Postmortems with Ayelet Sachto

Ayelet Sachto offers advice on creating an actionable, transparent, and blameless postmortem culture.

MP: Hello, and welcome to Episode 9 of the Google SRE Podcast, or as we affectionately refer to it, the Prodcast. This is actually going to be the final episode of our initial season. And last week we got to talk about incident management and on-call response, and this week we're moving on to what happens after the incident to close things out.

So here with us today to talk with us about postmortems is Ayelet. Would you like to go ahead and introduce yourself?

Ayelet: Yes, thank you. Happy to be here. I'm Ayelet Sachto. I'm currently a site reliability engineer in GKE (Google Kubernetes Engine) SRE team in London and formerly a strategic cloud engineer and leading PSO (Professional Services Organization) SRE efforts in EMEA. Incident management and postmortems are not new to me, as I've been living and bleeding production on a large scale for almost two decades now, developing and architecting large-scale application and data flow while implementing dev-ops practices and SRE technologies. Most of them cover production on-call, of course. And outside of my main role, I also volunteer in the tech community as a mentor, public speaker, and organizer.

Viv: Oh, you've been busy. Good busy.

Ayelet: Trying to be, yes. [Laughs]

MP: So for those of our listeners that might not be familiar with the term

"postmortem," could you give us your definition of what makes a postmortem a postmortem?

Ayelet: To answer what is postmortems or what makes a postmortem a postmortem, it's also important to understand why we write postmortems and what is the problem that we are trying to solve with postmortems.

Postmortems are a written record of an incident. They should include the actions taken to mitigate customer impact and resolve the incident, the stages that more often than not will be separated, the impact itself, the root causes, and—important to emphasize root cause or root causes and not just the symptoms—the follow-up actions to prevent it from reoccurring. Our goal with postmortem is to prevent incidents from reoccurring and at least reducing the likelihood of impact of future outages.

There are a lot of things that we can do in order to reduce the impact of incidents or to reduce the frequency of incidents. But the data that are coming from postmortems is crucial in order to understand what we can prioritize and what we should prioritize. Postmortem is our tool to learn from our failures. And unless we have some formalized processes of learning from these incidents in place, they will reoccur. So postmortems provide us a method to learn not only from our failures, but from others. And that's why it's really important also to share them globally and not hide them.

But to be able to learn from them, they need to be blameless because that will prevent side conversations about who did what and might be at fault, maybe. We don't want people to hide information or not to declare incidents because they are afraid of punishment. We want to encourage [the] culture that people are not afraid to take risks.

Viv: That makes sense. I like it. It's a lot of transparency around the process, just being really honest and open. And it sounds like when this goes well, then it does make for a stronger community and for a stronger product moving forward. So how do you make sure that those postmortems fill that? Like sometimes when you're like, "Oh, I'm going to document or record something afterwards," you just

dash off a couple of things because you're tired and then you're done, right? So how can we make sure we write good postmortems?

Ayelet: The TL;DR version of how we ensure that postmortems are written and those are kept with specific guidelines or standards, we need strong processes for that. And we need our systems and tools to be in place in order to make it easier for people. When you are asking what will be considered as a good postmortem, then we need to think about a checklist of things that are covered.

One will be the information that I mentioned, the milestone, the stages. And the timeline is very, very important; sometimes people are missing that. So they are writing when an incident started and when the impact may be mitigated, but they're not capturing all the stages.

And it's important to understand the time it took for each step, because with this information we can improve and understand where we have gaps. Was it in the time it took us to detect the issue, when we actually got alerted on it, or was it escalated by a customer? That is very important to understand. If we mitigate it really, really early on so we didn't have customer impact, but the incident itself was closed a few hours or after a day, that is also very important to understand because we actually mitigated. So we don't want to account for that time, but we do want to account for the follow-up actions and the time overall.

If we're talking about, also, things that we want to ensure in postmortems, we want to avoid any blameful language. We don't want to focus on people, as I mentioned; we want to focus on improvement and promoting an iterative and collaborative process. So it's important that whoever reviews— and yes, postmortems need to be reviewed— needs to make sure that we have the technical information, we have all the details, but we also have a language that is not pointing fingers, that is blameless, and encourages us to take those risks.

