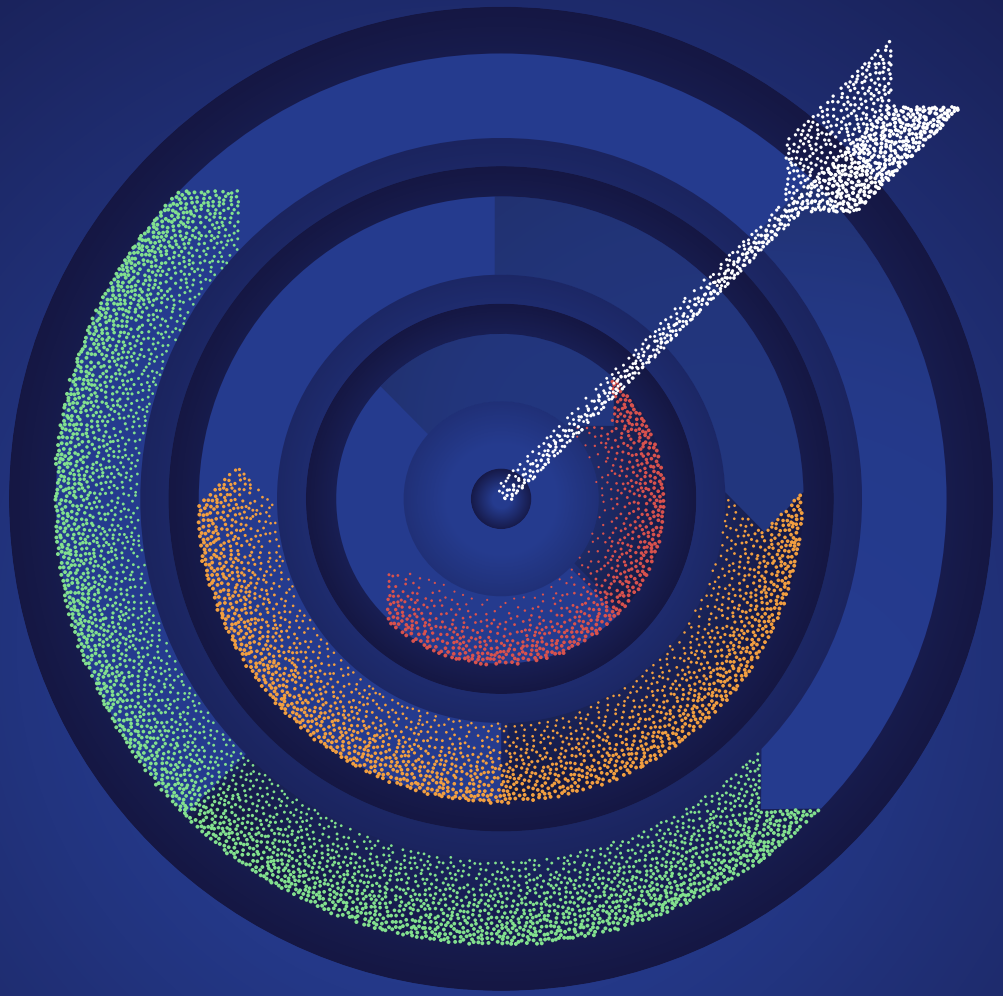
3 reasons

companies miss MTTR targets & how to boost your success rate



MTTR insights from the

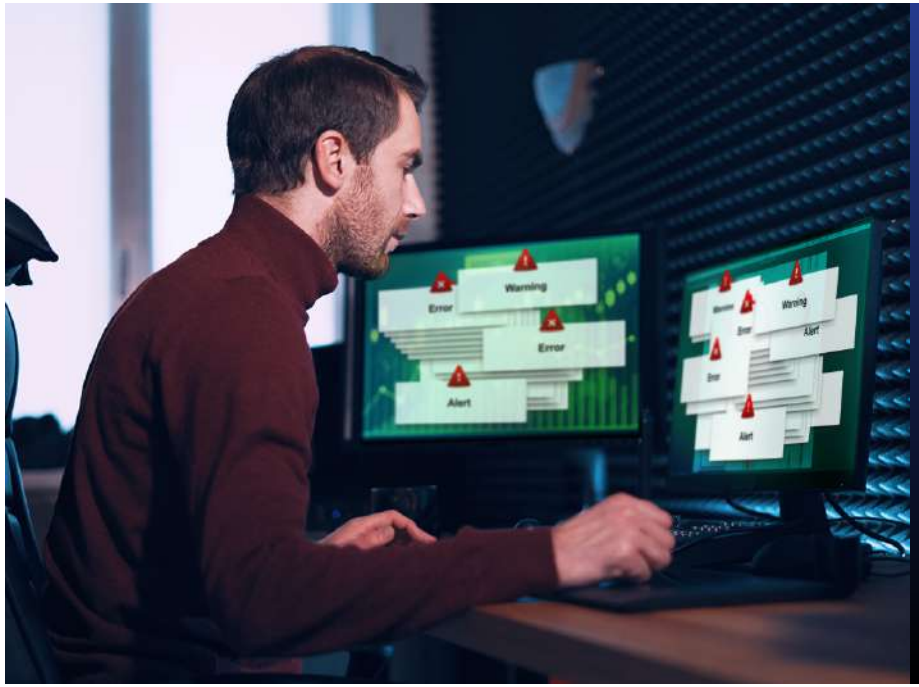
2023 Cloud Native Observability Report

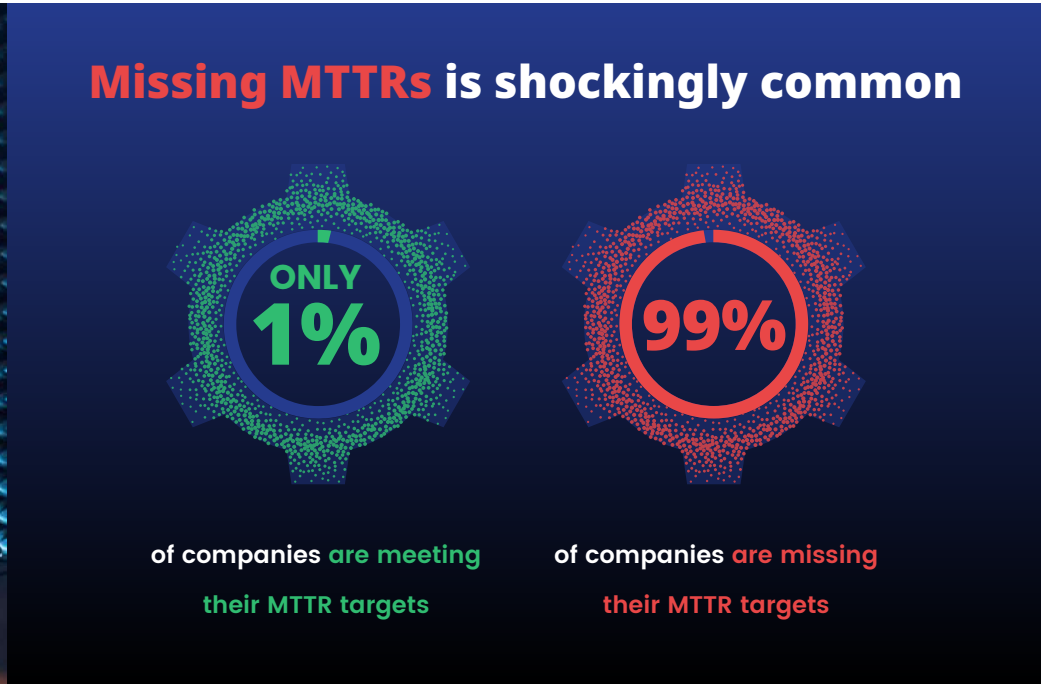
chronosphere

MTTR struggles

Companies everywhere are counting on cloud native architectures to enable engineers to deliver exceptional customer service more efficiently to grow top-line revenue and protect the bottom line. Yet when incidents occur, those same technical teams are finding it not only increasingly challenging, but downright impossible, to remediate issues quickly.

The recent **2023 Cloud Native Observability Report** survey of 500 engineers and engineering leaders at U.S.- based companies[1] revealed just how serious the mean time to repair (MTTR) problem has gotten.





[1]2023 Cloud Native Observability Report, January 2023.

Disappointingly,

only 7 of 500 respondents agree

they met or exceeded their MTTR goals. Put another way, very few respondents say their actual MTTR was the same or less than their target MTTR.

This ebook examines the top three reasons for this extraordinary gap and what your company can do to remediate customer issues faster.

1. MTTR is often poorly defined

The concept of MTTR dates back to the 1960s when it was used by the Defense Technical Information Center to assess the reliability and availability of military equipment. As system software and application performance has risen in importance, this term has been adopted by engineering teams. Unfortunately, it's morphed over time and no longer has a consistent definition.

MTTR can sometimes be defined as "mean time to restore," "mean time to respond," or "mean time to remediate."

