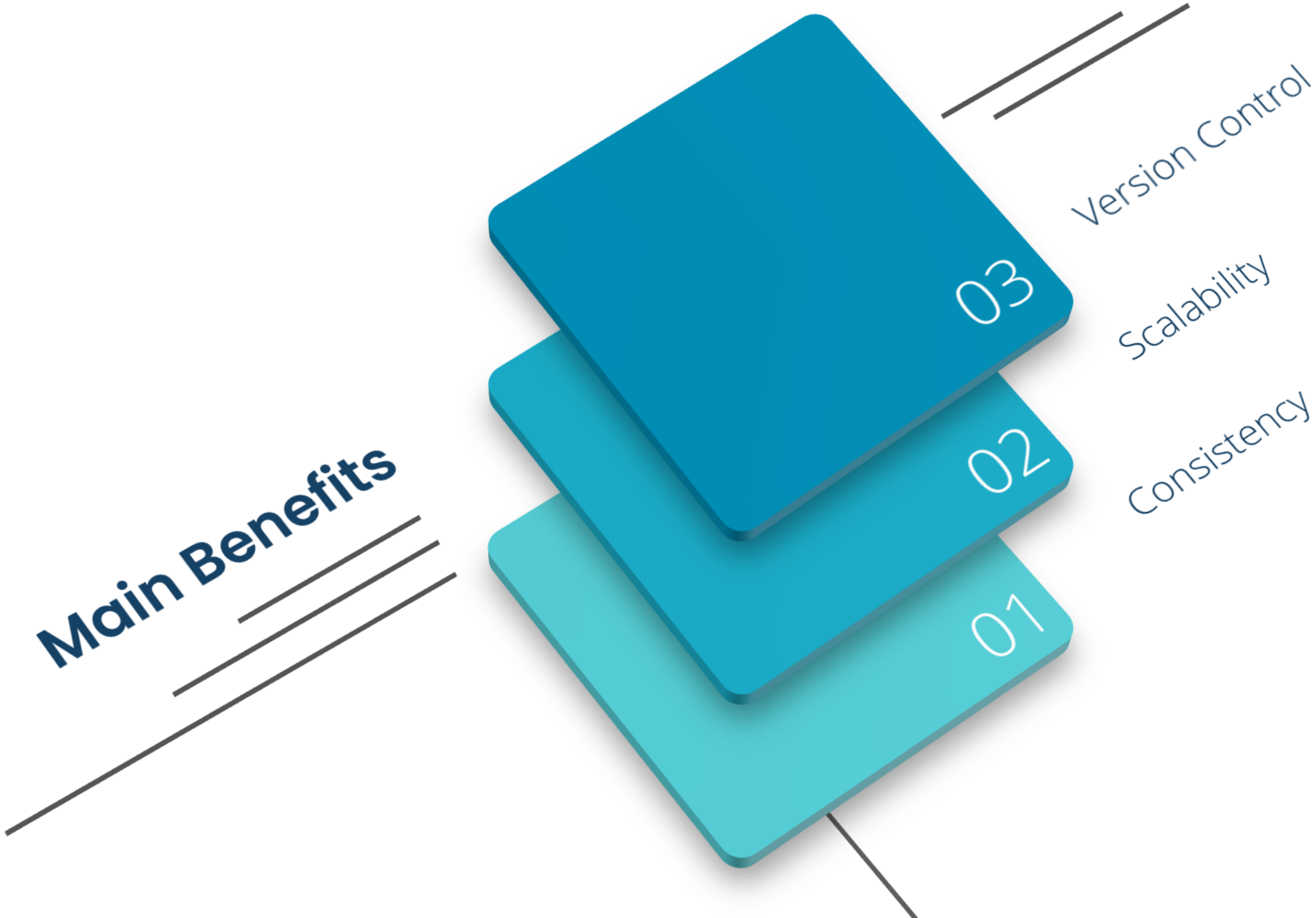
Introduction

Azure Infrastructure as Code (IaC) is a methodology that enables you to manage and provision your cloud resources in Microsoft Azure using code. Infrastructure as code (IaC) employs DevOps practices and versioning to define and deploy infrastructure, ensuring that the same environment is consistently generated every time it's deployed, just as the same source code consistently generates the same binary.