Another important point is that a good postmortem should include action items. Ideally, we want to translate our learnings from the postmortem to concrete action items. Otherwise it means that we are not improving. Think about it. How

can we fix the problem if we don't follow it up with action items?

MP: So I wanted to go a little bit more into blamelessness. Is that as simple as keeping language fact-focused, as in "this is what happened, an engineer pushed a config that caused," or is there more of a secret sauce to it, or is it really just as easy as recognizing that humans aren't perfect, humans make mistakes, and these are the facts of what happened?

Ayelet: I wish that it was easy. Like any cultural thing and any cultural transformation or change, it's more complicated than a checklist or making sure that our language is aligned with specific terminology.

But just for the favor of our audience, blamelessness is the notion of switching responsibility from people to systems and processes. We want to avoid that finger-pointing that I mentioned. For one of you who've been in an organization [where] the first question that the manager asks after an incident is "who did it?," probably you can recall, and hopefully not anymore, that you don't really feel empowered to make decisions and employees might feel like they are fearing for their job, they are avoiding taking any risks or changes, and we want to make those changes because we cannot improve without making changes, without taking risks. So if we want innovation, we want to take those risks. And if we want to take risks, we need to accept that failure will happen.

And instead of focusing on the people, we need to focus on the system and process that allowed it to happen. Actually, in one of Ben Treynor's emails, he outlined it really well that mistakes are a valuable opportunity to learn and improve. And if we are missing that opportunity, if we are not learning from our mistakes, we are taking the cost of the mistake, but without the benefit of learning from it. In that regard also, I mentioned the cultural impact and the cultural change in some cases. And blamelessness is also important to foster psychological safety in our team.

MP: So can you go into a little bit more detail about what it means to have psychological safety on a team, particularly in the context of SRE?

Ayelet: So we want people to feel safe enough to ask the right questions that may lead to identifying the root cause of an incident, for example. We want people to feel safe enough, not to hide incidents or problems because we want to improve. In order to demonstrate that, let me ask you something: have you ever had a question that you didn't ask or an idea that you didn't share with your team?

Viv: I've definitely not asked a lot of questions.

MP: Here and there, most certainly.

Ayelet: So I don't want to put you on the spot, but usually when people are not feeling safe enough not to ask questions or not to raise ideas, that can be a sign of a lack of psychological safety in that team. And psychological safety is a belief that while one will not be punished or humiliated for speaking up with ideas, questions, concerns, or mistakes, a culture of psychological safety makes it understood that things will break, failures will happen, and those breakage should be widely communicated.

And psychological safety is critical in order to prevent that incident [from] hiding. Because we want to improve, we don't want to hide things. Actually, a really good notion for that is thinking about it: if you are not asking the question, if you are not sharing your idea, you are actually preventing the rest of your team, the rest of the people from the opportunities to learn. Actually, I'm asking a lot of the time questions that for me in my head I'm like, "No, don't ask that, that sounds stupid!" Or "everybody knows that!" or something like that. But what I'm trying to do is actually ask myself, and what if it's not? What if one person in the team can benefit from that question? What if someone that maybe is more junior, maybe less vocal, maybe they are new to the team, maybe they don't feel safe enough to ask that question, what if they have the same question, but they don't feel safe enough to say it? I'm preventing them from the opportunity to learn. I'm thinking about imposter syndrome and self-doubt; I think that it's something that a lot of people in Google share in general. So what if your idea is not that stupid? What if actually nobody's thought about it or didn't think about that perspective?

So in order to grow, even if that idea sounds not as amazing as others, it's

important to voice them out. And more importantly, it's important for managers [and] organizations to create an environment that people feel safe enough and empowered enough to voice those.

Viv: I like it. And I'm really glad you touched on maybe the fear of speaking up or imposter syndrome or other things that might come from the person itself. In my case, I feel like my team does have an environment where folks are encouraged to ask questions and such, but I think it is still hard for me to do that sometimes because like you said, it is quite nerve-wracking and it's like, "What if everybody already knows this and I'm just the only one who's not caught up here?" Like you said, "what if this is stupid?" So it is tricky. Even once you have the environment, there are still hurdles and maybe you think you have a safe environment and it actually could be better. So lots of factors come into play.

