



# Beyond the Virtual War Room:

Automate incident management with Slack





19%

Increase in critical incidents

184

average microservices in an enterprise

## Incidents are more complex than ever

Massive transformations in software development over the past five to 10 years allow us to launch more often—and faster—with higher quality than ever before.

But there is a downside: Today, incidents are more common and more intricate. Two key reasons why:

- Modern technologies like Kubernetes, microservices, and API-driven architectures make **software more complex** and **increase the pace and volume** of releases, leading to more frequent incidents.
- A proliferation of best-in-class free and open source tools creates tool overload, never-ending **context switching**, and **additional silos** that slow down incident resolution.

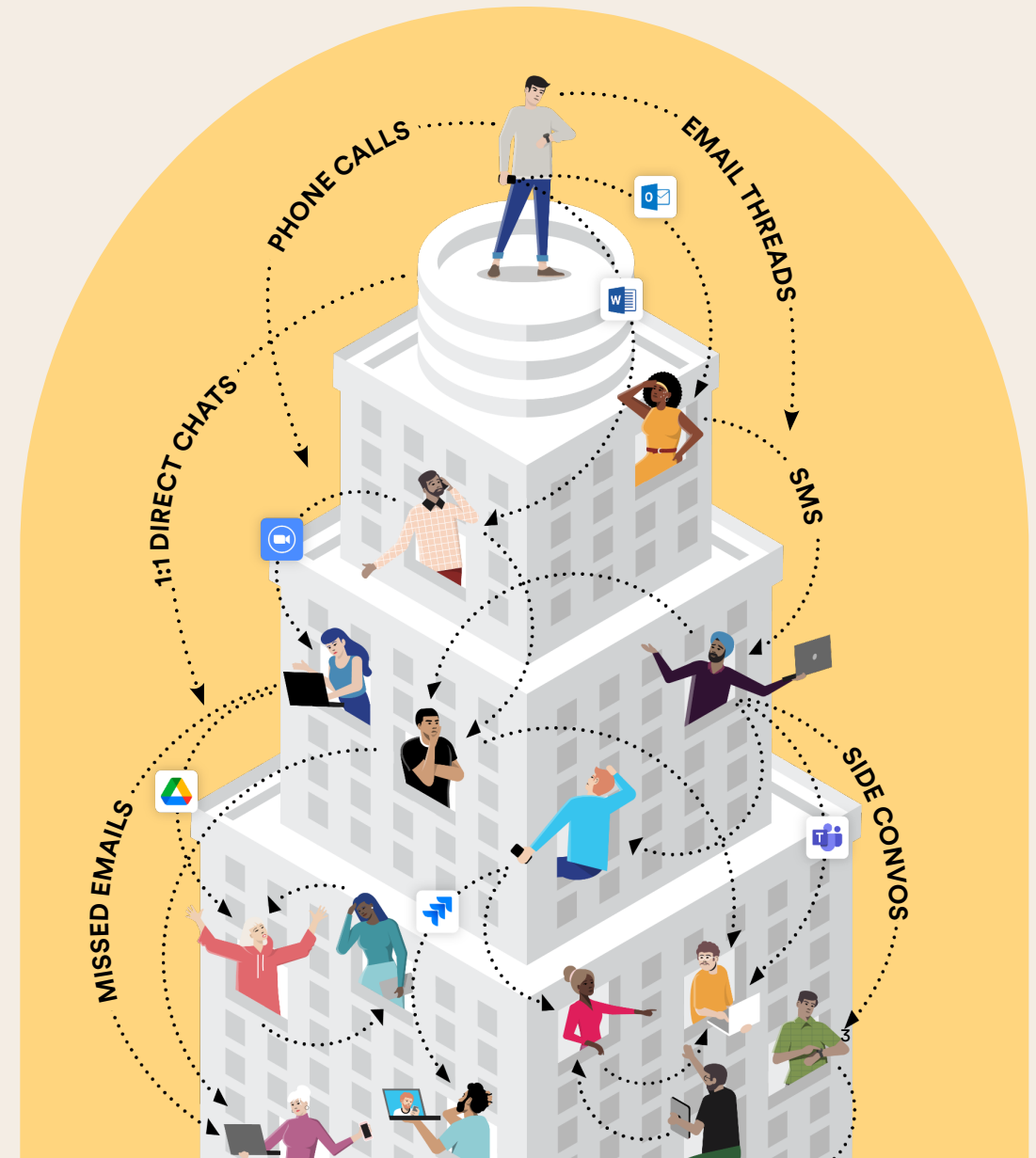
These unintended difficulties mean that enterprise developers and engineers must find new, more efficient ways to handle incident management seamlessly while keeping multiple teams and business lines in the loop.

