DOI:10.1145/3434224

Article development led by acmqueue
queue.acm.org

Why DevOps encourages us to celebrate outages.

BY THOMAS A. LIMONCELLI

The Time I Stole $10,000 from Bell Labs

IF IT WORKERS fear they will be punished for outages, they will adopt behavior that leads to even larger outages. Instead, we should celebrate our outages: Document them blamelessly, discuss what we've learned from them openly, and spread that knowledge generously. An outage is not an expense. It is an investment in the people who have learned from it. We can maximize that investment through management practices that maximize learning for those involved and by spreading that knowledge across the organization. Managed correctly, every outage makes the organization smarter. In short, the goal should be to create a learning culture—one that seeks to make only new mistakes.

I worked at Bell Labs in New Jersey from 1994 to 2000. I was a systems administrator on a team of people charged with maintaining thousands of computers and the network that connected them. It was intimidating to be surrounded by so many brilliant scientists and engineers, many of whom had written the textbooks I used in college.

One day, I had to make a configuration change to the central router. It is difficult to measure the size of a change. I could say it was a tiny change in that it affected only a few lines of the router's configuration file. On the other hand, it was a big change in that it impacted a network used by thousands of users. It was an important change because an important project was blocked waiting for it to be completed.

I typed the commands to alter the configuration, saved the new configuration, and checked the things I usually check. The change was a success … or so I thought.

Proud of myself, I moved on to other work. A little while later I couldn't connect to most machines on the network. Neither could anyone else. I panicked. Could my change have caused that? Impossible! That was nearly an hour ago.

No, it was definitely my change. There are some typos that don't show any ill effects right away. In this case, a cache was held for 45 minutes. At 46 minutes the router was a very expensive box doing nothing.

I reverted my change, and everything returned to normal.

My father used to joke about weather forecasters. For example, he would say that if they simply predicted that tomorrow the weather would be "about the same as today," they would be accurate 70% of the time where we lived, and perhaps 90% of the time in Los Angeles. By way of analogy, I often joke that during an outage, asking, "What was the last big change we made?" will make you look like a genius 70% of the time, and perhaps 90% of the time in Los Angeles.

Even though my change had been completed nearly an hour ago, it was certainly the most recent big change.

Learning The Wrong Lesson

Sitting at my desk, I did a little back-of-the-envelope math to calculate the cost

IMAGE BY ANDRE BOUKREEV

of this outage: number of people affected, estimated average Bell Labs salary, the likelihood people were at their computers at the time...

My calculations estimated that the outage cost the company about $10,000 and affected thousands of people.

I panicked.

I hid in my office.

Prayed that nobody would say anything or notice.

And guess what? Nobody did.

I dodged the bullet. Or, maybe someone else was blamed. I didn't care as long as I didn't get in trouble.

I learned an important lesson that day: Don't touch that particular configuration parameter and, if you do, always wait at least an hour before you declare success.

And *Then* It Happened

It may surprise you, but that outage was not the time I stole $10,000 from Bell Labs.

It was an honest mistake, a beginner's mistake. Chalk it up to the cost of learning on the job. While my fear and embarrassment were real, most likely those feelings were unfounded. I had an awesome boss who would have protected me. Plus, LANs were pretty unreliable back then, and most of the affected users probably took the outage in stride.

The stealing was what happened next.

A month later another person on my team made the exact same mistake. The outage was the same size, duration, and estimated $10,000 cost.

That outage definitely would have been prevented if I had shared what I learned from *my* outage. I knew it then, and years later I still believe it.

The stealing wasn't a result of my outage. It was how I responded to my outage. I had robbed the company of the opportunity to learn and improve.

Fear Drives Negative Behavior

If people are afraid they will be punished for outages, the result will be self-protective behaviors that have unintended negative side effects. These side effects can lead to more frequent and bigger outages.

Some of these negative behaviors include:

▸ *Hiding mistakes.* This blocks organizational learning and can rob the company of potential improvements.

▸ *Hiding problems.* People will intentionally hide a problem if there is a culture of shooting the messenger. This leads to problems being discovered only when they are too big to be invisible.

▸ *Ignoring small problems.* People will ignore a small problem out of fear that fixing it, which is often error prone, may lead to an outage that they will be blamed for. This leads to problems being addressed only when they are big enough, and expensive enough, that they can't be ignored.

▸ *Shutting down communication.* Fear has a chilling effect that prevents the open and honest communication required to work well as a team, and prevents teams from working well together.

▸ *Losing the best skilled people.* If they don't choose to leave the toxic culture, the toxic culture will force them out.

If your organization runs from one major disaster to another, maybe the problem is a corporate culture that unintentionally drives these behaviors.

Want a more reliable system? You will need a team that is highly skilled, communicates effectively, and fixes problems when they are small. Fear creates the opposite.