Introduction

Instead of manually configuring resources through the Azure portal or using PowerShell or Azure CLI scripts, IaC allows you to define your infrastructure in a declarative or imperative way using a programming language.



Main Benefits

01 Consistency

02 Scalability

03 Version Control

Azure IaC

Benefits

Consistency

IaC ensures that your infrastructure is always provisioned in a consistent and repeatable manner, reducing the risk of human errors and misconfigurations.

Version Control

IaC code can be stored in version control systems like Git, providing a history of changes and enabling collaboration among team members.

Scalability

IaC allows you to easily scale your infrastructure up or down by modifying the code, making it well-suited for dynamic workloads.

Reusability

Templates and configurations can be reused across multiple environments (development, staging, production), promoting best practices and reducing duplication of effort.

Auditability

IaC provides a clear audit trail of all changes made to your infrastructure, enhancing security and compliance.

Automation

Infrastructure deployment can be automated, enabling continuous integration and continuous deployment (CI/CD) pipelines.

Tools

Azure Resource Manager (ARM) Templates:

ARM templates are JSON files that define the desired state of your Azure resources and their relationships.

Each resource is described using a set of properties, such as name, type, location, and configuration settings.

ARM templates support parameters and variables to make your deployments dynamic and reusable across different environments.

They provide a way to define dependencies between resources, ensuring proper provisioning and sequencing.

ARM templates can be used directly via the Azure portal, Azure PowerShell, Azure CLI, Azure DevOps, or any other supported deployment method.

Terraform:

Terraform is an open-source IaC tool that supports multiple cloud providers, including Azure.

It uses a declarative configuration language (HCL - HashiCorp Configuration Language) to define your infrastructure.

Terraform configurations consist of resources, data sources, variables, and providers.

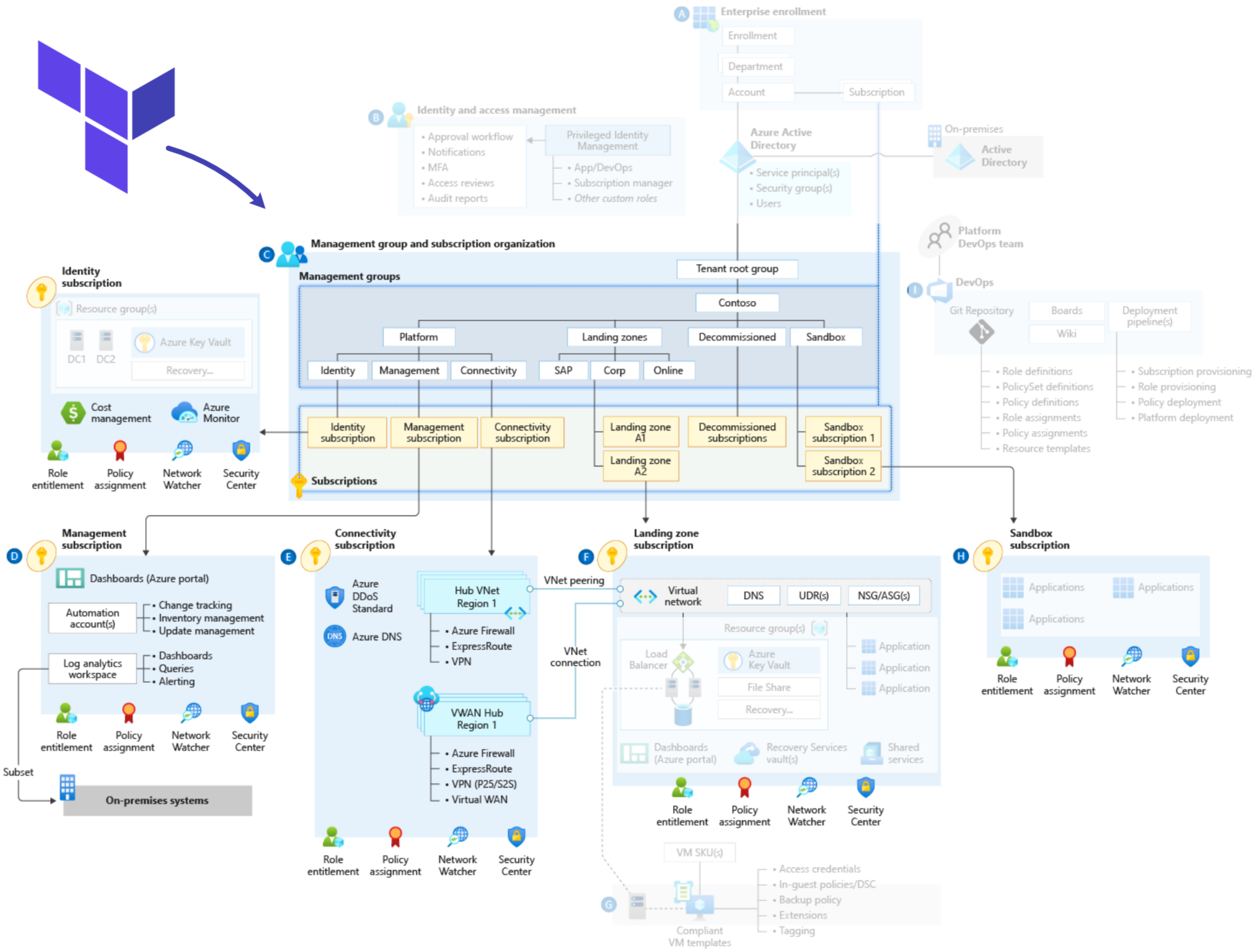
Providers in Terraform allow you to interact with different cloud platforms, and Azure is supported through the "azurerm" provider.

