WHITE PAPER

ACHIEVING ORDER THROUGH CHAOS ENGINEERING: A SMARTER WAY TO BUILD SYSTEM RESILIENCE

Abstract

Digital infrastructure has grown increasingly complex owing to distributed cloud architectures and microservices. More than ever before, it is increasingly challenging for organizations to predict potential failures and system vulnerabilities. This is a critical capability needed to avoid expensive outages and reputational damage.

This paper examines how chaos engineering helps organizations boost their digital immunity. As a leading quality engineering approach, chaos engineering provides a systematic, analytics-based, test-first, and well-executed path to ensuring system reliability and resilience in today's disruptive digital era.

Infosys®
Navigate your next

Introduction

Digital systems have become increasingly complex and interdependent, leading to greater vulnerabilities across distributed networks. There have been several instances where a sudden increase in online traffic or unforeseen cyberattacks have caused service failures, adversely impacting organizational reputation, brand integrity, and customer confidence. Such outages have a costly domino effect, resulting in revenue losses or, in some cases, regulatory action against the organization.

Thus, enterprises must implement robust and resilient quality engineering solutions that safeguard them from potential threats and help overcome these challenges. This is where 'chaos engineering' comes in.

Chaos engineering is a preventive measure that tests failure scenarios before they have a chance to grow and cause downtime in live environments. It identifies and fixes issues immediately by recognizing system weaknesses and how systems behave during an injected failure. Through chaos engineering, organizations can establish mitigation steps to safeguard end users from negative impact and build confidence in the system capacity to withstand highly variable and destructive conditions.

Chaos Engineering – A Boost to Digital Immunity

System resilience is about how promptly a system can recover from disruption. Chaos engineering is an experimentative process that deliberately disrupts the system to identify weak spots, anticipate failures, predict user experience, and rectify the architecture. It helps engineering teams redesign and restore the organization's infrastructure and make it more resilient in the face of any crisis. Thus, it builds confidence in system resiliency by running failure experiments to generate random and unpredictable behaviour.

Despite its name, chaos engineering is far from chaotic. It is a systematic, data-driven technique of conducting experiments that use chaotic behaviour to stress systems, identify flaws, and demonstrate resilience. System complexity and rising consumer expectations are two of the biggest forces behind chaos engineering. As systems becoming increasingly feature-rich, changes in system performance affect system predictability and service outcomes, which in turn, impact business success.

How Chaos Engineering is Different from Traditional Testing Practices

- **Performance testing** – It baselines application performance under a defined load in favorable environmental conditions. The main objective is to check how the system performs when the application is up and running without any severe functional defects in an environment comparable to the production environment. The potential disruptors uncovered during the performance tests are due to certain load conditions on the application.
- **Disaster recovery testing** – This process ensures that an organization can restore its data and applications to continue operations even after critical IT failure or complete service disruption.
- **Chaos testing** – During the chaos test, the application under normal load is subjected to known failures outside the prescribed boundaries with minimum blast radius to check if the system behaves as expected. Any deviation from expectations is noted as an observation and mitigation steps are prepared to rectify the deviation.

Quality assurance engineers find chaos testing to be more effective than performance and disaster recovery testing in unearthing latent bugs and identifying unanticipated system weaknesses.

5-step Chaos Engineering Framework

Much like a controlled injection, implementing chaos engineering calls for a systematic approach. The five-step framework described below, when 'injected' into an organization, can handle defects and fight system vulnerabilities.

Chaos engineering gives organizations a safety net by introducing failures in the pre-production environment, thereby promoting organizational learning, increasing reliability, and improving understanding of complex system dependencies.

1. **Prepare the process**

   Understand the end-to-end application architecture. Inform stakeholders and get their approval to implement chaos engineering. Finalize the hypothesis based on system understanding.

