Google Summer Of Code

Proposal

Improve The Usability Of cert-manager On Multiple Cloud Providers

By Arsh Sharma

Mentors: Jake Sanders (@jakexks), team-cert-manager@Jetstack

Table Of Contents

Table Of Contents 1

Abstract 2

Background 2

Project Details 3

- Spinning Kubernetes Instances In Different Clouds 3
- Adding Cloud Provider Specific Challenge Tests 5
- Running The Test Suite Periodically 6
- Documenting And Fixing Issues Found After Tests 6
- Fixing Other Bugs In Codebase 6

Schedule Of Deliverables 7

- Milestones 7
- Timeline 7
- Time Commitments 8

General Notes 9

About Me 9

- Personal Information and Contact Details 9
- Why Me? 10

Abstract

cert-manager is a Kubernetes add-on used extensively to automate the management and issuance of certificates from various issuers.

This project aims to capture the problems users often run into when deploying cert-manager on managed Kubernetes solutions on different clouds with the help of improved e2e testing.

Background

The most popular use case of cert-manager is with publicly trusted ACME certificates from different sources like Let's Encrypt. For the ACME Certificate Authority to verify that a client owns the domain, the client has to complete challenges.

When deploying cert-manager on different clouds users often run into problems because of there being slight differences in the way different cloud providers work. In the case of ACME Certificate Authorities this could cause unexpected behaviours while solving the challenges leading to non-desired results. Apart from this users also sometimes end up configuring things in a way which causes cert-manager to not function properly.

Past cases where this has happened:

1. https://github.com/jetstack/cert-manager/issues/3549

This project aims to improve the overall user experience of deploying cert-manager on different cloud providers by finding out and fixing the scenarios talked about above with the help of e2e tests.

Project Details

The following list defines the tasks with the proposed solutions that will be the basis for work during the project duration.

Spinning Up Kubernetes Instances In Different Clouds

Goals:

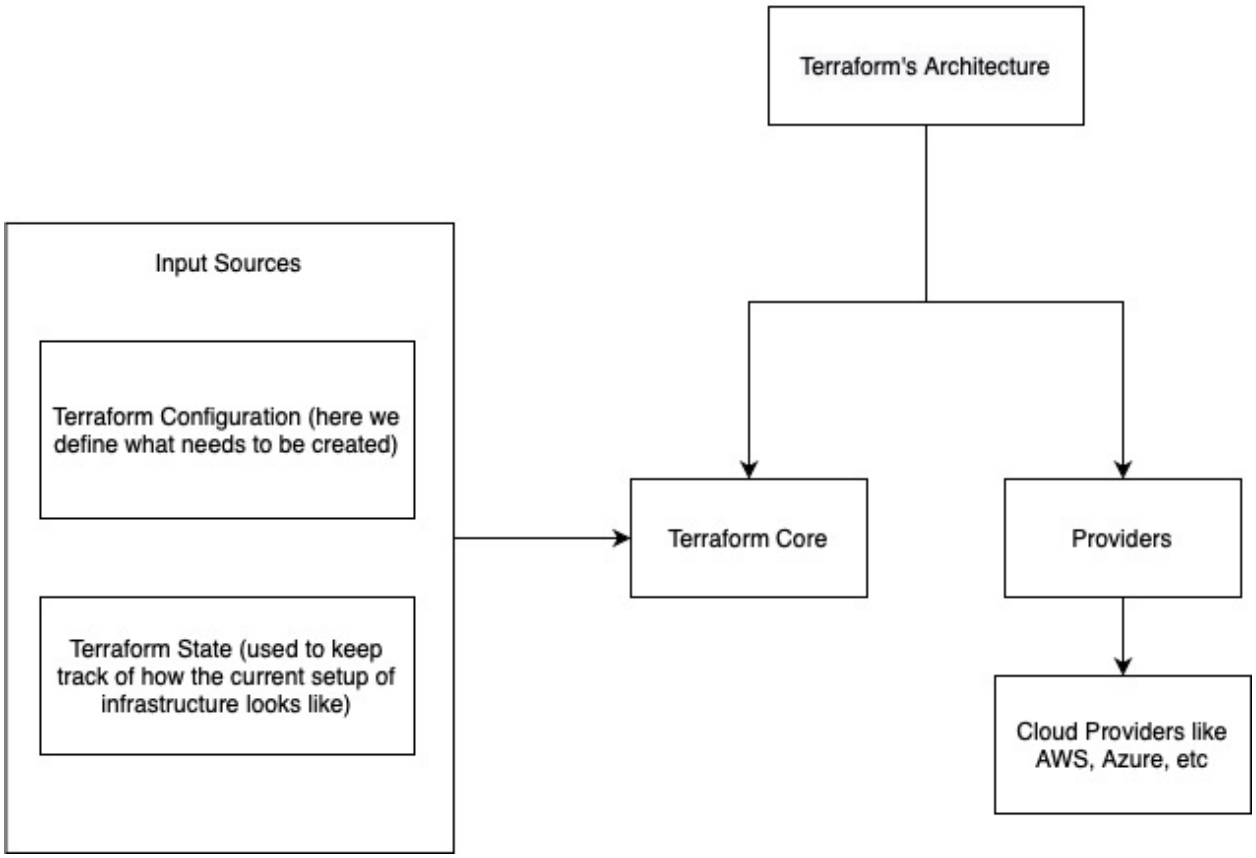
- Create infrastructure which uses different cloud providers' managed Kubernetes solutions, which will be later used to run the e2e tests.