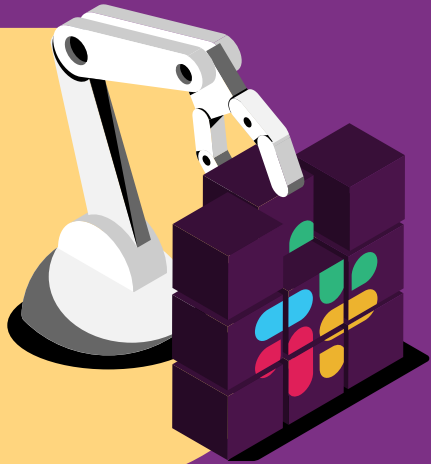
# So it's time to rethink the "war room"

The increasing complexity of incidents brings an urgent need to close the virtual "war room" for good. After all, it's called a war room for a reason: It creates panic, stress, and frustration. Consider the traditional approach:

- Someone in the Network Operations Center emails the on-call engineer. **Wait.**
- A team member investigates but needs others to assess severity. **Wait.**
- Responders slowly trickle in with no context. Then they interrupt the call. Information gets repeated. **Wait. Wait. Wait.**
- And when the incident gets elevated to a P1, five executives enter the war room. And the whole process starts all over again.

Every delay wastes the engineers' time, slows the response, and negatively impacts MTTR. What's worse, after resolution, engineers must spend even more time compiling fragmented information across recordings, emails, and chats for a debrief. Does that sound effective to you?





# Bring calm to the chaos with automation

Developers and operators deserve better than just a virtual war room. Slack uses automation, out-of-the-box integrations, and generative AI to centralize and speed up all things surrounding incident management.

## Slack offers three key capabilities:

### 1. Alerts and Real Time Response

- Huddles provide the ability to instantly connect over audio or video and share screens so you can get more done where you're already working to resolve the issue
- Slack provides out-of-the-box integrations with monitoring and incident response tools (PagerDuty, Datadog, AWS, etc)

### 2. Resolve Issues Quickly

- Slack lets you design workflows and lightweight custom apps that automate work and communications across both technical and business teams
- Slack Connect extends incident collaboration to vendors, partners, third-party developers, and even customers in a secure place
- Incident channels bring people together and deliver instant visibility
- Slack Canvas provides problem context quickly and links to the broader incident response team without flooding the main incident channel

### 3. Streamline Incident Reviews

- Generative AI app integrations draft communications and incident reviews automatically based on channel content
- Slack timestamps and preserves actions, decisions, and conversations during incidents to act as a timestamped audit trail for incident reviews

# The power of partnership

While automation and AI play leadership roles in creating streamlined incident management workflows, so, too, does integration.

Slack offers out-of-the-box integration with 2,600+ mission-critical tools that developers and engineers use every day, including:

- **Development platforms** like GitHub, GitLab, and Bitbucket
- **Monitoring and observability tools** like Datadog, AWS, Grafana, Dynatrace, New Relic, and Sentry
- **Incident management tools** like PagerDuty, Splunk On-Call, and OpsGenie
- **Customer service solutions** like Salesforce Service Cloud
- **Work management tools** like Jira, Asana, and Monday.com
- **Generative AI tools** like OpenAI, Anthropic, Cohere, and Jasper
- **Audio and video conferencing tools** like Microsoft Teams, Zoom, and Webex

Read on to see how Slack brings them all together and creates one seamless incident management experience.



# How it works: Incident detection

The following provides an example of how IT teams can use Slack for improved incident management.

