Blameless

THE COMPLETE GUIDE TO

Pragmatic Incident Command

A practical approach to coordinating, responding, and learning from incidents

VERSION 2

TABLE OF CONTENTS

03 **Introduction**

04 **Pragmatic Incident Command**

Incident Response Roles & Responsibilites
The Importance of the IC
Appointing the IC

09 **The Challenges of Incident Management**

Severity
Ownership
Observability
People Coverage
Conflict and Common Ground
Dealing with Management
Memorialization

19 **Additional Learning Resources**

Introduction

Our SRE and CRE teams put this guide together based on decades of experience in the trenches. The goal is to provide some practical advice on how teams can coordinate and respond to complex, dynamic incidents. After all, incidents are unplanned investments that surface valuable learnings for improvement.

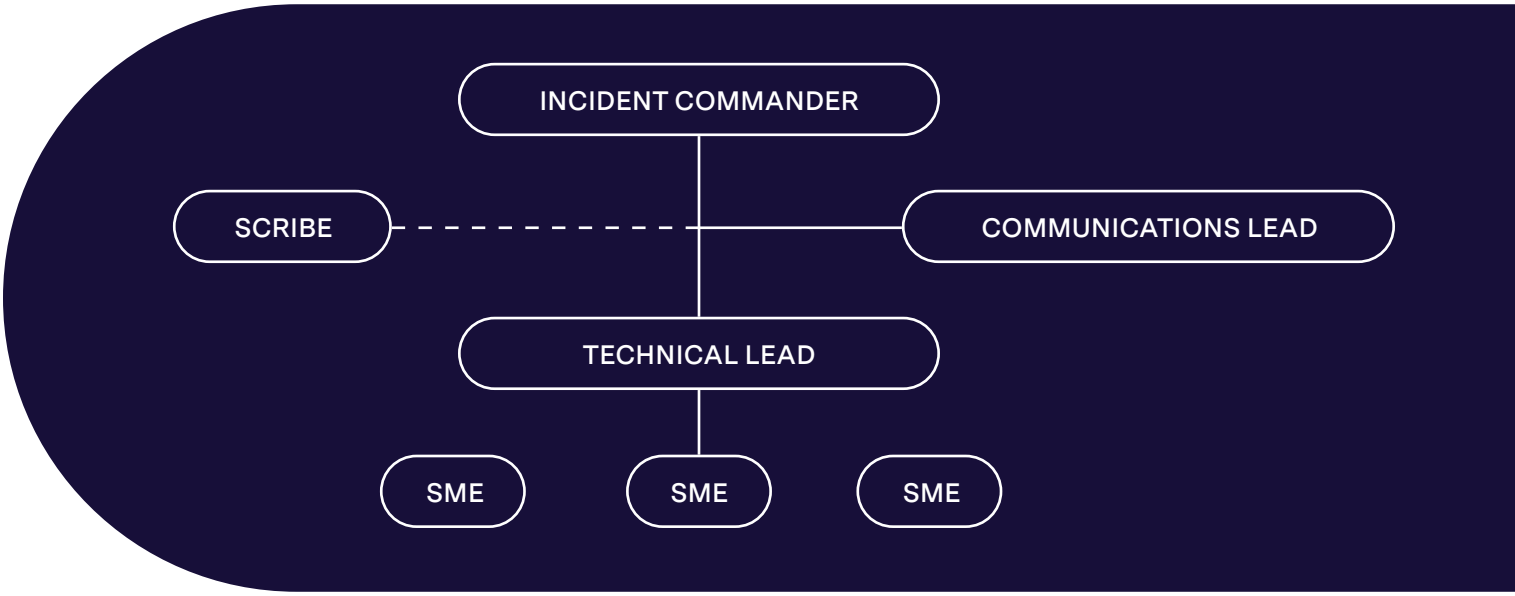
For the purposes of this guide, we define incidents as situations where there is a need for coordination among multiple people working on the same problem. There will be incidents where this is not the case. If you get paged in the middle of the night, go to your laptop, find the service and restart it, that's an incident, but the management part isn't as interesting because there's no burden of coordination. The type of incidents we will explore are those where many people are involved in a stressful situation.

Pragmatic Incident Command

This guide will walk through a pragmatic approach to coordinate and manage incidents given available resources, as well as ways to address common pitfalls and challenges. There are more formal ways to examine incident command and management, but what we're really focused on is how to get the basic skills to manage incidents more effectively.