Proposed Solution:

Terraform will be used for this. Terraform allows us to automate and manage our infrastructure, platform and the services that will run on that platform.

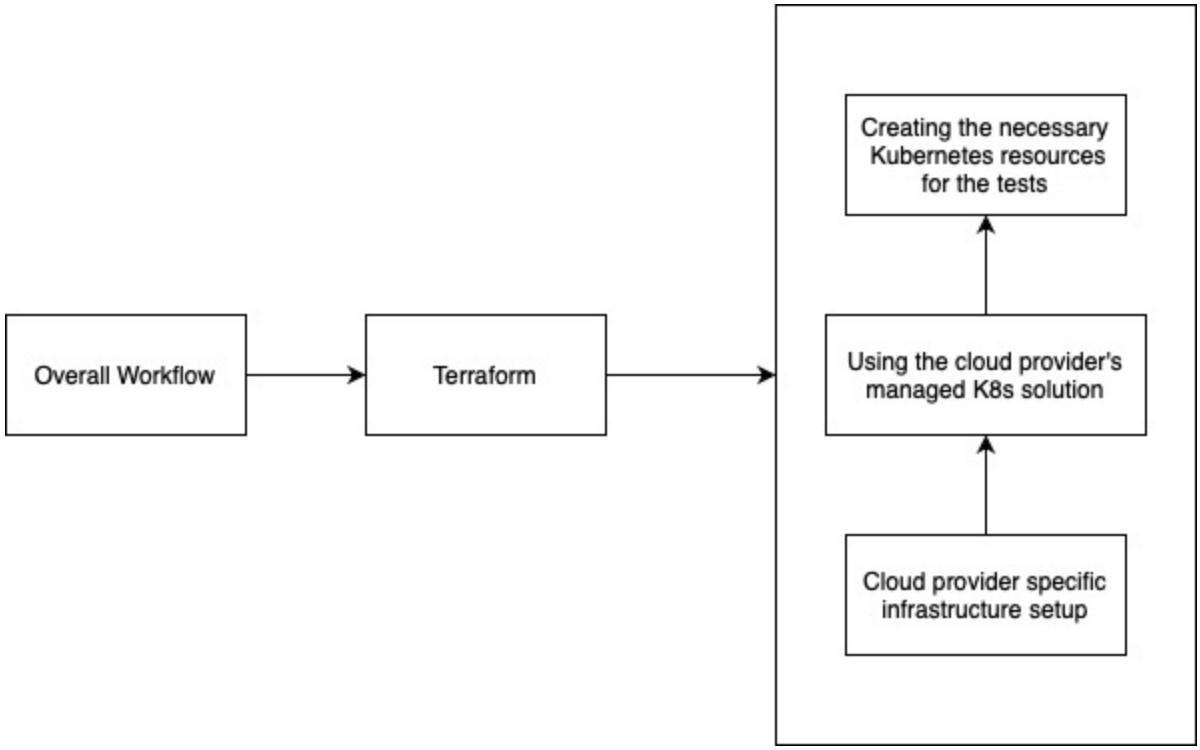
Terraform's architecture is made up of two main components. The first of the two is Terraform's core which uses two input sources: Terraform configuration (which we write to define what needs to be created) and Terraform state (current state of the infrastructure). The Terraform core then works towards matching the current state to the desired one which we described.

