How To Scale Chaos Engineering

Scaling Chaos Engineering is all about standards, processes, and efficiency. You need to build replicable testing practices that can be adopted by teams across your organization.



PHASE 1: PROVE VALUE WITH A SINGLE SERVICE



1. Select a critical service to maximize impact

For your first service, start with one where an outage would have the biggest customer impact to get the most value from your testing.

Common critical services:

① Checkout application ② Dilling data processor ③ Authorization verification



2. Define Health Checks from customer-impacting metrics

A **Health Check** monitors systems before, during, and after a test. Use metrics that impact users, such as the **four SRE Handbook signals**.

Common health checks:

✓ Latency
✓ Error rates
✓ Request rates



3. Discover and test dependencies

Map your dependencies, then test how your service reacts to dependency outages. These tests alone can take services from 99.9% to 99.99% uptime.

Common dependencies:

Databases
 API calls
 Content Delivery Networks



4. Start with the most common failures

Most outages are caused by the same **common failure modes**. Start with these failures, then you can add tests unique to your architecture.

Common failures:

Scalability
 Redundancy
 Latency



5. Interpret test results and take action

Address risks uncovered by failed tests by **analyzing results** and adding work to sprints. Use possible customer impact to prioritize fixes.

Common issues:

1 Timeout mismatches 1 Misconfigured autoscaling 1 Misconfigured failovers

PHASE 2: SCALE AND IMPROVE

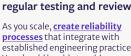
1. Roll out testing the same way for other services

Use the test suite to **onboard** other services across your organization following the same steps. Remember to point to your initial victories when talking to cautious or hesitant service

As you scale, create reliability processes that integrate with

2. Build processes around

established engineering practices. You should be able to address issues as part of your normal sprint instead of during incident

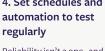


3. Use exploratory testing to fill in test coverage

Now that your teams are addressing the biggest issues, use **Chaos Engineering** to uncover the failures unique to your architecture, then add them to your standardized tests.

4. Set schedules and automation to test regularly

activity. Systems are constantly tests on a regular schedule to of your service reliability.



Reliability isn't a one-and-done shifting, so run the same group of catch changes and create a record



\$125M/yr

Saved by preventing outages



More reliability per hour of testing



50%

Reduction in downtime





Holiday sales were tremendous success without any major issues. Gremlin helps us to raise our bar.

-Lead Performance Engineer, Sephora

Sephora pulled off Black Friday without downtime. Want similar results?

Schedule a call with a reliability expert at **Gremlin.com**

Take our self-guided Gremlin product tour

