Gremlin

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:

- Checkout application
- Billing data processor
- 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:

- Timeout mismatches
- Misconfigured autoscaling
- 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 owners.

2. Build processes around regular testing and review

As you scale, **create reliability processes** that integrate with established engineering practices. You should be able to address issues as part of your normal sprint instead of during incident response.

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

Reliability isn't a one-and-done activity. Systems are constantly shifting , so run the same group of tests on a **regular schedule** to catch changes and create a record of your service reliability.

2X

More reliability per hour testing with Gremlin

50%

Reduction in downtime

75%

Lower MTTR

$125M/yr

Saved by preventing outages

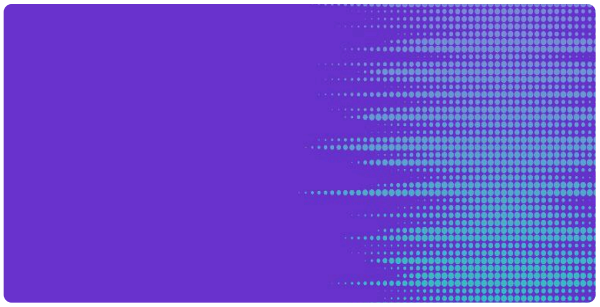
> Holiday sales were tremendous success without any major issues. Gremlin helps us to raise our bar.
>
> –Lead Performance Engineer, Sephora

Further Reading

Measuring the impact of your reliability work with reports

February 6, 2024 - 3 min read

Learn how Gremlin's built-in reporting tools track your reliability work, find high-priority risk in your environment, and demonstrate your progress towards greater reliability.



How the Gremlin agent fails safety

January 29, 2025 - 8 min read

Learn how Gremlin's built-in reporting tools track your reliability work, find high-priority risk in your environment, and demonstrate your progress towards greater reliability.

Reliability Intelligence: your reliability expert

August 10, 2025 - 4 min read

Gremlin's Reliability Intelligence combines Experiment analysis, Recommended Remediation, and an MCP Server to help teams increase reliability faster than ever.

Your reliability scorecard: How to measure and track service reliability

March 5, 2024 - 4 min read

Learn how Gremlin helps you track and manage your progress towards improved reliability with its comprehensive, built-in reporting tools.

Ready to scale reliability and meet your uptime goals?

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

Take our **self-guided Gremlin product tour**