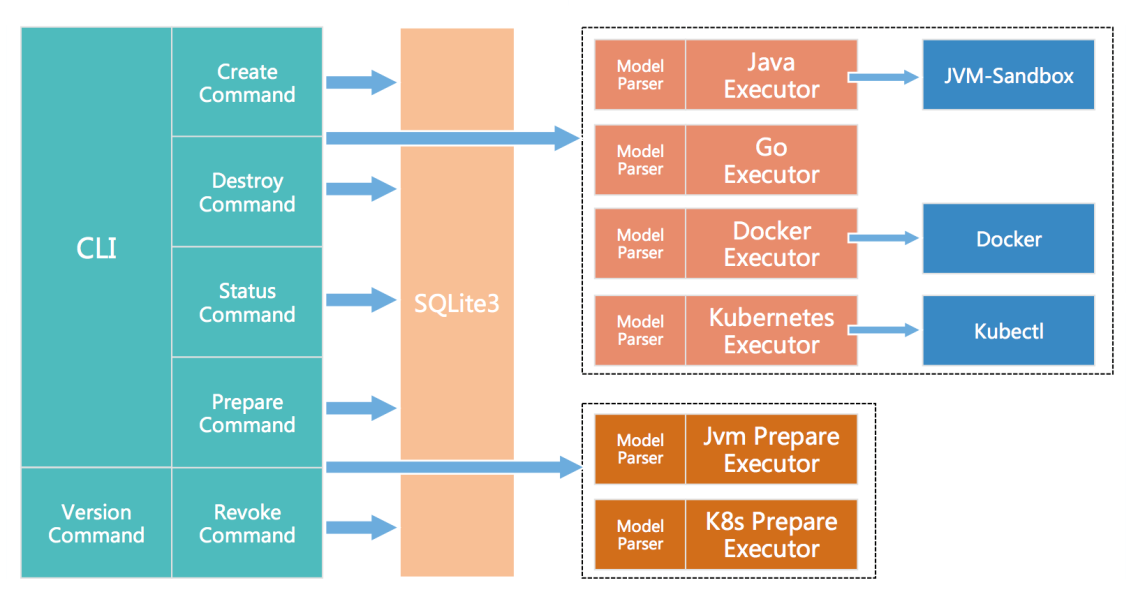
# Chaosblade

# **Introduction**

Chaos blade is a popular and easy-to-use chaos engineering toolkit that allows users to simulate various types of system failures and network errors in a controlled environment. It is designed to help organizations identify and mitigate potential issues in their systems before they cause real-world problems.

Chaos engineering is the practice of intentionally introducing controlled failures into a system to test its resilience and identify weaknesses. By simulating different types of failure scenarios, organizations can gain a better understanding of how their systems will perform under stress and identify areas that need improvement.

Chaos blade simplifies the chaos engineering process by providing a command-line interface that makes it easy to create and execute experiments. It includes a wide range of "chaos actions" that can be used to simulate failures, such as network latency, CPU overload, and file system errors. These actions can be applied to different parts of a system, such as a specific process or a network interface, allowing users to test the resilience of different components.



# Project Summary

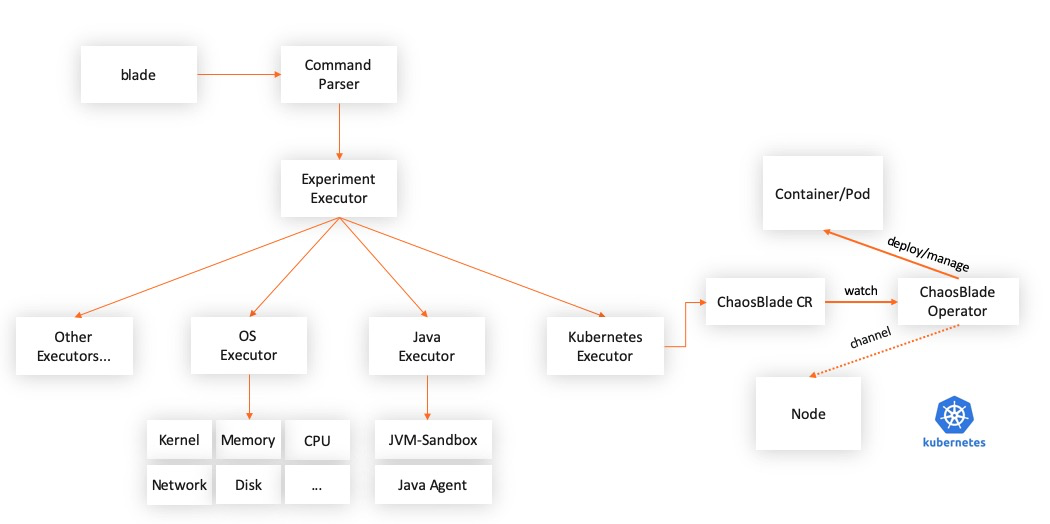
|  |  |
| --- | --- |
| Website | https://chaosblade.io/ |
| Organization/Foundation Name | Alibaba Cloud's Chaos Engineering Team |
| License | Apache-2.0 |
| Open/Proprietary | Open source project |
| Source Path(if open source) | https://github.com/chaosblade-io/chaosblade |
| Brief Description | Chaos blade is an open-source software tool for simulating system faults and chaos engineering experiments. Developed by Alibaba Cloud, it enables developers to identify potential issues in complex systems by introducing controlled failures. Chaos blade supports a wide range of operating systems and cloud platforms, and is widely used in the DevOps community. |

