PSCONF.EU

PowerShell Conference Europe

Microsoft Sentinel lifecycle management at scale

Fabian Bader

Antwerp 24

Many thanks to our sponsors:

@fabian_bader

Fabian Bader

- Cyber Security Architect glueckkanja AG
- Azure and Security MVP
- Blog: cloudbrother.info
- Organizer of
  - HH PowerShell UG
  - Purple Elbe Security UG

@fabian_bader

What to expect

- Based on real world experience
- Concept based
- Dos and don’ts
- No code sharing beyond what’s already available

What’s Microsoft Sentinel?

- A SIEM - Security Information and Event Management
- Azure cloud based solution
- Security log data from various systems is ingested
- Detects threats in your environment
- Must be „manually“ managed

The challenge as a MSSP

- Manage multiple Sentinel instances for multiple customers
- Deploy artifacts at scale
  - Analytics rules (detections)
  - Watchlists
  - …
- Have a standard way but adopt to the customer environment

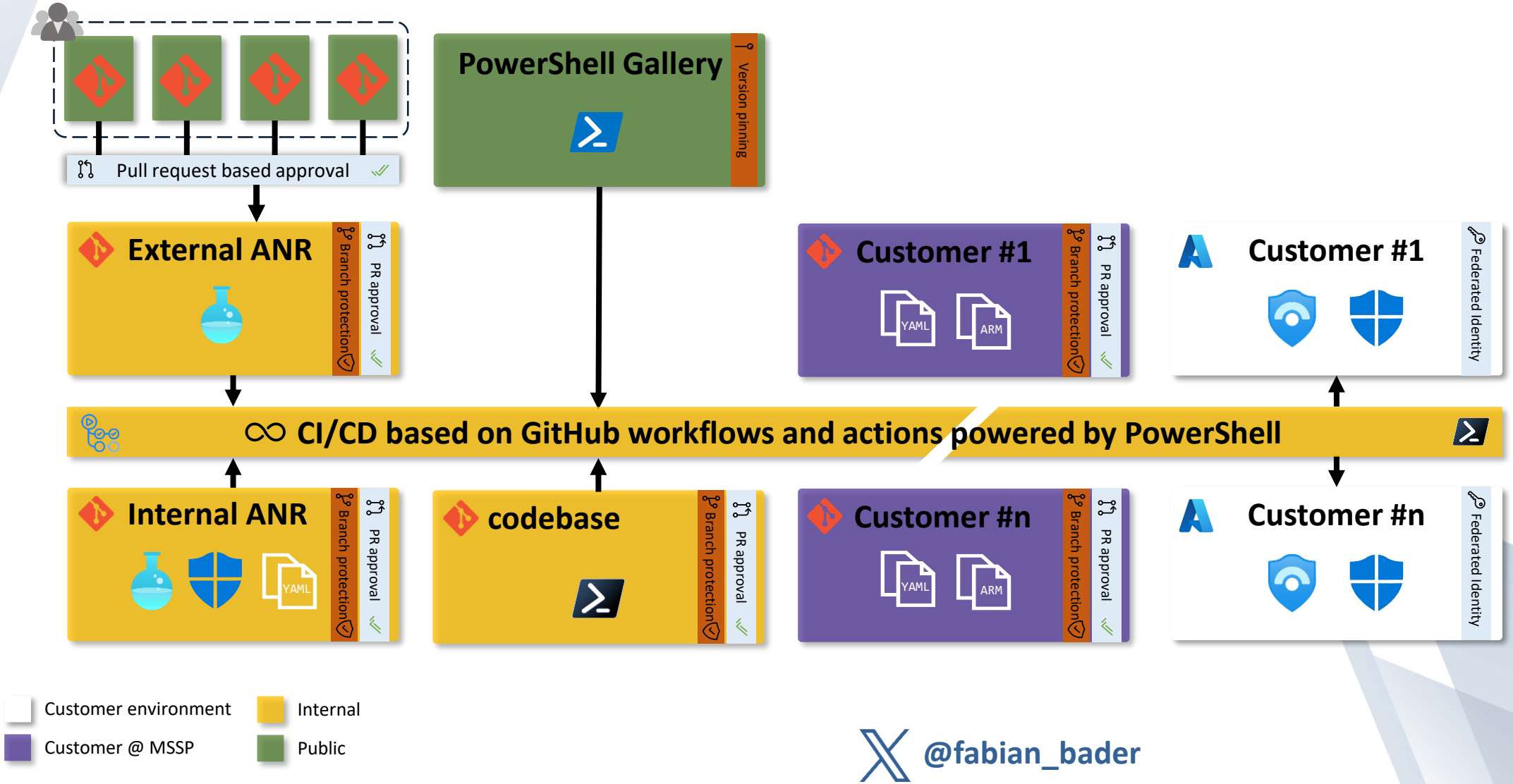
Feature Comparsion

Feature	Workspace Manager	Native GitHub integration
Analytics Rules (ANR)	✅	✅
Parser / Functions	⛔	⛔
Watchlists upload	⛔	⛔
Watchlist edit	⛔	⛔
Internal ANR Sources	🖱️ (manually)	🖱️ (manually per repo)
External ANR Sources	🖱️ (manually)	🖱️ (manually per repo)
Parameterization	⛔	⚠️
Change starttime of ANR	🖱️ (manually)	🖱️ (manually)
Meta-data store	⛔	⛔
Backup capability	⛔	⛔

