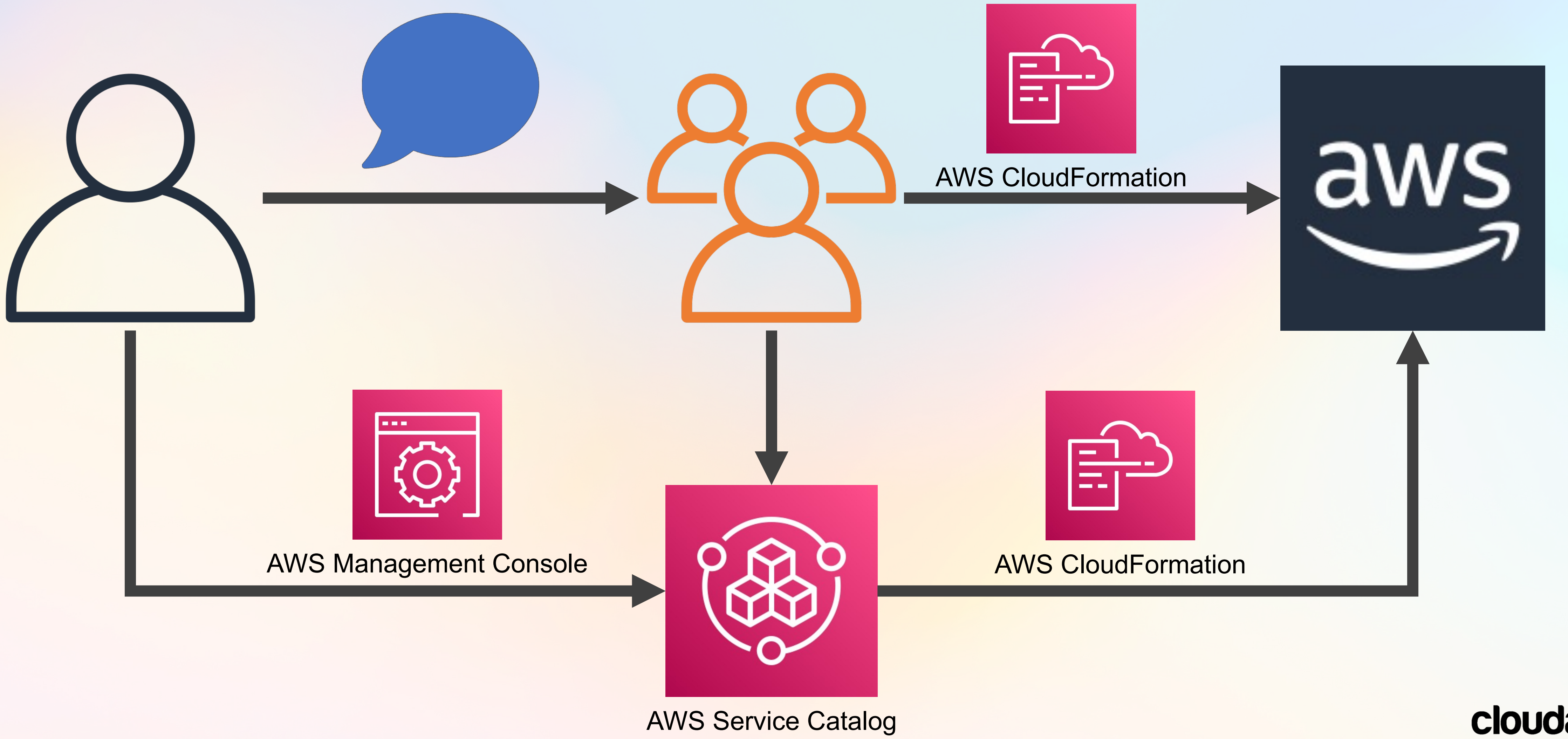
Policy as Code

Putting best practices in your repository

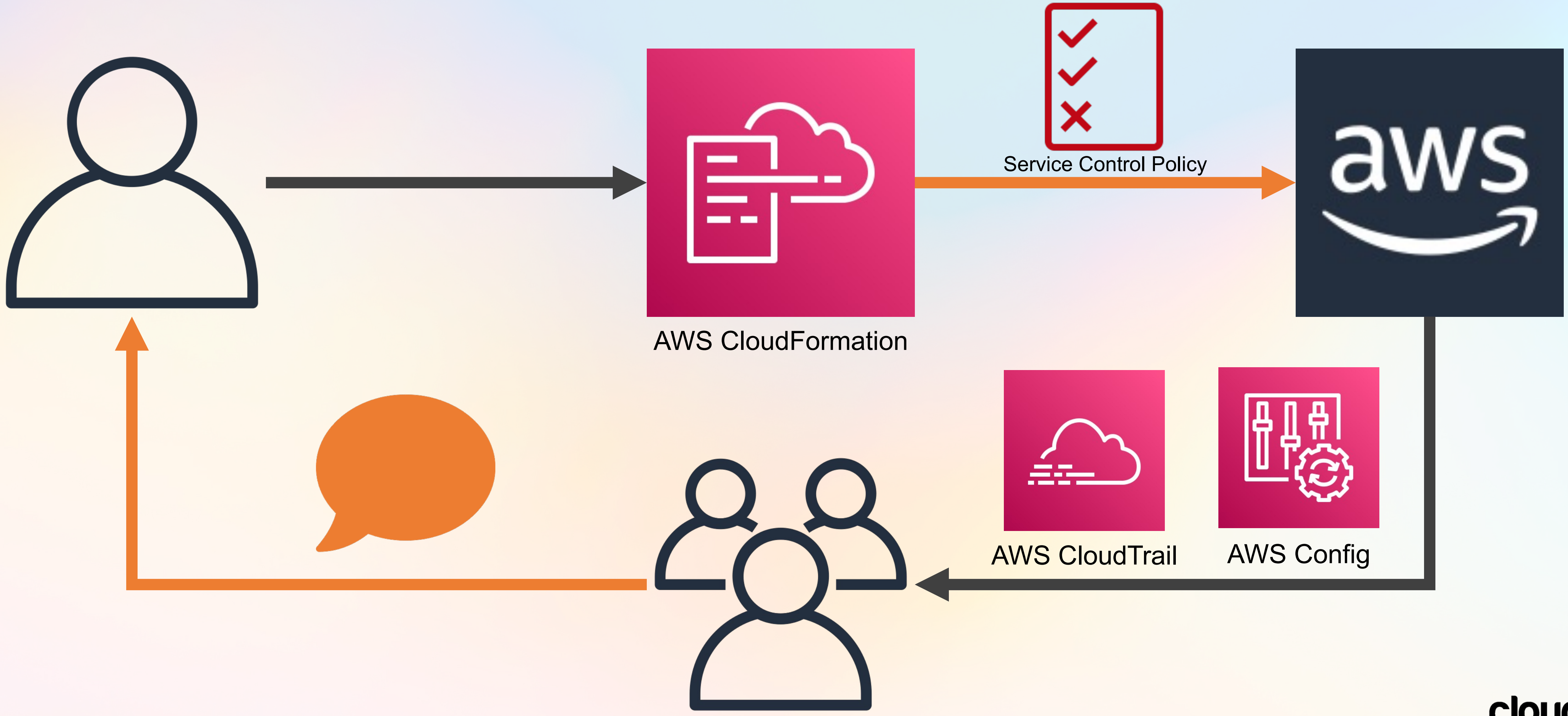
cloudar

What’s the problem?

