Cultural Shifts:

Fostering a Chaos First Mindset in Platform Engineering

CONF42

Sayan Mondal

About Me

Senior Software Engineer II at

Maintainer & Community Manager at Litmus

Sayan Mondal

@s_ayanide

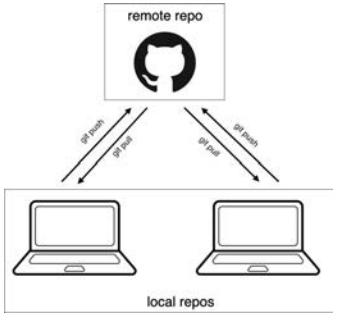
@s-ayanide

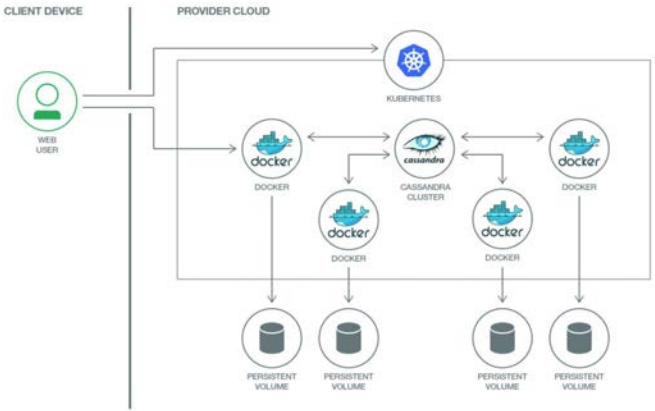
Agenda

- Core Components of IDP
- The Cloud-Native problem
- What and why of Chaos Engineering
- The Chaos First Principle
- How chaos plays a pivotal role in PE
- The future vision
- Tools in the market

- Hands on demo
- Actual chaos execution on infrastructure

Manufacturing software in the *Cloud Native era* is hard





- Runtime architecture, CI/CD, DevOps, Environments, SecOps, Configuration Management, Version Management, Testing, Observability, Analytics, SRE
- Devops goes to canary, etc
- Self Service and Policy Driven
- Zero Trust environment

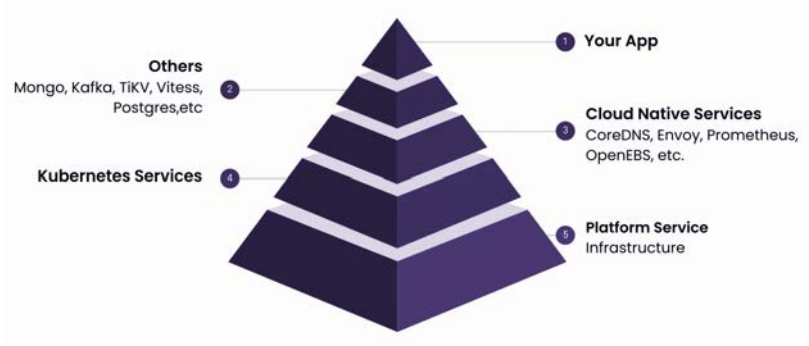
Core components of IDP

Core Components	Description
Application Configuration Management	Effectively handle application configuration in a reliable, dynamic and scalable way
Infrastructure Orchestration	Coordinate your infrastructure dynamically and intelligently, adapting to the context as needed.
Environment Management	Empower developers to generate new, fully provisioned environments on demand.
Deployment Management	Implement a delivery pipeline for Continuous Delivery
Role-Based Access Control	Manage roles to control who can do what in a scalable way

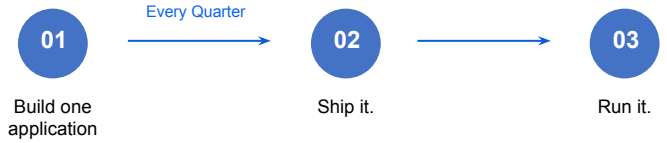
The *Cloud Native* problem

Proliferation of applications into micro services leads to a RELIABILITY challenge

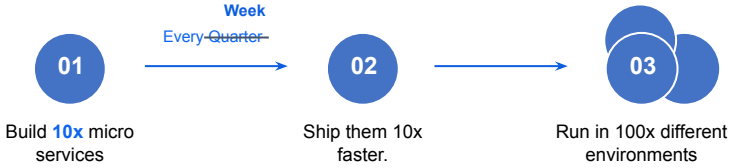
In cloud native, your code depends on hundreds of other microservices and runs on many platforms. The potential of being subjected to a dependent component failure is huge.