Ayelet: And actually the solution is exactly that, is asking questions, is actually encouraging others to ask questions as well. One of my favorite suggestions actually is to lead by example, and it's with questions, it's with other behaviors that you value, it's not just for managers or tech leads, what I'm going to say, but acknowledge that you also don't know everything, acknowledge that you might also need to ask questions, and model curiosity as well. This way, your team—and again, when I'm saying your team it's not just as leaders, it's not just as managers, it's for all of us; we are making the culture of our teams as well—encourage each other, and don't discourage each other from asking questions.

I mentioned before that actually creating psychological safety is not as simple and actually the idea of just asking questions sounds very simplistic, but it takes time and any change takes time. So for teams that maybe are not there yet, start simple, iterate, like we're doing with software engineering problems. It will take time, but the important part is that we'll improve with time and do some strides to [a] better position.

MP: There are a couple other things beyond blamelessness and psychological safety that I wanted to get to today. Jumping back a little bit, you talked about action items and follow-ups from postmortems. Is there a right way or wrong way

to do action items from a postmortem?

Ayelet: First of all, if you are actually taking action items, then you are in a good enough standing. But when we're saying "action items," those need to be concrete. And those need to be assigned, and ideally with an ETA. Now, every team is a bit different and the work processes are different, so it's possible that for one team it's okay to put it on their board; for another team it will be to actually assign it to a person; for other teams and other also action items, it can be that it will be to set a meeting in the calendar.

So there's no one way to actually create a follow-up, but you need to ensure that a follow-up will happen. And that happens usually again with assigning a specific person. And usually that person will not necessarily be the person that will resolve everything, but they can be the ones that will triage it, that will create the bug, that will set up the meeting, that will do some sort of an action that will promote that action item.

Another thing to consider is that if we have learnings in our postmortems, and we didn't really talk about all the parts in the postmortems, but in postmortems, we usually have what went well and what could be improved—and for us, usually, in Google, we have also where we got lucky. Usually we want to translate the thing that we need to improve, what could be better, and where we got lucky to some sort of an action item. So for example, if we didn't have a monitor alert or we didn't have an alert on something, but a developer looked at the dashboard exactly at this time because they developed a new feature, so we got lucky, but maybe next time we won't get lucky. So ideally one action item will be to create an alert to create a page. So next time, if someone will not look at the dashboard, we'll still catch it.

Viv: Who owns those action items in the sense of like, is that usually on one person, two people? Again, with the idea of not making someone feel like the incident is their fault could also extend into how you follow up, right?

Ayelet: So that's a tricky question because even ownership of postmortems can be different between teams and between organizations. So you're alluding also

that postmortems themselves need to have an owner and that owner needs to not just write it, but make sure that it's being reviewed and being approved and then being publicized—so shared globally, widely. And in that postmortem, they need to also assign the action items. Depends on the action items, depends on the organization, depends on the team. The action items may be assigned to, let's say, the rightful owner in the policy. It can be whoever is the subject matter expert. It can be based on a discussion. It can be part of the postmortem itself to identify who are the right people that need to be involved in the postmortem itself. So there is no one answer. And like we say in a lot of other things, it depends. And it's okay also to say, "We don't know who is the owner of implementing X, Y, Z." So the action items will be to create a bug and to start a discussion between those teams to make sure that it is being resolved by a specific date and time.

MP: I think there's another thing we've avoided talking about so far is the relationship between incidents and postmortems, because I know from my own personal experience, and I assume most people can imagine, you're not writing up a whole postmortem for every little outage that happens. So where is this tipping point where it was, "Oh, that was just a little outage," to "Oh, this was an incident, we need to do a postmortem for this." Where is that line? Is there even a clear line?

Ayelet: Again, a tricky question. The relationship between incidents and postmortems are not just, "We are opening postmortems for incidents." For incidents and severe enough incidents, we usually will open a postmortem, and I'll expand on it in a second, but also it goes both ways. So when we have a critical mass of postmortems, that can lead us to identifying patterns; that can lead us to actually reducing either incidents, again, both the volume of incidents and also the frequency of them. But in order to do that, we need to rely on postmortem data to prioritize what and how we should invest time and priorities.

Saying that, you did touch on an important point of *when* we should write a postmortem. Let's say that we have a bleeding money incident, Severity 1 incident. It's no brainer. If you open an OMG (Outage Management at Google), you'll write a postmortem, but there's also other scenarios that we want to write

postmortems that are not specifically for incidents or not incidents that are in a specific severity. And I mentioned, it depends, and here it also depends. Each team can define what are the criteria for them that they will write an incident.