Central teams create bottlenecks



GuardRails have long feedback loops

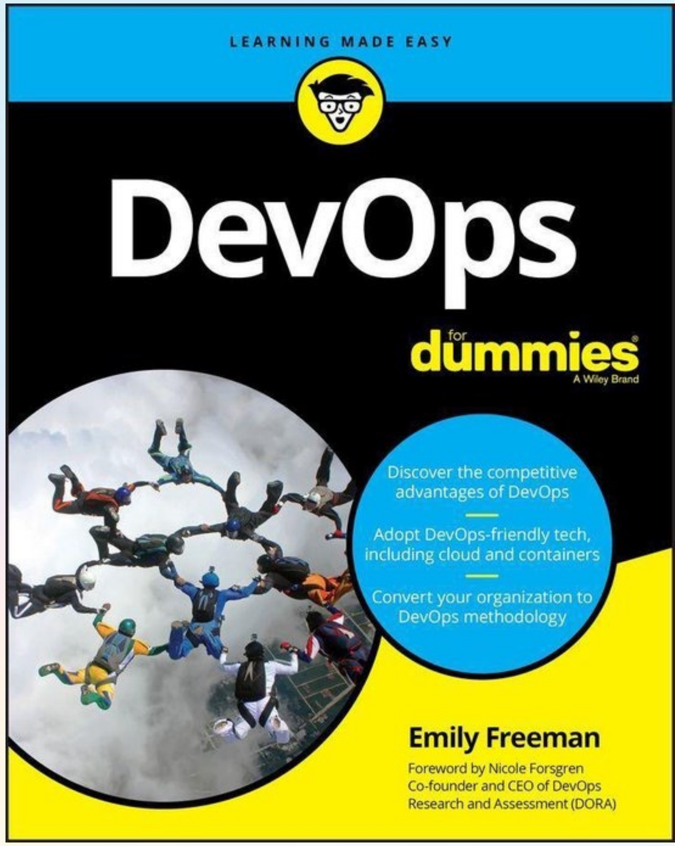


We need a new thing

or do we?

“Standardizing infrastructure configuration allows developers to stand up new infrastructure […] **without the assistance or approval of an operations specialist**, […] allowing them to take more ownership of their work.”

- Emily Freeman, DevOps for dummies

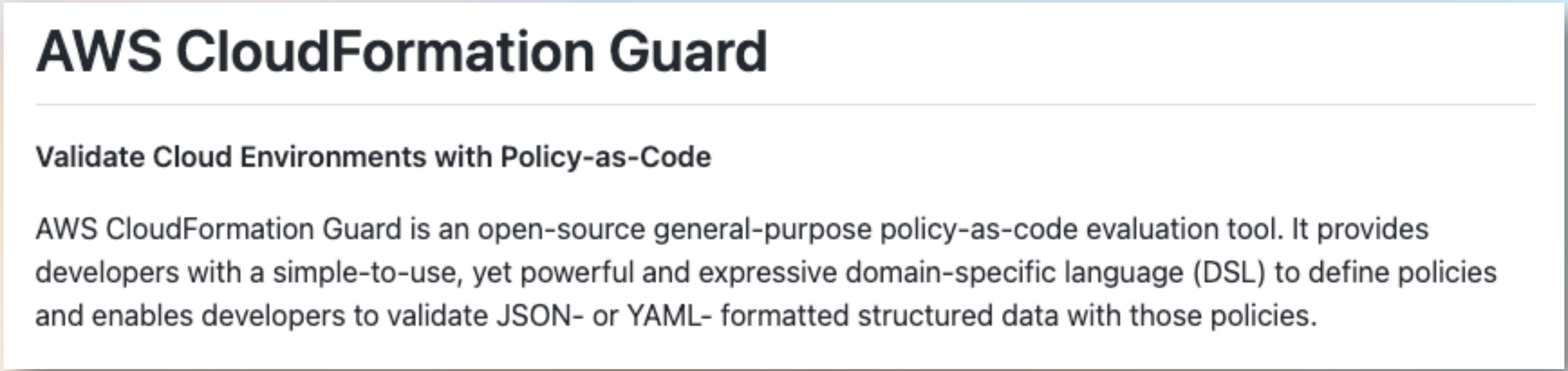

LEARNING MADE EASY
DevOps
for dummies
A Wiley Brand
Discover the competitive advantages of DevOps
Adopt DevOps-friendly tech, including cloud and containers
Convert your organization to DevOps methodology
Emily Freeman
Foreword by Nicole Forsgren
Co-founder and CEO of DevOps Research and Assessment (DORA)

cloudar

Everything is ~~awsome~~ **YAML**

Policy as Code

With (cfn-)guard


AWS CloudFormation Guard

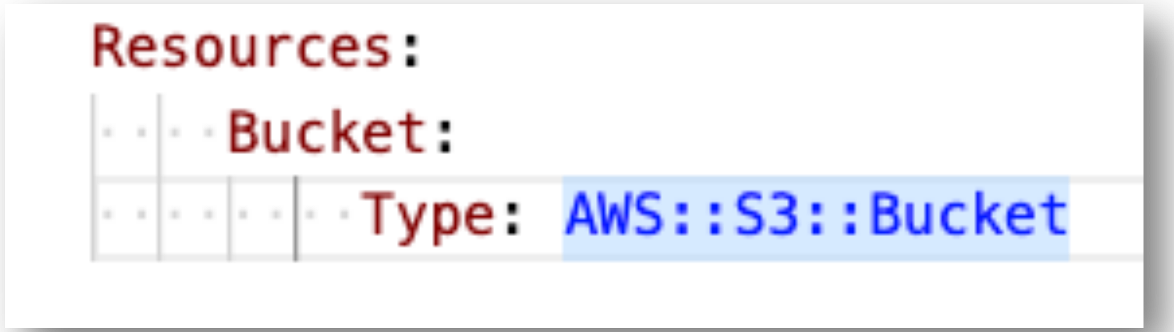
Validate Cloud Environments with Policy-as-Code

AWS CloudFormation Guard is an open-source general-purpose policy-as-code evaluation tool. It provides developers with a simple-to-use, yet powerful and expressive domain-specific language (DSL) to define policies and enables developers to validate JSON- or YAML- formatted structured data with those policies.

https://github.com/aws-cloudformation/cloudformation-guard

```
let s3_buckets = Resources.*[ Type == 'AWS::S3::Bucket']

rule S3_BUCKET_PUBLIC_ACCESS_PROHIBITED when %s3_buckets !empty {
  %s3_buckets.Properties.PublicAccessBlockConfiguration exists
  %s3_buckets.Properties.PublicAccessBlockConfiguration.BlockPublicAcls == true
  %s3_buckets.Properties.PublicAccessBlockConfiguration.BlockPublicPolicy == true
  %s3_buckets.Properties.PublicAccessBlockConfiguration.IgnorePublicAcls == true
  %s3_buckets.Properties.PublicAccessBlockConfiguration.RestrictPublicBuckets == true
  <<
    Violation: S3 Bucket Public Access controls need to be restricted.
    Fix: Set S3 Bucket PublicAccessBlockConfiguration properties for BlockPublicAcls,
    BlockPublicPolicy, IgnorePublicAcls, RestrictPublicBuckets parameters to true.
  >>
}
```