Frequently after a major outage or other problem, we see CEOs or politicians claiming they will "fire the responsible person." Congrats, dude. You just helped assure a future full of bigger, more frequent outages.

I'm not sure where this "fire someone" response comes from. It certainly makes good TV sitcom material. It definitely plays well at a news conference. It's doubtful, however, that MBA programs are teaching future executives that if they fire anyone who makes a mistake, eventually their company will employ only perfect people. On the contrary, firing everyone who makes a mistake will result in a company with no employees, or a company full of people waiting to be fired when management discovers that they are human. Yet, frequently CEOs and politicians are pressured to prove their seriousness by firing someone. How many times did pundits speculate when or who President Obama would fire during the stunted launch of the Affordable Care Act website?

Such toxic cultures make it difficult to hire the best. Word travels fast. If your company has a reputation for blaming and shaming, word will spread, and top talent will avoid you.

DevOps Celebrates Mistakes

DevOps culture has a more enlightened attitude about outages. Rather than hiding them or pretending they didn't happen, we document them. Rather than punishing anyone, we encourage responsibility and accountability.

It is irrational to believe that a complex system can be 100% free of outages. Therefore, punishing people or getting angry at someone because of an outage is irrational.

A more enlightened stance is to view each outage as an unplanned investment. I didn't create a $10,000 outage at Bell Labs. Bell Labs invested $10,000 in my education. To make the most out of that investment, the education should be put to the best use possible.

Learning from incidents does not magically happen. The desire may exist, but more is required. The shift from blame to learning requires a commitment from executives, management, and non-management alike. Executives must model blameless behavior and encourage learning. Management must create processes that enable learning. Project managers need to allocate space and time for these processes to happen. Everyone must learn to be more open and humble.

DevOps culture encourages writing a postmortem report to capture what happened and what was learned. Focusing on the question, "What was learned?" rather than, "Why did this happen?" or "Who's to blame?" creates a culture of learning and improvement.

Postmortems help us to be accountable. The word *accountable* literally means "to account for what happened"—that is, to tell the story. The postmortem should focus on a timeline of what happened and what was learned.

A postmortem report usually concludes with a list of what should be done to prevent similar events in the future. Each item in that bullet list is triaged like any other bug or feature request. New thinking in DevOps suggests that focusing on this list is a distraction from the learning process. Some organizations have started to separate out the process of identifying these follow-up projects by moving that discussion to a separate meeting conducted afterwards, often with a smaller group of people.

Dave Zwieback's excellent book *Beyond Blame: Learning from Failure and Success* discourages the use of the term *postmortem* and instead calls the process a *learning review*. A learning review can be used to analyze any event. There is as much to learn from success as from failure.

Large events (outages and successes) are chock-full of learning opportunities. Those involved should be encouraged to share what was learned even more widely by presenting on the topic. While at Google, I frequently saw SREs (site reliability engineers) travel to far-flung offices to give presentations about a recent outage and how the local team could leverage what was learned. Talk about the opposite of hiding in shame!

When an outage affects customers, a public version of the postmortem report should be made available. Public relations and legal departments will likely break into a sweat the first time this is suggested, but companies are learning that public postmortems actually build customer confidence and loyalty.

The best public postmortems present what has been learned in ways that are useful to customers. The highest compliment you can get is, "I learned so much from your public postmortem that it made me better at my job!" What customers mean by such a compliment is that either they have learned a practice they can adopt at their company, or they have learned previously obscured details about your product that help them do their jobs better when using your product. The loyalty this creates is priceless.

It is important that communication with the public be authentic. Sound like a human, not a press agent. Admit failure. Write in the first person and show real remorse. Avoid the temptation to minimize the full impact of the outage by saying, "We regret the impact it may have had on our users and customers." *May* have had an impact? There *was* impact! Otherwise, you wouldn't be sending this message. Say, "We apologize for the impact this outage had on our customers." Your legal and public relations departments may have trouble with this at first, but they need to learn that today's customers are astute judges of authenticity.

Conclusion

Obviously, I didn't literally steal $10,000 from Bell Labs. But I did rob my team of learning from my mistake in a way that could have improved the entire team. I learned my lesson, and I'm glad to have the opportunity to share it with you.

Nobody loves outages. They are inevitable, so we might as well make the most of them. Through blameless postmortems and other techniques we can create a culture where every outage results in the organization becoming smarter.

If we do it right, the only mistakes we make will be new mistakes.

If you would like to learn more about this subject, I recommend Zweiback's Beyond Blame: Learning from Failure and Success *and chapter 14 of* The Practice of Cloud System Administration, *the book I wrote with Strata Chalup and Christina Hogan.*

Thomas A. Limoncelli is the SRE manager at Stack Overflow Inc. in New York City. His books include *The Practice of System and Network Administration*, *The Practice of Cloud System Administration*, and *Time Management for System Administrators.* He blogs at EverythingSysadmin.com and tweets at @YesThatTom.