Microsoft

Azure Verified Modules (AVM)

Community Call, 3rd December 2025

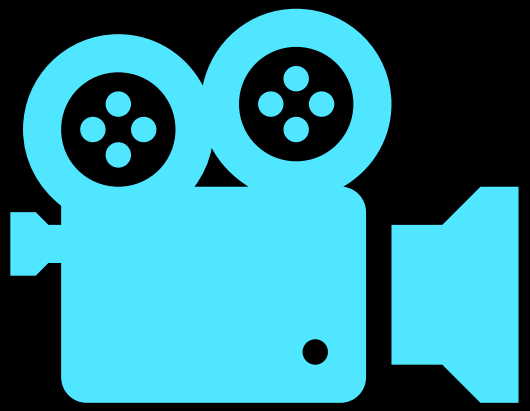
Speakers:

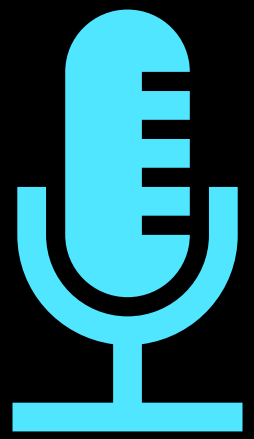
Microsoft: Charlie Grabiaud, Jack Tracey, Sebastian Gräf, Stephanie Yen, Steven Ma, Matt White, Jared Holgate, Erika Gressi, Alexander Sehr.

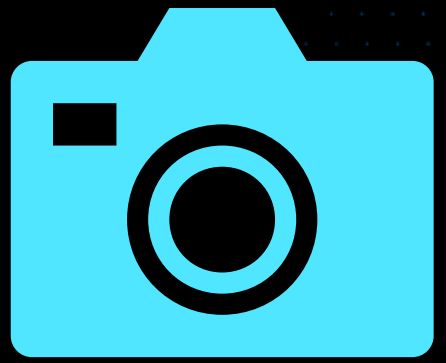
External: Tuna Cinsoy (Nykredit)

When you join this event, your name, email address and/or phone number may be viewable by other session participants in the attendee list. By joining, you're agreeing to this experience.

Also, this event will be recorded and shared publicly with others, including Microsoft's global customers, partners, employees, and service providers. The recording may include your name and any questions you submit to Q&A.







This meeting is being recorded

Meet the AVM Core Team

Technical SME's


Alex


Chris


Erika


Jack


Jared


Matt


Rainer


René


Sebastian

PM's


Charlie


Máté


Amit

Agenda

- AVM adoption at Nykredit (*Jack hosting Tuna Cinsoy*)
- Inner-sourcing (*Jack*)
- Copilot experiences for Infra-as-Code (*Sebastian* + *Stephanie*, *Steven*)
- Going v1 (*Jack*)
- AVM Terraform updates (*Matt*, *Jared*)
- AVM Bicep updates (*Erika*, *Jack*)
- Q & A

Nykredit Verified Modules

An Overview of Cloud Solutions IaC Initiative

Cloud Solutions
Azure Infrastructure

Agenda

01 What?

02 Why?

03 How?

04 Demo

01

Chapter 1: What?

The Definition

Nykredit Verified Modules is an initiative that abstracts away the intricacies of infrastructure deployment by providing `ready-to-use` IaC templates that are fully aligned with Nykredit’s Policies.

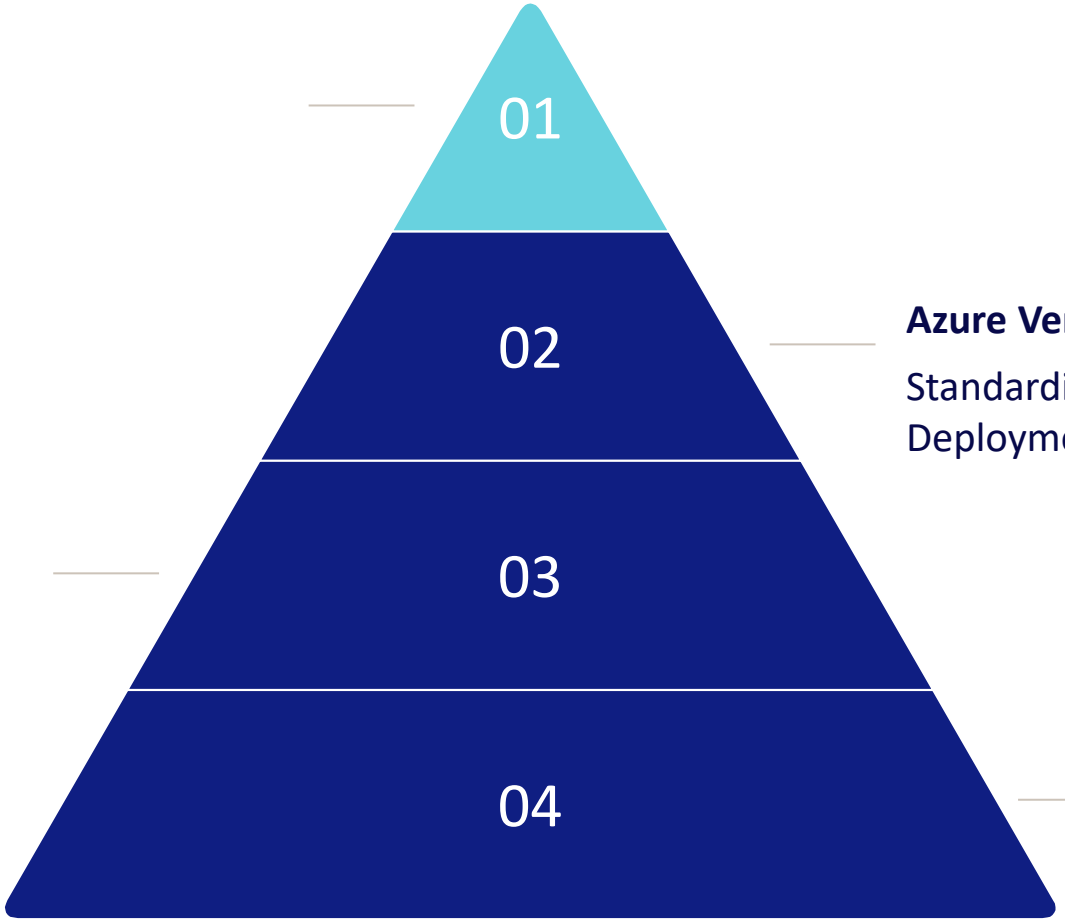
It offers:

- Off-the-shelf resource deployments
- Comprehensive documentation
- Maintenance by Cloud Solutions Team

Hierarchy of Solutions

Nykredit Verified Modules

Full Compliance with Nykredit's Policies



Azure Verified Modules

Standardized Way of Resource Deployment

Domain Specific Languages

Bicep, Terraform etc.

Azure Resource Manager

Central Control Plane for Resource Deployment

Role of Cloud Solutions

Value Proposal


```
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentParameters.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "parSubscriptionId": {
            "value": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
        },
        "parResourceGroups": {
            "value": [
                {
                    "parSolutionName": "premiumFuncApp",
                    "parEnvironment": "prod",
                    "parLocation": "germanywestcentral",
                    "parInstance": 1
                },
                {
                    "parSolutionName": "flexFuncApp",
                    "parEnvironment": "prod",
                    "parLocation": "germanywestcentral",
                    "parInstance": 1
                }
            ]
        },
        "parSubnets": {
            "value": [
                {
                    "parResourceGroupName": "nyk-dev-gwc-shared-rg",
                    "parSolutionName": "sa-inbound-conn",
                    "parEnvironment": "prod",
                    "parLocation": "germanywestcentral",
                    "parVirtualNetworkName": "nyk-dev-gwc-tunan-vnet-pw7wcl",
                    "parNetworkSecurityGroupResourceName": "nyk-dev-gwc-tunan-nsg-pw7wcl",
                    "parRouteTableResourceName": "nyk-dev-gwc-tunan-rt-pw7wcl",
                    "parAddressPrefix": "172.23.152.96/27",
                    "parInstance": 1
                },
                {
                    "parResourceGroupName": "nyk-dev-gwc-shared-rg",
                    "parSolutionName": "sa-inbound-conn",
                    "parEnvironment": "prod",
                    "parLocation": "germanywestcentral",
                    "parVirtualNetworkName": "nyk-dev-gwc-tunan-vnet-pw7wcl",
                    "parNetworkSecurityGroupResourceName": "nyk-dev-gwc-tunan-nsg-pw7wcl",
                    "parRouteTableResourceName": "nyk-dev-gwc-tunan-rt-pw7wcl",
                    "parAddressPrefix": "172.23.152.128/27",
                    "parInstance": 2
                },
                {
                    "parResourceGroupName": "nyk-dev-gwc-shared-rg",
                    "parSolutionName": "func-inb-conn",
                    "parEnvironment": "prod",
                    "parLocation": "germanywestcentral",
                    "parVirtualNetworkName": "nyk-dev-gwc-tunan-vnet-pw7wcl",
                    "parNetworkSecurityGroupResourceName": "nyk-dev-gwc-tunan-nsg-pw7wcl",
```


Single Source of Truth

Repeatable & Consistent Deployment

Simplicity

One Parameter File, Whole Subscription Coverage

Comprehensive Documentation

Ready-to-Use Modules

100% Policy Compliance

Deployments In Alignment with Policy Restrictions

02

Chapter 2: Why?