The screenshot displays a Slack interface for a channel named `# monitoring-alerts`. The channel header indicates it is for "Monitoring high-priority / high-volume cases" and has 74 members. A message from **Incident Automation** (APP) at 10:55 AM reports: **INC-00447: Login failure rate exceeded threshold has been created by an External Service (PagerDuty - AWS Cloudwatch)**. The message includes details: Description: Login error rate increasing on mobile app; Status: Open; Severity: High; Start Time: Thu Jun 15 2023 21:16:06 GMT. Below the message are buttons for "Send Update", "Declare Incident", "Close Incident", "Escalate", and "De-escalate".

A "A huddle happened" message at 11:03 AM shows that Matt Brewer, Zoe Maxwell, and Lee Hao were in a huddle for 7 minutes. Below this, a "Thread" section shows three replies from **Incident Automation** (APP) at 11:30 AM, each reporting a successful action: "Jira Issue: INC-00447 created", "Salesforce incident created", and "Clerk has notified Incident Leadership via SMS about this issue. They will be kept informed about any updates."

Overlaid on the right side of the screen is a "Declare Incident" popup form. The form contains the following fields:
 

- Summary:** Mobile app login errors
- Severity:** Critical
- Long Description:** Users can't login via mobile app, detected by monitoring tools and confirmed by customer service. Sev1 issue we need to resolve immediately.

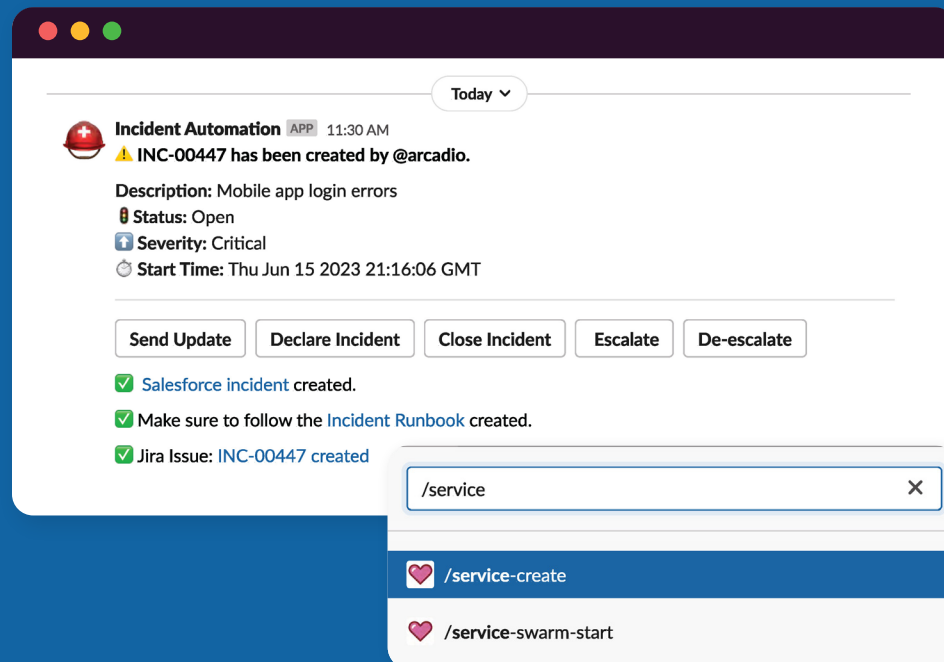
 The form has "Cancel" and "Create" buttons at the bottom.

An automated message within Slack, generated by **AWS** and **PagerDuty**, notifies everyone in the `#monitoring-alerts` channel immediately about a login failure on a company's mobile app. The team launches a Slack huddle to address the severity of the incident, quickly determines it's a Sev1 incident, and makes the final call in the thread, giving everyone visibility into key metrics.


With the click of the **"Declare Incident"** button inside the **Incident Automation** app, a responder enters minimal information into a popup box and formally declares the incident.