2. **Set up tools**

   Set up and enable chaos test tools on servers to run chaos experiments. Enable system monitoring and alerting tools. Use performance test tools to generate a steady load on the system under attack. Additionally, a Jenkins CI/CD pipeline can be set up to automate chaos tests.

3. **Run chaos tests**

   Orchestrate different kinds of attacks on the system to cause failures. Ensure proper alerts are generated for the failures and sent to the right teams to take relevant actions.

4. **Analyze the results**

   Analyze the test results and compare these with the expectations set when designing the hypothesis. Communicate the findings to the relevant stakeholders to make system improvements.

5. **Run regression tests**

   Repeat the tests once the issues are fixed and increase the blast radius to uncover further failures.

This step-by-step approach executes an attack plan within the test environment and applies the lessons/feedback from the outcomes, thereby improving the quality of production systems and delivering tangible value to enterprises.

Examples of Chaos Engineering Experiments

A chaos engineering experiment or a chaos engineering attack is the process of inducing attacks on a system under an expected load. An attack involves injecting failures into a system in a simple, safe, and secure way.

There are various types of attacks that can be run against infrastructure. This includes anything that impacts system resources, delays or drops network traffic, shuts down hosts, and more. A typical web application architecture can have four types of attacks run on it to assess application behavior:

- **Resource attacks** – Resource attacks reveal how an application service degrades when starved of resources like CPU, memory, I/O, or disk space
- **State attacks** – State attacks introduce chaos into the infrastructure to check whether the application service fails or whether it handles it and how
- **Network attacks** – Network attacks demonstrate the impact of lost or delayed traffic on the application. It is done to test how services behave when they are unable to reach any one of the dependencies, whether internal or external
- **Application attacks** – Application attacks introduce sudden user traffic on the application or on a particular function. It is done to test how services behave when there is sudden rise in the user traffic due to high demand.