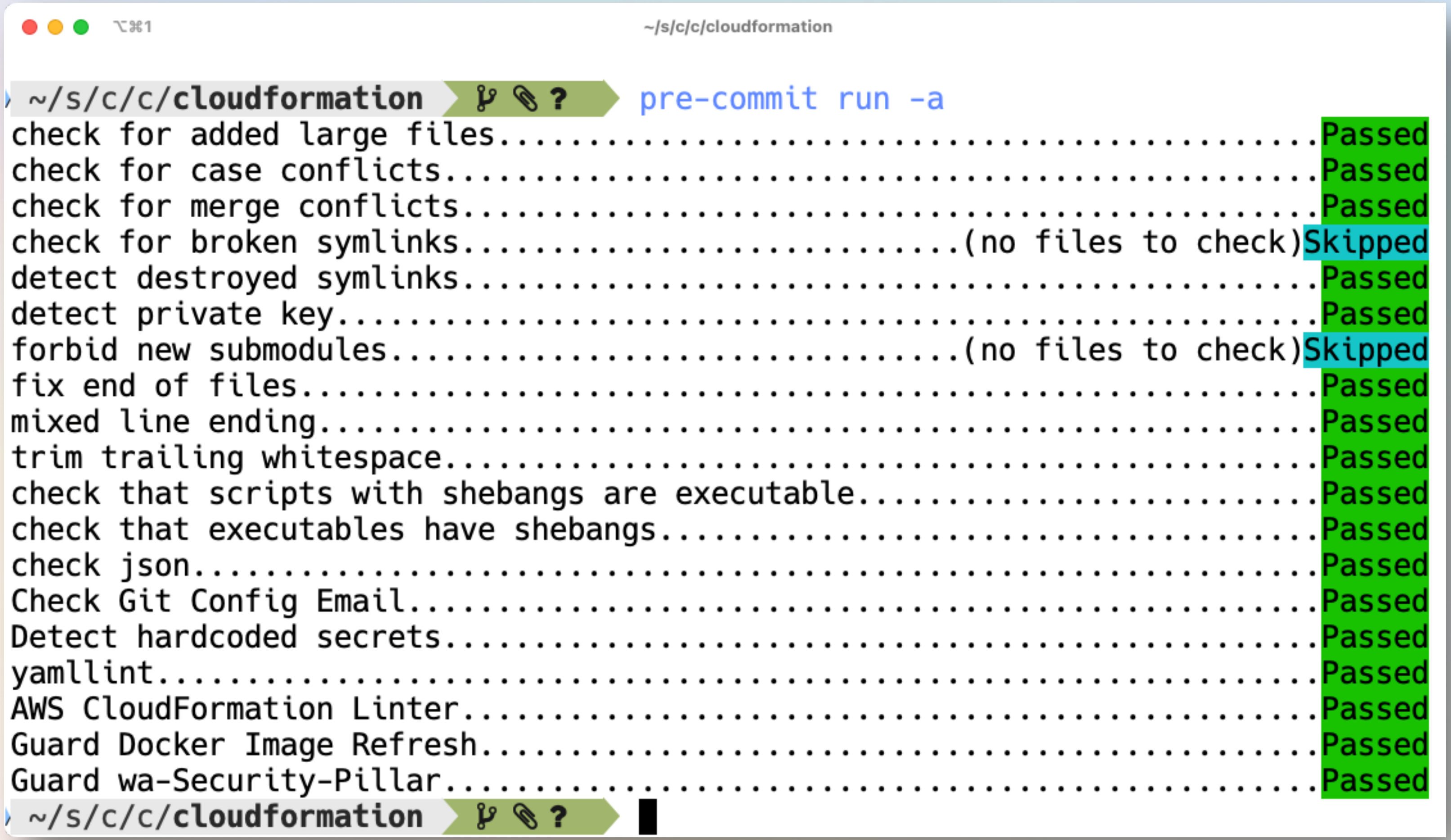
```
Resources:
    Bucket:
        Type: AWS::S3::Bucket
```

Creating a short feedback loop

Source code

pre-commit

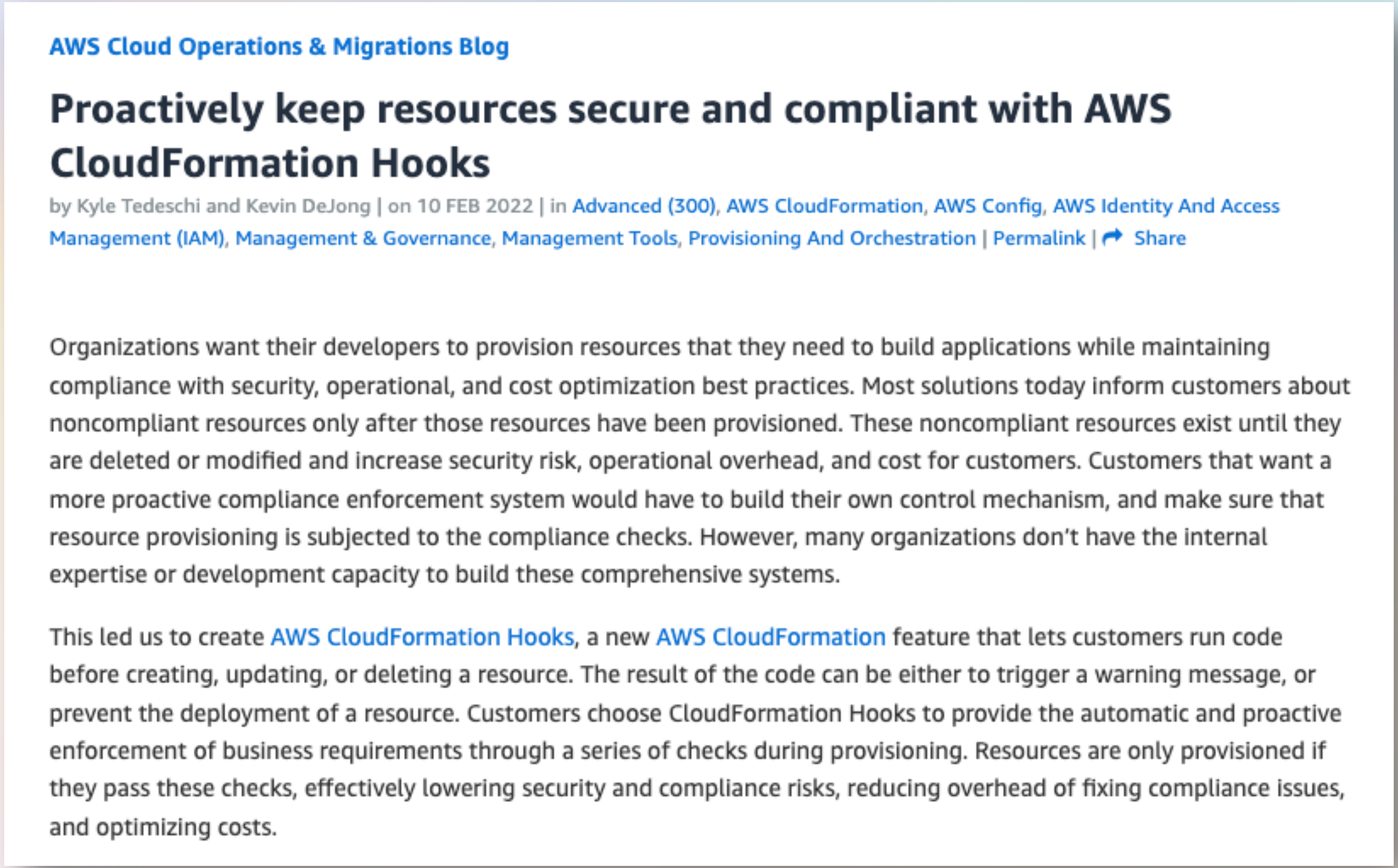
AWS CodePipeline



```
~/s/c/c/cloudformation  pre-commit run -a
check for added large files..............................................Passed
check for case conflicts.................................................Passed
check for merge conflicts................................................Passed
check for broken symlinks............................(no files to check)Skipped
detect destroyed symlinks................................................Passed
detect private key.......................................................Passed
forbid new submodules................................(no files to check)Skipped
fix end of files.........................................................Passed
mixed line ending........................................................Passed
trim trailing whitespace.................................................Passed
check that scripts with shebangs are executable..........................Passed
check that executables have shebangs.....................................Passed
check json...............................................................Passed
Check Git Config Email...................................................Passed
Detect hardcoded secrets.................................................Passed
yamllint.................................................................Passed
AWS CloudFormation Linter................................................Passed
Guard Docker Image Refresh...............................................Passed
Guard wa-Security-Pillar.................................................Passed
~/s/c/c/cloudformation
```