Legacy DevOps



Cloud-Native DevOps



Too many fault scenarios. Significant increase in service down potential because of a failure of a dependent service

Actual cost of *Downtimes*

79 Minutes

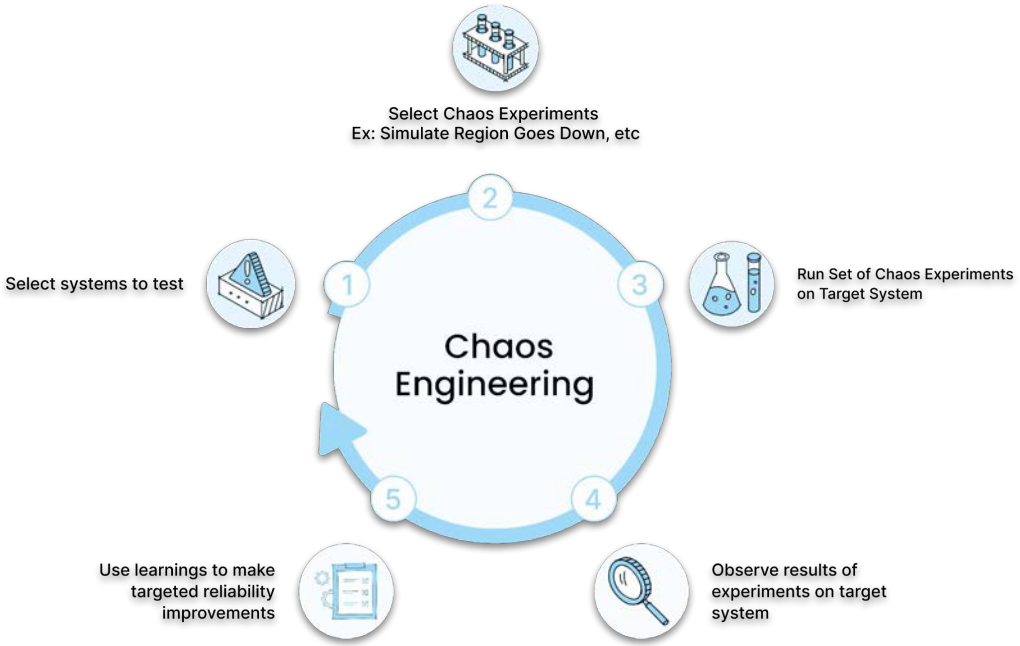
The average downtime for an outage was 79 minutes

$ 84,650

The cost of downtime averaged at $84,650 US per hour

What is *Chaos Engineering*

> **Chaos engineering is the process of testing a distributed computing system to ensure that it can withstand unexpected disruptions.**