Build a “new” solution

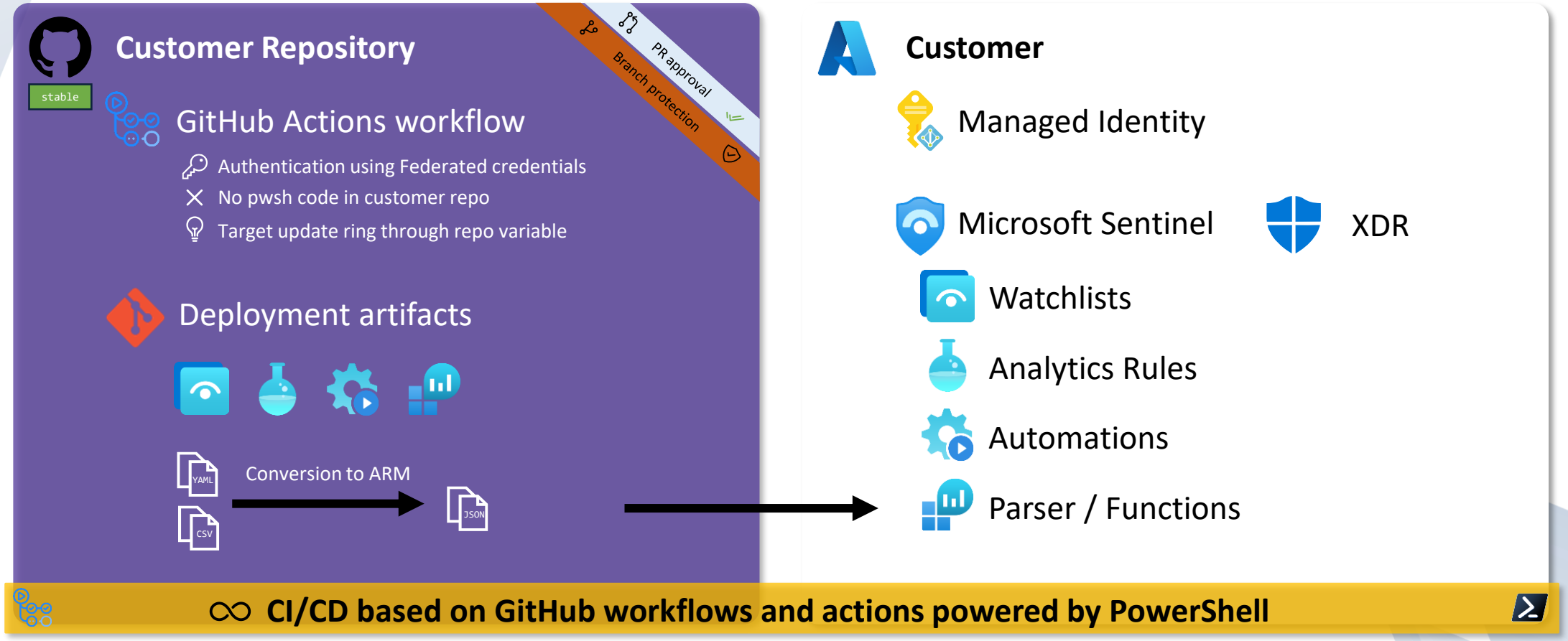
- Use what’s already available
- Build what’s missing
- Give back to the community

Our solution - CSOC Foundation



@fabian_bader

Deploy to the customer



Customer environment
Customer @ MSSP
Internal
Public

@fabian_bader

Which tools we use?

- Highly modified version of azure-sentinel-deploy.ps1
- SentinelARConverter (pwsh Module)
- SentinelEnrichment (pwsh Module)
- Sentinel Pester Framework
- Modified version of SecureHats/validate-detections
- Other pwsh Modules (powershell-yaml, Az)

Publicly shared modules

- SentinelEnrichment
  - Overwrite a Sentinel watchlist
  - Manage watchlists based on tags
  - Get information from
    - Microsoft Graph
    - Azure Data Lake
    - Azure DevOps API
  - Write data to Azure Data Collection Endpoint

Publicly shared modules

- SentinelARConverter*
  - Convert YAML based analytics rules into ARM templates
  - Modify properties on the fly
    - Start time
    - Severity
    - Analytics Rule name prefix
    - Parameterization: Variables and pre- and post KQL code
  - Validation of known deployment limitations (e.g. MITRE mapping)

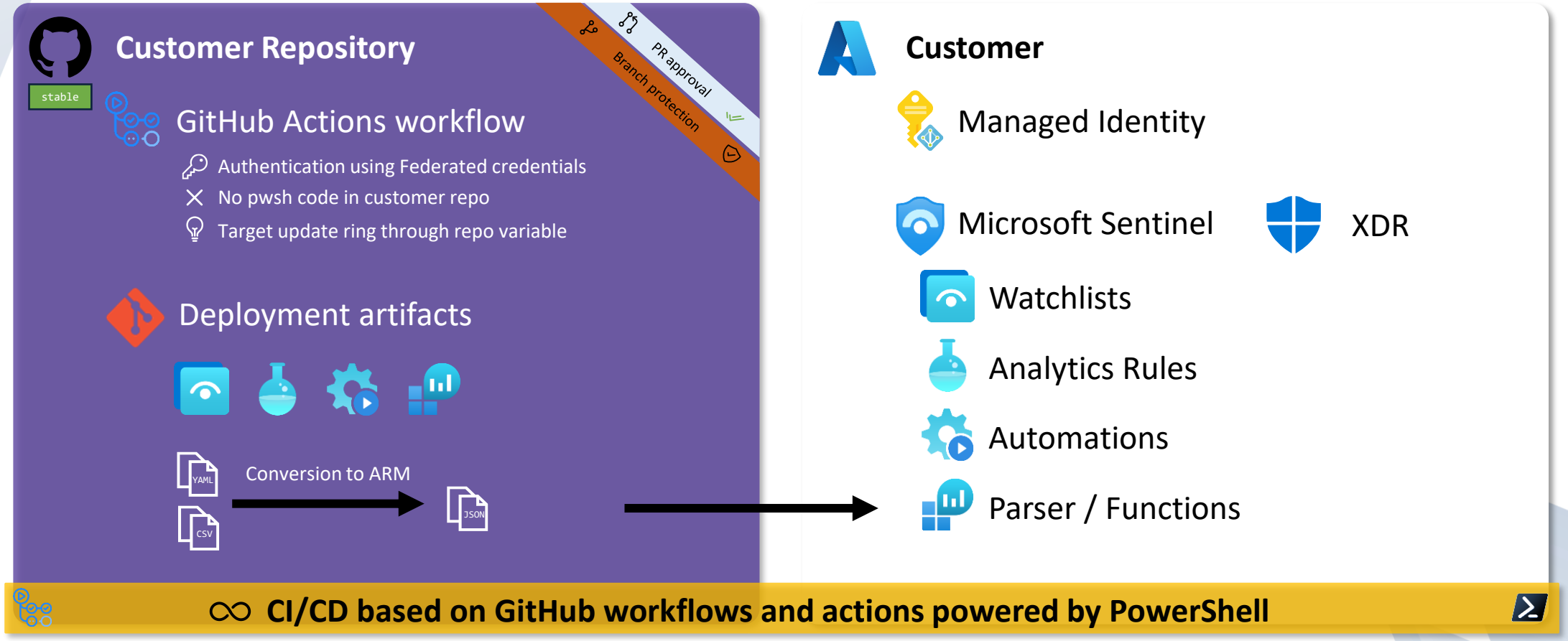
*not maintained by glueckkanja, personal project