Chaos engineering experiments on a typical web application

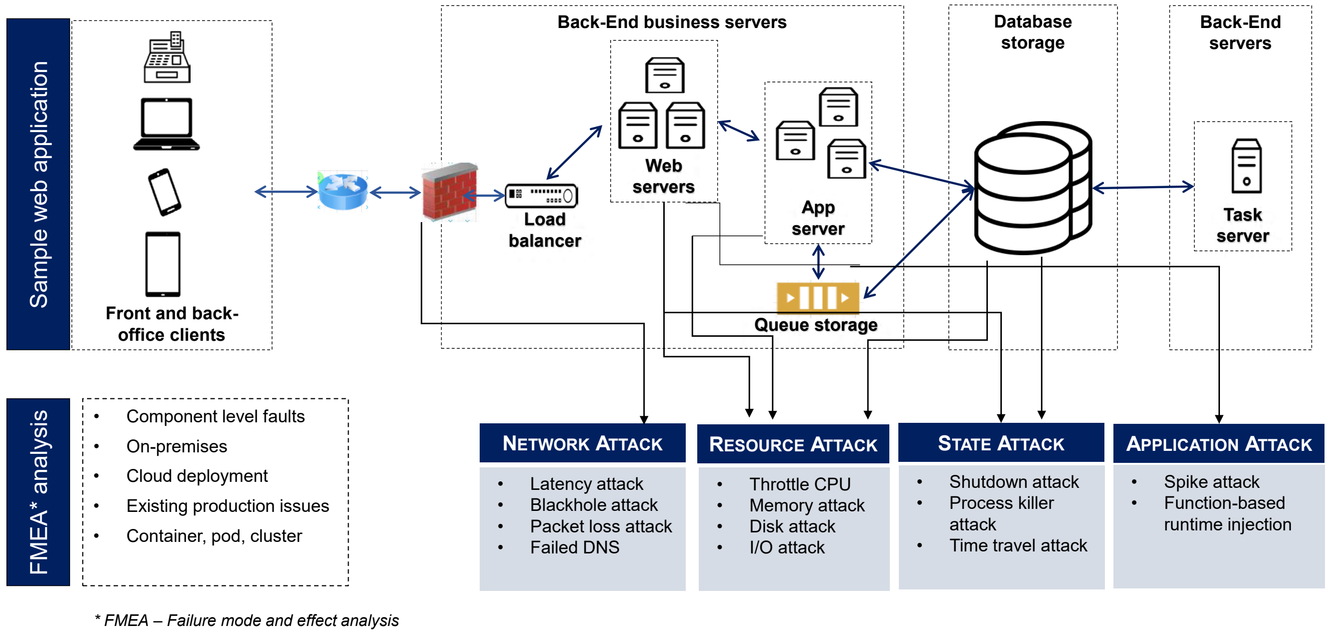

Figure 1 – Chaos engineering experiments on a typical web application

GameDay Concept

GameDay is an advanced concept of chaos engineering. It is organized by the chaos test team to practice chaos experiments, test incident response process, validate past outages, and find unknown issues in services.

In GameDay, a mock war room is set up and the calendar of all stakeholders is blocked for up to 2-4 hours. One or more chaos experiments are run on the system or service to observe the impact. All technical outcomes are discussed.

The team includes a 'General' who is responsible for conducting the GameDay, a 'Commander' who coordinates with all the participants, 'Observers' who monitor the GameDay tests and validate the deviations (if any), and a 'Scribe' who notes down the key observations.

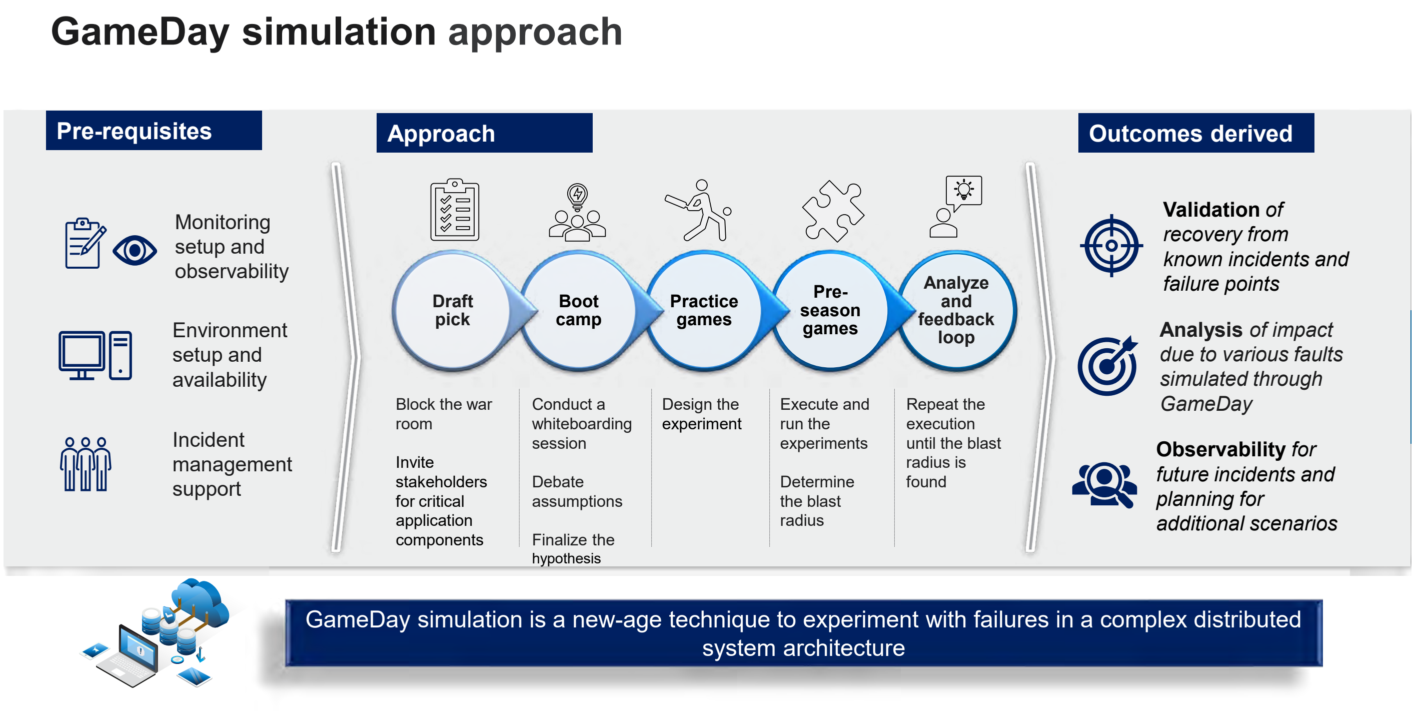
GameDay simulation approach
Pre-requisites
Monitoring setup and observability
Environment setup and availability
Incident management support
Approach
Draft pick
Boot camp
Practice games
Pre-season games
Analyze and feedback loop
Block the war room
Invite stakeholders for critical application components
Conduct a whiteboarding session
Debate assumptions
Finalize the hypothesis
Design the experiment
Execute and run the experiments
Determine the blast radius
Repeat the execution until the blast radius is found
Outcomes derived
Validation of recovery from known incidents and failure points
Analysis of impact due to various faults simulated through GameDay
Observability for future incidents and planning for additional scenarios
GameDay simulation is a new-age technique to experiment with failures in a complex distributed system architecture