# Project Details

## Key Features

* **Support for Open-Source Experimental Tool Hosting:**The platform can host mainstream experimental tools in the industry, such as its own chaosblade and external LitmusChaos. The chaos mesh experiment tool will also be hosted in the future.
* **A Wide Range of Experimental Scenarios:** Basic resources (such as CPU, memory, network, disk, process, kernel, and file), application services in multiple languages (such as Java, C++, JavaScript, and Golang), and Kubernetes platform (covering resources such as Container, Pod, and Node)
* **Automated Deployment of Experimental Tools:**Manual deployment of experimental tools is unnecessary. It can realize automatic deployment of experimental tools on hosts or clusters.
* **Unified User Interface of Chaos Experiments:** Users do not need to care about the use of different tools, and they can perform chaos experiments on the unified user interface.
* **Multi-Dimensional Experiment Mode:** It supports experiment orchestration from the dimension of hosts to Kubernetes resources and applications.
* **Integrated Cloud-Native Ecosystem:**It uses Helm for deployment management, integrates Prometheus for monitoring, and supports the hosting of cloud-native experimental tools.

## Architecture



* chaosblade is a chaos experiment management tool. It includes commands for creating experiments, destroying experiments, querying experiments, preparing experiment environments, and revoking experiment environments. It is an execution tool for chaos experiments. The execution methods include CLI and HTTP. It provides complete descriptions of commands, experimental scenarios, and scenario parameters. The operations are concise and clear.
* chaosblade-spec-go is the Golang definition of the chaos experimental model, which helps implement scenarios that use the Golang language.
* chaosblade-exec-os implements basic resource experimental scenarios, such as CPU, network, memory, and disk.
* chaosblade-exec-docker implements Docker container experimental scenarios, standardized by calling the Docker API.
* chaosblade-operator implements experimental scenarios on Kubernetes platforms. Chaos experiments are defined by the Kubernetes standard CRD mode. It is very convenient to use Kubernetes resource operations to create, update, and delete experimental scenarios, including using methods, such as kubectl and client-go. It can also be executed using the preceding chaosblade cli.
* chaosblade-exec-jvm implements Java application experimental scenarios. It is dynamically mounted using Java Agent technology without any access. It can be used at no cost, supports uninstallation, and can completely recycle every resource created by the Agent.
* chaosblade-exec-cplus implements C++ application experimental scenarios. It uses GDB technology to implement method-level and code line-level experimental scenario injection.

## Current Usage

Chaosblade is currently used by various companies and organizations, particularly those in the DevOps and cloud computing space. Some examples of companies that use Chaosblade include Alibaba Cloud, Tencent Cloud, and the Chinese ride-hailing company Didi Chuxing.

## Technical Details

1. **Architecture:** Chaosblade is a modular tool with a microkernel architecture. It includes a core module responsible f Architecture: Chaosblade is a modular tool with a microkernel architecture. It includes a core module responsible for the basic functionality of the tool, as well as a set of sub-modules that provide specific chaos experiment capabilities.
2. **Programming language:** Chaosblade is written in Golang, a popular programming language for developing high-performance applications.
3. **Operating system support:** Chaosblade supports a wide range of operating systems, including Linux, Windows, and macOS.
4. **Cloud platform support:** In addition to supporting on-premises systems, Chaosblade also supports cloud platforms such as Alibaba Cloud, Tencent Cloud, AWS, and Google Cloud Platform.
5. **Deployment:** Chaosblade can be deployed as a standalone binary or as a Kubernetes deployment, making it easy to integrate into existing DevOps pipelines.

### Project comparison

1. **Gremlin:** Gremlin is a cloud-based platform for chaos engineering that provides a range of chaos experiments, such as CPU spikes, network partitioning, and DNS failure. It supports a w Gremlin: Gremlin is a cloud-based platform for chaos engineering that provides a range of chaos experiments, such as CPU spikes, network partitioning, and DNS failure. It supports a wide range of platforms, including cloud providers like AWS, Azure, and Google Cloud. Gremlin also includes integrations with popular monitoring and alerting tools like Datadog and PagerDuty.
2. **Chaos Monkey**: Chaos Monkey is a tool developed by Netflix for testing the resiliency of its cloud-based systems. It randomly terminates instances in production environments to simulate failures and test th Gremlin: Gremlin is a cloud-based platform for chaos engineering that provides a range of chaos experiments, such as CPU spikes, network partitioning, and DNS failure. It supports a wide range of platforms, including cloud providers like AWS, Azure, and Google Cloud. Gremlin also includes integrations with popular monitoring and alerting tools like Datadog and PagerDuty.

### Any other information

Chaosblade is an open-source tool for injecting chaos experiments into distributed systems to test their resilience and reliability. The Chaosblade community is a group of developers, engineers, and users who contribute to the development, testing, and documentation of Chaosblade.

The community is inclusive and welcomes contributions from anyone interested in chaos engineering and distributed systems. Members of the community share their experiences using Chaosblade, discuss best practices, and help each other troubleshoot issues. The community also collaborates on new features and improvements to the tool.

### Reference / Acknowledgements

* **Chaosblade Website:** [**https://chaosblade.io/en/**](https://chaosblade.io/en/)
* **Chaosblade GitHub:** [**https://github.com/chaosblade-io/chaosblade**](https://github.com/chaosblade-io/chaosblade)
* **Chaosblade Documentation:** [**https://chaosblade.io/en/docs**](https://chaosblade.io/en/docs)
* **Chaosblade Community:** [**https://chaosblade.io/en/docs/community/**](https://chaosblade.io/en/docs/community/)
* **Chaosblade Slack:** [**https://chaosblade-io.slack.com/**](https://chaosblade-io.slack.com/)