Terraform maintains a state file that keeps track of the resources created, making it easier to manage updates and changes to your infrastructure.

Microsoft Azure Resource Manager

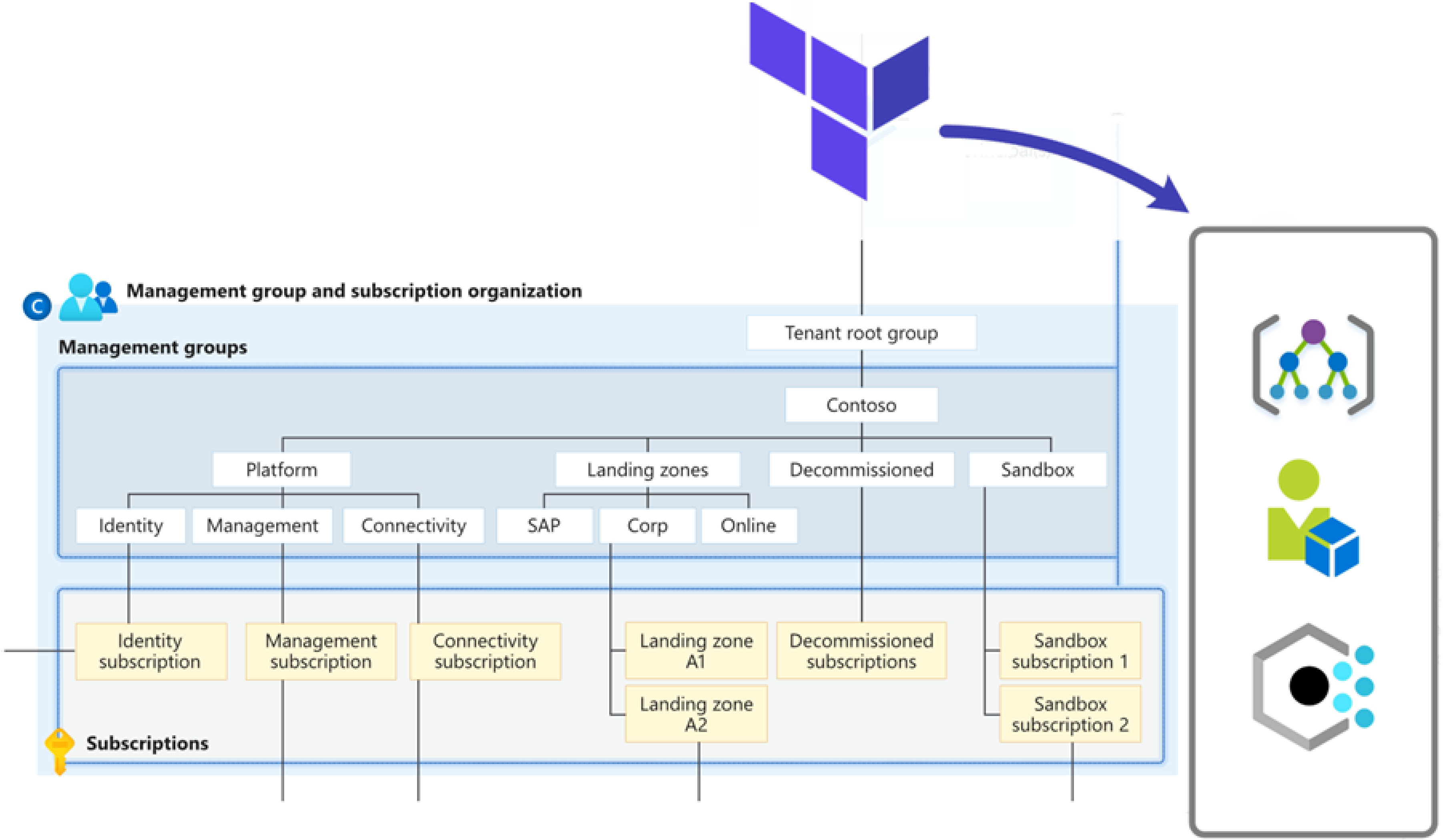
Terraform

GitHub


A
Enterprise enrollment
Enrollment
Department
Account
Subscription
B
Identity and access management
• Approval workflow
• Notifications
• MFA
• Access reviews
• Audit reports
Privileged Identity Management
• App/DevOps
• Subscription manager
• Other custom roles
Azure Active Directory
• Service principal(s)
• Security group(s)
• Users
On-premises
Active Directory
Platform DevOps team
I
DevOps
Git Repository
Boards
Wiki
Deployment pipeline(s)
• Role definitions
• PolicySet definitions
• Policy definitions
• Role assignments
• Policy assignments
• Resource templates
• Subscription provisioning
• Role provisioning
• Policy deployment
• Platform deployment
C
Management group and subscription organization
Management groups
Tenant root group
Contoso
Platform
Landing zones
Decommissioned
Sandbox
Identity
Management
Connectivity
SAP
Corp
Online
Identity subscription
Management subscription
Connectivity subscription
Landing zone A1
Landing zone A2
Decommissioned subscriptions
Sandbox subscription 1
Sandbox subscription 2
Subscriptions
Identity subscription
