

投稿日 2022/12/22

## 障害テストツール

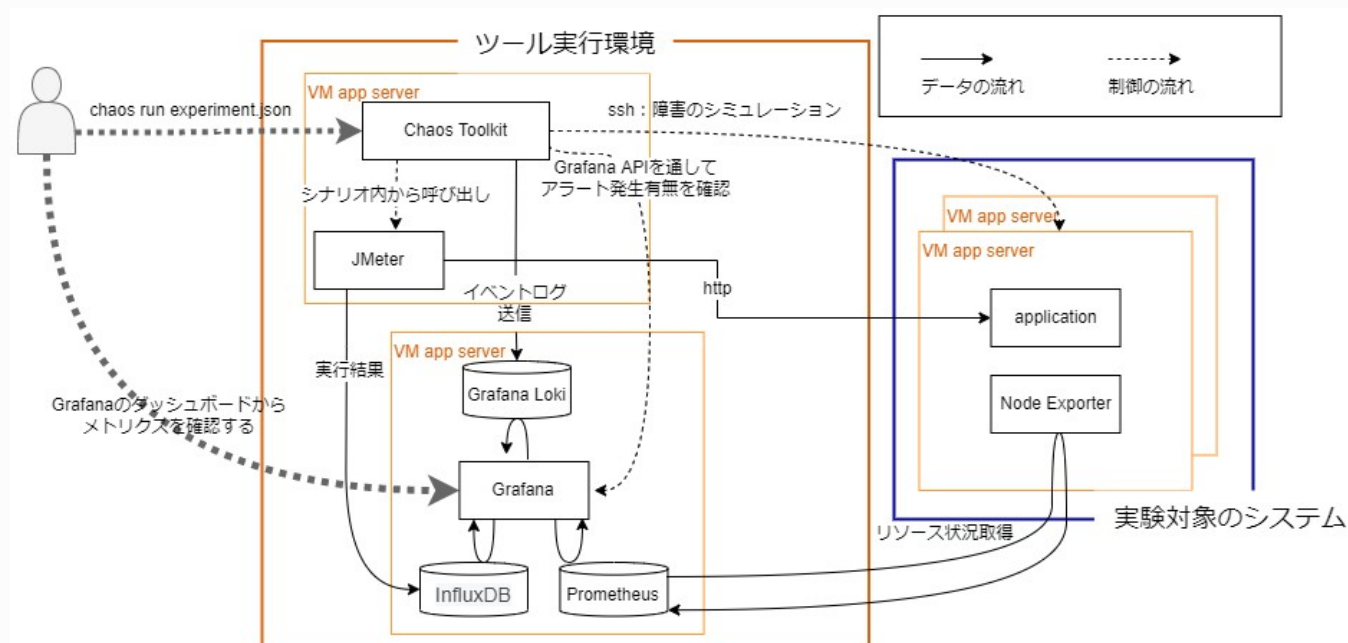
### はじめに

このツールは障害テストを行うための環境構築と基本的な障害のシミュレーションをサポートするツールです。障害テストをおこなうには障害のシミュレーションやメトリクス収集・監視が必要で、それらが可能なツールとなっています。コンテナ化しているため、オンプレやクラウドなどのプラットフォームに依存せず容易に構築できます。また、CPUなどのリソースに高負荷がかかっている状況やネットワーク障害などのシミュレーションが可能となっています。

### ツール概要

以下の図のような構成の障害テスト実行環境を構築できます。環境を構成する各ソフトウェアの役割は[こちら](#)を参照してください。

ツールを構成するソフトウェアはすべてOSSのため無償で利用でき、それらのソフトウェアをdocker composeコマンドで容易に立ち上げられるcompose.yamlファイルの提供もしています。



ツール構成図

### 本ツールにできること

本ツールではChaos Toolkitを利用して障害テストシナリオの記述・実行をおこないます。シナリオ内では障害のシミュレーションやJMeterシナリオの呼び出しなどが可能で、JMeter実行結果やNode Exporterから取得したリソース情報をを用いてメトリクス監視がおこなえます。

## 障害シミュレーション

---

テスト対象のシステムに対して以下の障害をシミュレーションできます。

- CPU負荷
- メモリ負荷
- ディスク負荷
- I/O負荷
- OSシャットダウン
- 時刻変更
- プロセスキル
- ネットワーク遅延
- パケットロス
- ネットワーク遮断