Slack automation then does the rest, opening the incident channel (`#INC-00047`), creating a **Jira** ticket and a **Salesforce Service Cloud** incident, and pushing a text message to executives. Key resources—an incident runbook, Jira ticket, Service Cloud incident and **Zoom** call link—get auto-bookmarked to the top of the channel. All these objects are tied directly to both the incident and the incident channel. Updates made to the incident by a user are propagated to all of the aforementioned tools automatically.

# How it works: Stakeholder collaboration




Today ▾


 **Incident Automation** APP 11:30 AM  
⚠️ INC-00447 has been created by @arcadio.

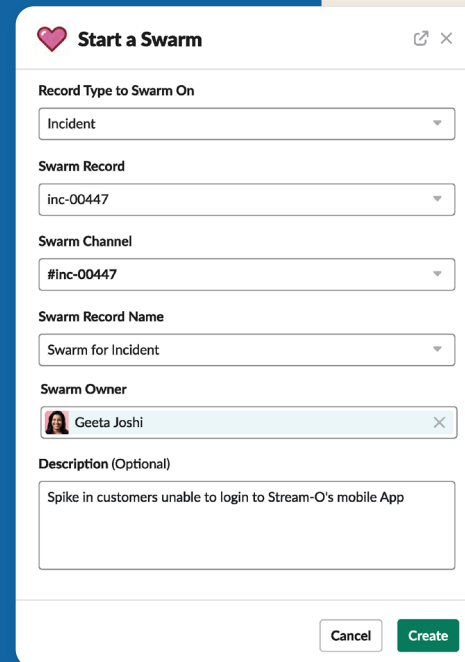
Description: Mobile app login errors  
📌 Status: Open  
🔴 Severity: Critical  
🕒 Start Time: Thu Jun 15 2023 21:16:06 GMT


[Send Update](#) [Declare Incident](#) [Close Incident](#) [Escalate](#) [De-escalate](#)

- ✅ Salesforce incident created.
- ✅ Make sure to follow the Incident Runbook created.
- ✅ Jira Issue: INC-00447 created

 /service-create

 /service-swarm-start




 **Start a Swarm** 🔗 ✕

Record Type to Swarm On

Swarm Record

Swarm Channel

Swarm Record Name

Swarm Owner  
 Geeta Joshi ✕

Description (Optional)

[Cancel](#) [Create](#)

The service team begins a swarm to discuss how they can communicate the incident with customers—all without interrupting the technical team's work. Once filled in, swarm details automatically get posted into the #INC-00047 channel.

# How it works:

## Stakeholder collaboration, continued

The screenshot displays the Salesforce Service Console interface for an incident titled "App Login Failure" (INC-00447). The console is divided into several sections:

- Incident Details:** Shows the subject "Mobile App Login Failure", description "Spike in customers unable to login to Stream-O's mobile App", and incident owner "Geeta Joshi" with a "High" urgency.
- Actions & Recommendations:** Provides options like "Change Case to Incident Owner" and "Finish Swarming".
- Swarm Record:** Shows a "Pinned Stream-O" with a "Service Cloud for Slack" app icon and a "Swarm request from @Geeta Joshi".
- Swarm Description:** Explains that when done collaborating, the swarm and records will be updated in Salesforce.
- Message:** A message from "Lee Hao" at 11:30 AM states, "This is impacting a lot of customers - can we work with the service escalation teams to get comms out?".

Overlaid on the console is a modal window titled "Add Banner to Experience Cloud Sites". The modal contains the following text:

Use banners to share updates and information with customers on your Experience Cloud sites. Select one or more related topics to get started.

The modal features a "Topics" dropdown menu with the following options:

- Stream-O AMER
- Stream-O EMEA
- Stream-O APAC
- Stream-O Global

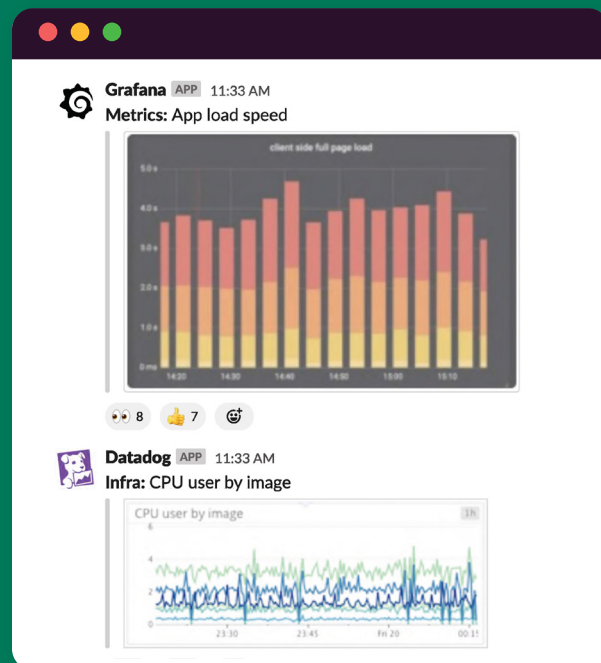
At the bottom of the modal are "Cancel" and "Next" buttons.

Customer-facing **stakeholders and agents** can view the incident record in **Salesforce Service Cloud** and join the swarm without ever leaving the service console. They can also collaborate on internal-facing and customer communications.

Once they determine the best approach, they can broadcast the latest status to customer-facing internal colleagues. They can also proactively notify customers by adding a banner to their website. And they can do it all right from the Salesforce Service Cloud.



# How it works: Incident troubleshooting



The figure shows two screenshots of a Slack interface. The top screenshot is a GitHub pull request for 'app\_fx/build\_repair' with a description 'Redirect endpoint to correct URL' and reviewers '@arcadio.buendia' and '@zoe.maxwell'. The bottom screenshot is a DeployBot confirmation dialog asking 'Do you want to deploy the application with ID streamo-app?' with 'Cancel' and 'Yes' buttons.

Back in Slack, the technical team starts troubleshooting. Team members easily share observability data—such as app load speed from **Grafana** or CPU utilization from **Datadog**—within the #INC-00047 channel, giving everyone instant visibility.

Once the fix is determined, the entire team sees the change being pushed in **GitHub**. They then can deploy the change directly in Slack using a custom **DeployBot** app, which is fully integrated with **GitHub**.

Once the build pipeline stages have passed, a team member deploys this change into production.

# How it works: Closing an incident

The screenshot displays a Slack interface with a thread for incident #inc-00447. On the left, a DeployBot message asks to deploy the application with ID streamo-app, with 'Cancel' and 'Yes' buttons. Below it, another DeployBot message shows a list of approved PRs and deployment steps, with 'No' and 'Yes' buttons. The main thread shows an Incident Automation message at 11:58 AM stating 'All Clear has been declared!'. Below this, three replies are shown: 'Jira Issue: INC-00447 updated.', 'Salesforce incident resolved.', and 'Clerk has notified Incident Leadership via SMS about this issue. They will be kept informed about any updates.' A 'Close Incident' dialog box is overlaid on the right, titled 'Close Incident' with a close button. It contains a text area with the note: 'All clear. A mobile app release from this morning accidentally pointed to the wrong endpoint. Resolved by re-deploying with correct URL.' and 'Cancel' and 'Submit' buttons.

With the incident resolved, automation in Slack makes it easy to create closure. A team member simply clicks the “**Close Incident**” button, adds their notes, and clicks submit.

Automatically, the **Jira** ticket is closed out, the **Service Cloud** incident is auto-resolved, and a final text message is pushed to executives. In addition, users can push a message to several different business stakeholder Slack channels, giving everybody visibility that the incident has been resolved.

# How it works: Root cause analysis

The screenshot shows a Slack thread with the following content:

- DeployBot** (APP) 11:38 AM: Do you want to deploy the application with ID streamo-app? [Cancel] [Yes]
- DeployBot** (APP) 11:40 AM: Select an approved PR: ACME-123 Combine sign-up steps 2&3 on...
  - ✓ Executing Merge pull request
  - ✓ Executing CircleBuild
  - CircleCI Build \$8821 - Completed after 0:42
  - Deploy to prod? [No] [Yes]
  - ✓ Push release to master
  - ✓ Deployment successful
  - ✓ Duration: 5 minutes
- Incident Automation** (APP) 11:58 AM: All Clear has been declared! 3 replies Last reply today at 11:58 AM