Why YAML?

```yaml
id: 6bb8e22c-4a5f-4d27-8a26-b60a7952d5af
name: Azure WAF matching for Log4j vuln(CVE-2021-44228)
kind: Scheduled
description: |-
  This query will alert on a positive pattern match by Azure WAF for CVE-2021-44228 log4j vulnerability exploitation attempt. If possible
   Refrence: https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-
severity: High
queryFrequency: 6h
queryPeriod: 6h
triggerOperator: gt
triggerThreshold: 0
tactics:
  - InitialAccess
query: "AzureDiagnostics\n| where ResourceProvider == \"MICROSOFT.NETWORK\" and Category in (\"ApplicationGatewayFirewallLog\", \"Frontdo
suppressionEnabled: false
incidentConfiguration:
  createIncident: true
  groupingConfiguration:
    enabled: false
    reopenClosedIncident: false
    lookbackDuration: 5h
    matchingMethod: AllEntities
    groupByEntities:
      - Account
      - IP
      - Host
      - URL
      - FileHash
    groupByAlertDetails:
    groupByCustomDetails:
eventGroupingSettings:
  aggregationKind: SingleAlert
entityMappings:
  - entityType: IP
    fieldMappings:
      - identifier: Address
        columnName: IPCustomEntity
suppressionDuration: 1h
```

```json
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "workspace": {
            "type": "String"
        }
    },
    "resources": [
        {
            "id": "[concat(resourceId('Microsoft.OperationalInsights/workspaces/providers', parameters('workspace'), 'Microsoft.Security
            "name": "[concat(parameters('workspace'),'/Microsoft.SecurityInsights/6bb8e22c-4a5f-4d27-8a26-b60a7952d5af')]",
            "type": "Microsoft.OperationalInsights/workspaces/providers/alertRules",
            "kind": "Scheduled",
            "apiVersion": "2022-09-01-preview",
            "properties": {
                "displayName": "Azure WAF matching for Log4j vuln(CVE-2021-44228)",
                "description": "This query will alert on a positive pattern match by Azure WAF for CVE-2021-44228 log4j vulnerability exp
                "severity": "High",
                "enabled": true,
                "query": "AzureDiagnostics\n| where ResourceProvider == \"MICROSOFT.NETWORK\" and Category in (\"ApplicationGatewayFirewa
                "queryFrequency": "PT6H",
                "queryPeriod": "PT6H",
                "triggerOperator": "GreaterThan",
                "triggerThreshold": 0,
                "suppressionDuration": "PT1H",
                "suppressionEnabled": false,
                "startTimeUtc": null,
                "tactics": [
                    "InitialAccess"
                ],
                "techniques": [],
                "alertRuleTemplateName": "2de8abd6-a613-450e-95ed-08e503369fb3",
                "incidentConfiguration": {
                    "createIncident": true,
                    "groupingConfiguration": {
                        "enabled": false,
                        "reopenClosedIncident": false,
                        "lookbackDuration": "PT5H",
                        "matchingMethod": "AllEntities",
                        "groupByEntities": [
                            "Account",
                            "IP",
                            "Host",
                            "URL",
                            "FileHash"
                        ],
                        "groupByAlertDetails": null,
                        "groupByCustomDetails": null
                    }
                },
                "eventGroupingSettings": {
                    "aggregationKind": "SingleAlert"
                },
                "alertDetailsOverride": null,
                "customDetails": null,
                "entityMappings": [
                    {
                        "entityType": "IP",
                        "fieldMappings": [
                            {
                                "identifier": "Address",
                                "columnName": "IPCustomEntity"
                            }
                        ]
                    }
                ],
                "sentinelEntitiesMappings": null,
                "templateVersion": null
```

Deploy to the customer



Customer environment
Customer @ MSSP
Internal
Public

@fabian_bader

Why GitHub actions

- Single repository for codebase and workflows
- Reusable workflows for all customers
- Central point of update
- Scoping and testing using branches
- Protection through CODEOWNERS and branch policies

