Ad Hoc

# The Ad Hoc Incident Response Playbook

Laying the foundation for a flexible, blameless incident response practice

Written by **Eric Isbell**, **Jason Williams**, **John French**, **Deirdre Holub**, **Bill Mill**, **Elissa Frankle Olinsky**, and **Craig Butler**

# Table of contents

# Don’t panic.

Software breaks all the time. Sometimes it breaks and is still usable; other times it breaks and systems grind to a halt. Since it is a certainty that software will fail, it’s essential your team is prepared and has a plan in place to respond.

This playbook is for government agencies and other civic organizations looking to develop and mature their practical understanding of incident response from a reactionary mindset to a more flexible, cross-functional, and blameless approach.

Follow our plays to better equip your organization and teams with practices that will lead to calmer, more effective incident response and ultimately more secure digital services.

## A flexible approach

At Ad Hoc, we believe that there is no "one size fits all" approach to incident response. A procedure that works for one organization may be wrong for another.

It is not our goal in this playbook to prescribe certain tactical tools or solutions, but rather to give you the strategies you can use to figure out what will be best for your situation. Focus on determining guiding principles based on your specific organizational needs, structure, and team resources. Guiding principles can then lead to flexible and enduring answers.

## A prepared mindset

For many teams, responding to security incidents feels frantic, chaotic, and stressful. Some teams might even let days or weeks go by before realizing there is a problem, which only complicates the response.

If we think of incident response like healthcare, preventative care and testing are equally as important as the emergency room. If we never invest time, effort, or energy before a medical emergency, then we run the risk of only ever reacting to emergencies instead of having the chance of preventing them.

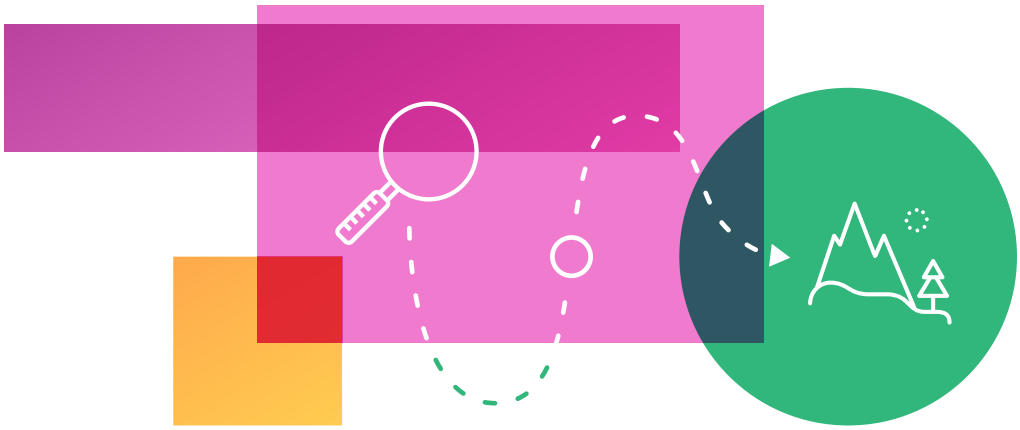
At Ad Hoc, we believe in prevention and emergency response. Software incidents can be incredibly costly, damaging, and harmful to organizations and their users. **The more your team can understand its systems, plan for various types of incidents, and practice responding to them, the easier it will be to respond to real-world issues.** And the less impact issues are likely to have.

## A cross-functional responsibility

At Ad Hoc, we take a cross-functional approach to building software, products, and services. That means that our entire team works together to determine how best to approach problems. We approach security (and therefore incident response) in the same way we approach accessibility or quality – as the responsibility of the entire team.

Every team member has a valuable and unique perspective that can contribute to creating resilient systems and remediating issues as they arise. Designers and researchers might have key insights into user behavior that can contribute to discovering the root cause of an incident. Product managers and writers may be able to coordinate communications with end users and stakeholders. And engineers can help update code and isolate activity.

Making incident response a cross-functional effort maximizes a team's resources. It allows teams to prevent or respond to incidents more efficiently while also improving stakeholder and user-facing communication during an incident. Cross-functional responses improve outcomes without placing all the burden on software engineers.



# **Play 1:** Survey the landscape of your service

> "He who defends everything defends nothing."
> — *Frederick the Great*

In this play, you will assemble key information about the landscape of your product or service, its security needs, and what is important to your organization so that you can make good decisions about procedures in later steps.

In many ways, this play is about understanding potential risks so that they can be prevented in the future. It is similar to understanding flood zones and historical weather data when buying or building a home. There may very well be risks or tradeoffs, but the goal of this play is to document and understand them so that you can start to make more informed decisions.