Figure 2 – GameDay simulation approach

Benefits of Chaos Engineering

To ignore chaos engineering is to embrace crisis engineering. Proactive QE teams have made chaos engineering a part of their regular operations by exposing their staff to chaos tests and collaboratively experimenting with other business units to refine testing and improve enterprise systems.

Chaos engineering delivers several benefits such as:

- **Reduced detection time** – Early identification of issues caused due to failures occurring in live environments, making it easy to proactively identify which component may cause issues
- **Knowing the path to recovery** – Chaos engineering helps predict system behavior in case of failure events and thus works towards protecting the system to avoid major outages
- **Being prepared for the unexpected** – It helps chart mitigation steps by experimenting with known system failures in a controlled environment
- **Highly-available systems** – Enables setting alerts and automating mitigation actions when known failures occur in a live environment, thereby reducing system downtime
- **Improved customer satisfaction** – Helps avoid service disruptions by detecting and preventing component outages, thereby enhancing user experience, increasing customer retention, and improving customer acquisition

Chaos engineering brings about cultural changes and maturity in the way an enterprise designs and develops its applications. However, its success calls for strong commitment from all levels across the organization.

Conclusion

System failures can prove very costly for enterprises, making it critical for organizations to focus on quality engineering practices. Chaos engineering is one such practice that boosts resilience, flexibility, and velocity while ensuring the smooth functioning of distributed enterprise systems. It allows organizations to introduce attacks that identify system weaknesses so they can rectify issues proactively. By identifying and fixing failure points early in the lifecycle, organizations can be prepared for the unexpected, recover faster from disruptions, increase efficiency, and reduce cost. Ultimately, it culminates in better business outcomes and customer experience.

About the Authors

Harleen Bedi

Senior Industry Principal

Harleen is a Senior IT Consultant with Infosys. She focuses on developing and promoting IT offerings for quality engineering based on emerging technologies such as AI, cloud, big data, etc. Harleen builds, articulates, and deploys QE strategies and innovations for enterprises, helping clients meet their business objectives.

Jack Hinduja

Lead Consultant

Jack Hinduja is a Lead Consultant at Infosys with over 15 years of experience in the telecom and banking sectors. He has led quality assurance and validation projects for enterprises across the globe. Jack is responsible for driving transformation in digital quality assurance and implementing performance and chaos engineering practices in various enterprises.