So some scenarios can be, we have an incident with customer impact. Those are [a] very common baseline of writing a postmortem. And usually there is a strong agreement on that. In Google, we have SLOs, service level objectives. If we are breaching those because of an incident, then we definitely need to write a postmortem as well. But there's also other scenarios. For example, what if we have a case of data loss and we don't have a direct impact on the customer yet? So it's a potential customer impact, but at the moment we don't. We will still want to create a record of what happened, how it happened to keep track, to triage it. We still want that learning. And we achieve that by having postmortems.

So other cases that we might want to have a postmortem, even if we didn't declare an incident, or we didn't declare, let's say, Priority 1 or Priority 0—depends on the organization—incident. As I mentioned, user-visible downtime usually comes with an incident with some sort of P0, P1, but it can be also the aggregation of the service. It can be also in a case when an on-caller intervenes. It can be also, we had a release and we had to roll back. We rolled back before we had a customer impact or the customer impact was so minor that we didn't open an incident, but we still want to keep track of that information. And maybe we routed traffic because of that. We might want to create a postmortem in case we have a lower priority incident, but with a resolution time that is above some threshold; they took longer to solve. In case we had the monitor failures—so in case our tools fails us—it's not customer impact, but in case we had a customer impact, we wouldn't know. So we want to understand. And in some organizations, [those] monitoring failures, those tools themselves, we'll also need to open an incident with some priority level.

So again, depends on the organization, depends on the team, but it's not one-to-one. There's a lot of cases that we would want to open a postmortem that maybe is not an incident with a specific priority or severity.

MP: The near-misses also generally are at this really broad category of, maybe

we want to take a closer look at this, even though it wasn't bad this time.

Ayelet: Yes, exactly.

MP: So I think, wrapping up here for today, what would be your final thoughts to those of us that want to write better postmortems in our day-to-day? What is the biggest piece of advice you have to offer us?

Ayelet: So actually one of my recommendations is not just writing a postmortem. A lot of organizations are writing a postmortem, are capturing those notes, but after that are not sharing them or not sharing it publicly enough, not widely enough. And it's a miss also. So let's say, you wrote a postmortem and it can be a very lean one, it can be the most amazing postmortem ever, if you are not sharing it, other people will not be able to learn from it.

So one of the things that I keep seeing with organizations that we work with is that they're actually writing postmortems. They are thinking about the language itself. They are trying to capture also action items and improve, but they are not sharing it. And usually in that step where people are talking about it—I'm actually sharing an example pre-Google time that actually because someone wrote a postmortem for an issue that was in a production system that was not under my, let's say, ownership at the time, and they shared it publicly, they shared it with R&D in general—I actually was able to prevent an incident in our own system because they shared a problem that we also encountered. And when we connected the dots, it actually showed the deeper problem in our systems. And we were able to actually solve the root cause for both of our systems.

So sometimes we're saying, we don't want to share it too broadly because it's not relevant for them, it's not the same system, it's not their ownership. And they are keeping that in the scope of the team or in the scope of even sometimes their PA, the organization, but others can learn from it and can improve from it. I do encourage every organization to share it as widely as possible and create the infrastructure to share it widely as well. Of course, in some cases, we need to redact some information, especially customer data and things like that, but it shouldn't be a block here. We should find a workaround for that as well. We have

solutions, in fact, for that.

MP: Well, that was some really great advice. I think that's definitely something that I'm going to want to incorporate more into my daily work, making sure that I'm looking out for postmortems from my peers and that I'm actually reading them thoroughly and trying to internalize that information, and also making sure anything I produce gets shared out and widely disseminated as well. So thank you so much for your time today. It's been really great.

Ayelet: Thank you for having me.

MP: And thank you so much for being our ninth and final guest on this first series of the Prodcast.

Viv: Yeah, this is a great way to wrap things up. We got great advice, we talked about postmortems, and maybe we'll write a postmortem for this Prodcast.

Ayelet: Actually, I recommend to do postmortems for a lot of things other than incidents. So retrospective is important.

Viv: Well, if we write one, we will certainly share it with you.

Ayelet: Looking forward to it.

MP: Thank you again.