## Start by observing and documenting the digital and physical aspects of your service or product.

Every meaningful incident response process must begin with building the team's shared understanding of the details of the system, product, or application they're supporting. That knowledge must also include what the biggest vulnerabilities are likely to be. If this process feels overwhelming, it might help to start by asking the engineers to make a simple diagram of how the system works.

Work with your team and organization to:

- Document your service, product, or system
- Define what would constitute an incident in this context
- Determine what types of incidents you are likely to encounter and what data might be sensitive or targeted by bad actors
    - For example, availability incidents, security incidents, data leaks, etc.

Try to define expectations for the expected operation of your system. This often takes the form of a Service Level Agreement or some other documented policy. These policies help you know when system performance dips below expected levels and might trigger an availability incident or what expected behavior should look like to users.

## Then, define what is important to your organization in terms of an incident response process.

A service that lets users browse product documentation will need a very different incident response policy than a service that lets users schedule physicians appointments. A small team or organization will have fewer resources available than a larger one but may also face

fewer numbers of risks. So, rather than starting with someone else's answers, define what is important to your organization, so that you can later find the best possible procedure for your situation.

Again, Ad Hoc recommends beginning with guiding principles. What is most important to your organization? Once you understand these principles, you can more easily prioritize and write procedures. In the case of incident response, a useful way to determine your organization's values is in terms of risks and potential damages in the case of an incident. In the following example, notice how the questions about data and storage lead naturally to questions about risk.

- What type(s) of data does your application or team utilize?
- How/where is it stored?
- What risk does this data present?
- What data or information is most likely to be targeted by outside actors?
- What data would be most damaging if it was accidentally exposed?
- What kind of breaches or exposures would put you or your stakeholders at risk of having to testify before Congress?

**Prioritizing the most vulnerable systems or data can help your team develop processes that achieve the highest value for the least effort.**

## Determine what incident response processes already exist, if any.

In many cases, your team may not be starting completely from scratch, so it might be helpful to take some time to understand what processes already exist.

- Do an audit of any existing incident response documentation, the outcomes of past incidents, or any relevant process documentation.
- Identify the biggest gaps in those processes based on what your team decided was the most valuable or important.

**If you're starting from scratch developing an incident response process, don't try to do everything all at once.** Identify what systems or data are most important to your team and prioritize from there.

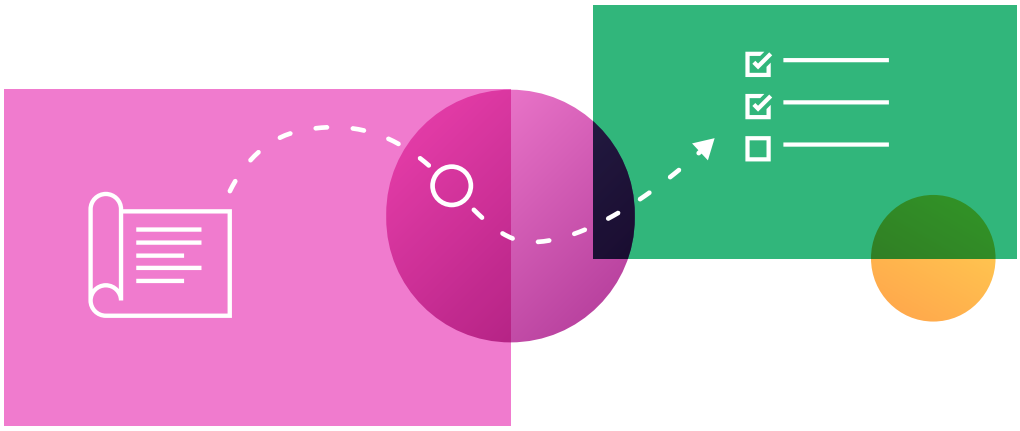
**Your team should consider running a threat modeling exercise where you try to anticipate what type of threats your system might attract.**

## Build a clear list of all relevant stakeholders and users.

A key part of understanding your system and preparing an incident response plan is documenting who might be impacted by incidents. This includes users who might rely on the functionality of the software to do things like make a doctor's appointment or file an insurance claim. This list should also include stakeholders who are responsible for the successful operation of that software for users.

Making this list ahead of time will help in the later stages of developing an incident communication plan.

- Document key stakeholders and try to include contact information in your documentation.
  - Consider checking with stakeholders during this process to verify their preferred method of communication during an incident.
- Document different users – both internal and external – who might be impacted during different types of incidents.
  - Check to see if you already have this information documented by a UX designer or researcher.



# **Play 2:** Be prepared, have a plan

> "Plans are useless, but planning is everything."
> — *Dwight D. Eisenhower*

Once your team has determined (1) the landscape of your application or service, (2) what is important to protect, and (3) what the team's shared definition of various incidents could look like, it is time to start developing and codifying the actual incident response plan.