Known Problems

Duplication of Effort

Parallel IaC development without cross-team coordination

Inconsistent Naming Convention

Lacking a unified standard across application landing zones

Support Bottleneck

Overwhelming support queues with preventable policy errors

High Maintenance Overhead

Teams burdened with manually updating decentralized codebases

Chain Flow – Butterfly Effect

Cloud Solutions take care of the intricacies of infrastructure deployment

Enhanced utilization

Wave 01

Ready-to-Use IaC Templates

Wave 02

Separation of Concerns

Wave 03

Boosted Solution Deployment

Developers focus **solely** on their applications

03

Chapter 3: How?

Journey to Well-Architected Application Landing Zone

Application teams evaluate their readiness for cloud

Application teams create their own IaC repository

Create issues for any type of feature requests and bug fixes

Evaluation
Validating Technical Fit

Onboarding
Network Foundation & SPN

Bootstrap
Setting Up Infra Version Control

Usage of NVM
Fill-in necessary params

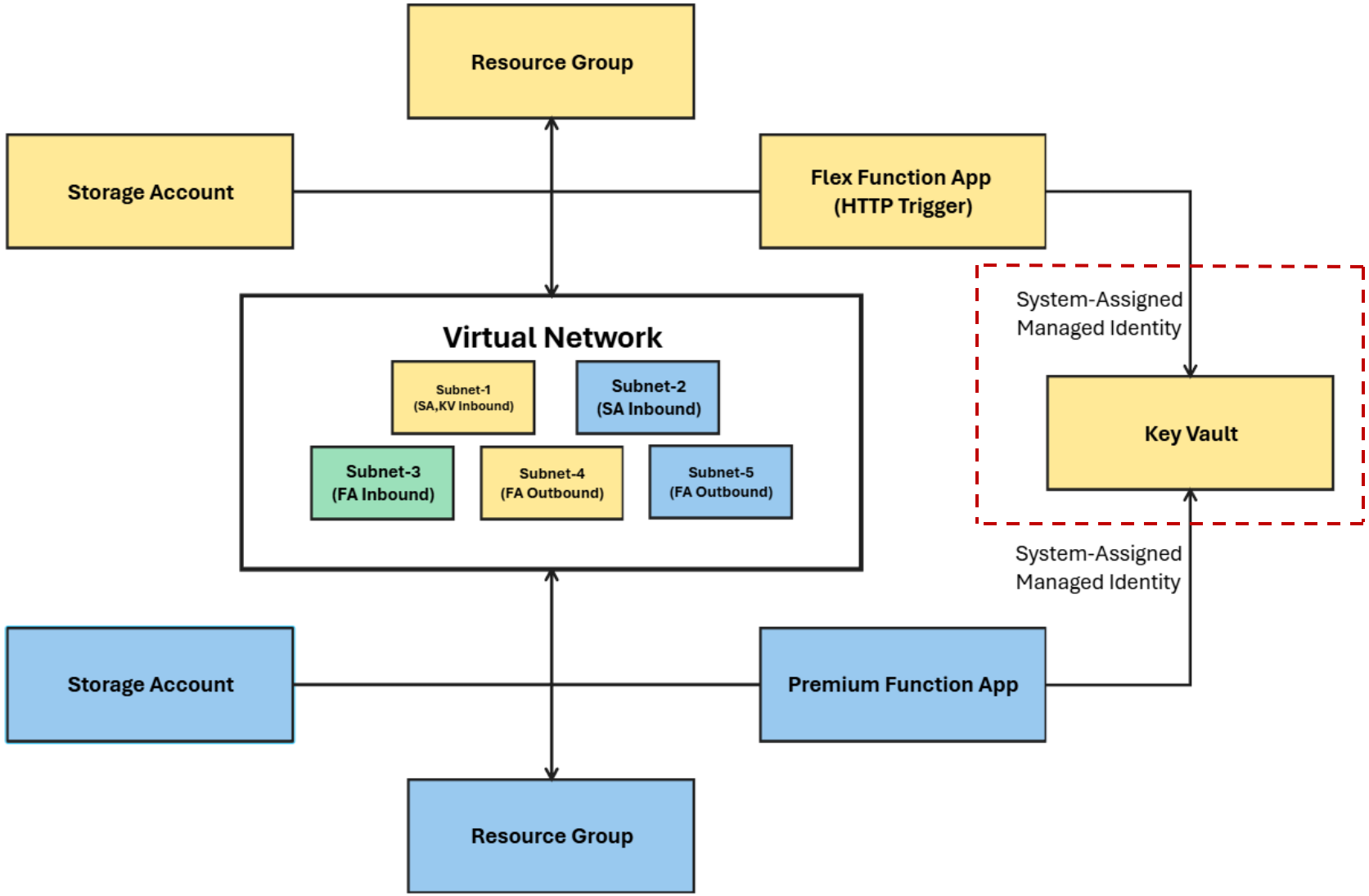
Inner-Sourcing
Facilitating Feedback

Cloud Solutions handle the application landing zone provisioning

No need for expertise – use with ready-to-go templates

04

Chapter 4: Demo


Resource Group
Storage Account
Flex Function App
(HTTP Trigger)
System-Assigned
Managed Identity
Key Vault
Virtual Network
Subnet-1
(SA,KV Inbound)
Subnet-2
(SA Inbound)
Subnet-3
(FA Inbound)
Subnet-4
(FA Outbound)
Subnet-5
(FA Outbound)
System-Assigned
Managed Identity
Storage Account
Premium Function App
Resource Group

Nykredit Verified Modules: Key Vault

This Bicep file is designed to create a key vault in a specified Azure subscription. It leverages the `key-vault` module from the Azure Verified Modules (AVM) library, and it is fully aligned with Nykredit's Policy Restrictions.

Prerequisites

- Ensure that the user or service principal that will execute the deployment command `az deployment group ...` has `AcrPull` role for the `nykverifiedmodules` ACR.
- Ensure that the user or service principal that will deploy the resources has necessary permissions for the target scope.

Version History

Version	Changes	Date
`0.5.4`	Enabled role assignments	2025-11-20
`0.5.3`	Replacing ID parameter with dynamic assignment	2025-10-13
`0.5.2`	Removed looping	2025-07-30
`0.5.1`	Naming module pointing to ACR	2025-07-30
`0.5.0`	Unique string generation for name global name uniqueness	2025-07-18
`0.4.1`	TargetScope set to RG	2025-07-08
`0.4.0`	Correcting target scope, user sets az context	2025-07-08
`0.3.0`	Multiple KV creation, naming convention	2025-07-08
`0.2.0`	Private Endpoint Integration	2025-07-03
`0.1.0`	Initial implementation using AVM module.	2025-07-01

Deployment Steps

1. **Prepare the Bicep File**

   ▼ Example Reference to the Module

```
param parSubscriptionId string
param parKeyVaults array

module modKeyVault 'br:nykverifiedmodules.azurecr.io/key-vault:v0.5.4' = [
for keyVault in parKeyVaults: {
    scope: resourceGroup(parSubscriptionId, keyVault.parResourceGroupName)
    params: {
    parSubscriptionId: parSubscriptionId
    parEnvironment: keyVault.parEnvironment
    parInstance: keyVault.parInstance
    parLocation: keyVault.parLocation
    parResourceGroupName: keyVault.parResourceGroupName
    parSku: keyVault.parSku
    parSolutionName: keyVault.parSolutionName
    parUniqueStringGenerator: keyVault.parUniqueStringGenerator
    parVnetName: keyVault.parVnetName
    parSubnetName: keyVault.parSubnetName
    parSubnetResourceGroupName: keyVault.parSubnetResourceGroupName
    parRoleAssignments: keyVault.parRoleAssignments
    }
}
]
```