The second part of Terraform's architecture is providers for specific technologies like AWS, Azure, Kubernetes etc. Each provider gives Terraform access to its resources for example through the AWS provider we can get access to AWS resources like EC2 instances, AWS Users, etc.



Overview of the architecture of Terraform

Terraform will be used to first create an infrastructure for a specific cloud provider (AWS, Azure, GCP, etc) using its managed Kubernetes solution (for example EKS in case of AWS). Then the required components (ingresses, services, etc) for testing cert-manager will be created using the cloud provider's managed Kubernetes solution.



Workflow which will be followed

Adding Cloud Provider Specific Challenge Tests

Goals:

- Add tests to solve the ACME Certificate Authorities' challenges for each specific cloud provider.

Proposed Solution:

e2e tests are present in the cert-manager repository. These tests will be integrated with the infrastructure which is created with Terraform. The goal here would be to solve HTTP-01 and DNS-01 challenges by using the cloud provider specific implementations of DNS and LoadBalancers. Once these tests are confirmed to be working they will be extended to cross-cloud DNS challenges: K8s hosted in one cloud, DNS provided by another.

Running The Test Suite Periodically

Goals:

- Ensuring that the tests are run periodically so that we can capture any new bugs which pop up.

Proposed Solution:

Once the test suite is completed, the existing periodicals configuration would be extended to include the test suite.

Documenting And Fixing Issues Found After Tests

Goals:

- Document test results and bugs.
- Fix bugs found (or provide workarounds in case fix not possible).

Proposed Solution:

The results found from running the tests would be properly analyzed. Any bugs which can be fixed, will be. Others will be well documented along with workarounds for them on the cert-manager website.

Fixing Other Bugs In Codebase

Goals:

- Fix bugs/issues in codebase

Proposed Solution:

If everything completes timely the remaining time would be utilized to fix existing bugs in the codebase. These would include issues logged on the cert-manager repository and any improvement in the code quality of the test suite. This is something I plan on continuing post the GSoC period also :)

Schedule Of Deliverables

Milestones

There are three major milestones:

- Setting up cloud providers for testing using Terraform.
- Integrating and automating the tests.
- Fixing bugs and documenting workarounds based on test results.

Timeline

Keeping these milestones in mind the following timeline will be followed.

Dates	Tasks
	Community Bonding Period Begins
May 17 - June 7	Finalizing the exact approach with mentors, Setting up the development environment, Preparing a draft infrastructure which will be used as a template for all other cloud providers.
	Community Bonding Period Ends
June 7 - June 16	Setting up infrastructure required for EKS (AWS) using Terraform. Creating the required resources in Kubernetes for the tests.
June 16 - June 28	Integrating tests for solving HTTP-01 and DNS-01 challenges for AWS.
June 28 - July 10	Extending the test suite to Azure
July 10 - July 16	This period has been kept for completing remaining work (if any). If everything proceeds smoothly till this then this period will be utilized to extend the test

	suite to GCP.
	First Phase Completed
July 16 - July 23	Finalize changes recommended by mentors and the community in the test suite (if any). Work on running the test suite periodically.
July 23 - August 6	Document test results, find fixes for bugs and provide workarounds for the bugs which can't be fixed (if any).
August 6 - August 16	If everything is completed timely then this time will be used to extend the approach to other cloud providers and/or fixing other bugs in the codebase.
August 16 - August 23	The final week has been left free for completing any remaining work (if any). This provides sufficient cushion for making sure that the timeline is followed. If everything gets completed smoothly before this period then this will be utilized for solving issues in the cert-manager repository.
	Program Ends

