# Escalation Policy Guide: Levels, Time Intervals, and Best Practices

Alerts that nobody answers turn small problems into long outages. This guide shows how an escalation policy fixes that, and shares five best practices.

UPDATED NOVEMBER 2026

TL;DR

An escalation policy is a set of rules that passes an alert to the next person when the first responder doesn't acknowledge it. It's built from four parts: escalation triggers, time thresholds, an escalation chain of levels, and communication channels. A separate policy for each severity keeps phone calls for critical incidents and sends minor ones to Slack.

## What is an escalation policy?

An escalation policy is a set of rules that decides who gets alerted about an incident, in what order, and how long each person has to respond before the next person is alerted. For example, when the primary on-call responder doesn't answer a phone call within 5 minutes, the secondary responder gets a call.

Without a policy, every alert rests on one person. If they miss it, the incident stays open and unresolved. An escalation policy sends the alert to the next person after a set time, so someone always responds and nobody has to decide at 2 a.m. who to call. It also lowers the mean time to acknowledge (MTTA), which is how long it takes someone to respond to an alert.

Escalation policies and on-call schedules are often confused, but they do different jobs. An on-call schedule decides who is responsible for responding to incidents now. An escalation policy decides who gets alerted next if that person doesn't respond. The schedule changes with every rotation, while the policy changes only when the team or the services change.

## What are the types of escalations?

There are three types of escalations.

**1. Hierarchical escalation:** the issue moves up to someone more senior, such as a team lead or an engineering manager. It is for incidents that need a decision only a senior person can make, like taking a service offline.

**2. Functional escalation:** the issue moves to the team with the right skills. A database error raised by support goes straight to the database engineers.

**3. Automatic (time-based) escalation:** rules pass the alert to the next level when nobody acknowledges it within a set time automatically. If the primary doesn't respond in 5 minutes, the secondary gets a call.

Automatic escalation makes sure someone responds. Hierarchical and functional escalation then bring in the people who can fix the problem.

### What are the core components of an escalation policy?

| Component | What it decides |
|---|---|
| Escalation triggers | When an escalation starts |
| Time thresholds | How long each level has to respond |
| Escalation levels | Who gets alerted, in what order |
| Communication channels | How each person gets the alert |

**1. Escalation triggers:** the conditions that start an escalation. A missed response time, a high-severity alert or a drop in payments can all be triggers.

**2. Time thresholds:** how long the policy waits before it alerts the next level, such as 5 minutes. The acknowledgement timeout is a second timer. It moves the incident on if someone responds but can't fix it within a set time.

**3. Escalation levels:** the people or teams alerted in order. Level 1 is the primary on-call, level 2 the secondary and level 3 an engineering manager. Spike's post on L1, L2 and L3 escalation covers how these levels work.

**4. Communication channels:** how each person gets the alert. Phone calls suit critical incidents at night. SMS and push notifications work as a backup, and Slack or Microsoft Teams suit daytime and low-priority alerts.

## What does an escalation policy look like during a real incident?

Here's how an escalation policy works for a login service. It has three levels: the primary on-call, the secondary on-call and the owner of the login service. Each person has 5 minutes to respond before the next one is alerted. Once someone responds, they have 30 minutes to fix the issue before the policy brings in the next level.

At 1:10 a.m., login fails and the monitoring tool raises an alert. The primary gets a phone call and an SMS.

Let's say the primary is asleep and doesn't respond. Then the policy calls the secondary at 1:15 a.m. The secondary picks up and starts looking into it.

By 1:45 a.m., the secondary still hasn't found the cause. The 30 minutes are up, so the policy calls the owner of the login service. The owner finds an expired security certificate and renews it. Logins work again at 2:05 a.m., 55 minutes after the first alert.

Nobody had to decide who to call next. The policy alerted the next person automatically.

### What happens if nobody responds?

When Repeat Escalation is enabled, the policy starts again from level 1 after a short pause of about 15 minutes. It repeats until someone responds or it reaches a set limit, such as 5 rounds. If an alert goes unanswered for five rounds, the on-call schedule probably needs fixing.

## How do you create an escalation policy in 5 steps?

1. **Determine your triggers:** Triggers decide which alerts start this policy. Severity is the simplest trigger, so all SEV 1 alerts from the payments service could start it. Alerts can also be matched by their title, such as any alert with keywords like login fail, database, etc., in it. Spike's post on severity and priority-based routing shows how this works.
2. **Add an on-call schedule or a person at each level:** When a level points to the on-call schedule instead of a named person, the policy always alerts whoever is on-call. If the rotation hands over on Monday, the new person starts getting alerts that day and nobody has to edit the policy. Later levels can also point to a specific person, such as the owner of a service.