2. **Prepare Parameters File**

   ▶ Example Parameters of the Module

3. **Deploy the Bicep file using the Azure CLI**

   ▶ Example Deployment Commands

```
param parKeyVaults array

import { roleAssignmentType } from 'br/public:avm/utl/types/avm-common-types:0.4.0'

var varRoleAssignments roleAssignmentType[] = [
  {
    principalId: functionAppModule[0].outputs.outSystemAssignedIdentityPrincipalId
    roleDefinitionIdOrName: 'Key Vault Secrets Officer'
  }
]

module modKeyVault 'br:nykverifiedmodules.azurecr.io/key-vault:v0.5.4' = [
  for keyVault in parKeyVaults: {
    scope: resourceGroup(parSubscriptionId, keyVault.parResourceGroupName)
    params: {
      parSubscriptionId: parSubscriptionId
      parEnvironment: keyVault.parEnvironment
      parInstance: keyVault.parInstance
      parLocation: keyVault.parLocation
      parResourceGroupName: keyVault.parResourceGroupName
      parSku: keyVault.parSku
      parSolutionName: keyVault.parSolutionName
      parUniqueStringGenerator: keyVault.parUniqueStringGenerator
      parVnetName: keyVault.parVnetName
      parSubnetName: modSubnet[0].outputs.outName
      parSubnetResourceGroupName: keyVault.parSubnetResourceGroupName
      parRoleAssignments: varRoleAssignments
    }
  }
]
```

```
"parKeyVaults": {
    "value": [
        {
            "parResourceGroupName": "rg-flexFuncApp-prod-gwc-01",
            "parSolutionName": "msft-demo",
            "parEnvironment": "prod",
            "parLocation": "germanywestcentral",
            "parSku": "premium",
            "parInstance": 1,
            "parUniqueStringGenerator": "MicrosoftExternalCommCallDemoFlexFA",
            "parVnetName": "nyk-dev-gwc-tunan-vnet-pw7wcl",
            "parSubnetResourceGroupName": "nyk-dev-gwc-shared-rg"
        },
        {
            "parResourceGroupName": "rg-premFuncApp-prod-gwc-01",
            "parSolutionName": "msft-demo",
            "parEnvironment": "prod",
            "parLocation": "germanywestcentral",
            "parSku": "premium",
            "parInstance": 1,
            "parUniqueStringGenerator": "MicrosoftExternalCommCallDemoPremFA",
            "parVnetName": "nyk-dev-gwc-tunan-vnet-pw7wcl",
            "parSubnetResourceGroupName": "nyk-dev-gwc-shared-rg"
        }

    ]
}
```

kv-premi-mu2-prod-gwc-01 | Access control (IAM)

Key vault

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Access policies
- Resource visualizer
- Events
- Objects
  - Keys
  - Secrets

Add | Download role assignments | Edit columns | Refresh | Delete | Feedback

Check access | **Role assignments** | Roles | Deny assignments | Classic administrators

Number of role assignments for this subscription

22 4000

fa-flex-ieg-dev-gwc-01

Type : **All** | Role : **All** | Scope : **All scopes** | Group by : **Role**

All (1) | Job function roles (1) | Privileged administrator roles (0)

Name	Type	Role	Scope	Condition
Key Vault Secrets Officer (1)				
fa-flex-ieg-dev-gwc-01	Managed identity	Key Vault Secrets Officer	This resource	None

tak

Microsoft

Inner-sourcing AVM

Jack Tracey

Common asks

How can we make AVM modules (Bicep/Terraform) available privately within our organization?

How can we customize AVM modules for our organization's security and requirements?

How can we build and publish our own AVM-aligned modules privately?

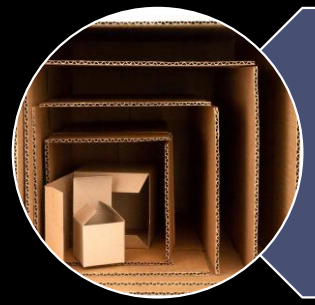
The answer...

