Litmus Chaos

Introduction

LitmusChaos is an open source Chaos Engineering platform that enables teams to identify weaknesses & potential outages in infrastructures by inducing chaos tests in a controlled way. Developers & SREs can practice Chaos Engineering with LitmusChaos as it is easy to use, based on modern Chaos Engineering principles & community collaborated. It is 100% open source & a CNCF project.

LitmusChaos takes a cloud-native approach to create, manage and monitor chaos. The platform itself runs as a set of microservices and uses Kubernetes custom resources to define the chaos intent, as well as the steady state hypothesis.

Project Summary

Website	https://litmuschaos.io/
Organization/Foundation Name	Cloud Native Computing Foundation (CNCF)
License	Apache License 2.0
Open/Proprietary	Open
Source Path	https://github.com/litmuschaos/litmus.git
Brief Description	LitmusChaos is a Cloud-Native Chaos Engineering Framework with cross-cloud support.

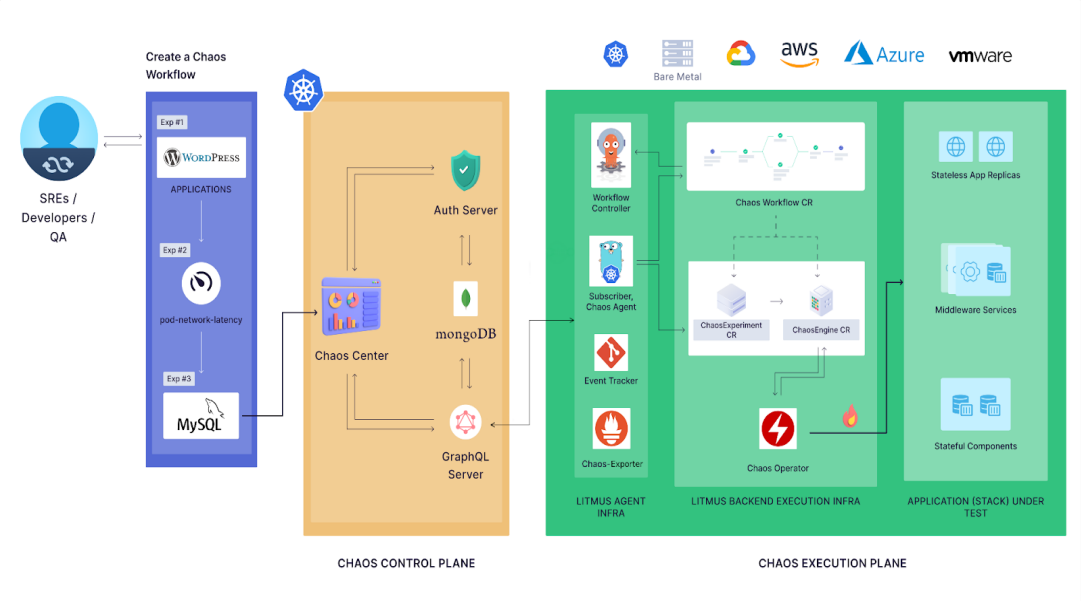
Project Details

Key Features

- Chaos Control Plane: A centralized chaos management tool called chaos-center, which helps construct, schedule and visualize Litmus chaos workflows
- Chaos Execution Plane Services: Made up of a chaos agent and multiple operators that execute & monitor the experiment within a defined target Kubernetes environment.

- ChaosExperiment: A resource to group the configuration parameters of a particular fault. ChaosExperiment CRs are essentially installable templates that describe the library carrying out the fault, indicate permissions needed to run it & the defaults it will operate with. Through the ChaosExperiment, Litmus supports BYOC (bring-your-own-chaos) that helps integrate (optional) any third-party tooling to perform the fault injection.
- ChaosEngine: A resource to link a Kubernetes application workload/service, node or an infra component to a fault described by the ChaosExperiment. It also provides options to tune the run properties and specify the steady state validation constraints using 'probes'. ChaosEngine is watched by the Chaos-Operator, which reconciles it (triggers experiment execution) via runners.
- ChaosResult: A resource to hold the results of the experiment run. It provides details of the success of each validation constraint, the revert/rollback status of the fault as well as a verdict. The Chaos-exporter reads the results and exposes information as prometheus metrics. ChaosResults are especially useful during automated runs.

Architecture



The Litmus architecture can be segregated into two parts:

- Control Plane: Contains the components required for the functioning of Chaos Center, the website-based portal for Litmus.
- Execution Plane: Contains the components required for the injection of chaos in the target resources.

Current Usage

- For Developers: To run chaos experiments during application development as an extension of unit testing or integration testing.
- For CI/CD pipeline builders: To run chaos as a pipeline stage to find bugs when the application is subjected to fail paths in a pipeline.
- For SREs: To plan and schedule chaos experiments into the application and/or surrounding infrastructure. This practice identifies the weaknesses in the deployment system and increases resilience.

Technical Details

Litmusctl CLI requires the following things:

- kubeconfig - litmusctl needs the kubeconfig of the k8s cluster where we need to connect litmus chaos delegates. The CLI currently uses the default path of kubeconfig i.e. ~/.kube/config.
- kubectl - litmusctl is using kubectl under the hood to apply the manifest.

Reference/ Acknowledgements

All the informations are from https://docs.litmuschaos.io/