cloudar

Reusing Rules - Preventive



AWS Cloud Operations & Migrations Blog

Proactively keep resources secure and compliant with AWS CloudFormation Hooks

by Kyle Tedeschi and Kevin DeJong | on 10 FEB 2022 | in Advanced (300), AWS CloudFormation, AWS Config, AWS Identity And Access Management (IAM), Management & Governance, Management Tools, Provisioning And Orchestration | Permalink | Share

Organizations want their developers to provision resources that they need to build applications while maintaining compliance with security, operational, and cost optimization best practices. Most solutions today inform customers about noncompliant resources only after those resources have been provisioned. These noncompliant resources exist until they are deleted or modified and increase security risk, operational overhead, and cost for customers. Customers that want a more proactive compliance enforcement system would have to build their own control mechanism, and make sure that resource provisioning is subjected to the compliance checks. However, many organizations don't have the internal expertise or development capacity to build these comprehensive systems.

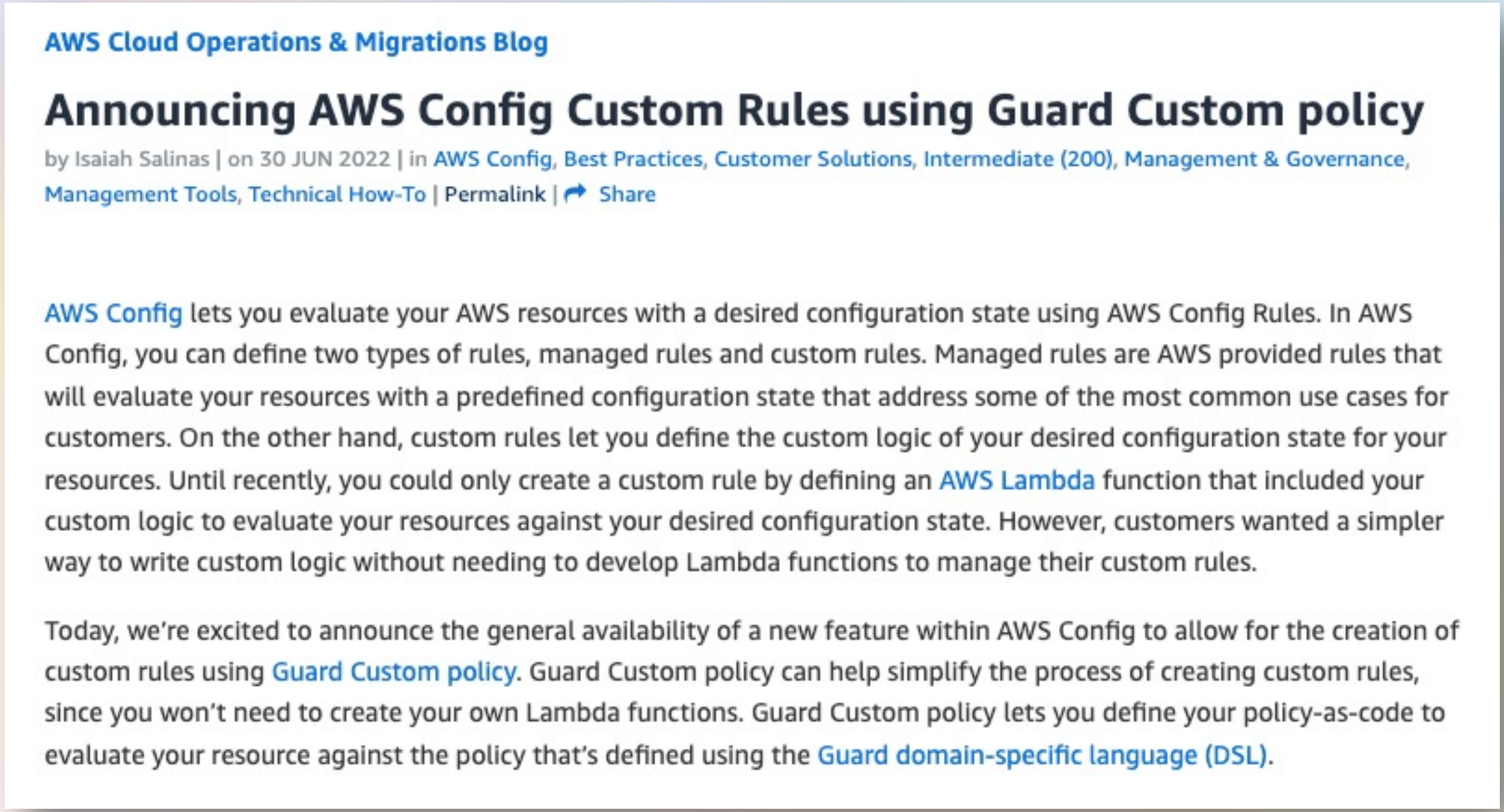
This led us to create AWS CloudFormation Hooks, a new AWS CloudFormation feature that lets customers run code before creating, updating, or deleting a resource. The result of the code can be either to trigger a warning message, or prevent the deployment of a resource. Customers choose CloudFormation Hooks to provide the automatic and proactive enforcement of business requirements through a series of checks during provisioning. Resources are only provisioned if they pass these checks, effectively lowering security and compliance risks, reducing overhead of fixing compliance issues, and optimizing costs.

https://aws.amazon.com/blogs/mt/proactively-keep-resources-secure-and-compliant-with-aws-cloudformation-hooks/

AWS CloudFormation

cloudar

Reusing Rules - Detective



AWS Cloud Operations & Migrations Blog

Announcing AWS Config Custom Rules using Guard Custom policy

by Isaiah Salinas | on 30 JUN 2022 | in AWS Config, Best Practices, Customer Solutions, Intermediate (200), Management & Governance, Management Tools, Technical How-To | Permalink | Share

AWS Config lets you evaluate your AWS resources with a desired configuration state using AWS Config Rules. In AWS Config, you can define two types of rules, managed rules and custom rules. Managed rules are AWS provided rules that will evaluate your resources with a predefined configuration state that address some of the most common use cases for customers. On the other hand, custom rules let you define the custom logic of your desired configuration state for your resources. Until recently, you could only create a custom rule by defining an AWS Lambda function that included your custom logic to evaluate your resources against your desired configuration state. However, customers wanted a simpler way to write custom logic without needing to develop Lambda functions to manage their custom rules.

