Managing Incidents

ANDREW STRIBBLEHILL AND KAVITA GULIANI

Andrew Stribblehill has been a Site Reliability Engineer at Google since 2006 and has worked on ad serving, database, billing pipeline, and fraud detection teams. Before joining Google, he studied theoretical physics at Durham University after which he specialized in high-performance computing for the university. stribb@google.com

Kavita Guliani is a Technical Writer for Technical Infrastructure and Site Reliability Engineering at Google Mountain View. Before working at Google, Kavita worked for companies like Symantec, Cisco, and Lam Research Corporation. She holds a degree in English from Delhi University and studied technical writing at San Jose State University. kguliani@google.com

Effective incident management is key to limiting the disruption caused by an incident and restoring normal business operations as quickly as possible. Without gaming it out in advance, and with adrenalin running through the veins, principled incident management can go out of the window in real-life situations.

This article walks you through a pen portrait of an incident that spirals out of control due to ad hoc incident management practices, outlines a well-managed approach, and reviews how the same incident might have looked with well-functioning incident management.

Unmanaged Incidents

Put yourself in the shoes of Mary, the on-call engineer for The Firm. It's 2 p.m. on a Friday and your pager has just exploded. Black-box monitoring tells you that your service has stopped serving any traffic in an entire datacenter. With a sigh, you put down your coffee and set about the job of fixing it. A few minutes into the task, another alert tells you that a second datacenter has stopped serving. And then a third, out of five. Worse, there is more traffic than the remaining two datacenters can handle; they start to overload. Before you know it, your service is overloaded and unable to serve any requests.

You stare at the logs for what seems like an eternity. There are thousands of lines that suggest there's an error in one of the recently updated modules, so you decide to revert the servers to the previous release. When you see that this hasn't helped, you call Josephine, who wrote most of the code for the now-hemorrhaging service. Reminding you that it's 3:30 a.m. in her time zone, she blearily agrees to log in and take a look. Your colleagues Sabrina and Robin start poking around from their own terminals. "Just looking," they tell you.

Now one of the suits has phoned your boss and is angrily demanding to know why he wasn't informed about this "business-critical service's total meltdown." Independently, the vice presidents are nagging you for an ETA and to understand "How could this possibly have happened?" You would sympathize but that takes cognitive effort that you are holding in reserve for your job. They call on their prior engineering experience and make irrelevant but hard-to-refute comments: "Increase the page size" sticks in the mind.

Time passes and the two remaining datacenters fail completely. Unbeknown to you, the sleep-addled Josephine had called Malcolm. He had a brainwave: something about CPU affinity. He felt certain that he could optimize your remaining server processes if he could just deploy this one simple change to the production environment, so he tried it. Within seconds, the servers restarted, picking up the change. And died.

The Anatomy of an Unmanaged Incident

Note that everybody in the above picture was doing their job, as they saw it. How could things go so wrong? Here are some hazards to watch out for:

Sharp Focus on the Technical Problem

We tend to hire people like Mary for their technical prowess. So it's unsurprising that she was busy making operational changes to the system, trying valiantly to solve the problem. She wasn't in a position to think about how to mitigate the problem: the technical task at hand was overwhelming.

Poor Communication

For the same reason, she was far too busy to communicate clearly. Nobody knew what others were doing. Business leaders were angered; customers frustrated; the would-be volunteers switched off or, at any rate, were not used effectively.

Freelancing

Malcolm was making changes to the system with the best of intentions. But Malcolm hadn't coordinated his actions with anyone, perhaps because no one, not even Mary, was actually in charge of troubleshooting. Whatever the case, his changes made a bad situation far worse.

Elements of Incident-Management Process

Incident-management skills and practices exist to channel the energies of enthusiastic individuals. Google's incident-management system is based on the Incident Command System [1], known for its clarity and scalability.

A well-designed incident-management process has the following features:

Recursive Separation of Responsibilities

It's important to make sure that everybody involved in the incident knows their role and doesn't stray onto someone else's turf. Somewhat counterintuitively, this allows people more autonomy than they might otherwise have since they need not second-guess their colleagues.

If the load on a given member becomes excessive, they need to ask the planning lead for more staff, then delegate work, up to and including the creation of sub-incidents. Less extreme delegation involves role leaders factoring out noisy components to colleagues who report high-level information back up to them.

There are several distinct roles that are both easy to identify and sufficiently separable that they can be worked on by different people:

Incident Command: The incident commander holds the high-level state about the incident. They structure the incident response task force, assigning responsibilities according to the need. De facto, they hold all positions that they have not delegated. If appropriate, they can suggest ways around problems to Ops, thus helping them to avoid fixating on unhelpful parts of the problem space.

Operational Work: The Ops lead works with the incident commander to respond to the incident by applying operational tools to the task at hand.

Communication: People filling this role are to be the public face of the incident-response task force. They might take on the synopsis-writing component of maintaining the incident document; they almost certainly send emails periodically to keep others aware of the progress made.

Planning: Dealing with longer-term issues than Ops, people in the Planning role support Ops, match offers of support with roles, file bugs, and track how the system has diverged from the norm so that it can be reverted once the trouble is over.