Inner-sourcing*

*Beware it can become complex

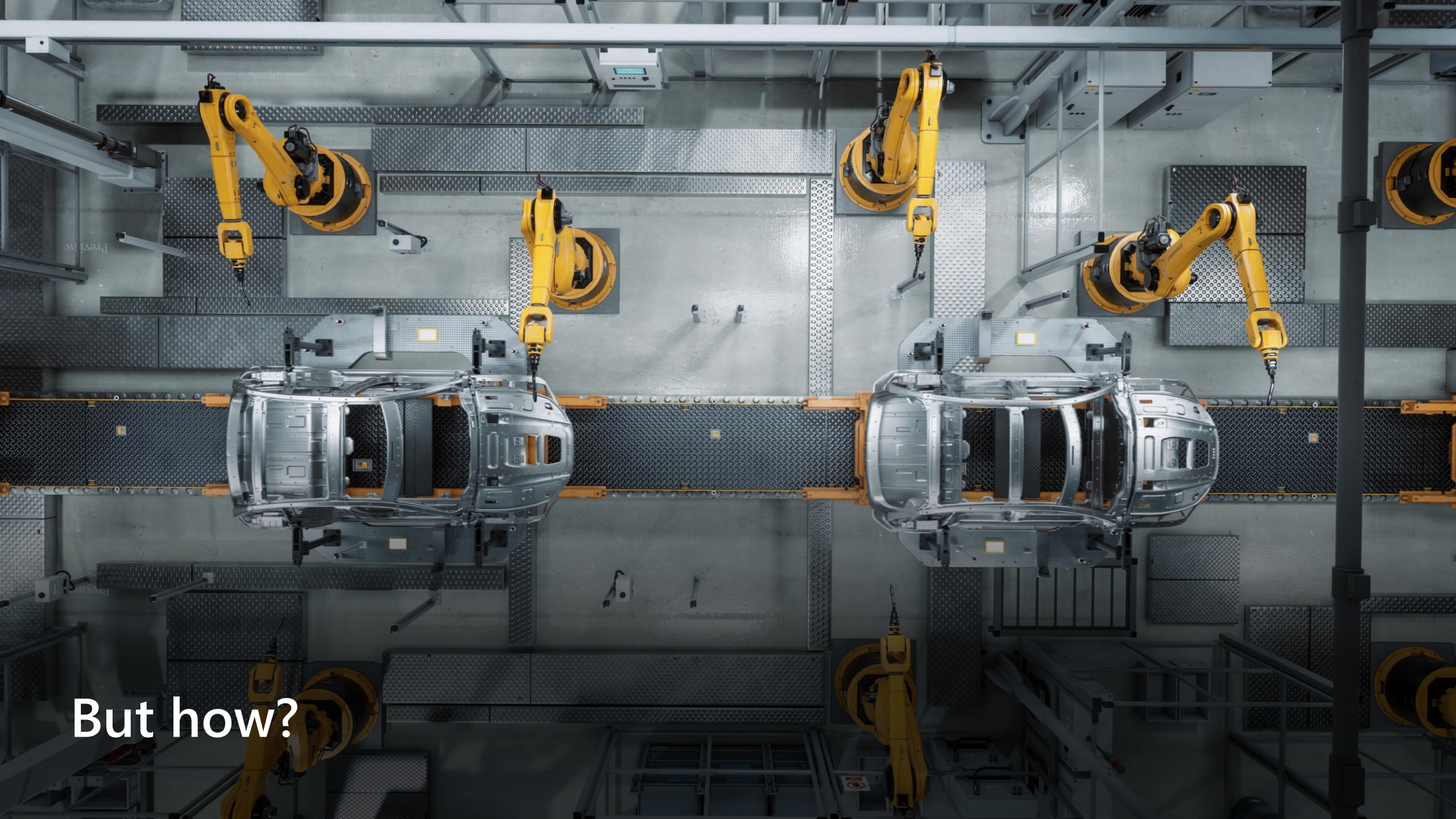
But first, some principles ⚖️

AVMs **MUST** remain as-is and unaltered when inner-sourced



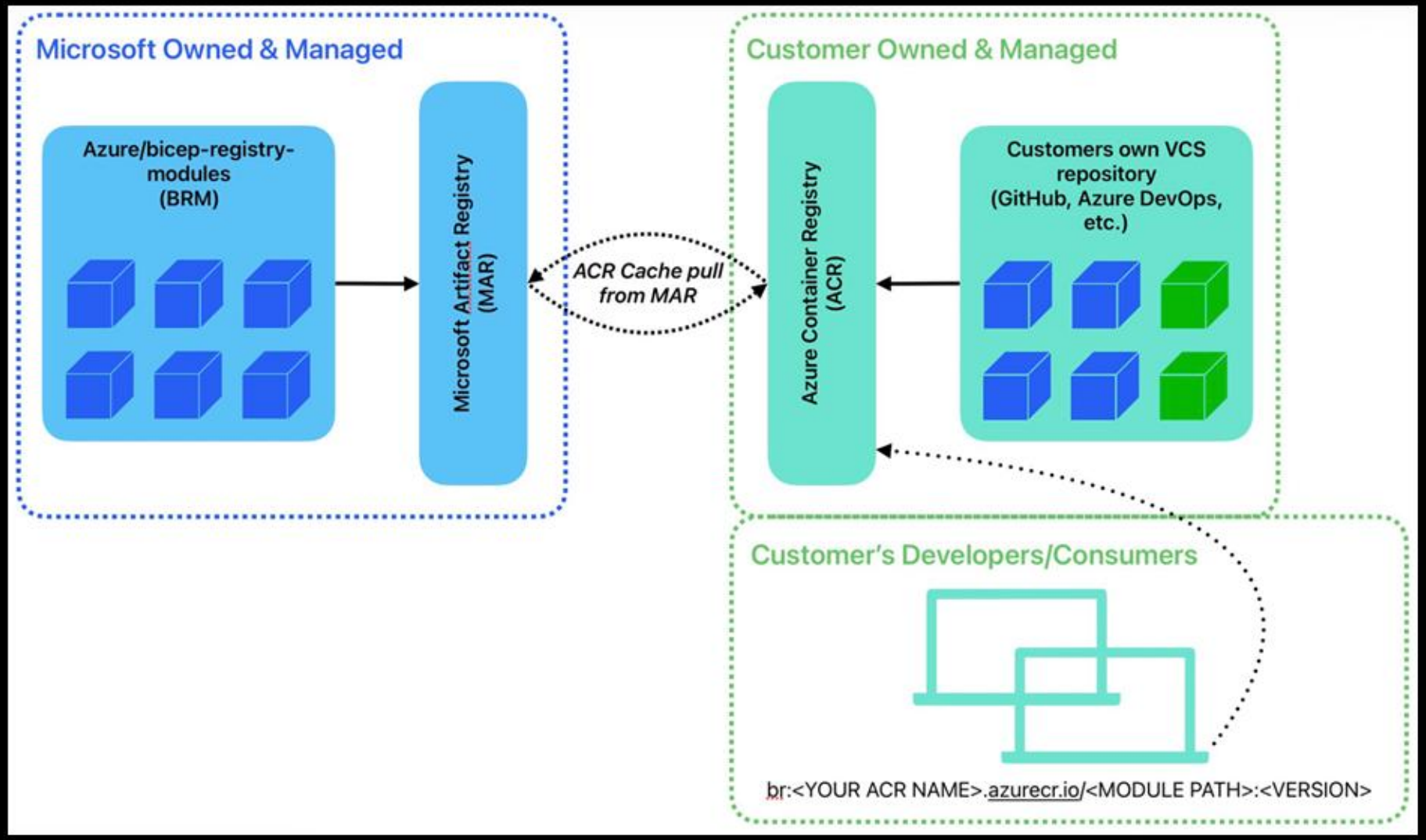
To customize AVMs you **SHOULD**, wrap the original modules in your own, limit what's exposed, and set organization-specific defaults and publish these

Your organizations modules **SHOULD** use the AVM classifications and comply with the respective specifications



But how?

Bicep | Recommended Architecture Overview



File Edit Selection View Go Run Terminal Help

bicep-avm-innersource-demo

demo.bicep 1 | bicepconfig.json

bicepconfig.json > ...

```
1  {
2    "moduleAliases": {
3      "br": {
4        "public": {
5          "registry": "bcpavminnersourcedemo-aahydgfpgce4ebcj.azurecr.io",
6          "modulePath": "."
7        }
8      }
9    }
10 }
11
```