Today, we're excited to announce the general availability of a new feature within AWS Config to allow for the creation of custom rules using Guard Custom policy. Guard Custom policy can help simplify the process of creating custom rules, since you won't need to create your own Lambda functions. Guard Custom policy lets you define your policy-as-code to evaluate your resource against the policy that's defined using the Guard domain-specific language (DSL).

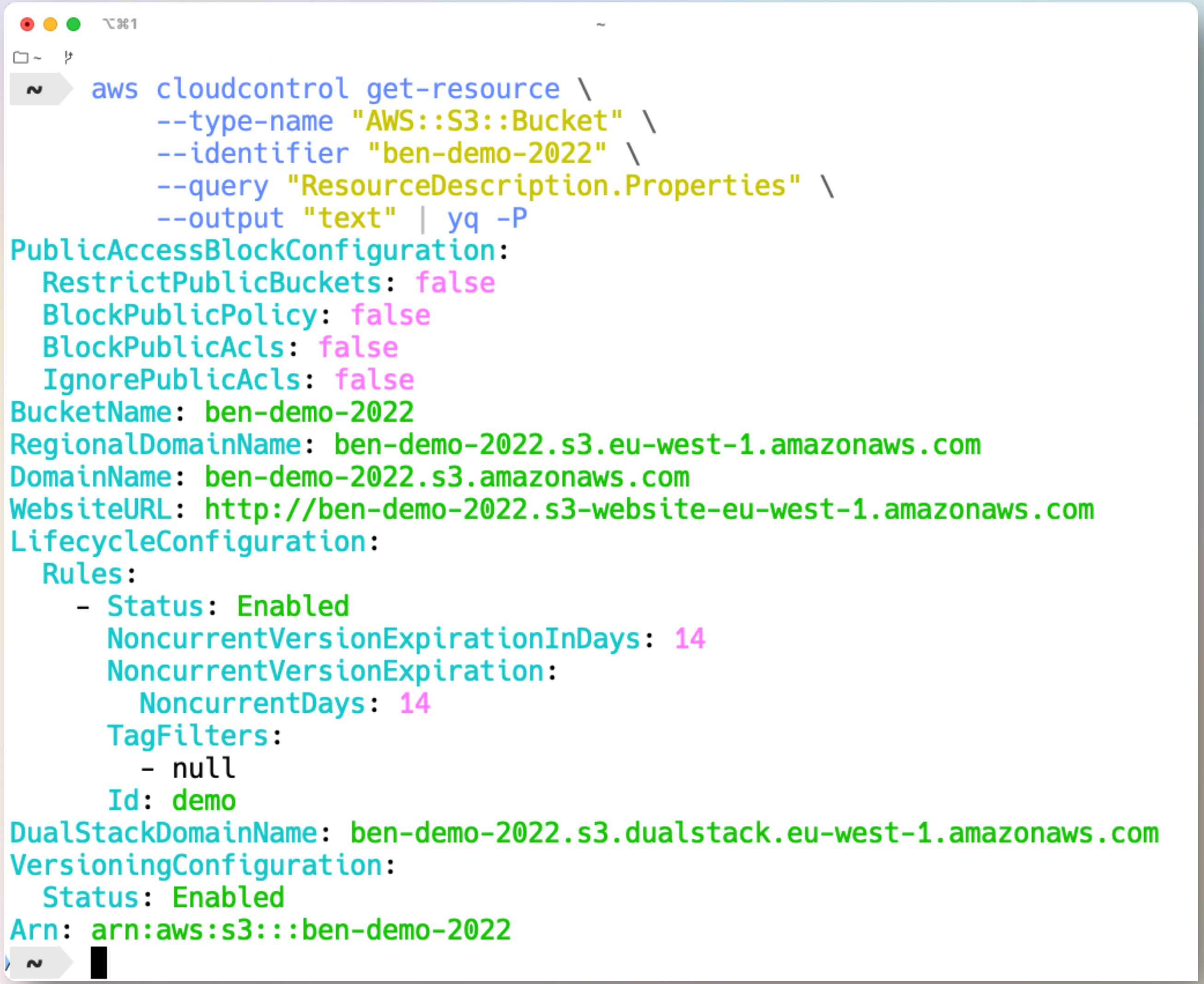
https://aws.amazon.com/blogs/mt/announcing-aws-config-custom-rules-using-guard-custom-policy/

AWS CloudFormation

AWS Config

cloudar

Reusing Rules - Detective



```
~ aws cloudcontrol get-resource \
        --type-name "AWS::S3::Bucket" \
        --identifier "ben-demo-2022" \
        --query "ResourceDescription.Properties" \
        --output "text" | yq -P
PublicAccessBlockConfiguration:
  RestrictPublicBuckets: false
  BlockPublicPolicy: false
  BlockPublicAcls: false
  IgnorePublicAcls: false
BucketName: ben-demo-2022
RegionalDomainName: ben-demo-2022.s3.eu-west-1.amazonaws.com
DomainName: ben-demo-2022.s3.amazonaws.com
WebsiteURL: http://ben-demo-2022.s3-website-eu-west-1.amazonaws.com
LifecycleConfiguration:
  Rules:
    - Status: Enabled
      NoncurrentVersionExpirationInDays: 14
      NoncurrentVersionExpiration:
        NoncurrentDays: 14
      TagFilters:
        - null
      Id: demo
DualStackDomainName: ben-demo-2022.s3.dualstack.eu-west-1.amazonaws.com
VersioningConfiguration:
  Status: Enabled
Arn: arn:aws:s3:::ben-demo-2022
~
```