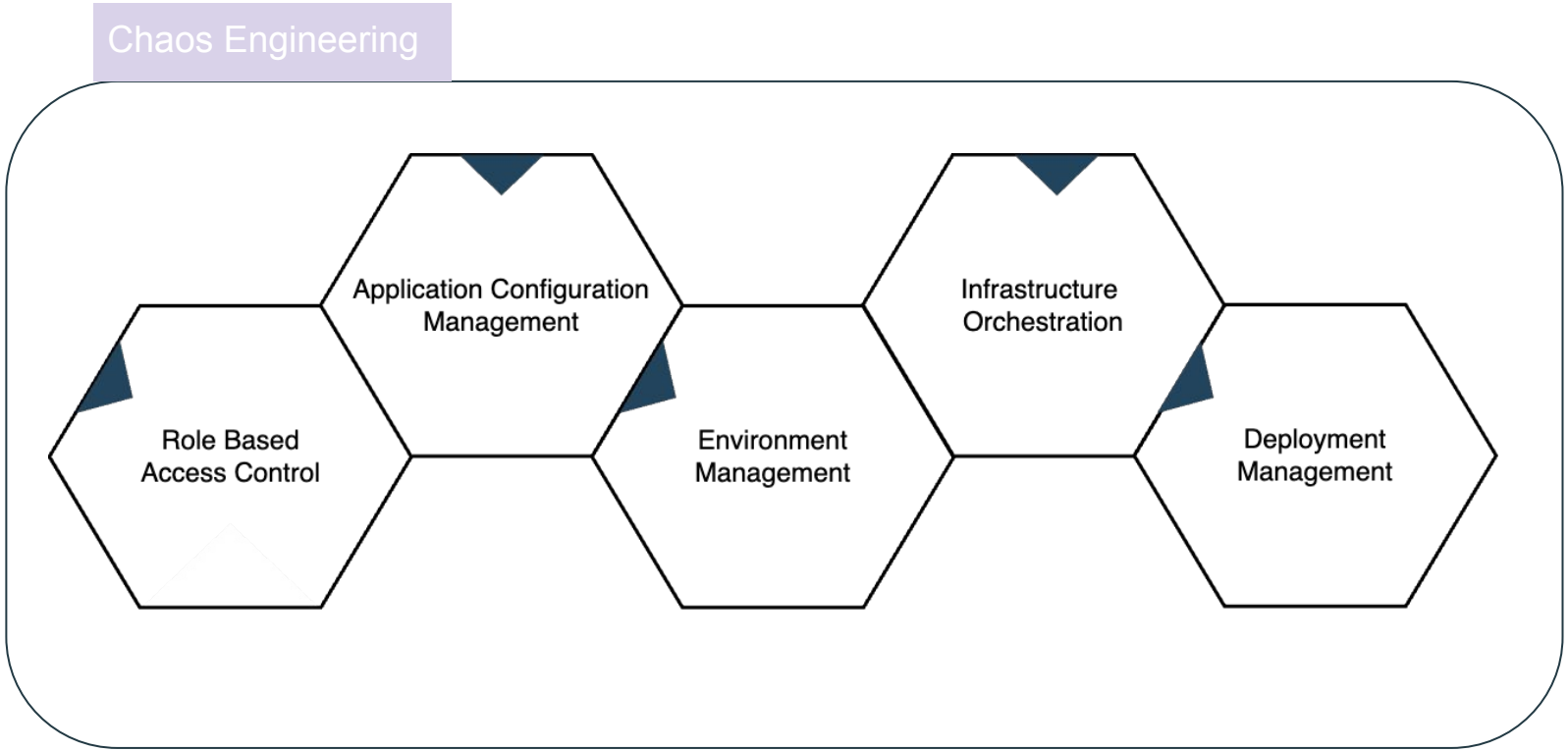


Chaos First *Principle*

“

The Chaos First Principle in Platform Engineering advocates for designing systems with the expectation of failure as a fundamental assumption rather than an exceptional event. Embracing this principle involves deliberately introducing chaos and disruptions into the platform infrastructure to proactively identify weaknesses and enhance resilience

Chaos Engineering as part of *Platform Engineering*



How it plays a *Pivotal Role*

By regularly conducting Chaos Engineering experiments, Platform Engineers gain insights into how the platform behaves during unexpected events

Capacity Planning and Scaling

By introducing controlled chaos, engineers can observe how the system handles increased loads, resource constraints, and unexpected spikes in traffic

Cultural Shift towards Resilience

By embracing failure as a natural part of system development, engineers become more proactive in addressing weaknesses and designing systems that can withstand unforeseen challenges

Continuous Improvement

Chaos Engineering promotes a mindset of continuous improvement within Platform Engineering teams

Backstage is an open platform for building developer portal. It restores order to your infrastructure and enables teams to ship high quality products

Litmus Chaos is an Open Source Cloud-Native Chaos Engineering Framework with cross-cloud support. It is a CNCF Incubating project with adoption across several organizations.

The *Vision*

Define and execute

Define and execute the chaos experiments

Define and execute Chaos scenarios
Identify appropriate scenarios

Chaos as a service

Service catalog to enable Platform Engineers to discover and apply chaos

Self service
Experiments are in Git just like code

***Integrate* into CI/CD systems**

Execute automated and controlled chaos experiments across prod/non-prod environments

Execute chaos experiments with push button
make chaos repeatable process

Enable *observability* for Chaos and Automated evaluation
Chaos metrics used to assess impact and manage SLOs/Errors