## Keep the incident response plan simple and clear.

The actual incident response plan is simply a document or a series of documents. You can think of it as the jumping-off point for all things related to incident response. We will help expand on this throughout the playbook, but at a high level, it should try to achieve the following functions:

- Outline the team's understanding of the system and potential vulnerabilities.
  - For example, the document can outline the difference between an incident in which a downstream service causes an outage for users versus an incident where there are unauthorized users in an administrator environment. There may be many more types of incidents as well.
- Link to documents for the team to use for communication in the event of an incident.
- Define and communicate roles, expectations, and instructions.
- Help new team members quickly understand how to be involved in the incident response process.

This document should not be unnecessarily complex. The simpler it is, the more likely it is to function correctly when an incident occurs. Incidents can often be incredibly stressful situations, filled with multiple parties going different directions at once searching for the root cause and trying to establish communications to stakeholders. By avoiding complexity in your plan, you can reduce the friction of spinning up an incident response effort and protect your team from additional stress.

## Document your incident response plan in a visible, shared environment.

Before we get into the specifics of what your incident response plan entails, it is important to convey that the plan must be a tangible, documented artifact, **not something that lives in the heads of team members**.

The incident response plan must remain up to date, as a living document that all team members have the ability to access and inform.

Where is the best place to store your incident response plan so that everyone on the team (including stakeholders) can easily find it, reference it, or update it?

**Be careful not to store the incident response template somewhere that it may itself be impacted during an incident!**

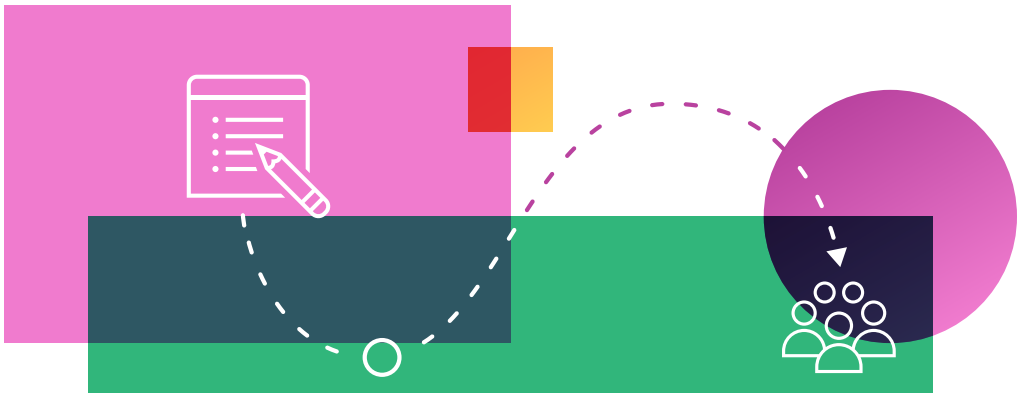
## Create an additional, easy-to-use template for future incident response recordkeeping.

In addition to having an overall policy for incidents, it will be important to create easy ways to start responding when an actual incident occurs. Have clear instructions about how to make a copy of the document. Provide instructions or fields about what information needs to get collected so the team can start investigating. The day of the incident may be chaotic, so instructions must be extremely clear.

Check the appendix at the end of this playbook for a sample incident response template to get you started.

This document must collect information quickly so that the team can quickly jump in and understand the timeline and status of an incident.

**Make the template yours. Our approach may not suit all of your needs!**



# **Play 3:** Define the team roles

> "There are no small roles, only small actors."
> — *Konstantin Stanislavski*

There are many different ways to organize teams and assign roles to respond to incidents. We will cover one possible approach of many. **Not every team will have the resources or team members available to fill all of the roles we will cover in this section.** Teams can combine, edit, or remove roles depending on what works best for your situation.

You can think about this section like a box full of LEGOs. Think about what you need to build and use the pieces you have available to you to construct the best team. You can always iterate, change, and swap out pieces as you go!

**Here is how our teams structure incident roles.**

## Incident Commander (IC)

The Incident Commander's job is to coordinate the team's response. They take charge and become the main "point person" or "team captain" for the incident. The Incident Commander can be anyone on the team. Your team should strive to make the process clear enough that anyone feels comfortable inhabiting that role when needed.