EXPLORER

OPEN EDITORS 1 unsaved

- demo.bicep 1
- bicepconfig.json

BICEP-AVM INNERSOURCE DEMO

- bicepconfig.json
- demo.bicep 1

OUTLINE

TIMELINE

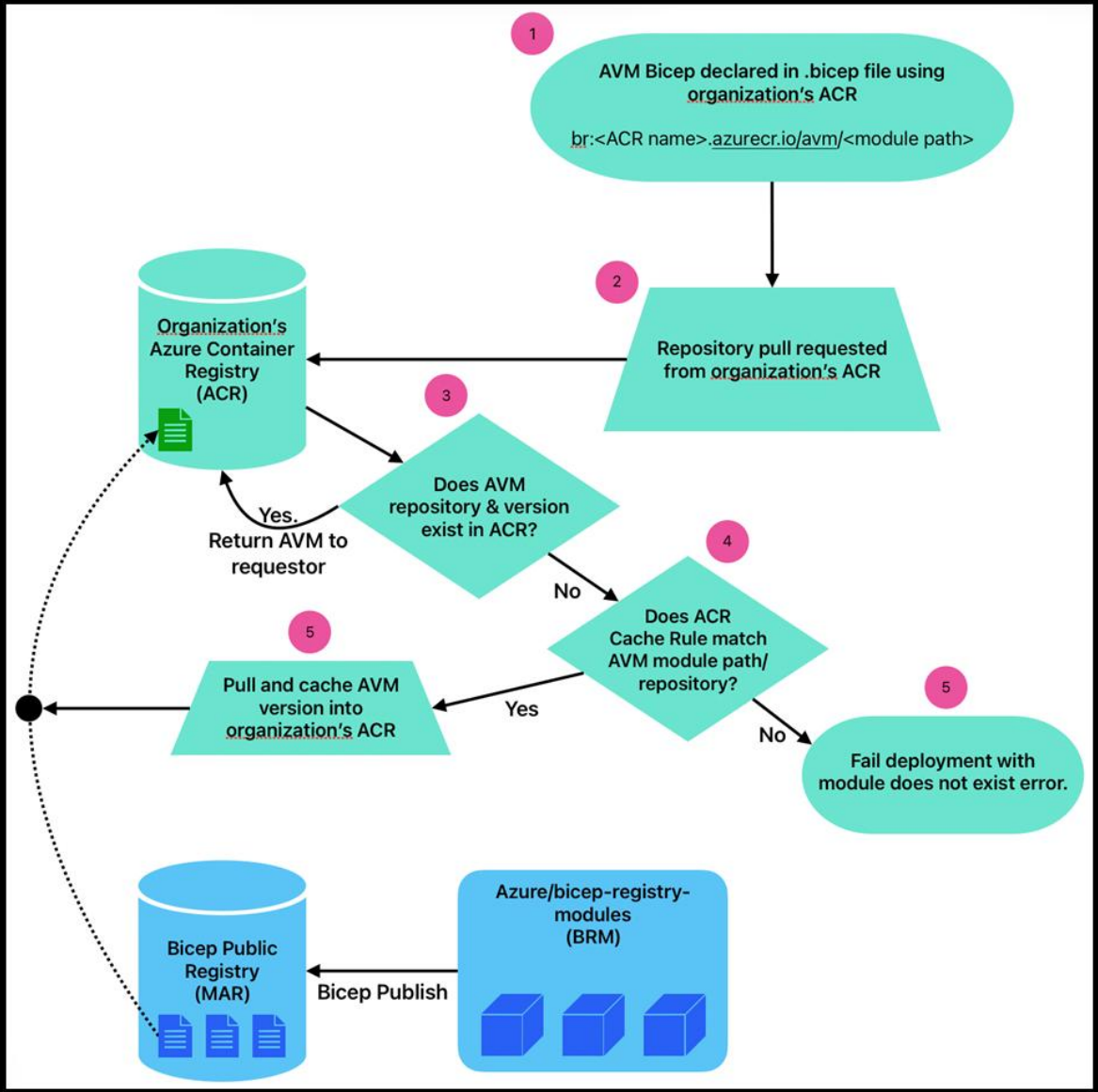
METADATA

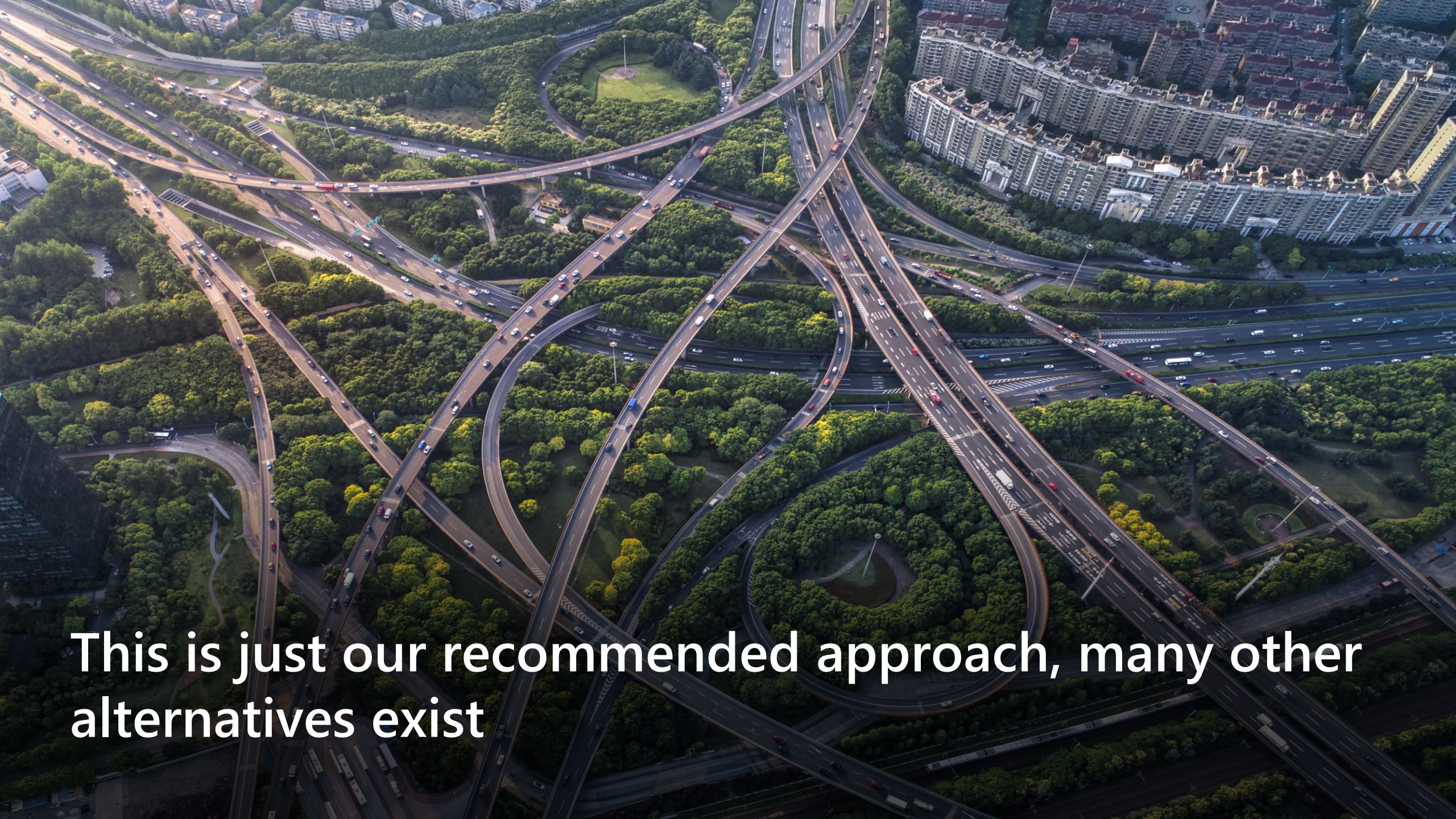
CODETOUR

1 0 3 Live Share

Ln 11, Col 1 Spaces: 2 UTF-8 CRLF JSON with Comments

Bicep | Recommended Architecture Workflow





This is just our recommended approach, many other alternatives exist

Terraform | What you should know…

There are many options to choose from

- HCP Terraform/Terraform Enterprise
- Private Registries from 3rd parties (just a list of some, not recommendations, you must assess for your needs 👍):
  - JFrog, env zero, Spacelift, Tapir, Terralist, GitLab, many more…
- Anywhere accessible from Terraform as per docs
  - GitHub, Azure DevOps, Git etc…

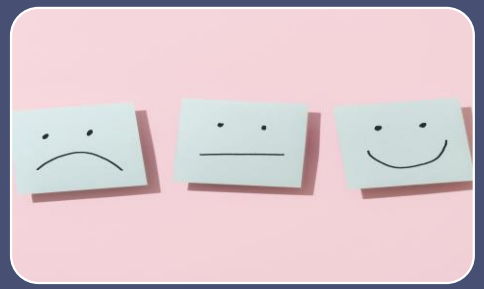
Modules that cross-reference other AVMs require additional work

- You will need to find, replace the `source` inputs in the module blocks that contain cross-references to other AVMs on the Public Terraform Registry before then publishing/storing them in your private registry/hosting solution
  - e.g.
    - from: `source = "Azure/avm-xyz-123/azurerm"`
    - to: `source = "<your private registry/hosting solution path>"`

What you can expect from us

We will document and publish this guidance and notes to the AVM website

- ETA: January 2026 (latest, hopefully before)



We will continue to listen to feedback and make enhancements in this space

- Please use the AVM repo and issues to give us feedback 👍
  - Feel free to also propose PRs attached to your issue also ❤️

For transparency and clarity, our primary focus will remain on adding features and publishing them to the public registries

Microsoft

Copilot experiences for Infra-as-Code

Sebastian Gräf, Stephanie Yen, Steven Ma

AI-accelerated IaC development

GitHub Copilot

Tool	Where	Sync/Async	Scope
Code Completion	Editor (inline)	Sync	Single line/block autocomplete
Chat (Ask, Plan)	Sidebar/inline	Sync	Q&A, explain, generate snippets
Edits	Editor	Sync	Multi-file changes with preview
Agent Mode	***Editor***	***Sync***	***Multi-step tasks with tool use***
Coding Agent	***GitHub Issues***	***Async***	***Creates branch + PR from issue***
Code Review	***GitHub PRs***	***Async***	***Reviews PR, leaves comments***
CLI	Terminal	Sync	Shell command help (gh copilot)
GitHub.com	Browser	Sync	PR summaries, issue triage