Incident Response Roles & Responsibilities

The image below is a structure that shows up a lot in IT. It's somewhat based on the traditional ICS practices and structure in dealing with physical incidents, but it's been simplified.



At the top of the structure, you have the Incident Commander (IC). The IC's job is to coordinate the incident, and their ultimate goal is to bring the incident to completion as fast as possible. When the incident has started, generally the person that's first paged is by default the IC, and responsible for helping to kick off the triage process. That doesn't mean that they have to remain the IC, but until someone takes over, the role must be filled. There may not be other people around, so they may have to subsume all of the above roles until others show up in the incident.

A Communications Lead is in charge of communications leadership, though for smaller incidents, this role is typically subsumed by the IC. A Technical Lead is an individual who is knowledgeable in the technical domain in question, and helps to drive the technical resolution by liaising with Subject Matter Experts.

We also have a dotted line here for the Scribe, who is often a person that's maybe not completely active in the incident, but who is transcribing key information during the incident. With today's tools, this role could be automated through bots that execute tasks such as grabbing log files and highlighting key information in the channel. Regardless of how it's done, taking notes during an incident is incredibly important to get the full value of your incident.

Like John Allspaw says

> "Incidents are unplanned investments in learning for your organization.

What that means is each time you have an incident, it reveals something about your system that can be improved or made more robust. That may involve teaching people things, or improving the product or service, but there's always something to be learned. It's harder to learn if the team is solely acting on memory. By having a scribe keep track of what's going on, or doing everything in a Slack channel, you have chat logs to refer back to, and a timeline to get everybody aligned on the same narrative as you put together the post-incident report.

The Importance of the Incident Commander

The above framework can scale to very large incidents. For example, ICS has been used to support the California wildfires, or taking care of people at major events.

However, while the Pragmatic Incident Command approach shares many common qualities, it is not about command-and-control. In most incident scenarios, there may not be enough extra people in the incident to take on the above roles, and the responsibilities all get rolled into one and boiled down to coordinating the response. The main takeaway is that there needs to be a central point of coordination in an incident in order to support adaptability.

The buck ultimately stops with the Incident Commander, defined as the person responsible for coordinating and managing the response. The IC commonly has several responsibilities:

COORDINATES RESPONSE: The IC is always the point of coordination. This is the case even if they are also the one jumping into the system, taking care of communicating with management, trying to figure out what's going on, making changes to fix the issue, and writing a post-incident report to share with the team. The central point of coordination is key to minimizing chaos.

PUSH OUT STATUS UPDATES TO STAKEHOLDERS: This tends to be one of the most important tasks. It can be hard to prioritize as a responder, when you're deep in the weeds of trying to resolve an issue. But it is incredibly important because others get anxious when they don't know what's going on. Status updates add a level of professionalism, calm, and rhythm to the incident that would be hard to come by otherwise.

CONFLICT RESOLUTION: The IC also attempts to take care of conflict resolution. The IC engages the right expert(s), takes in their advice, and makes it happen. But, ultimately, decisions need to be made to move things forward.

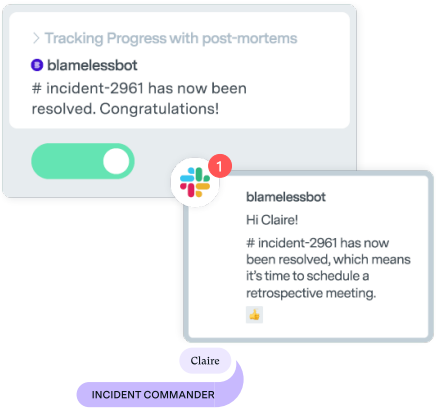
MAINTAIN FOCUS: Don't let nosy managers bug your engineers. Managers should go to the IC with requests so that engineers/responders can focus on their job, because it's really hard to troubleshoot things under stress.

DELEGATE, DELEGATE, DELEGATE: The main takeaway for an IC is to delegate as much as possible. If there is somebody on the team that's available to go and look at logs or help triage, the IC must delegate that task so they can focus on the coordination and drive a path toward resolution.