物理電源の切断など、物理的な事象を含んだシナリオ作成や実行は本ツールでは対応不可能です。

## トランザクション投入

---

JMeterを利用してトランザクション投入をおこなえます。JMeter実行結果はInfluxDBに蓄積され、レイテンシー、エラー率とスループットを計測できます。

## メトリクス収集・監視

---

JMeter実行結果とNode Exporterで取得できるリソース情報をそれぞれInfluxDBとPrometheusに収集し、それらを利用したメトリクス収集・監視が可能です。

## ツール実行環境の構築

---

### 前提・制約

---

ツールは大きく分けて障害テスト実行環境構築ツールと障害シミュレーションツールの2つがあります。

障害テスト実行環境構築ツールは[Docker Engine](#)と[Docker Compose](#)がインストールされたLinux系のOSで動作することを想定としています。

障害シミュレーションツールはテスト対象システムのサーバにSSHでログインし障害をシミュレーションするコマンド（stress-ng, tc, iptablesなど）を実行するため、障害をシミュレーションする対象のサーバはツール実行環境からSSH可能で必要なコマンドが実行可能なLinux系OSを対象としています。詳しくは障害シミュレーションツールの[README](#)を参照してください。

## 環境構築手順

---

以下の手順にしたがってツールの環境構築をおこなってください。

- [環境構築手順](#)

## ツール実行方法

---

以下の手順を参考にしてツールの実行をおこなってください。

- [障害テスト実行手順](#)

## ツールカスタマイズ

---

本ツールで利用するGrafanaやChaos Toolkitは独自に拡張が可能です。

## 障害シミュレーションのパターン追加

---

Chaos Toolkitは容易に拡張可能で、任意の操作（障害シミュレーションなど）を追加できます。

Chaos Toolkitにはあらかじめ用意された拡張があり、[AWS](#)や[Azure](#)などさまざまなプラットフォームに対応することも可能です。その他に既存の拡張としてこういったものが用意されているかは[Chaos Toolkitのドキュメント](#)で確認できます。

ユーザ側で拡張を作成することも可能で、主に以下の3つの方法で拡張を行うことができます。

1. Pythonの関数を呼び出す
2. 任意のプロセスを実行する
  - Linuxコマンドを呼び出したり、GoやRustで書いたコードをコンパイルしたバイナリファイルを実行するなど
3. HTTPリクエスト
  - 特定の操作を行うHTTPエンドポイントを作成しChaos Toolkitからリクエストを送信する

詳細は以下のドキュメントを参照してください。

- [Learn the basic of extending the Chaos Toolkit](#)
- [Extending Chaos Toolkit with Python](#)

## 収集・監視するメトリクスの追加

Grafanaを任意のデータソースと連携してメトリクスの収集・可視化を行うことが可能です。本ツールではGrafanaをInfluxDBやPrometheusと連携して[Golden Signals](#)を可視化できます。


独自のメトリクス監視を追加する際は以下の例を参考にしてください。

- [アプリケーション監視](#)
- [リソース監視](#)


/\* Recommend \*/

## 「ソフトウェアテスト」のおすすめ記事はこちら

この記事に関連する記事もお読みください。

 ソフトウェアテスト

TIS AIChatLab：MagicPodをつかった自動テスト導入戦略

2025/02/26  8

 ソフトウェアテスト

システムの耐障害性を高めるためのアプローチ

2022/12/22  6 ソフトウェアテスト

障害テスト計画ガイド

2022/12/22  13

最近投稿された記事も用意しました。

 ソフトウェアテスト

TIS AIChatLab：MagicPodをつかった自動テスト導入戦略

2025/02/26  8

 ソフトウェアテスト


DialogPlay：AIテスト自動化プラットフォーム『MagicPod』の活用

2024/10/30  22 ソフトウェアテスト


Sales Driveの安定リリースを支える自動テスト：MagicPodの活用状況

2023/12/22  14

「ソフトウェアテスト」で最も読まれている記事を以下にまとめています。

 ソフトウェアテスト

全体テスト計画ガイド

2018/10/01  35

 ソフトウェアテスト

テスト種別＆テスト観点カタログ

2018/10/01  19 ソフトウェアテスト

性能テスト計画ガイド

2022/04/05  21