*Note: The Incident Commander will often change throughout the duration of a longer incident. Your team should have clear rules about how and when an IC takes charge, and how to delegate that authority to the next IC.*

### What might an IC do?

- Formally declares the incident in communication channels.
- Copies and begins filling out an incident response template so the team has a shared place to start documenting details about the incident, their attempts to resolve that incident, and next steps.
- Handles updates and communicates with stakeholders to ensure the response team can focus solely on investigating the incident.
- If additional roles are necessary, it is the IC who should assign them.
- Adds updates to ongoing documentation, especially for smaller teams that don't have dedicated roles for this task.
- Formally declares when an incident is resolved based on the agreed upon criteria in the team's incident response plan.
- Should be responsible for making sure that this is balanced appropriately. If the immediate needs of the service's users are not being met, they should direct more effort towards stabilization efforts.

### What should an IC NOT do?

- They should not spend time investigating the incident's cause. Instead, they support the response team that is responsible for the investigation. This is to ensure the engineers and responders can concentrate on resolution and allows the Incident Commander to field all items that could be disruptive to resolution.

In smaller teams, the Incident Commander will also handle communication with stakeholders, as in the below role. In larger teams, this responsibility becomes a role of its own.

### How should you choose an Incident Commander?

Often, the first Incident Commander might be the first person to realize that a situation meets the team's criteria for declaring an official incident. Once an incident is declared, the team might want to select an Incident Commander who is not needed to triage or fix the incident, but has strong system knowledge and communication skills to coordinate with the team and stakeholders.

## Communicator

Larger teams who have enough people might benefit from delegating communication responsibilities to an official Communicator.

### What might a Communicator do?

- Digital services often have many stakeholders interested in the incident. The Communicator's job is to convey the status of the team's investigation to them at regular intervals. This saves the IC burden during stressful times.
    - The Communicator should be prepared to explain the incident to a non-technical audience.

- They should act as backup Incident Commander in case of point of failure.
- They should be a sounding board for the technical team to walk through issues.

### How should you choose a Communicator?

The IC or the shared team should select a Communicator who has prior knowledge of key stakeholders and user groups. This person should have the ability to quickly draft messages and updates to a non-technical audience.

**Pick a fixed interval for communication that works for your team; for example, the communicator may inform the stakeholders that they will send out a status email every 15 minutes. In this example, the update cadence gives the stakeholders a chance to be informed but to also spread word to groups or individuals who might depend on the service and need to relay this to their customers. The cadence allows for predictability, and for trickle-down customer service, when everyone has to relay information to their customers.**

## Incident Recorder

The Incident Recorder's job is to help the investigators (i.e. response team) to record the actions they take as they take them. In the event that your team doesn't have enough people to have a dedicated Incident Reporter, this can be a distributed task.

### What might an Incident Recorder do?

- They should help document what the team is attempting. Documentation will help to understand the cause and impact of the incident.

- They should capture what information the team is gathering that is relevant to the cause or impact.
- They should capture the actions the team is taking to reduce the impact or resolve the incident.

Recording the team's actions effectively and accurately is critical to your post-mortem. If you don't know what happened, and when, it is very difficult to learn from your team's responses later.

### How to choose an Incident Recorder?

The Incident Recorder should be someone who is comfortable documenting the response team's attempts at resolving the issue, so some technical experience or background may be helpful.

## Investigators

The Investigators' job is to respond appropriately to the incident. They must (1) stabilize the problem and (2) determine the cause of the incident. Once they understand the cause, they must also (3) fix the problem. Often an organization can have a team of Investigators collaborating on these things together.

What the investigation and triage of the incident looks like can vary widely depending on its nature and scope. For example, Investigators may enable maintenance features of the product or application, update status pages, or disable intensive features of the application to stabilize it. Investigators might also conduct research to determine the scope of impacted users or data that may have been exposed in a leak while also attempting to ensure the leak is fixed.

Investigators should be capable of understanding and unpacking the root cause of the incident, identifying possible solutions to stabilize the application, and identifying necessary steps to address any

consequences and formally close the incident. Often these steps might need strategic prioritization depending on the circumstances of the incident. Investigators should be able to weigh trade-offs between when to spend time investigating root cause versus making sure the application or systems have been restored. Some teams split these responsibilities or adjust what is most pressing during the incident response process. The more monitoring, observability, and flexibility you've built into your app, the easier it will be for Investigators to have the information they need to make these decisions.