Appointing the IC

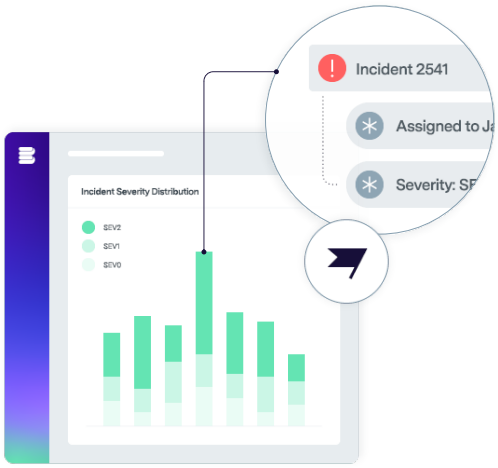
Typically, an individual who is most willing to take ownership of the outcome of an incident as well as the outcome, can be a good candidate to become an IC. Those who have significant depth in the system as well as institutional knowledge are also typically great candidates.

A good practice for larger organizations is to create an official IC rotation, with a group of individuals who are on-call to support incidents that cross teams or hit customer-facing KPIs. For smaller organizations, appointing ICs often may be done more informally, but the main takeaway again is simply to ensure that there is a central point of coordination during incidents.



The Challenges of Incident Management

Incidents are often fluid, and oftentimes IC's will need to think on their feet. There are many challenges and potential failure points, compounded by the fact that incidents are inherently extremely stressful situations. However, anyone can prepare by arming themselves with best practices to address key incident management challenges.



Blameless can help you quickly and easily determine incident severity based on the source of the alert or the service impacted. Incident commanders can either select severity manually, or Blameless can establish severity automatically based on information in your observability solution or service catalog.

Severity

Once an incident is detected, the first point of human intervention is typically when it pages out. When that happens, instinctively your first reaction may be to start working on the problem, log into systems, and pull up dashboards. But the first thing you need to know is the customer impact, and more specifically, what is the impact to your customers, the business, and to the bottom line. This is because the prime directive of incident response is to ensure the customer going through as little duress as possible. In other words, you want the incident to end as soon as possible. From there, you can then triangulate into what is the severity of an incident.
It's important to consider that what different organizations call Sev0's typically mean very different things. They have a lot in common, but

what is the most IMPORTANT thing to keep coming back to is the customer IMPACT.

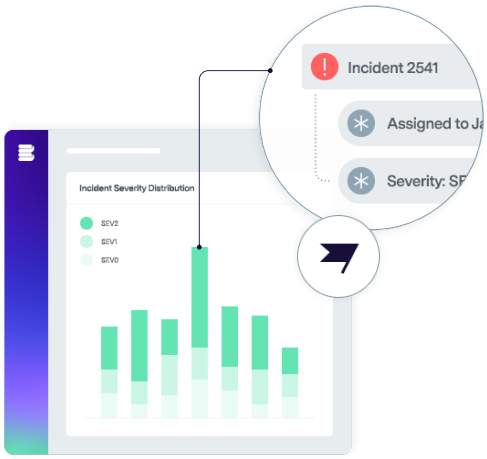
That is how you map from the signals from your technical systems back to one of those severities, which helps you drive the appropriate organizational response.

The concept of the severity level is really about what response you want to get from your organization during the incident. Customer impact is also relative. For example, for the Head of Sales, the customer impact could be Sev0 because they're afraid of losing an account, even though the

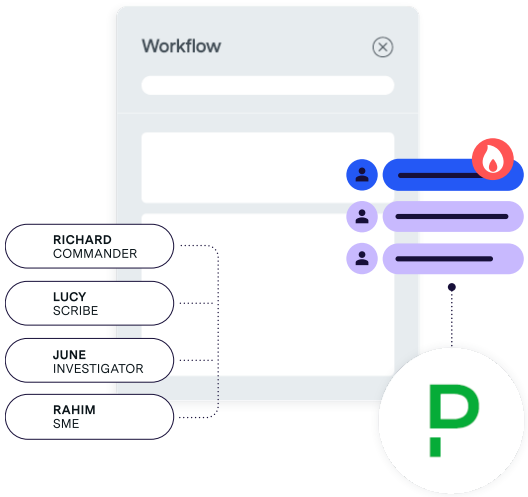
customer in question views something more as an annoyance instead of a major issue. As such, it's important to be mindful of the information that's coming in.

In the example above, the Head of Sales could be calling a Sev0 because they want all hands on deck to fix the problem, but it's important to qualify whether the incident really needs all hands on deck. Prioritization is essential; if every incident is a Sev0, no incidents are a Sev0.