3. **Choose the alert channels for each level:** Phone calls work best for critical incidents. A simultaneous Slack message at level 1 tells the rest of the team at the same time. Low-priority policies can skip phone calls and use email or mobile app alerts.
4. **Decide on the time intervals:** There are four time intervals to set: the wait before the first alert, the time to acknowledge, the time to fix after acknowledging and how often the policy repeats if nobody responds. Critical incidents need short time intervals: no wait time, 5 minutes to acknowledge, 30 minutes to fix and a repeat every 15 to 30 minutes.
   Low-priority incidents can have longer time intervals. A 3 to 5 minute wait before the first alert lets issues that fix themselves close without alerting anyone. After that, a few hours or the next working day to acknowledge is usually enough and the policy rarely needs to repeat.
5. **Send a test alert:** A test alert shows whether the calls go through, whether the time intervals are right and whether the alert reaches the last level. Consider running one during working hours and again after any change to the schedule.

## 5 escalation policy best practices

1. Define your severity levels before building the policy. Severity describes the incident's impact on the users. Critical incidents could be when users can't use a core feature such as login or payments. A major incident might mean a feature is broken for some users, and a minor incident means little or no user impact. Each level then gets its own policy. Spike's posts on escalation policies for critical incidents and low-priority incidents cover both in detail.
2. Pass context along with the alert. When an incident moves to the next level, the new responder shouldn't have to start from scratch. Short notes on what's been checked and a link to the runbook let them pick up where the last person stopped.
3. Set response times that match the severity. Critical incidents usually get an immediate alert. Whereas minor incidents can wait a few hours, they still need a time limit so they don't remain unattended.
4. Match the alert channel to the urgency. A critical incident needs an alert that's hard to miss, so a phone call works best. Sending a call and a Slack message together makes it even less likely to be missed. A minor incident can go to Slack, Microsoft Teams or email instead, because it can wait.
5. Use an on-call schedule at level 1. That way the first responder isn't always the same person, and the load is shared evenly across the team.

## What should you look for in escalation policy software?

| Feature | Why it matters | In Spike |
|---|---|---|
| Phone call alerts | Reaches the on-call person for critical incidents, even at night | Yes |
| Response time limits | Alerts the next person if nobody acknowledges, or if someone acknowledges but can't fix it in time | Yes |
| Repeat alerts | Starts the policy again when nobody responds, so the alert doesn't get missed | Yes |
| Separate policies by severity | Critical incidents can call people while minor ones go to chat | Yes |
| On-call schedule sync | Alerts whoever is on duty, so nobody has to edit the policy after each rotation | Yes |
| Slack and Microsoft Teams alerts | People can see and acknowledge alerts in the chat tool they already use | Yes |

In Spike, you build an escalation policy one level at a time. Each level says who gets alerted and how: a person or an on-call schedule, by phone call, SMS, email, push notification, Slack or Microsoft Teams. If nobody acknowledges, the policy moves to the next level and keeps repeating until someone does. On top of these basics, Spike adds four features that make policies easier to run:

- **Slack mentions:** an alert can mention @channel, @here or one teammate, or even no mention.
- **Escalation history:** every change to a policy is saved, along with who made it.
- **Wait time:** a short pause before the first alert, so issues that fix themselves don't alert anyone.
- **Simultaneous alerts:** one level can alert several people and channels at the same time.

## Set up your first two escalation policies today!

An escalation policy doesn't have to be perfect on day one. Two policies are enough to start: one for critical incidents and one for low-priority ones. The critical policy calls people and moves to the next level within minutes. The low-priority policy sends a Slack message or email and gives people more time to respond. Link both to the on-call schedule and send a test alert. Real incidents will then show what to adjust.

Spike lets you set up that first policy and hear a real test call within minutes. Create your first escalation policy today!

## FAQs

### How many levels should an escalation policy have?

Three or four levels suit almost any service: the primary, the secondary, a specialist or service owner, and a manager.

### When should managers or executives be included in escalations?

Managers and executives should be included when an incident goes beyond what the on-call team can handle. This covers incidents that affect critical business operations, need a decision above the team's authority, or need several teams to work together to resolve.

### What if an incident resolves itself before anyone responds?

The policy stops as soon as the incident resolves, so nobody else gets alerted. A wait time of 3 to 5 minutes before the first alert also keeps these short-lived issues from waking anyone up. The incident should still be logged and reviewed, though.

### How often should you review an escalation policy?

Every quarter is a good rhythm. A team change, a new service or a major incident are good reasons for an extra review in between.

### Do support and IT teams need an escalation policy too?

Yes. Support and IT teams handle urgent issues too, such as a customer who can't log in or an office network that goes down. Without an escalation policy, these issues can sit with nobody assigned and miss their service level agreement (SLA).

### What metrics show an escalation policy is working?

A working policy means fast responses, few false alarms and a team that isn't burdened. Mean time to acknowledge (MTTA) shows how fast people respond. The escalation rate shows how often alerts pass the first level, where a high rate means missed alerts. The false alarm rate shows how many alerts needed no action, and out-of-hours alerts per person show whether the load is fair.

## Need help setting this up?

Write to support@spike.sh, or book a call with our team at spike.sh/demo, and we will help you set up Spike.

Read this online: spike.sh/escalation-policies-guide