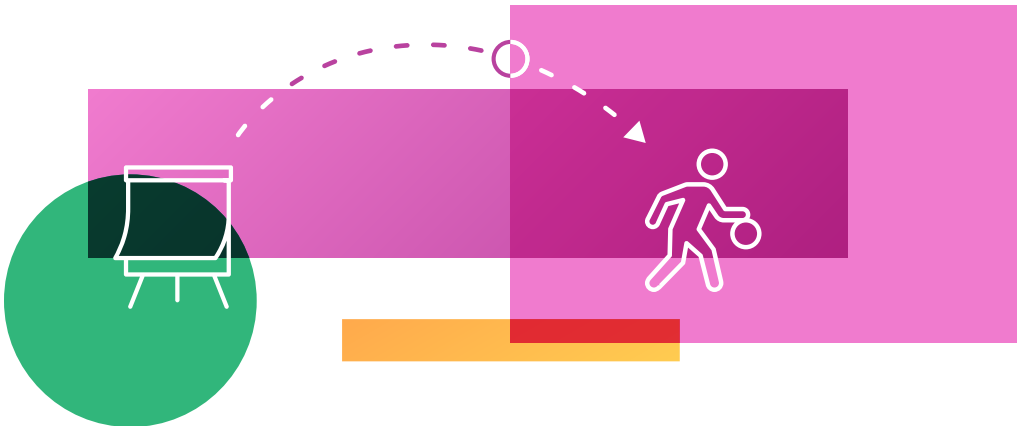
### What might an Investigator do?

- Stabilize the application or product.
- Find the root cause and remediate or fix the problem.
- When possible, Investigators should discuss what they're doing somewhere public, in writing, such as in a shared Slack channel. That way, the Incident Reporter and Communicator can effectively monitor the existing discussion so as to keep stakeholders up to date without distracting the Investigators.

### How to choose the Investigators?

The Investigators can be assigned based on who has the most domain knowledge or experience with the impacted systems, often engineers who are responsible for the day-to-day operations of the system.

**All of these roles are in service of resolving an incident that you and your team may encounter. But the roles themselves are not the end goal; adapt them as needed to your team and your process.**



# **Play 4:** Use it or lose it

> "We were on a break!"
> — *Ross Geller*

A lot of things worth doing take practice. If you're learning to speak a new language, a textbook will only get you so far – you also have to practice speaking, especially in community with others who speak the same language.

Making your team's incident response processes second-nature is similar. The entire team must "learn a new language" together. Practice is critical. The best incident response plan in the world won't help your team respond to an incident if it isn't practiced.

## How do you practice an incident response process?

### Schedule regular incident response check-ins.

The exact cadence can be up to your team, but at the very least, collaboratively review the incident response processes several times per year. This will help your team adapt to changes in product, new upstream or downstream services, etc.

- You can use these check-ins to confirm that specific processes within your incident response process are still useful.
- These check-ins help the team internalize the practices and principles that will drive a real incident response as second nature.
- Past incidents will serve as good jumping off points for considerations about what to focus on in the future.
- For more mature teams, use these meetings as an opportunity to validate your Service Level Objectives (SLOs) and Service Level Agreements (SLAs) and ways you can automate and improve aspects of your system.

### Hold a threat modeling session.

Threat modeling is an exercise that allows your team to anticipate and prioritize potential threats or risks to your system. By better understanding where incidents are likely to occur, or what parts of your system are likely to be targeted, your team can more easily provide the most value and security to your system. Threat modeling sessions are particularly important for new teams, since the team will need to collaboratively hypothesize what might happen once their new application is online..

### Run at least one or two full "game day" exercises per year.

A game day is a team exercise that simulates responding to different types of incidents in as realistic a way as possible. Game days make it possible for teams to be able to respond calmly when real-world incidents occur.

At Ad Hoc, we believe that game days:

- Should involve the whole team, often including customers and other stakeholders.

- Should have a dedicated “chaos master” or “chaos team” designed to plan and implement the simulated incidents.
- Should actually be kind of fun for the team.
- Ideally simulate at least two different types of likely incidents.
- Should practice both the investigation and communication aspects of incident response.

**If you need help getting started with running game days, we've done some writing on what game days can look like!**

### Be prepared with a plan for who gets contacted and when.

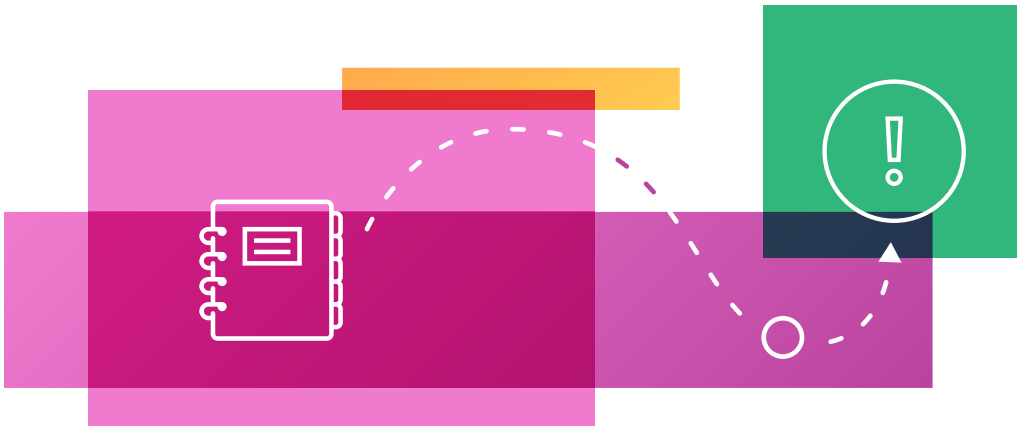
Incidents can be stressful and chaotic. You don't want to waste time tracking down contact information, providing the same feedback to multiple stakeholders, etc. So build a communication plan and practice using it during your game day activities:

- One of the first steps in an incident response is to explicitly identify who will be playing the roles defined above. In an after-hours incident, it may be clearer who the people are because they have been paged. When incidents happen during the workday, team members may all want to pitch in, but the team will waste time if people do not volunteer or get assigned to be the Incident Commander, Communicator (if applicable), etc.
- Keep a contact list up to date. Not just for team members, but also:
  - **Key stakeholders** within the business or customer.
  - **Shared services** that you may need to use (example: if separate teams handle your build/deploy infrastructure, content delivery network, or feature flag service).
  - **Other systems** with which you integrate. How can you contact them after hours? How can you initiate an incident for their systems?

- **End users**: If necessary, how can you message outage information to your application's users?

Ideally, this contact list should be set up in terms of roles/systems rather than individuals; an email that initiates an incident for an integration is better than the email address for that team's technical lead. The plan should contemplate that people may be on vacation, away from their keyboards, or otherwise unavailable.

- With the key stakeholders, discuss and agree upon what the cadence and nature of your communication with them will be.
  - Which stakeholders need to be notified if an incident occurs and when it is resolved?
  - Are there stakeholders who need to be given status during an incident? If so, which roles? How often should they get updates? In what forum?
  - Are there mitigation steps that require communication of approval from the stakeholder (example: putting up a maintenance page)?



# **Play 5:** Applying your process to a real incident

> "For the things we have to learn before we can do them, we learn by doing them."
>
> — *Aristotle*

Our initial plays are aimed at establishing or improving an incident response process. Now we want to spend some time talking about real incidents and how to apply the principles and resources we have outlined here.

> **Step 1:** Stay calm and decide your approach.
>
> **Step 2:** Root cause analysis.
>
> **Step 3:** Plan for handoffs and transitions.
>
> **Step 4:** After the incident.

## Step 1: Stay calm and decide your approach

First, we'd advise everyone to stay calm. **Don't panic, and remember your training.**

Before rushing to investigate the issue, take some time to decide what are the appropriate actions to take first. In an emergency, physicians like to say "Slow is smooth and smooth is fast." Be thorough and methodical in your actions; a common source of incident response failure is immediate overcorrection.

Sometimes, your system's failure is well-understood. More often, it's because something has occurred that you didn't expect, which means that you have some flaw or issue in the mental model of what's happening in your system.

- Make sure that you have, as fully as possible, considered the cost of any actions you take before you take them.
- Have a bias towards inaction. Make sure that more than one person agrees before any consequential action (such as restarting a service or modifying a configuration) is taken.
- Start with containment.
  - Before you dig in to try and solve the root cause of the problem, ask if there are any actions you can take that would immediately benefit the users of your system.
    - Update the relevant status pages.
    - Can you enable a warning message on your service?
    - Are there feature flags you can flip to direct users to an alternate backend, or disable a problematic part of your service?

After focusing on containment, move to the next step to flesh out your investigation of the issue.

**Be sure to record all actions taken and the reasoning behind them.**

## Step 2: Root cause analysis

As your team moves from the initial investigation into a more tactical response, it's important to be sure that the team is capturing and documenting any decisions or theories as you go. This documentation will make it easier for your team to make evidence-based decisions in real time.

- When software breaks, there are often numerous possibilities as to why. Documenting what the team is seeing in the system as they investigate will help you collect evidence to confirm or deny assumptions as you go without jumping to conclusions.
- Despite the urgency that often surrounds incidents, it's important for your team to approach your root cause analysis deliberately and make sure that any updates that are given to stakeholders or users are backed up by evidence to avoid confusion.

**As you go, continue to remind yourself and the team not to jump to one conclusion.**

Making pre-determined assumptions without all the facts can cost your team precious time and resources during an incident. It can lead to investigations that need to restart from the ground up, which can negatively impact the team's momentum and morale and weaken stakeholder trust in the team's response. It is always better to take the time to confirm a hypothesis before taking hard-to-reverse actions.