This also means that while ICs do make the call on severity with input from efforts, severity levels are not worth debating over as they can always be changed after the fact. The time spent debating over the right severity level is better spent on resolving the incident as quickly as possible to restore service and minimize customer impact. If the incident is resolved in a timely fashion, the individual pushing for the elevated severity won't be worrying about the semantics; they'll simply be happy that it's been addressed.



An incident's severity can be used to trigger a cascading set of automatic steps defined in a runbook. Blameless allows these workflows to be established ahead of time, and the workflow can be adjusted if and when severity changes or specific incident tags are added.



Blameless can be used to automatically recruit service owners by integrating with on-call tools like Pagerduty or OpsGenie. That same information can also be drawn out of your custom service catalog using webhooks.

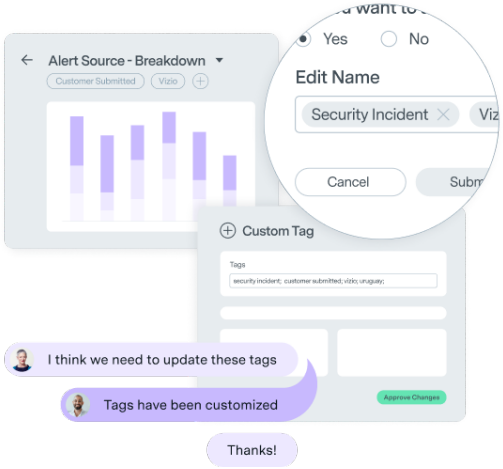
Ownership

Another common challenge in incident management is identifying ownership. For example, if you discover that a service is running slowly and is starting to reject requests, how do you understand who owns that service, who is the right subject matter expert to page, and who to escalate to?

Sometimes you may have to dig into code repositories and track down who's done the most commits, then go back to the org chart and trace it down. This is one of those things you can start to solve ahead of the incident, but during an incident, it's critical to have just enough clues to be able to find the right, responsible person to help drive the incident forward.

An important thing to note is that the owner in question is not only relative to the service identified as the problem, but also relative to other interdependent services that might be affected. It's important, but not always possible, to kind of have a good idea of a chart or graph of who's responsible for each service. In some organizations, given the complexity of microservice architectures, this can change frequently.

Ownership can also be fluid based on who wants to take ownership, who really cares about the service, or who is willing to take on responsibility or things like communicating with stakeholders. Members of our SRE team have recalled incidents where they resorted to Twitter to find other crossvendor folks or experts in the community via Twitter in order to help push forward the incident resolution. Part of ownership is being resourceful in finding the people and resources online to get the problem solved.



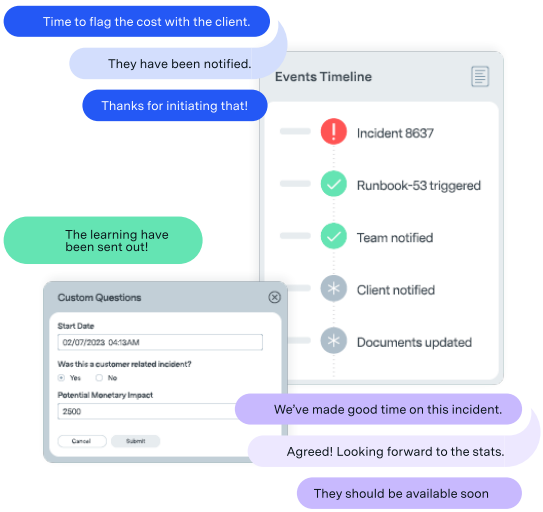
Blameless Reliability Insights Dashboards can be used to easily monitor alert quality, on-call schedule, MTTx metrics or any other items relevant to improving your incident management program. These can be customized and shared with the right personnel to make the data they contain immediately actionable.

Observability

We define observability as how tooling and instrumentation come together to provide visibility into technical systems. At a high level it looks at things such as who has access to the log files and the alerting system. A little bit deeper, it's about how quickly you can get to the data as well as the quality of the data.

Your log files are somewhat definitive, but it's also important to look at your alerting structure. Alerts may not always be set up to be actionable or to be triggerable, so part of alert observability is understanding whether a particular alert is significant at a given time. It's also important to understand where the right dashboards, metrics, telemetry and logs are, and how to get to them. To minimize ad hoc work during incidents, such considerations should be made ahead of time, and observable
platforms should be embedded in your runbooks or wikis to ensure a
repeatable, consistent process.