Resource group(s)
DC1
DC2
Azure Key Vault
Recovery...
Cost management
Azure Monitor
Role entitlement
Policy assignment
Network Watcher
Security Center
D
Management subscription
Dashboards (Azure portal)
Automation account(s)
• Change tracking
• Inventory management
• Update management
Log analytics workspace
• Dashboards
• Queries
• Alerting
Role entitlement
Policy assignment
Network Watcher
Security Center
Subset
On-premises systems
E
Connectivity subscription
Azure DDoS Standard
Azure DNS
Hub VNet Region 1
• Azure Firewall
• ExpressRoute
• VPN
VWAN Hub Region 1
• Azure Firewall
• ExpressRoute
• VPN (P2S/S2S)
• Virtual WAN
Role entitlement
Policy assignment
Network Watcher
Security Center
VNet peering
VNet connection
F
Landing zone subscription
Virtual network
DNS
UDR(s)
NSG/ASG(s)
Resource group(s)
Load Balancer
Azure Key Vault
File Share
Recovery...
Application
Application
Application
Dashboards (Azure portal)
Recovery Services vault(s)
Shared services
Role entitlement
Policy assignment
Network Watcher
Security Center
G
VM SKU(s)
Compliant VM templates
• Access credentials
• In-guest policies/DSC
• Backup policy
• Extensions
• Tagging
H
Sandbox subscription
Applications
Applications
Applications
Role entitlement
Policy assignment
Network Watcher
Security Center