Centralized code

Internal codebase

Code | Blame | 89 lines (81 loc) · 3.29 KB

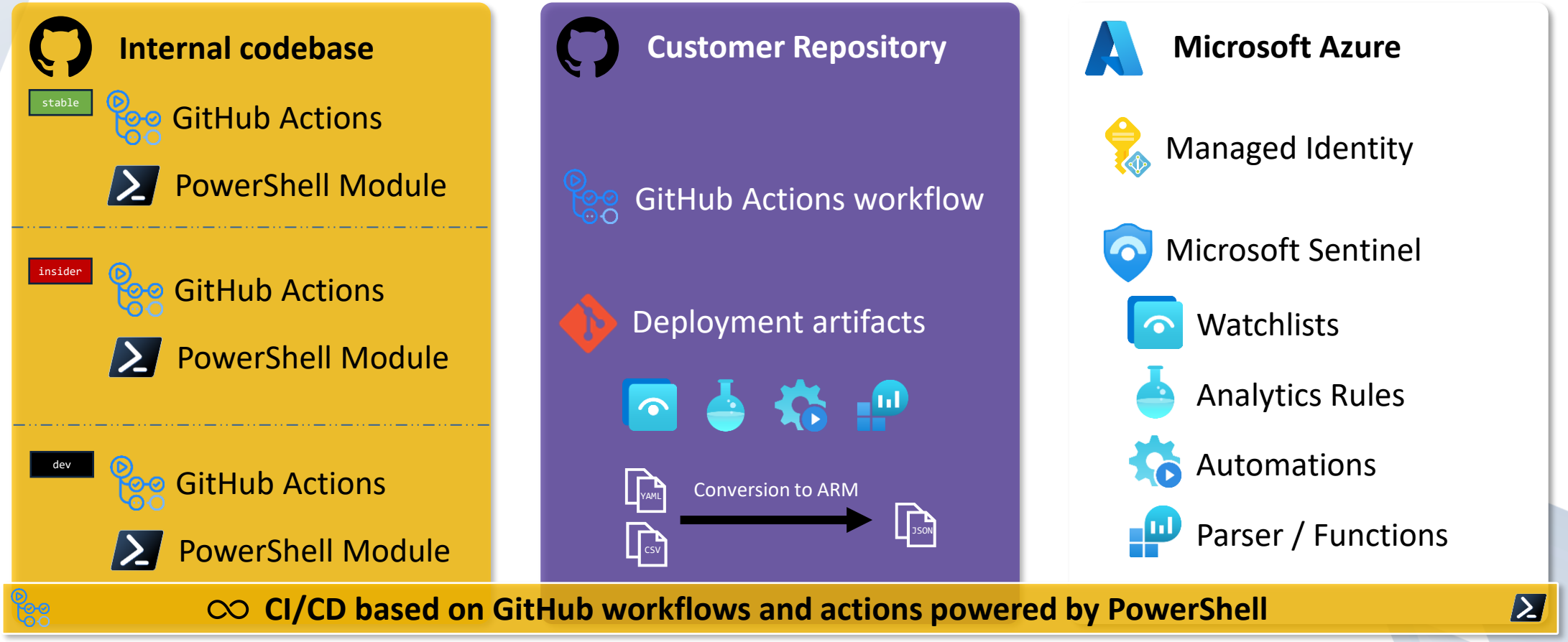
```yaml
name: Deploy-GkgabCsocSentinelArtifacts
description: [redacted]
inputs:
  AZURE_CLIENT_ID:
    description: "Client Id of Azure AD Federated Credential"
    required: true
  AZURE_TENANT_ID:
    description: "Tenant Id of the customers Sentinel environment"
    required: true
  AZURE_SUBSCRIPTION_ID:
    description: "Subscription Id of the customers Sentinel environment"
    required: true
  GITHUB_TOKEN:
    description: "Path to the CSOC Foundation code"
    required: true
  GITHUB_APP_ID:
    description: "App Id for GitHub to access Codebase"
    required: true
  GITHUB_PRIVATE_KEY:
    description: "Private Key for GitHub to access Codebase"
    required: true

runs:
  using: "composite"
  steps:
    - name: Generate token
      id: generate_token
      uses: azure-resources/github-app-token@v1
      with:
        app_id: ${{ inputs.GITHUB_APP_ID }}
        private_key: ${{ inputs.GITHUB_PRIVATE_KEY }}

    - name: Get foundation code
      shell: pwsh
      run: |
        [redacted]

    - name: Checkout
      uses: actions/checkout@v2

    - name: OIDC Login to Azure Public Cloud with AzPowershell (enableAzPSSession true)
      uses: azure/login@v2
      with:
        client-id: ${{ inputs.AZURE_CLIENT_ID }}
        tenant-id: ${{ inputs.AZURE_TENANT_ID }}
        subscription-id: ${{ inputs.AZURE_SUBSCRIPTION_ID }}
```

Customer Repository

Code | Blame | 68 lines (59 loc) · 2.05 KB

```yaml
name: Deploy Sentinel artifacts to customer

env:
  FOUNDATION_CODE: ${{ vars.FOUNDATION_CODE }}
  UPDATE_RING: ${{ vars.UPDATE_RING }}

on:
  workflow_dispatch:

  schedule:
    - cron: "0 11 * * *"

  push:
    branches: [ main ]
    paths:
    - "Sentinel/AnalyticsRules/**"
    - "Sentinel/AutomationRules/**"
    - "Sentinel/Parsers/**"
    - "Sentinel/Playbooks/**"
    - "Sentinel/Workbooks/**"

permissions:
      id-token: write
      contents: write
      pull-requests: write

jobs:
  Deploy-GkgabCsocSentinelArtifacts-insider:
    if: ${{ vars.UPDATE_RING != 'stable' }}
    environment: prod
    runs-on: ubuntu-latest
    env:
      directory: '${{ github.workspace }}/Sentinel'
      branch: 'main'
      rootDirectory: '${{ github.workspace }}/Sentinel'
      githubAuthToken: ${{ secrets.GITHUB_TOKEN }}
      smartDeployment: 'true'

    steps:
    - uses: csoc-foundation/csoc-foundation-codebase/Deploy-SentinelArtifacts@dev_fabian
      with:
        AZURE_CLIENT_ID: ${{ secrets.AZURE_CLIENT_ID }}
        AZURE_TENANT_ID: ${{ secrets.AZURE_TENANT_ID }}
        AZURE_SUBSCRIPTION_ID: ${{ secrets.AZURE_SUBSCRIPTION_ID }}
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
        GITHUB_APP_ID: ${{ secrets.APP_ID }}
        GITHUB_PRIVATE_KEY: ${{ secrets.APP_PRIVATE_KEY }}
```