Measure the impact of inducing chaos
Define and validate Chaos SLOs

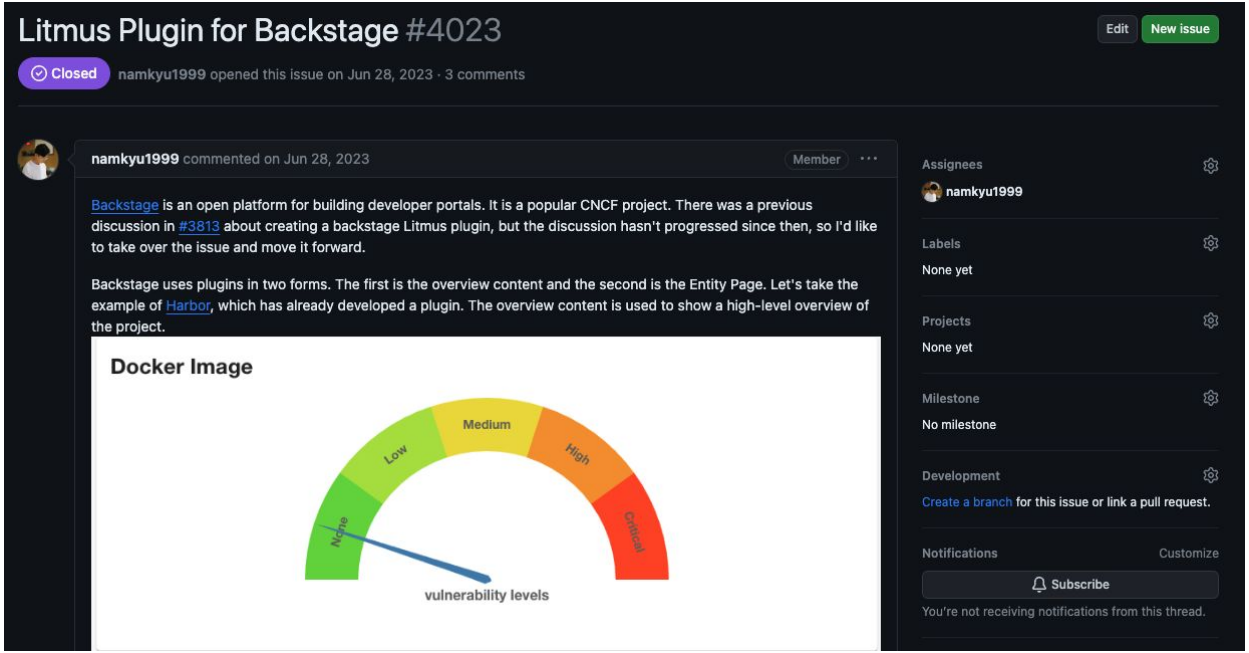
Power of *Open Source*

All credits to

Namkyu Park

https://github.com/namkyu1999

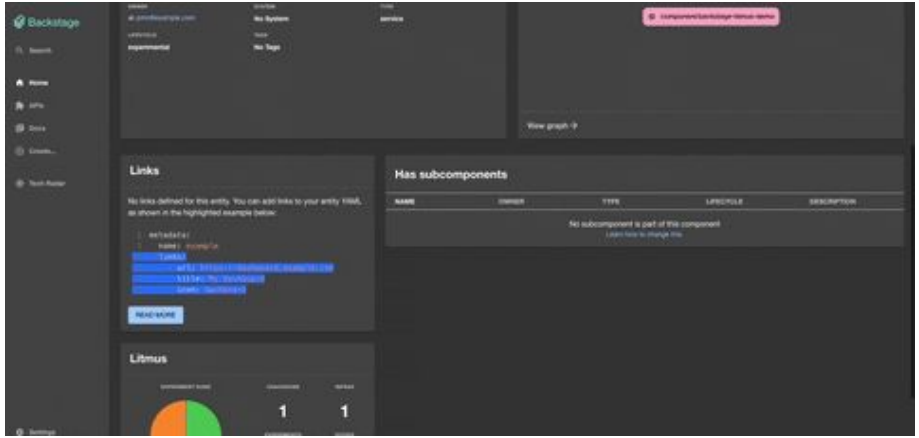
```
yarn add backstage-plugin-litmus
```



https://github.com/litmuschaos/backstage-plugin

Hands on demo

```yaml
proxy:
 '/litmus':
   target: 'your-own-litmus-ui-url'
   changeOrigin: true
   headers:
     Authorization: 'Bearer ${LITMUS_AUTH_TOKEN}'
litmus:
  baseUrl: 'your-own-litmus-ui-url'
  apiToken: ${LITMUS_AUTH_TOKEN}
```



entities.yaml

```yaml
---
apiVersion: backstage.io/v1alpha1
kind: Component
metadata:
  name: backstage-litmus-demo
  description: An example of a Backstage application.
  ## append here
  annotations:
    litmuschaos.io/project-id: your-own-project-id
  ##
spec:
  type: service
  owner: john@example.com
  lifecycle: experimental
```

Future *Roadmap*

Maturity Model

Define a maturity model for Chaos Engineering in Platform Engineering to help organization access their current level

Industry Standards

Contribute to the development of industry standards and best practices fostering a common understanding and adoption

Define Guardrails

Ensure Chaos is conducted in a safe and compliant manner by defining guarding policies applicable to the system under stress

Chaos Budgeting

A framework that defines the acceptable level of disruption or downtime for different components helping team allocate resources effectively

Thank You

Follow Litmus on

/LitmusChaos

/litmuschaos

Contact me on

@s_ayanide

/s-ayanide