These tools should all be ready to go in your toolbox, with the right levels of access for all the people who need to work on an incident. It's important to have dashboards, reports, logs, etc. all set up ahead of time, and to have some practice with using them to troubleshoot problems.



Blameless connects directly to your on-call management tools to help ensure that your incident responders aren't unnecessarily bombarded with calls to support an incident. When incident severities change, the messaging protocol changes as well to make sure everyone has the right level of investment

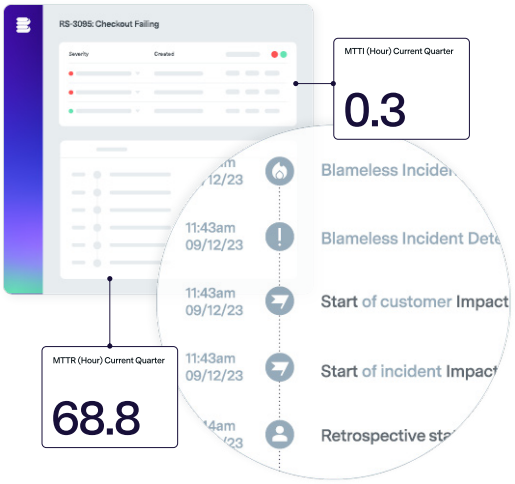
People Coverage

In the context of incident response, people should be thought of as a potential single point of failure (SPOF). This is also important to ensure proper load balancing and sustainability; no one person can be accountable for a system 24x7. For example, if you know you'll be out of commission at a certain time, but you happen to be on call, there is a certain level of responsibility to identify how you can still address the issue or find a backfill.

A few common questions that are important to answer include:

- Do your on-call staff know the rules of engagement of being oncall?
- Do your on-call staff have connectivity during their shift? (wifi, authentication, etc.)
- Do your team members know what hours they are on-call so they can ensure access to everything they need?
- If the primary on-call can't be reached, what does the escalation path look like?

It's also important to have a monitoring system that is set up properly, so that on-call staff don't wind up in pager hell responding to alerts that are not actionable or important. Time should be spent tuning the alerting structure so that on-call staff can still maintain work-life balance. Avoiding pager fatigue is important, as it creates downstream problems such as people ignoring alerts due to low signal:noise.



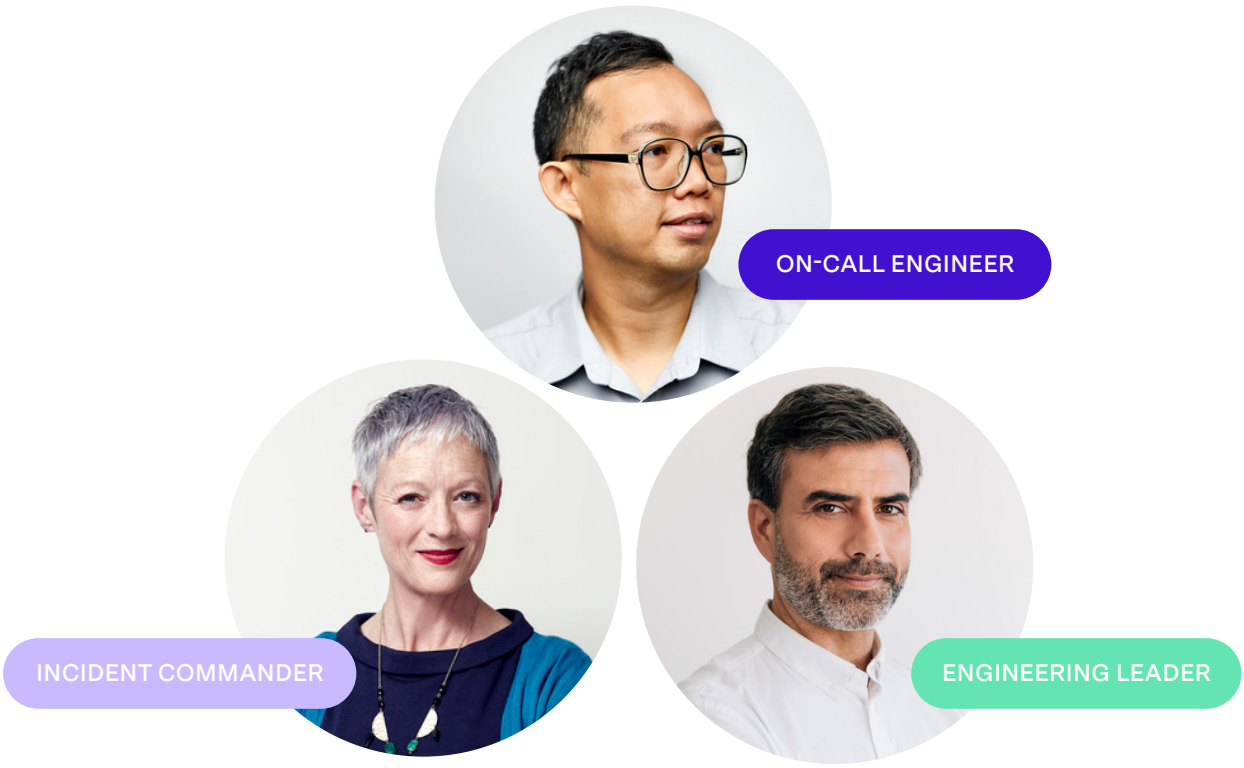
Every relevant detail arising during the incident is captured automatically on the incident timeline. This makes it easy to examine the role that conflicts played in the outcome of things like time to resolve.