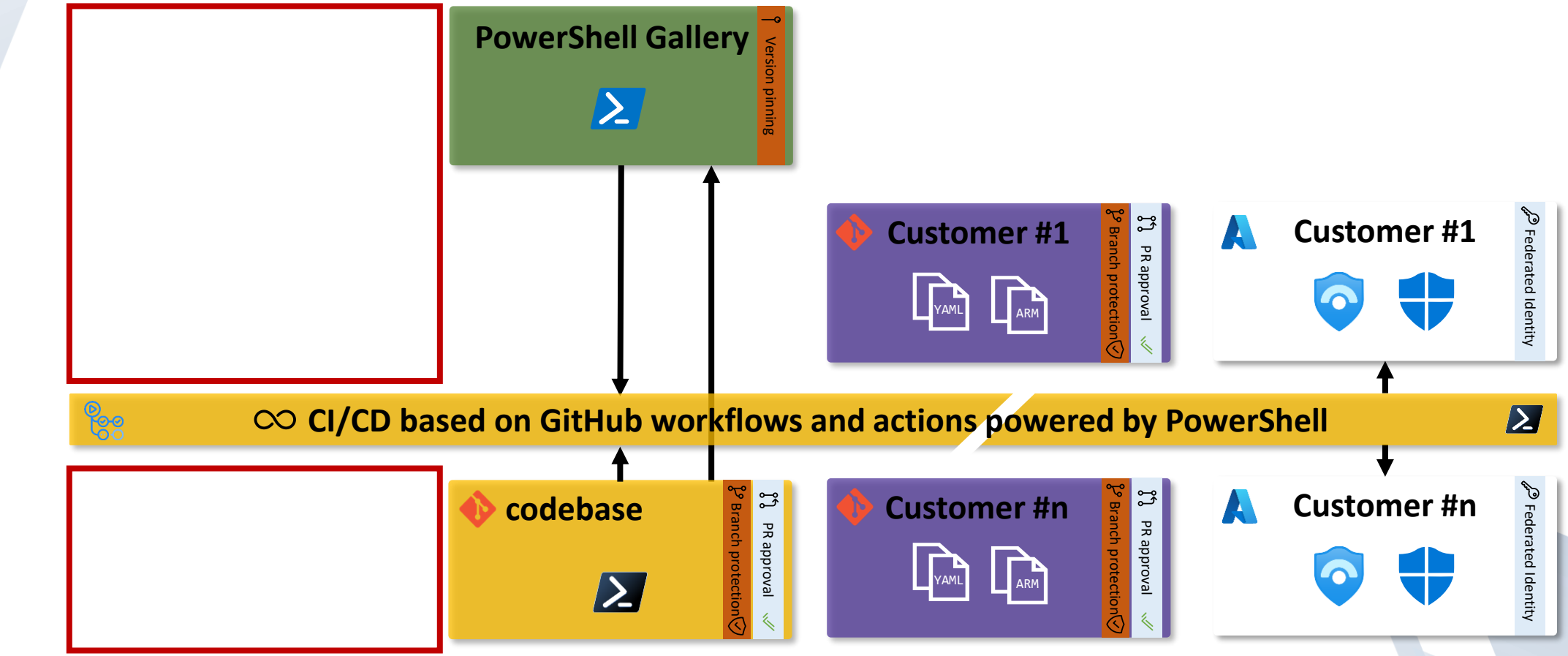
Centralized code



Customer environment
Internal
Customer @ MSSP
Public

@fabian_bader

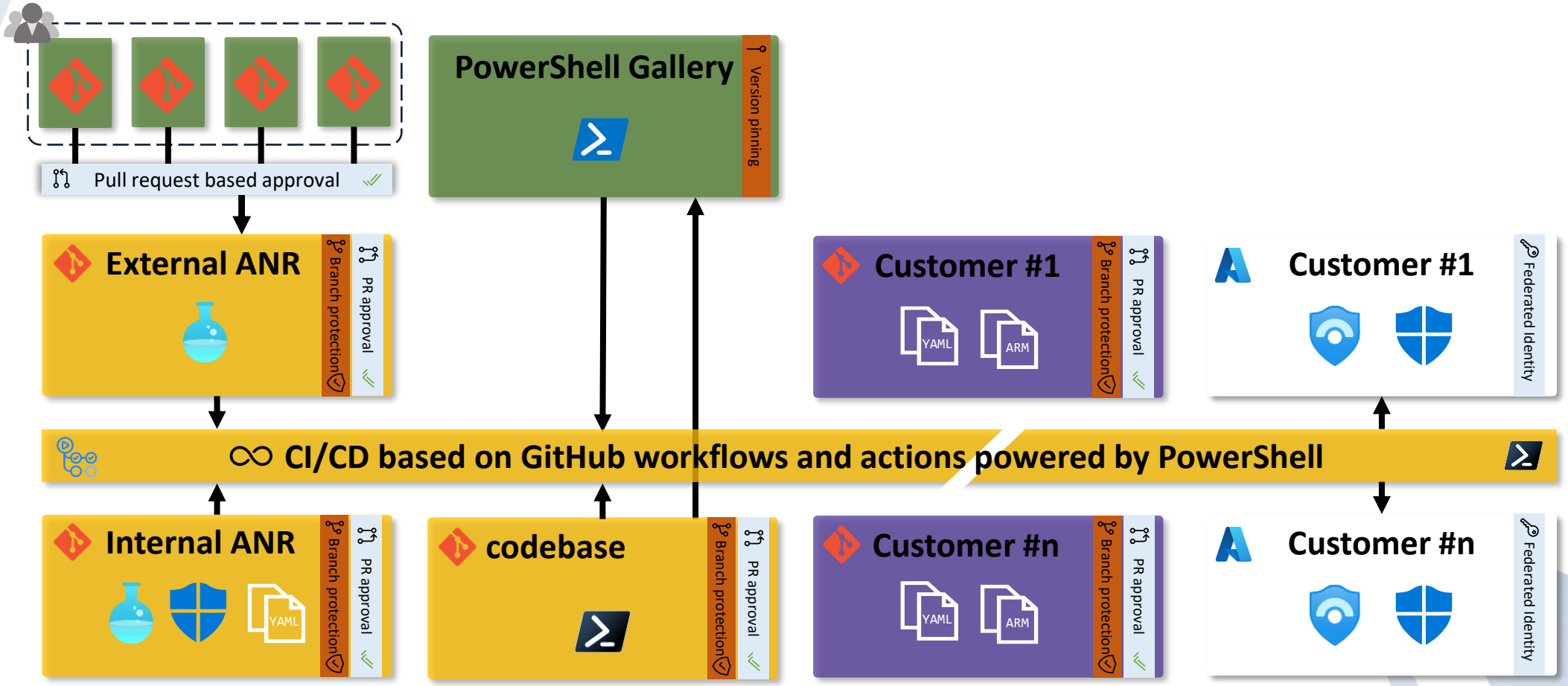
But what about detections?