Here are some tips and tricks and questions your team can ask when performing your root cause analysis:

- Given the incident that is occuring, have the Investigators discuss possible causes. You often will have some sort of error message, logging, error message, or reporting that will help give some initial indication of the impact.
- Decide as an investigation team whether you want to focus on the most likely root cause or divide into smaller groups if there are multiple possibilities for what is causing the incident.
- Investigators can examine things like Splunk logs, AWS console messages, and any other tools at your disposal to get more information about the history of or interactions with the impacted parts of your system.

Once your team has identified one or more potential root causes (ideally with evidence and/or justification), determine what steps you would need to take to proceed and be sure to validate at every step.

Once you understand your root cause, the team can move to solving the problem.

## Step 3: Plan for handoffs and transitions

### Communication is key during an incident.

Stakeholders want to know what is happening and when the problem will be fixed. Team members need to be able to share findings, questions, and actions in a calm, focused environment. Having a plan for how you manage communications during an incident is key to resolving it efficiently.

**Use the communication plan you initially developed with your team to make sure communication is happening as expected.**

## During the incident, manage internal and external threads.

A common failure of incident response processes is having Investigators trying to concentrate on diagnosing the problem while being buffeted by questions from stakeholders. A successful incident response process eliminates this difficulty by maintaining separate communication channels: internal and external.

Ideally there is no overlap between the two channels, with stakeholders reviewing only the external thread while the team exclusively sees only the internal thread. As an example, the external thread could be an email chain while the internal thread is in a group chat.

Often stakeholders want to initiate a conference or video call to "get everyone in the same war room," but that's the last thing the Investigators need. If a call is required, the Incident Commander or Communicator should join in their role as conduit between the team and the external world, while the team continues to move forward with focus.

**In the external channel, the Incident Commander or Communicator is keeping the stakeholders in the know.**

In the forum and cadence agreed upon in the communication plan, the Incident Commander or Communicator role provides updates to the stakeholders. Key items to consider sharing include:

- What is the impact of the incident? How many users are affected? Is there a workaround? This information will likely emerge over the course of the investigation.
- What steps have the team taken to contain the impact of the incident?
- What actions is the team taking to resolve the issue?

It's important not to get into the weeds here; the stakeholders won't care what is showing up in the logs nor will it be helpful for them. But, if the team is putting up a maintenance page or turning off some feature, the stakeholders need to know and may need to approve. Even just confirming that the team is still investigating is helpful to assure stakeholders that the work is ongoing. When your service is down, radio silence can quickly lead to panic.

Also, be careful about offering up hypotheses on the problem, possible solutions, or timelines for resolution to stakeholders before the team has had a chance to validate them. Don't provide false hope or paint an unnecessarily gloomy picture. Communicate accurately, candidly, and clearly.

**In the internal channel, the communication is 100% focused on resolution.**

The investigators will use their separate internal to share what they are seeing in monitoring data and logs, derive hypotheses as to the root cause, and provide status or questions for the IC/Communicator to share with the stakeholders. Relevant links, screenshots, logs, etc. should be shared in the channel.

If there are topics unrelated to solving the problem, try to capture those elsewhere so the communication in the internal channel serves as a record of the resolution. Where possible, investigators should "think out loud" so the IC/Communicator doesn't have to ask them for status and can just derive it from the conversation.

For this reason, a text-based channel, like a group chat, is often preferable to a shared Zoom call. It also helps if others have to transition onto the incident resolution team: they can follow the history rather than having to have someone take time to catch them up.

In addition, it's best to overcommunicate while you are working on an issue. If someone says "Somebody should turn off the ability to log in," it's easy for people to assume someone has done that. Explicitly state when you take an action.

### Incident Commander handoffs

Depending on the duration of the incident, or due to unforeseen circumstances, a team member or IC may need to drop off.

**If an Investigator needs to drop off:**

- If needed/possible, flag in another engineer to assist. This action becomes critical if there is only one person investigating.
- Give a brief summary of what has happened and the latest that we know. The new Investigator can then review the prior communication.

**If an IC/Communicator needs to drop:**

- Give a brief summary of what has happened and the latest that we know. The new IC can then review the prior communication.
- Communicate clearly to all stakeholders that the IC/Communicator is changing. Confirm who they will be talking to for future updates.

## Step 4: After the incident

### After the incident is over, take a deep breath (and maybe a nap).

More often than not, incident response can be a difficult process. After completing one, you should feel some sense of accomplishment. Take some time to collect yourself and give your team time to recover. When you're ready, come back to study what happened so that you can ensure your team is stronger and more resilient than it was before the incident.

### Conduct an incident post-mortem.

Post-mortems are an essential part of a mature and effective incident response process. They allow your team the mental and physical space to really think about what happened and why – and what might prevent this or something else from going wrong in the future.