AWS CloudFormation

AWS Config

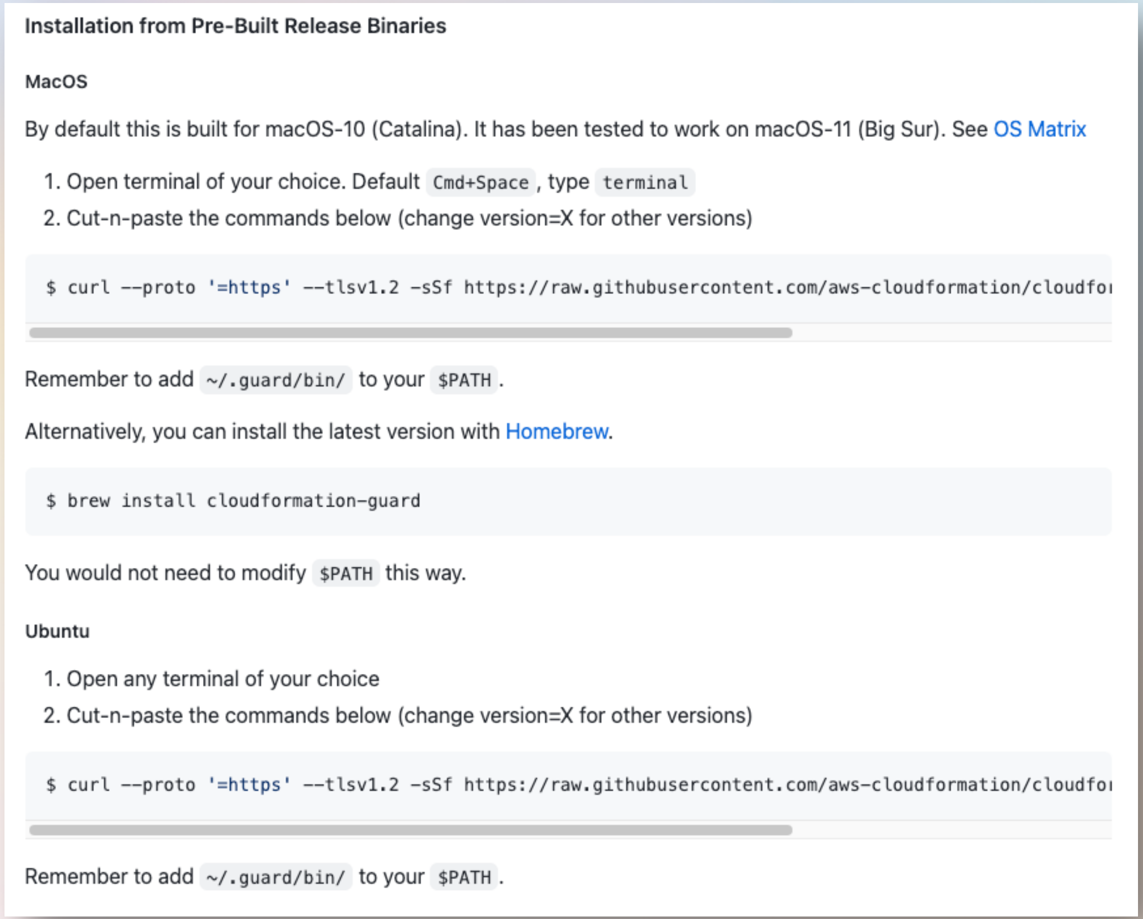
AWS Cloud Control API

cloudar

How to get started

and related problems

Installation is harder than it should be



Installation from Pre-Built Release Binaries

MacOS

By default this is built for macOS-10 (Catalina). It has been tested to work on macOS-11 (Big Sur). See OS Matrix

1. Open terminal of your choice. Default `Cmd+Space`, type `terminal`
2. Cut-n-paste the commands below (change version=X for other versions)

```
$ curl --proto '=https' --tlsv1.2 -sSf https://raw.githubusercontent.com/aws-cloudformation/cloudfo
```

Remember to add `~/.guard/bin/` to your `$PATH`.

Alternatively, you can install the latest version with Homebrew.

```
$ brew install cloudformation-guard
```

You would not need to modify `$PATH` this way.

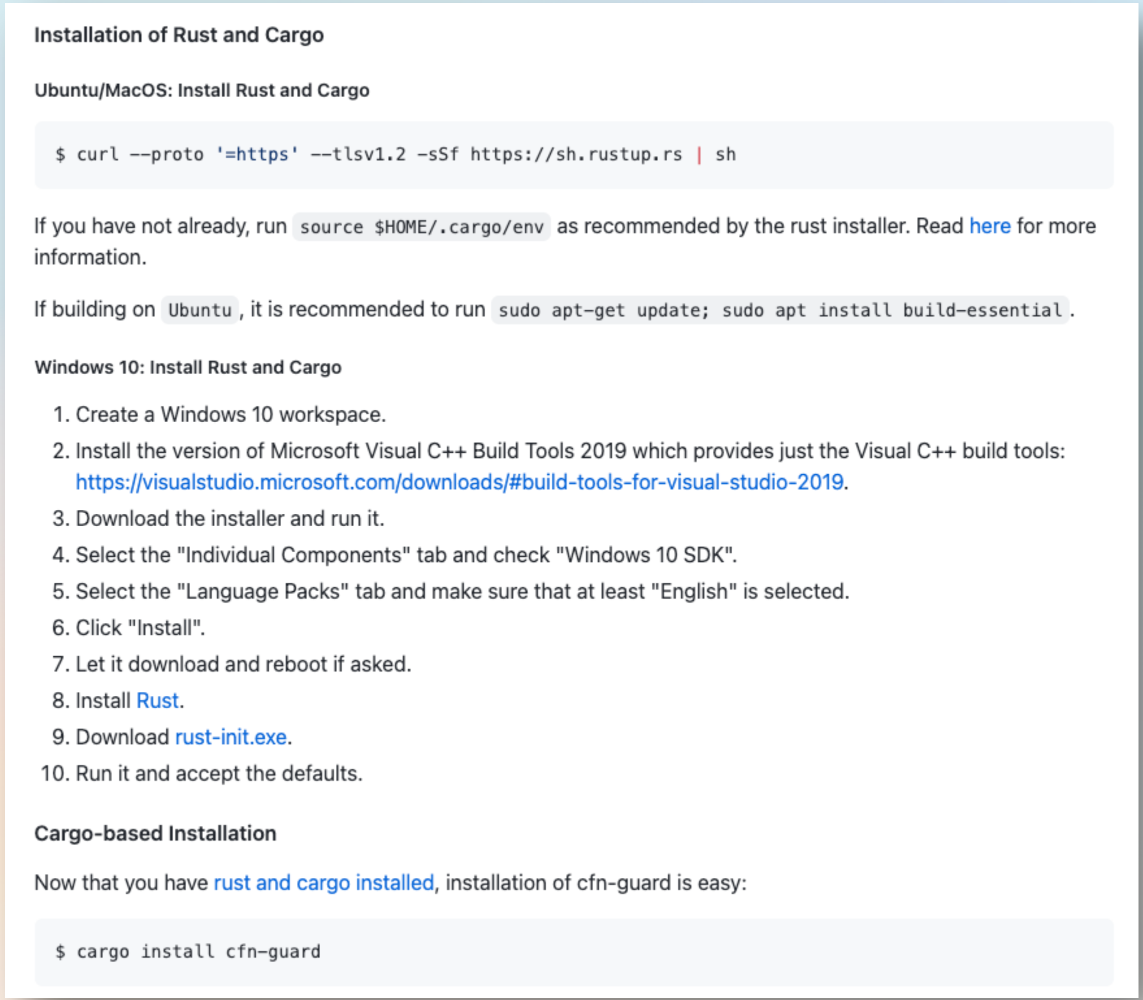
Ubuntu

1. Open any terminal of your choice
2. Cut-n-paste the commands below (change version=X for other versions)

```
$ curl --proto '=https' --tlsv1.2 -sSf https://raw.githubusercontent.com/aws-cloudformation/cloudfo
```