Customer environment
Customer @ MSSP
Internal
Public

@fabian_bader

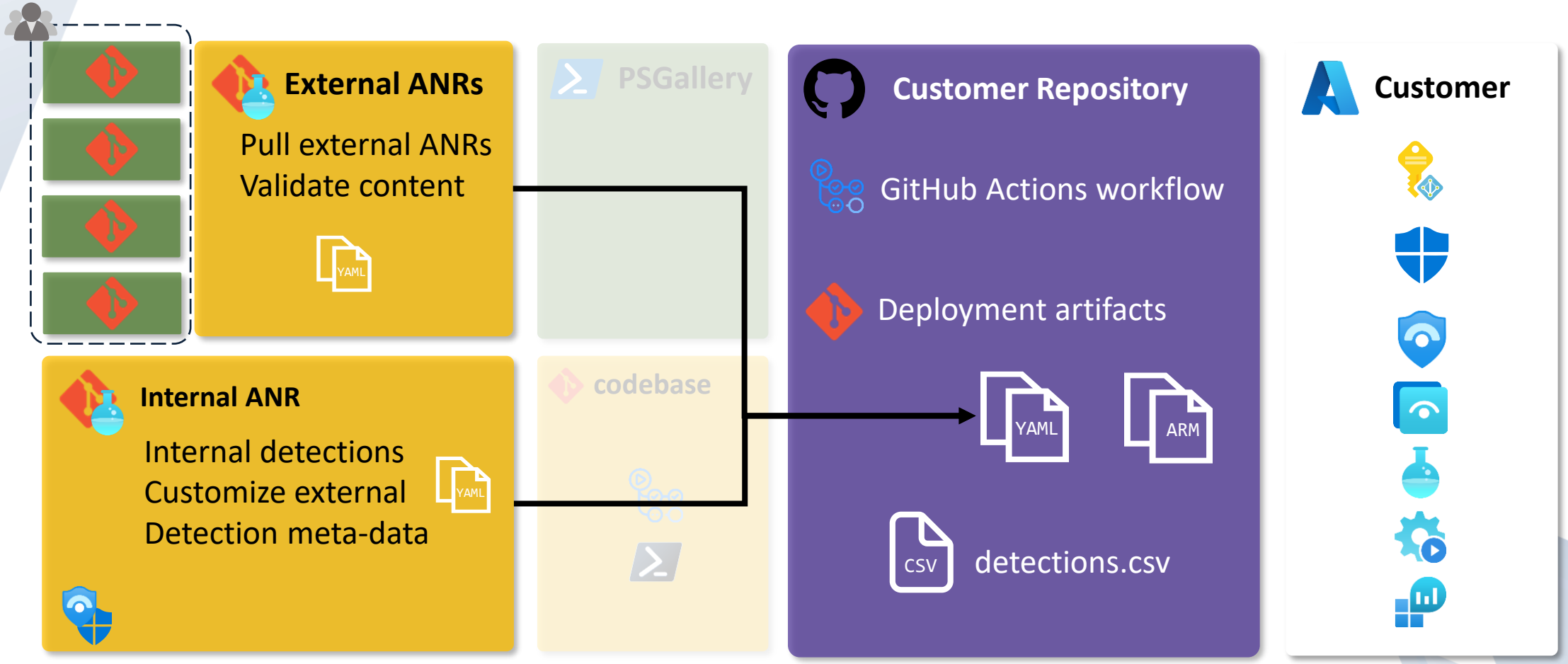
Centralized detection management



Customer environment
Customer @ MSSP
Internal
Public

@fabian_bader

Centralized detection management



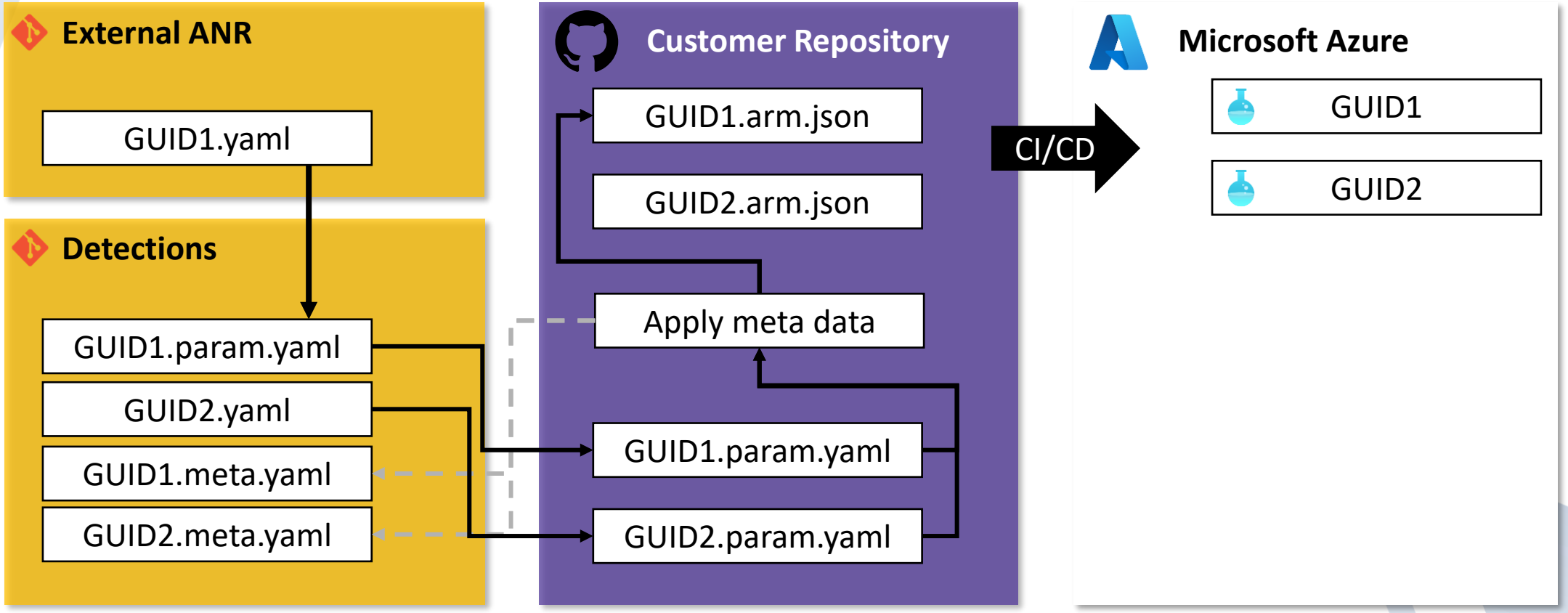
Customer environment
Internal
Customer @ MSSP
Public

@fabian_bader

Why do parametrization?