- Always approach your post-mortem with a "blameless" approach, meaning that you are not there to point fingers or attack anyone. You're there as a team to help prevent this from happening again in the future. If someone on your team made a mistake, have empathy, and together build a plan to help prevent it in the future.
    - Making software is an incredibly complex thing done by necessarily imperfect human beings. Mistakes will happen! The important thing is to try to learn to not make the same kind of mistakes repeatedly.
- The post-mortem shouldn't just be about what caused the incident, but also focused on your Incident Response. What went well? What didn't go well? Where were you lucky? An incident provides a valuable opportunity to iteratively improve not just your code but also your ability to monitor your application and respond to problems.
- Post-mortems should not result in simple, vague reassurances such as, "we need to watch out for that," or "we won't do that again." They should result in specific actions and plans. Wherever possible, create specific tickets to refactor code, add monitoring or alerting, add unit/integration tests, or change specific code review processes.

**Teams who run effective post-mortems will be better positioned to apply lessons learned during an incident to future work. This will improve the resiliency of their applications/systems.**

# **Play 6:** No matter what, take care of each other

> "A little consideration, a little thought for others, makes all the difference."
>
> — *Eeyore*

Going through an incident can put a lot of weight and pressure on teams. At Ad Hoc, we believe that laying a foundation of trust, respect, inclusion, and empathy can really strengthen a team's ability to both respond to incidents and learn and grow from them.

Having a team that trusts each other is as important an element of incident response as having a comprehensive suite of monitoring tools. A high-trust team can delegate actions to the person with the right knowledge base, work through a problem and possible solutions objectively, and help each other stay calm in chaos.

## Share the responsibility of incident response as a team.

It's important to have trust and understanding within a cross-functional team. If engineers and researchers never collaborate or share knowledge during day-to-day work, then it is unlikely to expect that they would collaborate well during an incident.

You should work to create a team culture that encourages everyone to ask questions without judgment and be explicit in creating a culture that communicates that every team member has a role to play in incident response. Say it out loud and practice it.

Many teams feel that security and incident response is the responsibility of specific subsets of a broader team. But Ad Hoc believes that products and services will be more resilient if everyone is empowered to understand concepts like how to spot incidents, basic information about what types of incidents can occur, and how to respond to them.

## Incident response must truly be blameless.

In a culture of shame, teammates are disincentivized from calling attention to a possible incident for fear of being blamed for causing it. A culture that prioritizes getting the system back to nominal over blaming the person who caused it encourages people to raise concerns to the team. This open communication allows triage and resolution to start faster.

## Enable teammates to be human by encouraging explicit handoffs.

Whether in a live incident or a gameday, someone's kid will need to be picked up from school or someone's chickens will escape from their coop. Having documentation accessible to everyone and many people able to take on each role enables your IC to hand off communications while they take care of pressing family needs. When these things come up – and they will! – the person who must leave should be able to clearly ask for relief, with someone else clearly taking on that role.

## Practice the care work of incident response in your gamedays.

Our chaos masters don't schedule a break for lunch on our game days. In a live incident, the outage won't break for lunch, either. Not having pre-scheduled breaks forces the team to figure out how to take breaks together. Practice your clear handoffs and your gratitude for the person who discovered the incident during your gamedays. But most importantly, celebrate one another on your teams every day, so when the going gets tough, you've built the muscle of taking care of each other.

APPENDIX

# Sample incident response template

## Incident summary

**Date:** *7/15/2023*

**Status:** *[choose: Open / Closed]*

**Project incident report #:**

**Executive summary:** *Write a short, easily understandable description of the issue and the business, i.e, how will this affect customers or peering partners*

**This issue is an:** *[choose: Incident / Event]*

---

## Incident details

**Author(s):** *[Insert name], [Insert name]*

**Response team:** *[Insert name], [Insert name]*

**Root cause:** *[Insert cause (i.e. phising attempt asking to buy gift cards for CEO)]*

**How this was detected:** *Notification of the issue was given to {Name} regarding the issue occurring, with Eric & Chris being tagged in to the system to revoke the Home Depot Gift Cards*

**Resolution:**

**Timeline: HH:MM $action**

- ***7/15/2023, 10:32am:*** *User informed of incident occurring*

**Action items:**
All action items are to be documented in a corresponding ticket and includes the label #Ad_Hoc_Example

- Action Item:
- Owner:
- Ticket Number:
- Additional notes:

---


**Ad Hoc** is a digital services company that helps the government better serve people. We collaborate closely with our partners to solve the right problems and deliver software solutions that work. In doing so, we give agencies confidence that their tools will effectively meet the demands of their mission and the needs of their users.

Work with Ad Hoc to ensure your agency has a comprehensive plan in place to prevent system emergencies from harming your digital services. Learn more at **https://adhoc.team/incident-response** or contact us at **hello@adhoc.team**.