Remember to add `~/.guard/bin/` to your `$PATH`.

https://github.com/aws-cloudformation/cloudformation-guard#installation



Installation of Rust and Cargo

Ubuntu/MacOS: Install Rust and Cargo

```
$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

If you have not already, run `source $HOME/.cargo/env` as recommended by the rust installer. Read here for more information.

If building on `Ubuntu`, it is recommended to run `sudo apt-get update; sudo apt install build-essential`.

Windows 10: Install Rust and Cargo

1. Create a Windows 10 workspace.
2. Install the version of Microsoft Visual C++ Build Tools 2019 which provides just the Visual C++ build tools: https://visualstudio.microsoft.com/downloads/#build-tools-for-visual-studio-2019.
3. Download the installer and run it.
4. Select the "Individual Components" tab and check "Windows 10 SDK".
5. Select the "Language Packs" tab and make sure that at least "English" is selected.
6. Click "Install".
7. Let it download and reboot if asked.
8. Install Rust.
9. Download rust-init.exe.
10. Run it and accept the defaults.

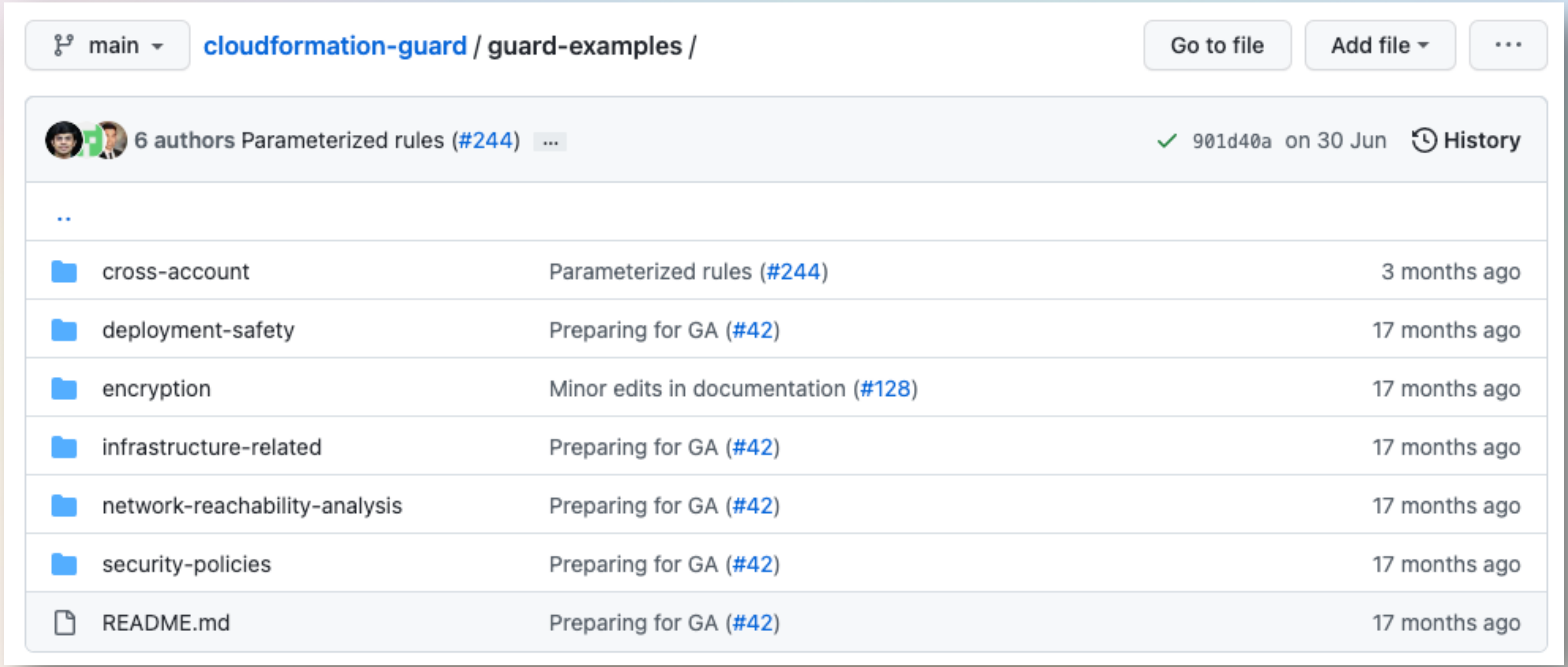
Cargo-based Installation

Now that you have rust and cargo installed, installation of cfn-guard is easy:

```
$ cargo install cfn-guard
```

cloudar

How to find rules?


main cloudformation-guard / guard-examples /

Go to file | Add file | ...

6 authors Parameterized rules (#244) ... ✓ 901d40a on 30 Jun History

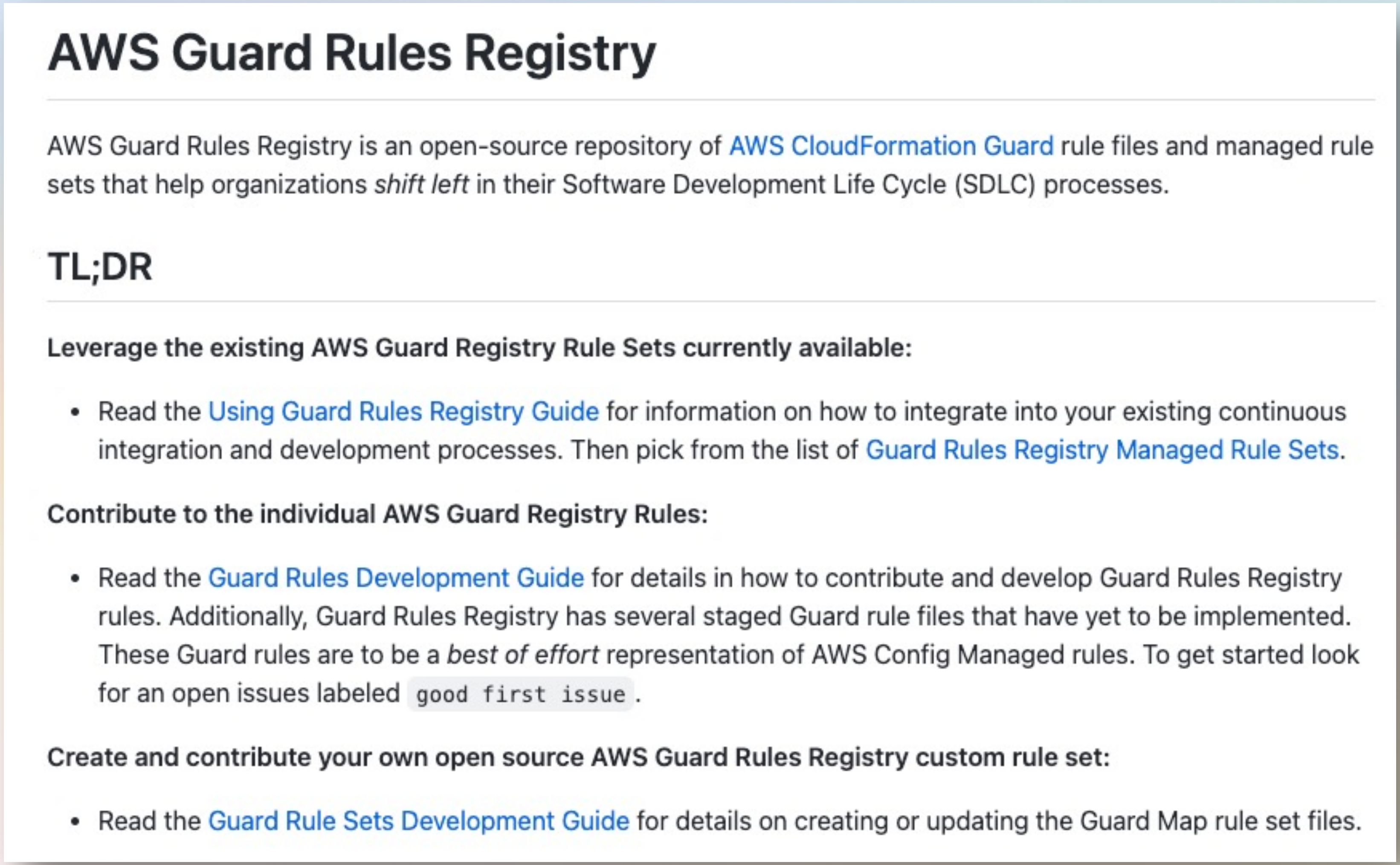
..		
cross-account	Parameterized rules (#244)	3 months ago
deployment-safety	Preparing for GA (#42)	17 months ago
encryption	Minor edits in documentation (#128)	17 months ago
infrastructure-related	Preparing for GA (#42)	17 months ago
network-reachability-analysis	Preparing for GA (#42)	17 months ago
security-policies	Preparing for GA (#42)	17 months ago
README.md	Preparing for GA (#42)	17 months ago

https://github.com/aws-cloudformation/cloudformation-guard/tree/main/guard-examples

AWS Guard Rule Registry

A decent first step

cloudar



AWS Guard Rules Registry

AWS Guard Rules Registry is an open-source repository of AWS CloudFormation Guard rule files and managed rule sets that help organizations *shift left* in their Software Development Life Cycle (SDLC) processes.

TL;DR

Leverage the existing AWS Guard Registry Rule Sets currently available:

- Read the Using Guard Rules Registry Guide for information on how to integrate into your existing continuous integration and development processes. Then pick from the list of Guard Rules Registry Managed Rule Sets.

Contribute to the individual AWS Guard Registry Rules:

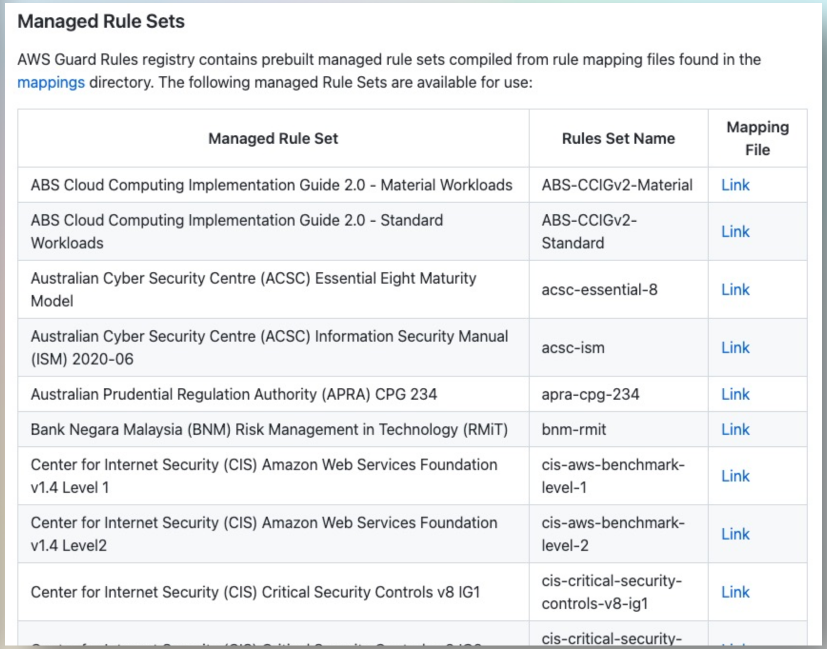
- Read the Guard Rules Development Guide for details in how to contribute and develop Guard Rules Registry rules. Additionally, Guard Rules Registry has several staged Guard rule files that have yet to be implemented. These Guard rules are to be a *best of effort* representation of AWS Config Managed rules. To get started look for an open issues labeled `good first issue`.

Create and contribute your own open source AWS Guard Rules Registry custom rule set:

- Read the Guard Rule Sets Development Guide for details on creating or updating the Guard Map rule set files.

Features:

- Pre-build rules and rule sets
- Docker Image for use in ci/cd
- Individual rule supression



Managed Rule Sets

AWS Guard Rules registry contains prebuilt managed rule sets compiled from rule mapping files found in the mappings directory. The following managed Rule Sets are available for use:

Managed Rule Set	Rules Set Name	Mapping File
ABS Cloud Computing Implementation Guide 2.0 - Material Workloads	ABS-CCIGv2-Material	Link
ABS Cloud Computing Implementation Guide 2.0 - Standard Workloads	ABS-CCIGv2-Standard	Link
Australian Cyber Security Centre (ACSC) Essential Eight Maturity Model	acsc-essential-8	Link
Australian Cyber Security Centre (ACSC) Information Security Manual (ISM) 2020-06	acsc-ism	Link
Australian Prudential Regulation Authority (APRA) CPG 234	apra-cpg-234	Link
Bank Negara Malaysia (BNM) Risk Management in Technology (RMiT)	bnm-rmit	Link
Center for Internet Security (CIS) Amazon Web Services Foundation v1.4 Level 1	cis-aws-benchmark-level-1	Link
Center for Internet Security (CIS) Amazon Web Services Foundation v1.4 Level2	cis-aws-benchmark-level-2	Link
Center for Internet Security (CIS) Critical Security Controls v8 IG1	cis-critical-security-controls-v8-ig1	Link
[illegible]	cis-critical-security-	[illegible]

https://github.com/aws-cloudformation/aws-guard-rules-registry

Challenges

Challenges (1/2)

- Where do you store your rules?
- Combining custom rules with Open Source rules
  - Running cfn-guard from docker with local rule file
  - Building custom docker images
  - Running natively
- Integrating with CloudFormation
  - Limited examples for hooks
    - https://github.com/aws-cloudformation/community-registry-extensions/tree/main/hooks/S3_BucketVersioningEnabled
  - Per-account / per-region setting

cloudar

Challenges (2/2)

- Integrating with other AWS Services
  - Config does not support all resources
  - CloudControl API has it’s own limitations
- Integrating with Posture Management Vendors
  - Same rules in different tools

Thank you!

Ben Bridts
ben@cloudar.be
@BenBridts | @WeAreCloudar
www.cloudar.be

aws partner network

Premier
Consulting Partner

- Solution Provider
- Public Sector Partner
- MSP Partner
- Immersion Day Partner
- DevOps Competency
- Migration Competency
- Government Competency
- Well Architected
- AWS Lambda

cloudar