Instructions + Agents + Prompts

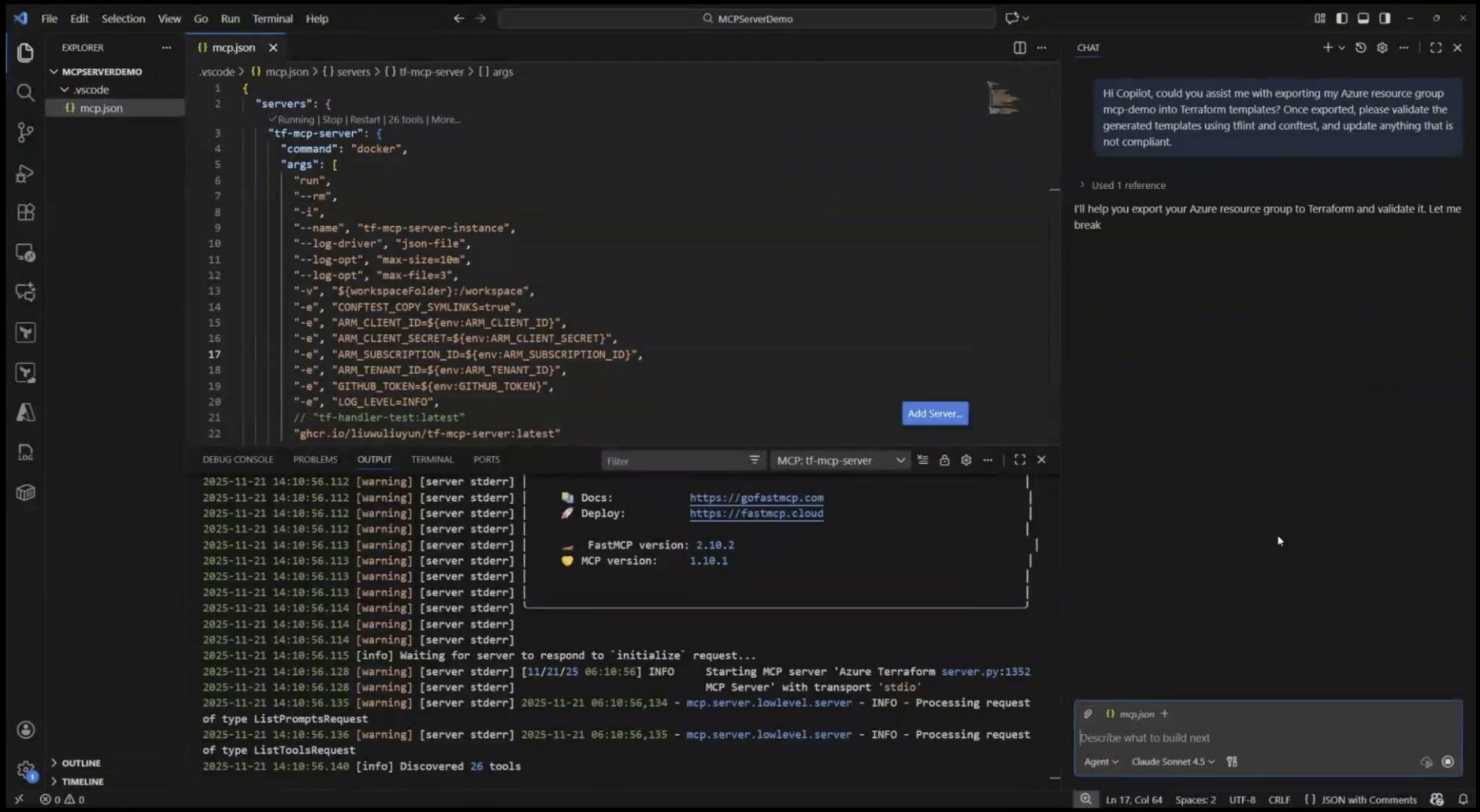
https://github.com/github/awesome-copilot

