Focus on customer requirements

# Meaningful availability: How many nines do you actually need?

Kolton Andrus
Co-Founder and CEO, Gremlin

How do you define availability in a way that makes sense? People typically talk in terms of nines of availability, which expresses system availability (uptime) as a percentage of total system time. Availability at 99% is OK as far as it goes, but that equates to more than seven hours of downtime per month.

Hitting 99.9% availability is better, at just under nine hours of downtime a year.

> *Companies aiming for high availability (HA) talk about four nines (less than an hour of downtime a year), which is what we strived for during my time at Netflix.*

Five nines availability, which means only five minutes of downtime per year, is ambitious. There is probably not one company that currently hits this standard. But as more and more critical services come online, it's reasonable to assume that the industry will shift toward higher standards, especially given the costs of downtime.

For most companies, getting from two nines to three will have a notable business impact. And doing so requires little effort; it's on the scale of taking an hour every couple of weeks to run an experiment and then fixing/enhancing the system based on your findings. This will provide significant improvement and create more reliable systems.

One way to do this is via chaos engineering, which involves inflicting bugs intentionally to test system resilience and learn from the experience. Moving to the next level is more difficult, both to achieve and maintain. So the big question is: How many nines are enough for your situation?

Here's how to answer that question.

## What do your customers want?

Ultimately what your end users and customers care about is whether your site is available when they want or need to use it. At the same time, though, nines are an easy shorthand way to talk about the real issue: reliability.

You need to track reliability somehow, but you want to do so while keeping the emphasis where it should be: on customer satisfaction and experience.

It isn't the nines that matter so much as the end-user experience, so let's move the discussion beyond "How many nines do I need?" to "How can I best serve my customers in ways that they care about?"

For many companies, achieving three nines and then growing and maturing the practices within that level is enough to keep most customers happy and unaware of system problems, depending on how you implement those practices when downtime occurs.

Very few organizations ever achieve four nines, and the ones that do often have strong reasons for needing that level of high availability (think air-traffic control systems).

Companies want to please their customers and limit expenses (including penalties for not meeting SLAs that talk about uptime/downtime). They aren't thinking about "getting one more nine" so much as how to keep customers happy, and site reliability is a big part of that.

## Long-term vs. short-term outages

Customers rarely notice outages that last mere seconds, but longer-duration outages definitely get noticed—and both contribute equally to your calculation of nines of availability.

That difference needs to be accounted for in your thinking. A networking blip that causes a momentary stutter while communication is rerouted as the system self-heals causes only a small customer impact, even if it happens more frequently than you would like.

In contrast, one long outage of even just 10 or 15 minutes can cause a well-known or popular site extensive public relations damage. This is true even when the total downtime for both systems is identical at the end of a week/month/year.

The central focus to the questions around reliability should boil down to this: Are you taking care of your customers? Are you meeting their expectations? You must think about how your customers' experience reliability, in order to have a valuable conversation. Customers trying to check a bank balance or make an airline reservation just want to log in, do what they need to do, and go on with their lives.

You might be able to ignore momentary blips, but outages or even scheduled maintenance windows that go on for longer periods of time are becoming more and more unacceptable.

## Be proactive with chaos engineering

Customers don't care how many nines of uptime your database has if they don't receive the desired information because of cloud-internal networking latency. They don't care that you had a sudden and unexpected traffic surge because of a pandemic and a market crash. They just want to check the value of their investment account.

Preventing a negative customer impact is the path to providing good customer experience. And chaos engineering experimentation can help you find and solve problems before any customer is ever affected.

The Herculean task is taking the first step by running that initial experiment. Consider setting aside one hour for your first game-day experiment.

Test something simple that you are sure you have done right, then run a chaos experiment to prove it. If your system passes, great! Then consider automating the experiment to run during the build process in your CI/CD pipeline so you know you never have a regression.

Then, consider picking something small that you suspect could be a problem, and keep going. Follow this pattern for six months, running a small internal game day once every few weeks. After six months, look back and decide whether it was worth it. Your nines will improve, and so will customer experience.

*Don't miss my presentation at DevOps Enterprise Summit London — Virtual, running June 23-25, 2020. I will share insights from my years at Amazon and Netflix, and from a wide array of customers, to help you address the question of how many nines of availability is enough. I'll describe what each level of availability looks like, the challenges faced at each stage, and the trade-offs required to achieve the next nine of uptime.*

## Keep learning