- External rules globally
  - Change severity
  - Change run interval
  - Append filter ability based on e.g. watchlists
- Internal rules per customer
  - Replace variables in NRT rules (e.g. break glass)
  - Overwrite global settings
  - Minor query changes to accommodate the customer env.

Parametrization



@fabian_bader

Dos and don’ts

- Analytics rules run immediately after deployment
  - Avoid deploying all customers at once
- Analytics rules that run once a day should not run in all environments at once

```
PS C:\> Convert-SentinelARYamlToArm -StartRunningAt
```

- Always sort JSON data before committing 

```
PS C:\> Invoke-SortJSONObject
```

Dos and don’ts

- Establish a meta-data store for all detections
  - Suitable for production
  - Prerequisites
  - Deployment scope
- Use meta-data information in your automation

Dos and don’ts

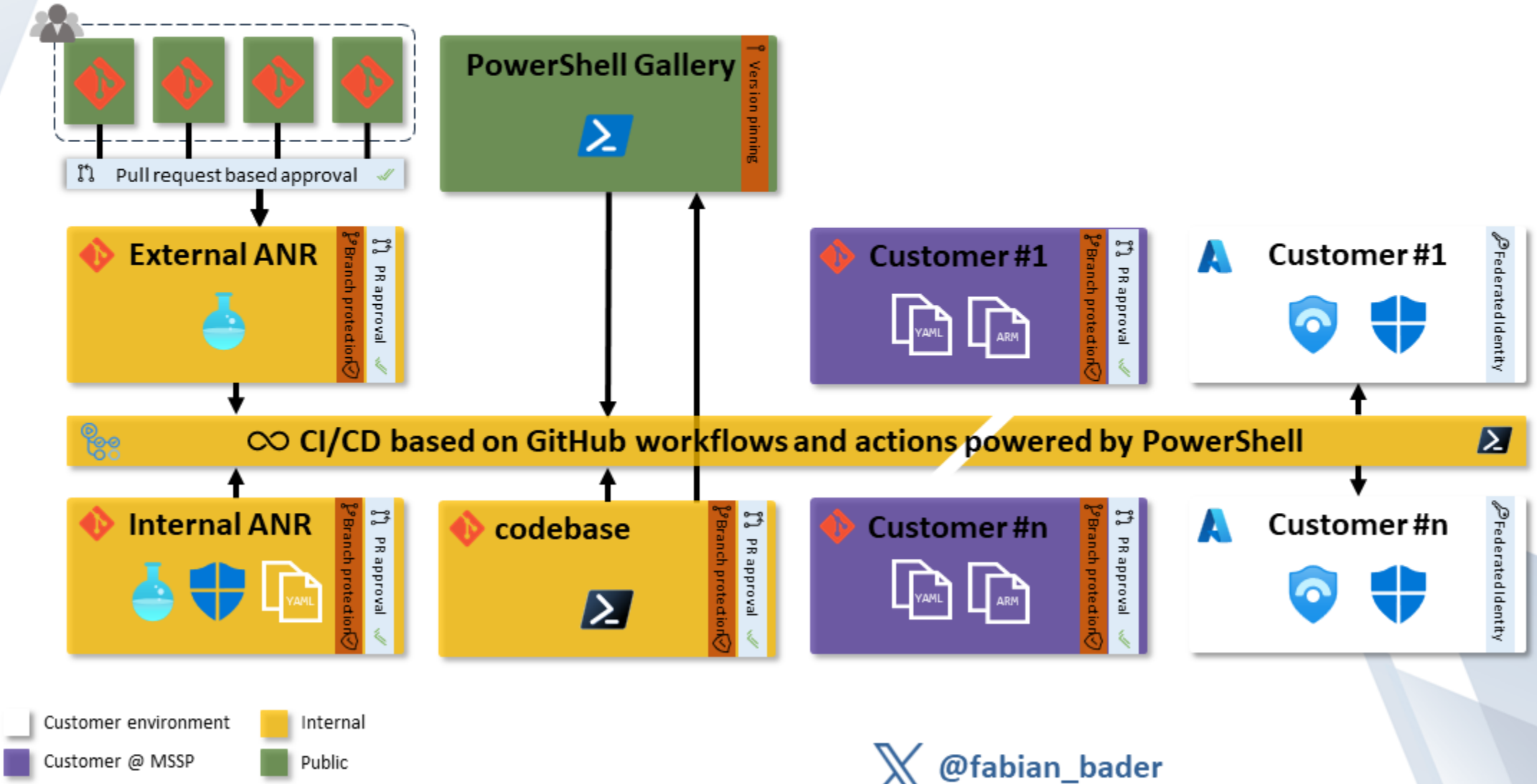
- Get telemetry
  - Deployment success
  - Alert noise in production
  - Closure codes (FP/TP)
- Get feedback/listen to your analysts!

Feature Comparsion

Feature	CSOC Foundation	Workspace Manager	Native GitHub integration
Analytics Rules (ANR)	✅	✅	✅
Parser / Functions	✅	⛔	⛔
Watchlists upload	✅	⛔	⛔
Watchlist edit	✅	⛔	⛔
Internal ANR Sources	✅	🖱️ (manually)	🖱️ (manually per repo)
External ANR Sources	✅	🖱️ (manually)	🖱️ (manually per repo)
Parameterization	✅	⛔	⚠️
Change starttime of ANR	✅	🖱️ (manually)	🖱️ (manually)
Meta-data store	✅	⛔	⛔
Backup capability	✅	⛔	⛔
XDR Custom Detections	✅	⛔	⛔