Microsoft

Demo

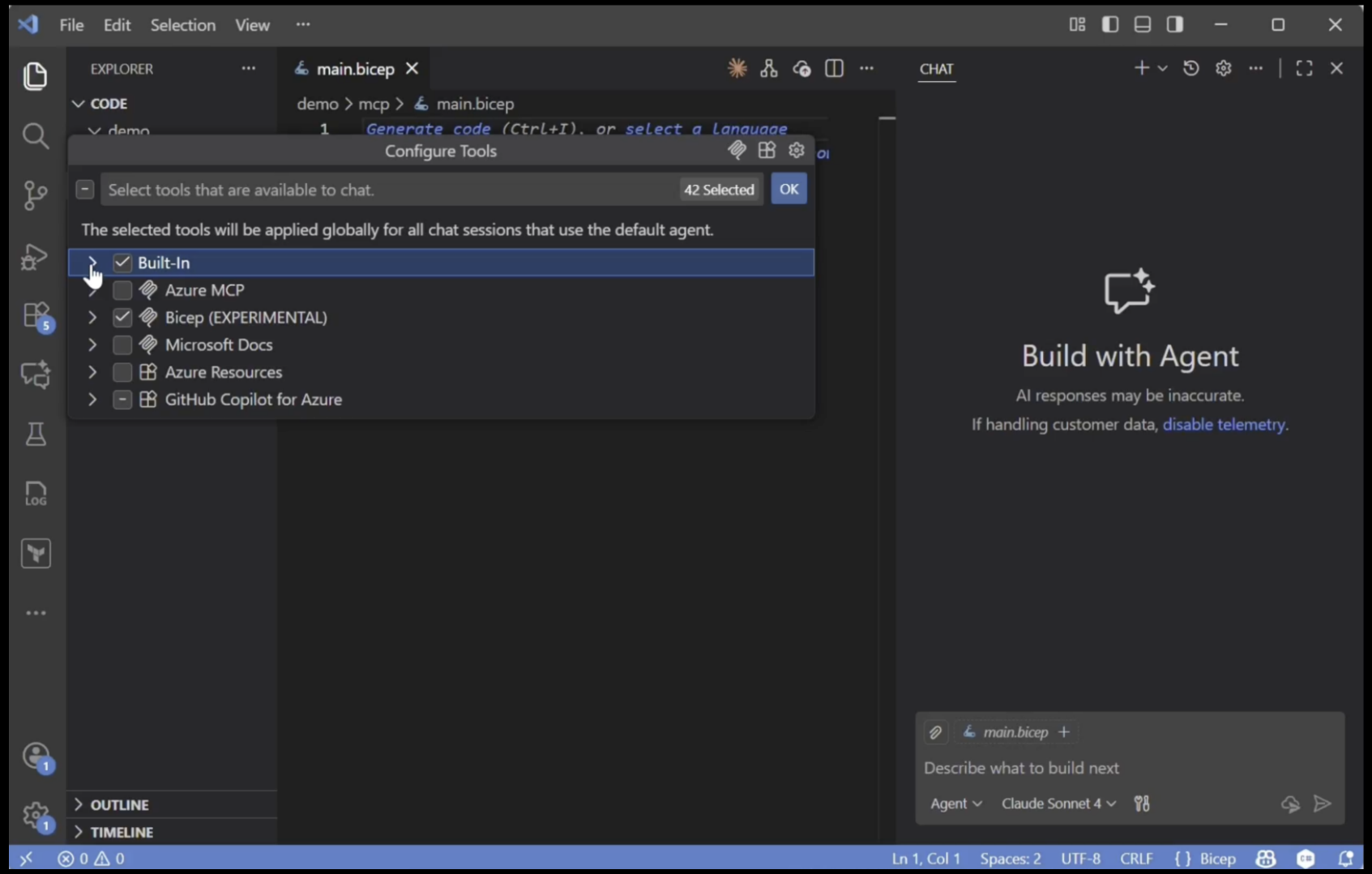
Sebastian Gräf

Azure Terraform MCP Server



MCPDemo1.mp4

Bicep MCP Server



Bicep MCP Server AVM Demo.mp4

Microsoft

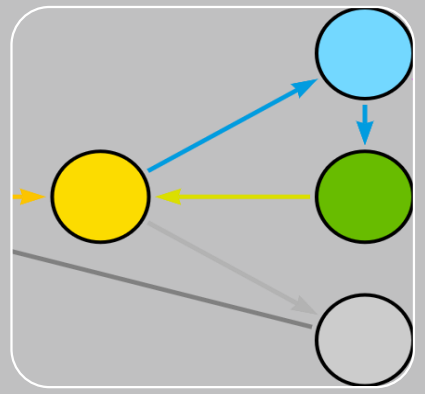
Going v1.X.X

Jack Tracey

Let’s talk about the elephant in the room…

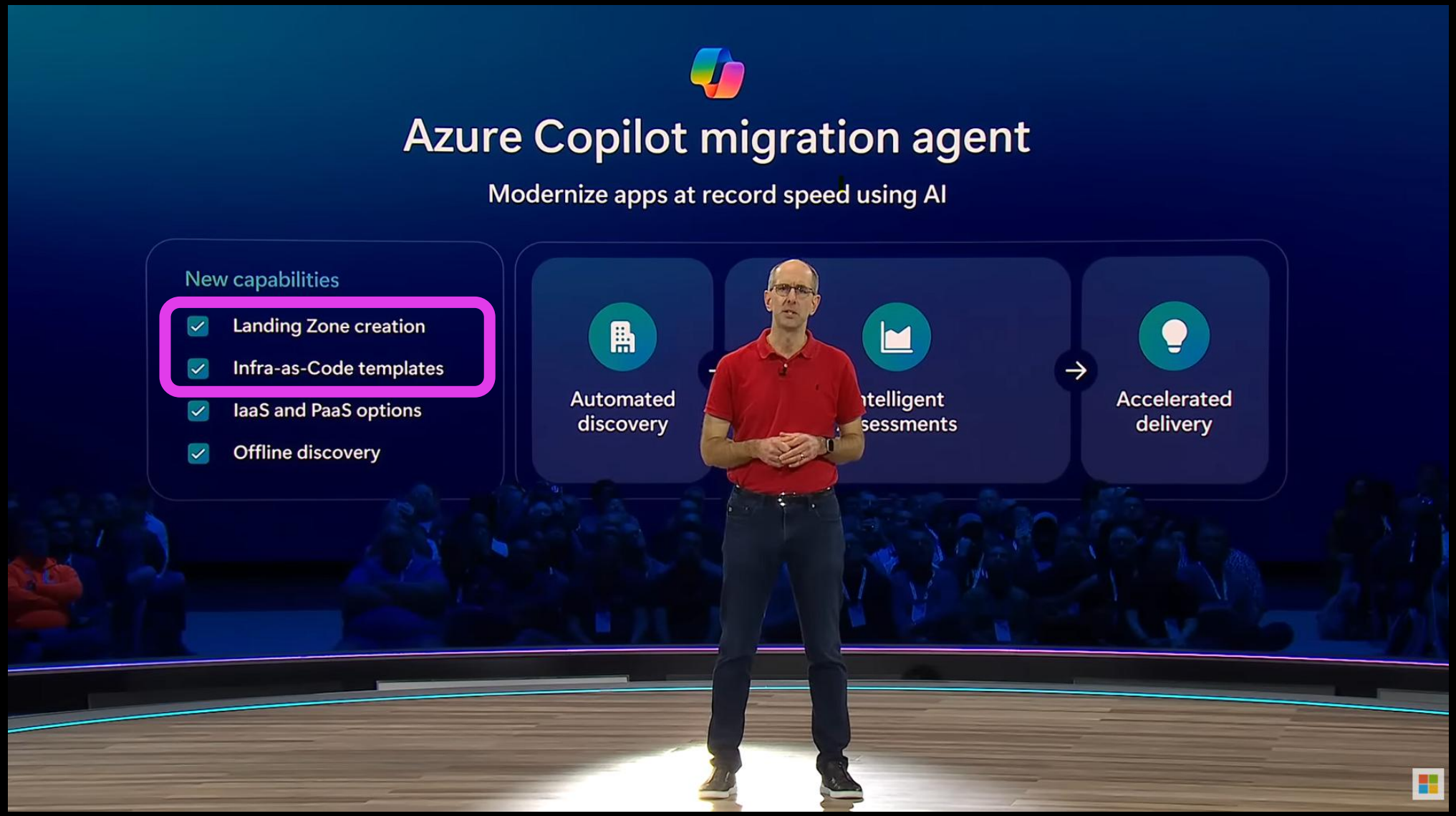

We are still making progress, just a bump in the road

Big ticket items we've been working on to get to v1



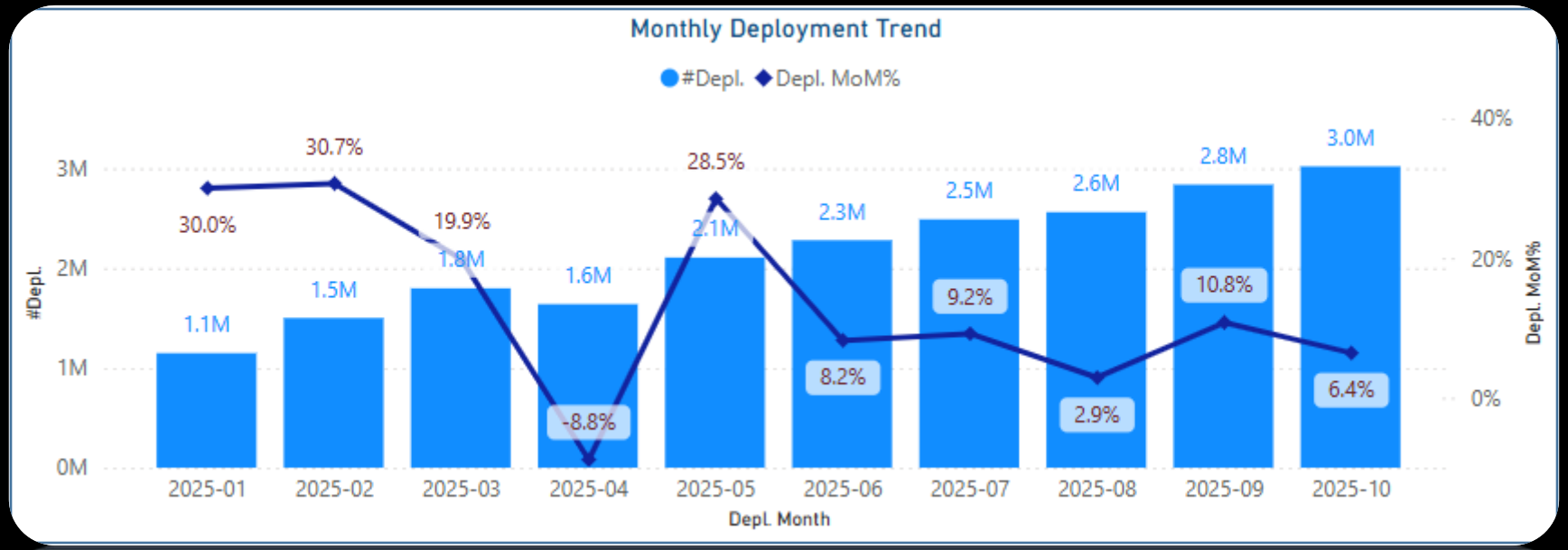
Child Modules	Utility Module specifications	CMK support for Managed HSM	Service Groups shared interface	Specifications coverage in tests
✓	✓	✓	▶	▶

Products are building upon AVM more 👍



And remember… 💡

AVM Modules today **are ready for production** use and **are being used by many in production** at **organizations of all shapes, sizes, and industries**!



And remember... 💡

V1 != ready for production ❌

V1 == stability to specs and no planned breaking changes ✅

Bicep is still v0.X.X but has been supported since v0.3 in production by everyone

Terraform only went V1 in 2021 but got to v0.15.X before it got there. And again, used in production by everyone

So, when…?

Microsoft

AVM Terraform updates

Jared Holgate, Matt White

azapi for all the things

What

- azapi will be mandated for all resource modules
- azapi will be preferred for pattern modules
- This includes all interface resources, azurerm will be removed altogether

Why

- azapi is now a mature and robust provider
- azapi does not need a human to implement every feature and bug fix
- azapi is owned by Microsoft
- azapi has preview and GA features from day one
- azapi has resource feature parity and consistency with Bicep
- azapi has advanced retry and timeout capabilities
- azapi supports write-only attributes for any resource

How

- Specs and automation are being updated
- The following slides have some heads up…

State migration with moved blocks

- We'll used moved blocks to make the experience as seamless as possible in 99% of cases
- There may be some edge cases (yet to be seen) where the azurerm or module implementation has fundamental issues. In these cases we may need to provide instructions to customers to add their own moved blocks.

Resource_group_name -> parent_id

- azurerm provider is limited to a single subscription per provider instance and expects the resource_group_name
- azapi does not have this limitation
- azapi expects the full id of the parent (usually a resource group in our case)
- For consistency we will call this parent_id
- We don’t want to use a data source for the subscription ID

Interfaces

- @Matt is working on a utility module called terraform-azure-avm-utl-interfaces which will do the majority of the work for you
- All interfaces will need to be migrated from azurerm to azapi equivalents. moved blocks will migrate the state

azapi specifics to consider

- Resource outputs should be limited to the minimum needed to avoid noisy plans, set response_export_values to an empty list by default!
- Use replace_triggers_refs to force destroy and create where necessary
- Use retry instead of time_sleep for eventual consistency problems

terraform test in avm pr-check

- We now run terraform test for root and submodules in CI / CD
- Support Unit and Integration tests
- Build as part of the avm-ptn-sub-vending migration – check it out!

Microsoft

AVM Bicep updates

Alexander Sehr, Erika Gressi, Jack Tracey

Child Modules

- New modules (45 in total)
- Documentation
  - BCPRMNFR3 - Child resources structure
  - Child Module Publishing
- Ready for requests

api-management/service
- api
- api-version-set
- api/diagnostics
- api/policy
- authorization-server
- backend
- cache
- diagnostics
- identity-provider
- logger
- named-value
- policy
- portalsetting
- private-endpoint-connection
- product
- product/api
- product/group
- subscription
- workspace

azure-stack-hci/cluster
- arc-setting/extension

document-db/database-account
- sql-role-assignment
- sql-role-definition

event-hub/namespace
- eventhub

key-vault/vault
- access-policy
- key
- secret

network/virtual-hub
- route-map

network/virtual-network
- subnet

storage/storage-account
- blob-service/container
- blob-service/container/immutability-policy
- file-service/share
- local-user
- management-policy
- queue-service/queue
- table-service/table

web/site
- config
- slot

Child Module Publishing

Latest updates

- Contributor guidelines
- 40+ child modules published

Contributing
Bicep Modules
Contribution Flow
Child Module Publishing


```
module subnet 'br/public:avm/res/network/virtual-network/subnet:'
```

This module deploys a Virtual Network Subnet.