Conflict & Common Ground

Sometimes conflict is unavoidable in an incident, and time is wasted arguing and fighting. Sometimes it's easier just to acquiesce in the moment, because if you are indeed correct, eventually evidence will show that to be the case. One way to approach conflict is with the example of Aikido, where you can move out the way and let that energy go past. The intent is to stay focused on what it is that you're trying to do, which is to try to bring the incident to an end as soon as possible. However, if you have a really strong idea of what the solution or next steps should be in an incident, it may be time to engage in conflict to move the incident forward.

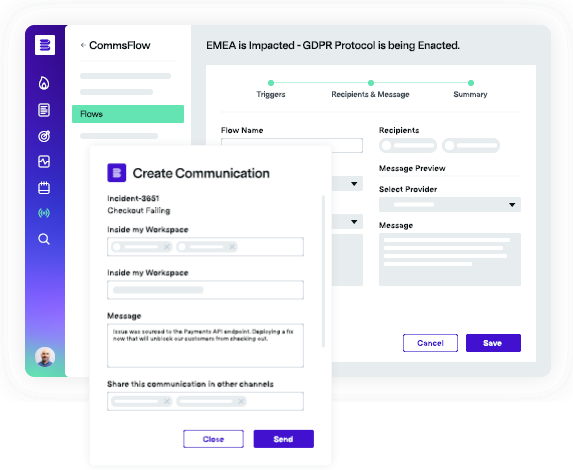
Regardless of if you deflect or engage in conflict, one of the most important notions for ICs to keep in mind is that of staying centered and calm. If you are working through an issue with people that are very excited and activated, they could pull you up into their mental state as well, making it more difficult for you to stay completely focused. Coming to the table with evidence and facts is also important, so that team members remain levelheaded and stay aligned on the common goal.

Another key principle is that of establishing common ground. This refers to finding out what context and knowledge each person has, and how that overlaps in a Venn diagram. When teams get stuck in an incident, they can start to understand how each individual carries different information, and how that causes miscommunications and conflicts. There's an extension of this called Fundamental Common Ground Breakdown.



The game ‘Telephone’ is a good example of this. Say you go and whisper to someone, “The file server’s on fire.” Then they tell the next person, “The server’s on fire.” Then that person tells the next, “There’s a fire.” Information has been added or changed in a totally unintentional way.

The result of this is that each person has a different model of what the problem is. Without the same context, difficulty arises when teams try to arrive at a solution and move forward together. There’s a communication breakdown because they don’t share all the same information. Common ground is thus a great lens through which to examine how incidents flow, and where things get stuck.