**Thread #inc-00447**

- Incident Automation** (APP) 11:30 AM: Salesforce incident resolved.
- Incident Automation** (APP) 11:30 AM: Clerk has notified Incident Leadership via SMS about this issue. They will be kept informed about any updates.
- Incident Automation** (APP) 11:30 AM: Confluence: Root Cause Analysis [INC-00447](#) created
- ChatGPT** (APP) 11:30 AM: A summary of the root cause analysis for #inc-00447 is now drafted in the document

#INC-00447

**Root Cause Analysis Summary**

Incident Title	Mobile app login errors
Severity	Critical
Description	Users can't login via mobile app, detected by monitoring tools and confirmed by customer service. Sev1 issue we need to resolve immediately.
Slack Incident Channel	<a href="#">Slack Incident Channel</a>
Last Update	Seems to be an endpoint issue. Changing the URL for logins now and will report back in 15 minutes if resolved.
Close Notes	All clear. A mobile app release from this morning accidentally pointed to the wrong endpoint. Resolved by re-deploying with correct URL.
Jira Ticket	<a href="#">Ticket to Jira Issue</a>
Salesforce Incident	<a href="#">Salesforce Incident Record</a>

**Problem Statement:** Beginning at 11:30am PST, users in AWS US regions were not able to login to the mobile app on both iOS and Android. This lasted for 43 minutes, until corrective action was taken.

**Root Cause:** A deployment from March 23 2023 in the day hardcoded user endpoints to a specific production URL. When a network change occurred this morning to enable dynamic URLs through load balancing, users were unable to access the new production URLs and could not login to accounts on their mobile apps.

**Corrective Actions:**

- Redirect Prod1 & Prod2 environments in AWS West2 to AWS East1 temporarily
- Remove hardcoded URL and replace with value from ITSM

Mere moments after an incident is closed, a root cause analysis document is auto-created in **Confluence** and auto-populated with incident basics.

Even better, a generative AI tool like **ChatGPT** or **Claude** can instantly draft an incident summary in the documentation tool of your choice, saving teams hours of work and allowing them to take preventive actions sooner to ensure the incident doesn't happen again.

# Automated incident management with Slack lets developers:

- Assemble a team as fast as possible all in one place—no more separate emails, DMs, and texts
- Get everyone up to speed in an instant with the context they need to begin troubleshooting immediately, not in minutes or hours
- Eliminate blaming and bias
- Create a single source of truth for all incident details
- Complete accurate and comprehensive incident reviews in days, not weeks, with the power of generative AI and automation
- Learn from mistakes and act on improvement opportunities faster to prevent similar future incidents
- Spend more time innovating and less time putting out fires

With Slack, users enjoy:

---

**17%**

faster time to  
detect incidents


**19%**

reduced mean-time-  
to-repair incidents

---

**86%**

agree Slack makes it easier  
to share learnings





## Getting started: 3 simple tips for smoother incidents today

**1. Integrate and automate.** Integrate your monitoring tools with our out-of-the-box Slack app integrations to create real-time alerts in the channels where your developers and engineers spend the most time. Next, integrate with other tools that can help you automate part or all of your incident response.

**2. Use #incident-channels to supplement video war rooms or huddles.** You don't have to automate all steps of the process at once. If you still rely on a real-time "war room," use channels to supplement what you already use (Teams, Zoom, or even a Slack huddle), and start building automation incrementally while creating a searchable archive of the incident.

**3. Get out of DMs and use threads.** Posting messages in threads will give your entire team transparency and neatly organize information so you're not flooding the channel or overwhelming a larger team.

# Stop waiting for better incident management

Software and security incidents don't happen on anyone's schedule. But they don't have to cause widespread panic, either.

By bringing automation and generative AI to the world of incident management, Slack offers a clear path through the chaos, so engineers and developers can solve incidents faster and get back to their more essential work.

If you're already using Slack, [start automating your incident management processes](#). If you're new to Slack and would like to learn more, start by learning more about [automation and workflows](#) with Slack.