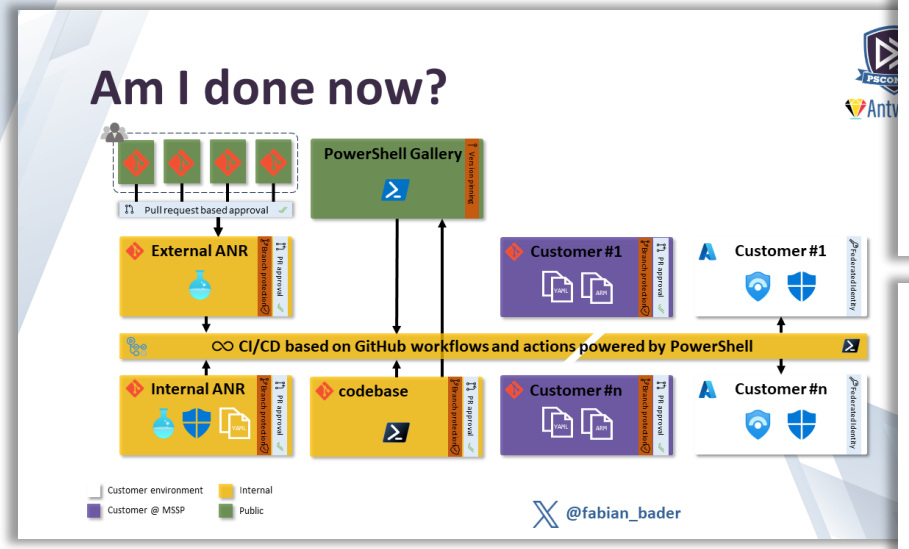
@fabian_bader

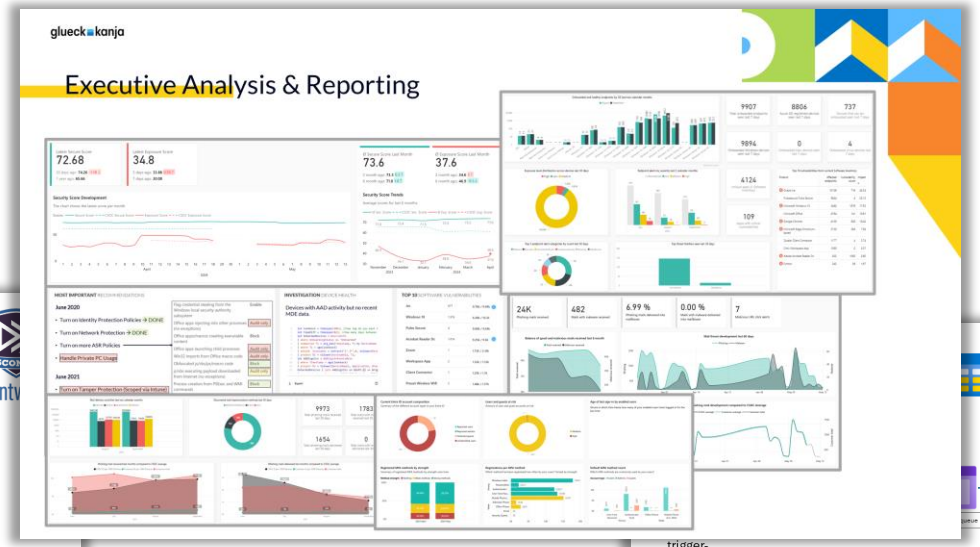
Am I done now?

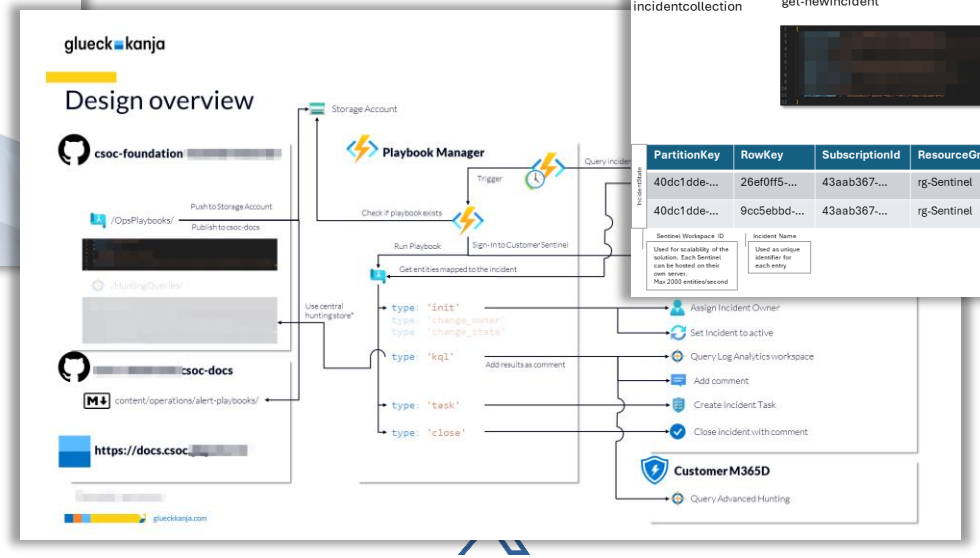


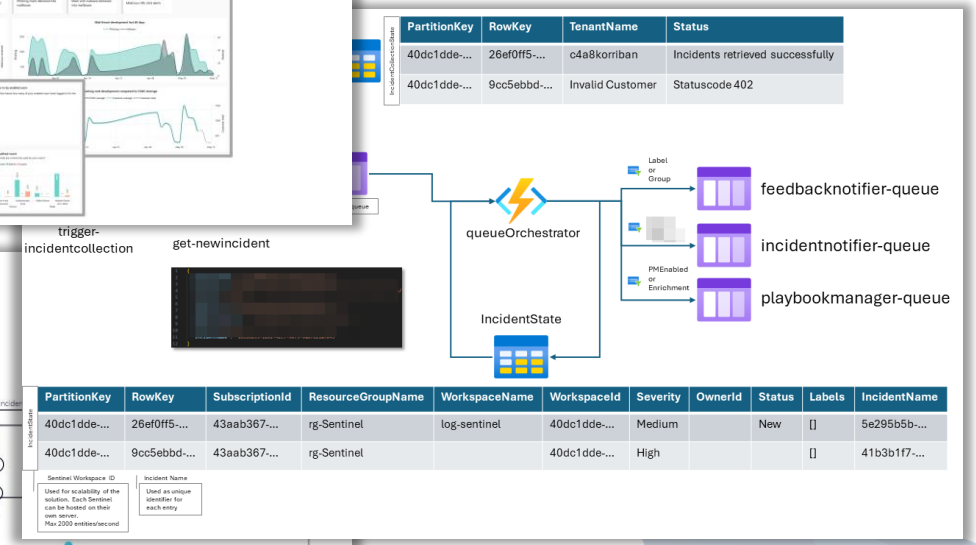
@fabian_bader

Am I done now? No!









Q&A

15 minutes

@fabian_bader