Time Commitments

- The above timeline is tentative and is for providing a rough idea of the planned project work. Full efforts will be made to the stick to it. A much more detailed schedule will be shared with the mentors during the community bonding period after finalizing the exact approach for the project.
- The expected time commitment for GSoC this time is 18 hours/week. I have no prior commitments during the coding period so will easily be able to contribute for the necessary time (even more if need be).
- My college summer break starts from May 13 and classes resume from August 10 (tentative). If the classes do not resume online then my contributions might be slightly limited post college reopening. But if that happens I will complete my project work before this date by putting in extra time.

- I am currently working as a Cloud Native Computing Foundation Mentee under the Linux Foundation Mentorship Program which ends on May 31. This coincides **partially** with the community bonding period. However this won't be a problem at all since I will be free much before the actual GSoC coding period starts (June 7). Moreover the majority of my mentorship work will be done by May 17 so I will be able to actively contribute during the community bonding period also.

General Notes

I firmly believe that communication is one of the most important aspects of open source programs like GSoC. To make sure that the project status is communicated properly, I will be undertaking the following steps:

- Participating in cert-manager daily standups and bi-weekly community meetings.
- Publishing a bi-weekly blog post detailing the work done, problems faced and how they were resolved.
- Maintaining a google doc with **daily** updates. Even if I'm stuck on some issue I will be writing that down in the doc. I believe this is a very effective way of not only maintaining accountability but also ensuring smooth asynchronous communication.
- Contacting the mentors daily to keep them in the loop of the progress of the project.

About Me

Personal Information and Contact Details

Name: Arsh Sharma
Email: arshsharma461@gmail.com
Slack nickname: Arsh
GitHub Username: RinkiyaKeDad
University: Indian Institute of Technology (BHU), Varanasi

Timezone: GMT +05:30 (IST - Indian Standard Timezone)
Website: arshsharma.netlify.com
Twitter: @RinkiyaKeDad

Why Me?

I have been contributing to the cloud native ecosystem for quite some time now. I have experience writing code in Go. I'm also *currently* working as a CNCF intern where my project is to develop utilities to evaluate dependency updates to Kubernetes. I have previously also interned at Nirmata where I worked on a CNCF sandbox project: Kyverno which is a Kubernetes native policy management engine.

I have also looked at the cert-manager codebase and opened the following pull requests till now:

1. https://github.com/jetstack/cert-manager/pull/3771
2. https://github.com/jetstack/cert-manager/pull/3765
3. https://github.com/jetstack/cert-manager/pull/3815
4. https://github.com/jetstack/cert-manager/pull/3835

I will be working on more issues and sending more PRs in the coming weeks.

I love contributing to open source projects because of the amazing people I get to meet and learn from. I've been contributing to open source for about a year now and I'm familiar with version controls systems like Git. Some other past open source contributions:

1. https://github.com/kubernetes/k8s.io/pull/1789
2. https://github.com/kubernetes/kubernetes/pull/100792
3. https://github.com/aerogear/graphback/pull/2153
4. https://github.com/matrix-org/matrix-react-sdk/pull/5442
5. https://github.com/creativecommons/ccsearch-browser-extension/pull/310

I also love cloud native tech and writing Go code. Other than that I also like to write blogs related to the cloud native ecosystem.

Post GSoC Plans

I’m **not** applying for GSoC under any other organization this year since I am heart warmed by the incredibly welcoming folks I've found in the cert-manager team and I really want to be a part of it.

Post the GSoC period I would love to continue contributing by adding more features to not only the test suite but also to other parts of cert-manager. Once the project is completed, I’ll try to provide patches and bug fixes for it since I want to contribute to the project in some or another way. Apart from that, I’ll always be a part of the cert-manager community and will be following as well as contributing towards its development.