View Documentation

0.1.1
0.1.0

New Bicep Child Module Proposal

Want to publish the child module of a Bicep resource module? Let us know!

```
child module should not contain a [ version.json ] file unless explicitly allowed for publishing.
main module version should be increased if the child version number has been increased.
The telemetry parameter should be present & have the expected type, default value & metadata description.
```

Remaining challenges

- ✅ Scalable approach towards onboarding

Managed HSM CMK encryption

- Integrated and validated on **16** resource modules
- Usage example integrated into the README, skipped from continuous validation for cost optimization

Example 4: ***Using managed HSM Customer-Managed-Keys with User-Assigned identity***

This instance deploys the module with Managed HSM-based Customer Managed Key (CMK) encryption, using a User-Assigned Managed Identity to access the HSM key.

You can find the full example and the setup of its dependencies in the deployment test folder path [/tests/e2e/cmk-hsm-uami]

> **Note**: This test is skipped from the CI deployment validation due to the presence of a `.e2eignore` file in the test folder. The reason for skipping the deployment is:

```
ured at all times, which would incur significant costs for contributors.
```

Consistent Features & Extension Resources (Interfaces)

Consistent Features & Extension Resources (Interfaces)

The following table shows which Bicep resource modules have which consistent features and extension resources (interfaces) implemented as defined in the Bicep Interfaces specification.

#	Module	RBAC	Locks	Tags	Diag	PE	CMK	CMK-mHSM	Identity
28	avm/res/compute/disk-encryption-set	✅	✅	✅			✅	✅	✅
37	avm/res/container-registry/registry	✅	✅	✅	✅	✅	✅		✅

https://azure.github.io/Azure-Verified-Modules/indexes/bicep/bicep-resource-modules/#consistent-features--extension-resources-interfaces

Managed HSM CMK encryption - implementation

- CMK **shared interface**: common to key vault vault and managed HSM
- Keep input interface (common type), **avoiding breaking changes**
- Complexity within the module, **transparent** to the user
- **Least privilege**: Does not require RBAC at managed HSM (mHSM) resource level for the deployment identity

CMK shared interface

```
customerManagedKey: {
  keyVaultResourceId:
  keyName:
  keyVersion:
  autoRotationEnabled:
  userAssignedIdentityResourceId:
}
```

CMK conditional implementation

```
var isHSMManagedCMK = split(customerManagedKey.?keyVaultResourceId ?? '', '/')[?7] == 'managedHSMs'
```

```
encryption: !empty(customerManagedKey)
  ? {
      keySource: 'Microsoft.KeyVault'
      keyVaultProperties: {
        keyVaultUri: !isHSMManagedCMK
            ? cMKKeyVault!.properties.vaultUri
            : 'https://${last(split((customerManagedKey!.keyVaultResourceId), '/'))}.managedhsm.azure.net/'
```

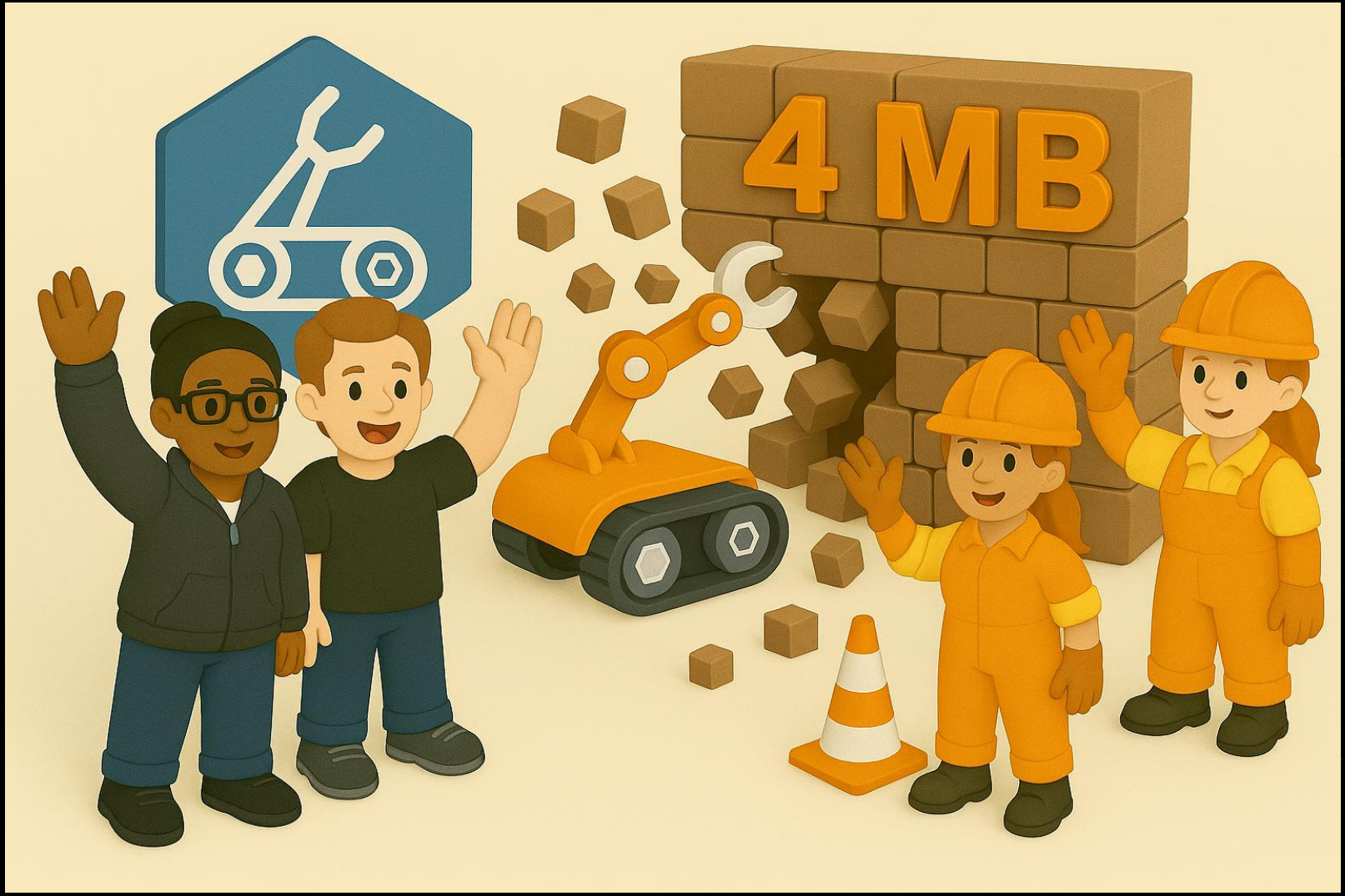
Outcome

→ **KV CMK encryption**

→ **mHSM CMK encryption**

https://azure.github.io/Azure-Verified-Modules/specs/bcp/res/interfaces/#customer-managed-keys

An update on the 4 MB ARM Limit



Original Issue: Bicep linked templates - 4Mb limit · Issue #5890 · Azure/bicep

Q & A

Questions from GH

pazdedav 4 days ago

Guidance on using the Migration Agent featured in this video that creates platform landing zone vs. the AVM path for doing the same. *Question: Was the AVM team involved in building the code behind agent-generated platform LZ? Or does it use AVM modules?*

vjmanda 7 hours ago · edited by vjmanda

- AzureRM vs AzAPI
  - Why the shift?
  - Road map of when this move might happen
  - Likely impact on those already using AVMs based on AzureRM
  - How to migrate state from AzureRM following adoption of AzAPI?

Getting Involved – aka.ms/AVM

AVM is open for everyone to contribute

- Devolved ownership, not centralised!
- AVM welcomes contributors from all over the world!

Learn

- AVM Resources – aka.ms/AVM/resources
  - Labs, blog posts, podcasts, videos and more
- Leverage aka.ms/AVM/specs & aka.ms/AVM/contributing
- Stay informed – aka.ms/avm/monthly/latest

Contribute

- Identify which proposed modules you would like to contribute to:
  - Bicep AVM modules looking for contributors
  - Terraform AVM modules looking for contributors
- Propose a new module: aka.ms/AVM/ModuleProposal