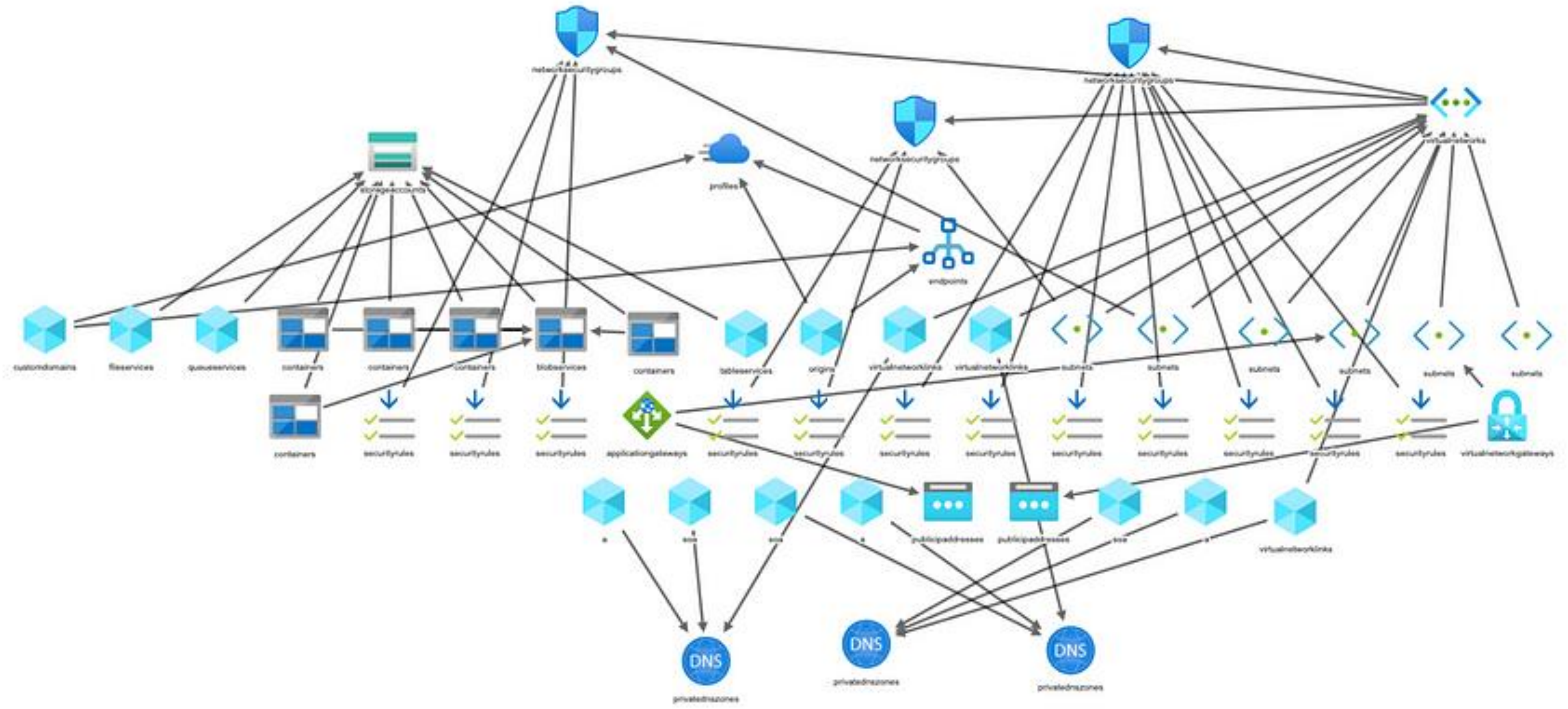
Azure IaC

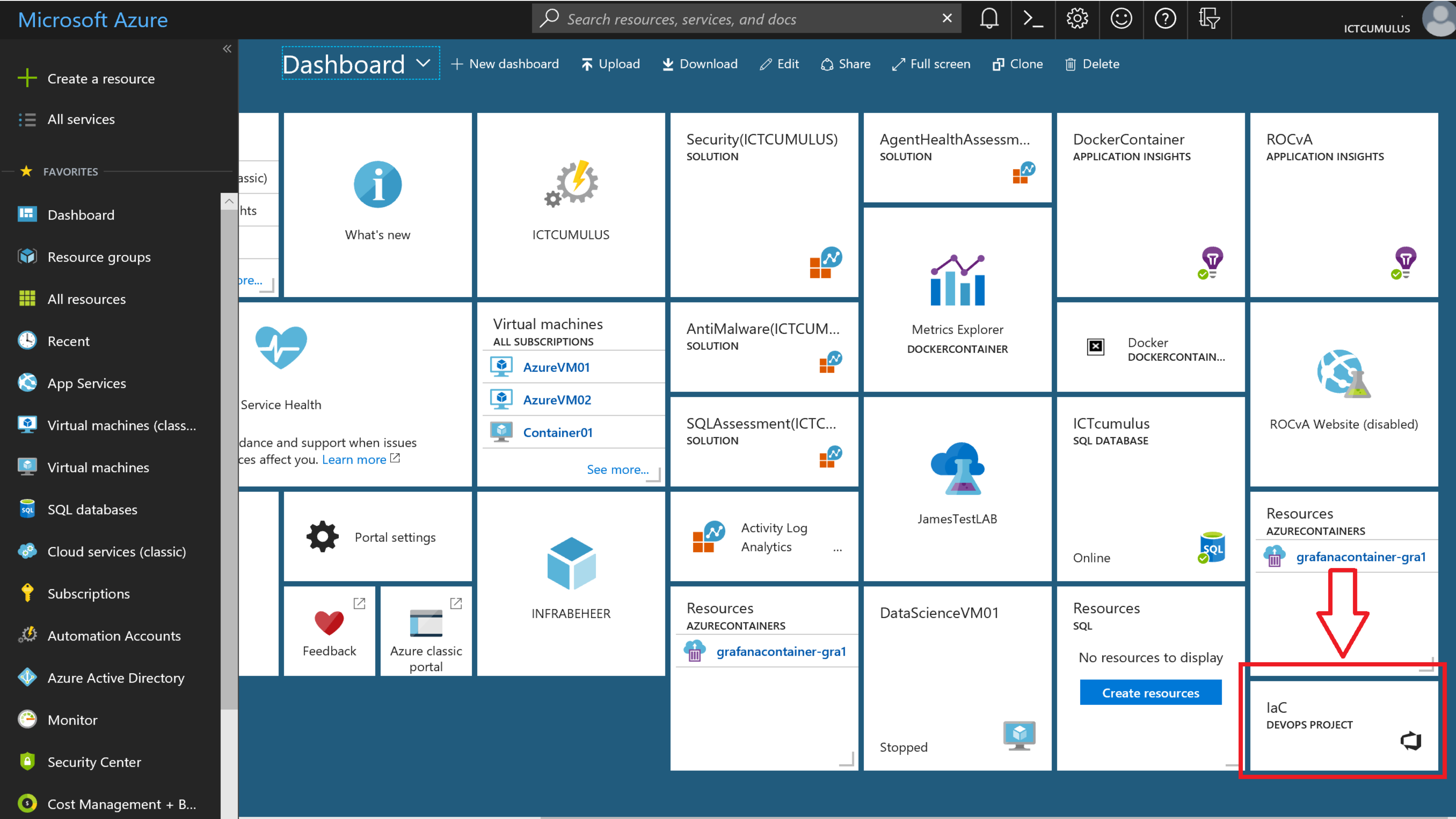
Tools



Azure IaC

Visualisation





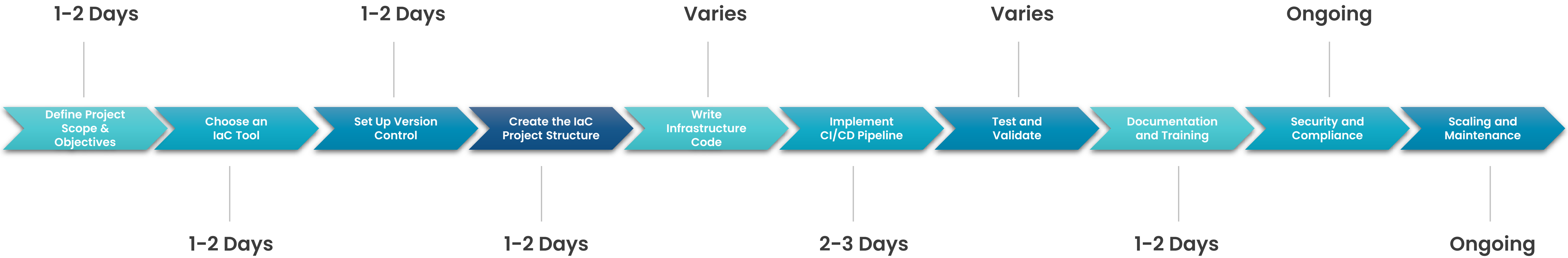
dance and support when issues
ces affect you. Learn more

No resources to display

Deployment Timeline

Setting up an Azure Infrastructure as Code (IaC) project involves several steps, and the duration can vary depending on the complexity of your infrastructure, your familiarity with IaC tools, and the specific requirements of your project.

Here's a general guideline for setting up an IaC project and a rough estimate of the time required:



Define Project Scope & Objectives
1-2 Days

Choose an IaC Tool
1-2 Days

Set Up Version Control
1-2 Days

Create the IaC Project Structure
1-2 Days

Write Infrastructure Code
Varies

Implement CI/CD Pipeline
2-3 Days

Test and Validate
Varies

Documentation and Training
1-2 Days

Security and Compliance
Ongoing

Scaling and Maintenance
Ongoing

Methodology

Netways Cloud Migration

Assessment and Planning:

1.Understand the existing IT landscape, applications, and dependencies.
2. Identify the business objectives and requirements for the migration.
3. Assess the current infrastructure and workloads to determine their suitability for migration to Azure.
4. Categorize applications based on their complexity, criticality, and interdependencies.

Cloud Readiness Assessment:

1. Evaluate the readiness of the existing applications and workloads for the cloud.
2. Identify any modifications or refactoring needed to make applications cloud-ready.
3. Consider regulatory and compliance requirements.

Design and Architecture:

1. Create a target architecture for Azure deployment.
2. Define the appropriate Azure services to be used for each workload.
3. Address security and compliance concerns in the architecture.

Data Migration Strategy:

1. Plan the migration of databases and data to Azure.
2. Choose the appropriate data migration tools and methods.
3. Consider data security and privacy during the migration.

Pilot Migration:

1. Select a representative workload or application for a pilot migration.
2. Execute the pilot migration to identify potential issues and learn from the process.
3. Use the pilot migration as a basis for adjusting the migration strategy if needed.

Testing and Validation:

1. Perform testing to ensure that applications function correctly in the Azure environment.
2. Validate the performance and scalability of the migrated workloads.
3. Conduct security and compliance checks.

Full-Scale Migration:

1. Execute the migration of all identified workloads and applications to Azure.
2. Monitor and manage the migration process to address any issues that arise.

Optimization and Cost Management:

1. Continuously optimize the Azure environment for performance and cost efficiency.
2. Implement governance policies to control costs and resource usage.

Post-Migration Support:

1. Provide support to end-users and address any post-migration issues.
2. Train the IT team on managing and operating the Azure environment effectively.

Thank you

Netways Presales Team

sales@netways.com

www.netways.com

in www.linkedin.com/company/netways