A Recognized Command Post

Interested parties need to understand where they can interact with the incident commander. In many situations, taking the incident task force members into a known room is effective; scrawl "War Room" on the door if that helps. Others may prefer to remain at their desk but keep a weather eye on email and IRC.

We have found IRC to be a huge boon in incident response. It is very reliable and can be used as a log to scroll back through, which is invaluable in keeping detailed state changes in mind. We've written bots that log the traffic (helpful for postmortem analysis) and others that report interesting events such as alerts to the channel. IRC is also a convenient medium over which geographically distributed teams can coordinate.

Live Incident State Document

The most important responsibility of the incident commander is to keep a living incident document. This can be in a wiki but it helps if it's editable by several people concurrently. Most teams at Google use Google Docs for this, although Google Docs SREs use Google Sites: after all, depending on the software you are trying to fix as part of your incident-management system is unlikely to end well.

See Appendix 1 for a sample incident document. They can be messy but they must be functional, so be sure to make a template first so that it's easy to start your incident documents. Keep the most important information at the top. Retain them for postmortem analysis and, if necessary, meta-analysis.

Clear, Live Handoff

It's essential that the post of incident commander be clearly handed off at the end of the working day. If outgoing and new people are not both in the same location, a simple and safe way is to update the new incident commander over the phone or video call. Once the new incident commander is fully apprised, the outgoing commander should explicitly make the handoff ("You're now the incident commander; okay?") and not leave the call until there is firm acknowledgment.

A Managed Incident

Mary is into her third coffee of the day: it's 2 p.m. The pager's harsh tone surprises her and she gulps the drink down. Problem. A datacenter has stopped serving traffic. She starts to investigate. Shortly another alert fires and the second datacenter out of five is out of order. Since this is a rapidly growing issue, she knows that she'll benefit from the structure of her incident-management framework.

Mary snags Sabrina. "Can you take command?" Nodding her agreement, Sabrina quickly gets a rundown of the story so far, writes it up as an email, and sends it to the prearranged mailing list. Sabrina recognizes that she doesn't know the impact yet, so she asks Mary. "None so far. Let's just hope we don't lose a third datacenter." She copies it into a live incident document, marking it as the summary.

When the third alert fires, Sabrina sees it among the debugging chatter on IRC and quickly follows up to the email thread with an update: the VPs are being informed about the high-level status of the incident without being bothered with the minutiae. She seeks out an external communications representative so they can start drafting their messaging. "How's it going, Mary?" asks Sabrina. "Want me to find the developer on call?" "Sure."

By the time Josephine logs in, Robin has already volunteered to help out. Sabrina reminds him and Josephine that they are to keep Mary informed of any actions they are taking unless Mary delegates to them. They quickly learn the current situation by reading the incident document.

Mary has, by now, tried the old binary release and found it wanting: she mutters this and Robin updates IRC to say that it didn't work. Sabrina pastes it into the document.

At 5 p.m., Sabrina starts finding replacement staff to take on the incident: she and her colleagues are about to go home. Sabrina updates the incident document. At 5:45 she holds a brief phone conference to make sure that everyone's aware of the current situation. By 6 p.m., they're all tired, and they hand off their responsibilities to their colleagues in the remote office.

When to Declare an Incident

It is better to declare an incident early and then find a simple fix and shut it down than to have to spin up the incident-management framework hours into a burgeoning problem. Set clear conditions for declaring an incident. My team follows these broad guidelines. If any of the following is true, it's an incident:

- Do you need to involve a second team in fixing the problem?
- Is it a customer-visible outage?
- Is it an unsolved issue even after an hour's concentrated analysis?

Incident-management proficiency atrophies quickly. So what is an engineer supposed to do? Have more incidents? Fortunately, the incident-management framework can be followed for other operational changes that need to span time zones and/or teams. If you use it frequently as a regular part of your change-management procedures, it'll be easy to pick up when needed. If your organization performs disaster-recovery testing (you should: it's fun), incident-management should be part of that process. We often game out the response to an on-call issue to further familiarize ourselves.

Resource

[1] https://www.osha.gov/SLTC/etools/ics/what_is_ics.html.

Appendix 1: Incident Template

Incident <WRITE TITLE HERE>: yyyy-mm-dd

Incident management info: http://<your-internal-site>/incident-management-cheat-sheet

~TWO-LINE SUMMARY; COMMS-LEAD TO MAINTAIN CONTINUALLY.

Status: active
Command post(s): #incident on IRC

Command Hierarchy (all responders)

- Current Incident Commander: XXX
  - Operations lead: XXX
  - Planning lead: *role filled by Command*
  - Communications lead: *role filled by Command*
- Next Incident Commander: *undefined*

Detailed Status (updated at yyyy-mm-dd hh:mm UTC by $user)
UPDATE EVERY 4 HOURS AND AT HANDOFF

Exit Criteria

TODO list and bugs filed

Incident timeline, most recent first: times are in UTC
yyyy-mm-dd
hh:mm   USERNAME: XXX