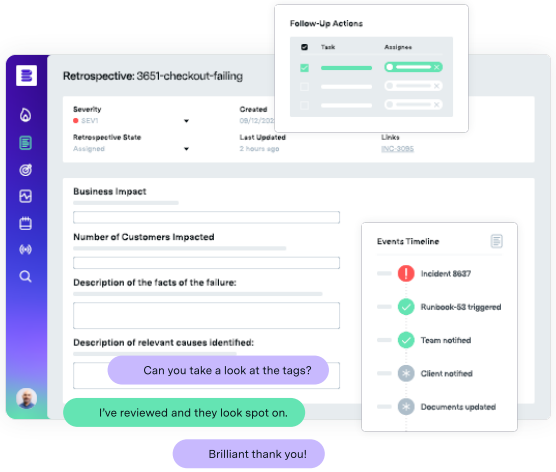
During the heat of an incident, it's important for incident responders to have space to work without interruptions from upper management. Blameless allows teams to set up automated recurring communication to executive leaders via Commsflow, keeping their incident response channel exclusive to the response team.

Dealing with Management

Managers can also unintentionally create friction during incidents. An important thing to keep in mind is that usually they're trying to help. Inherently, managers tend to carry more responsibility for the actions of their teams, and for the systems that they own. They thus have more fear activation when things are down, as they have to respond to their own management.

So they may try to interject and help. Sometimes that's good, and indeed helpful. However, the other side of that is that humans naturally follow hierarchical models. An interesting problem that arises is when managers unintentionally have the response team chase after different scenarios and 'dead ends'. The manager may be trying to help, but their authority could skew the incident communication in unproductive directions, through no fault of their own or that of the team.

However, knowing how to engage managers during incidents is also deeply important, because they're often the points of escalation when no one else is responding. Because they have authority, if they do start getting involved in an incident, as an IC the best thing to do is message them privately to lend their authority to you, as people are going off in directions that may not be productive. Of course, this is also contextual based on your organization and manager in question. But while the IC may not have organizational authority, the IC title confers some level of authority and must come with the ability to help others understand that you are the central point in command. This often can be difficult to get used to, but gets easier with time and practice.



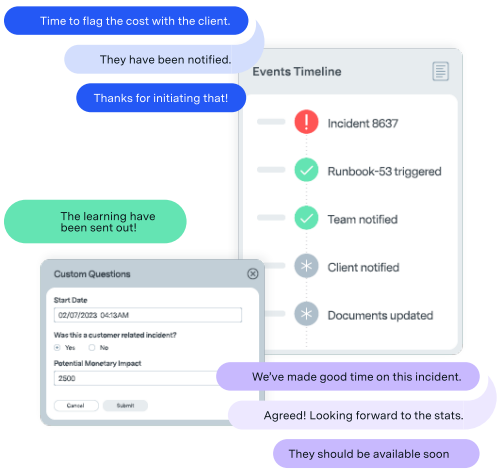
Blameless helps your team more effectively capture learnings from every incident, then leverage that new knowledge to build better products. Retrospectives can be easily collaborated on and follow up actions pushed directly into your ticketing system.

Memorialization

The practice of memorialization is key to support post-incident learning. It is the act of gathering evidence while the incident is ongoing, typically done by a Scribe that is either assigned or automation-based. Evidence can comprise chat logs and communication, snippets of log files, pictures of a graph; these are all things that need to be gathered together postincident, and which are basically facts.

Sometimes in the beginning of the incident, the response team may have no idea of what's going on or what the nature of the root problem is, so it's important to gather evidence so that team members can come back to later. Most importantly, when the incident is over, this evidence is crucial to really delve deep and work on solution policies, with the goal of preventing the same time of incident from happening again in the future.

Teams who work collaboratively on retrospectives tend to save 3 hours on average in their postmortem process. Our customers also have 3X retrospective completion after adopting Blameless.



Blameless

Additional Learning Resources

Having a strong IC is key for successful incident response and resolution. The good news is that, just as with any other craft, it can be honed with study and practice.

If you're interested in learning more, here are some additional resources we recommend and love:

NONVIOLENT COMMUNICATION BY MARSHALL ROSENBERG: Has a lot of really good techniques for managing conflicts.

THE PARADOX OF TOLERANCE: Sometimes you may run into personal issues during an incident, but it's essential to nip intolerance in the bud.

THE ICS RESOURCE CENTER: Comprehensive resources from FEMA walking through the ICS framework for incident management.

PAGERDUTY'S INCIDENT RESPONSE DOCUMENTATION: Thorough documentation based on PagerDuty's own internal incident response process.

NEW RELIC'S BEST PRACTICES FOR TRAINING ICS: Additional insights such as the 'three flows' concept of controlling the flow of emotions, information, and analysis.

VERSION 2