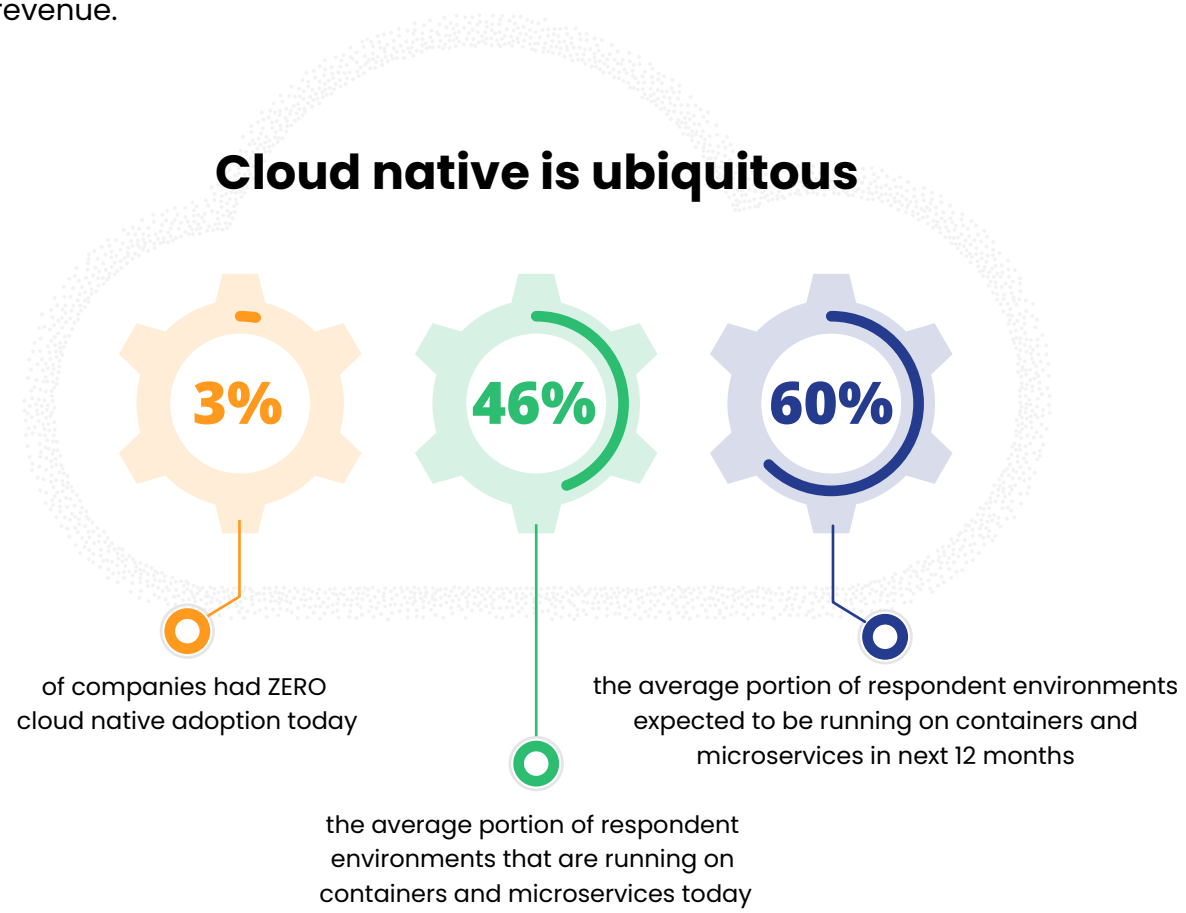
However, MTTR is properly defined as "mean time to repair" across industries ranging from telco to manufacturing to technology. It is the measurement of how long it takes from an issue occurring to when the issue is fixed.

Because different companies start and stop the MTTR clock at different points — some measure the time it takes to get systems back to some operational semblance or the time it takes to completely fix the underlying problem — your company can only benchmark repair time against itself; you can't benchmark MTTR against your peers.

But inconsistency in metrics measurement is just part of why, on average, companies responding to the **2023 Cloud Native Observability Report** are targeting and missing the MTTR mark so badly.

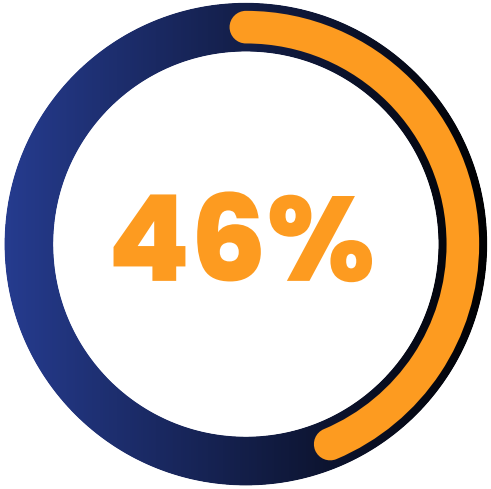
2. Cloud native complexity is more challenging than anticipated

For the past few years, enterprises have been adopting cloud native architectures for key business benefits, namely to increase efficiency and revenue.



Cloud native adoption is growing 30% YoY

What portion of your overall environment is cloud native **TODAY?**



What portion of your overall environment will be cloud native **IN 12 MONTHS?**

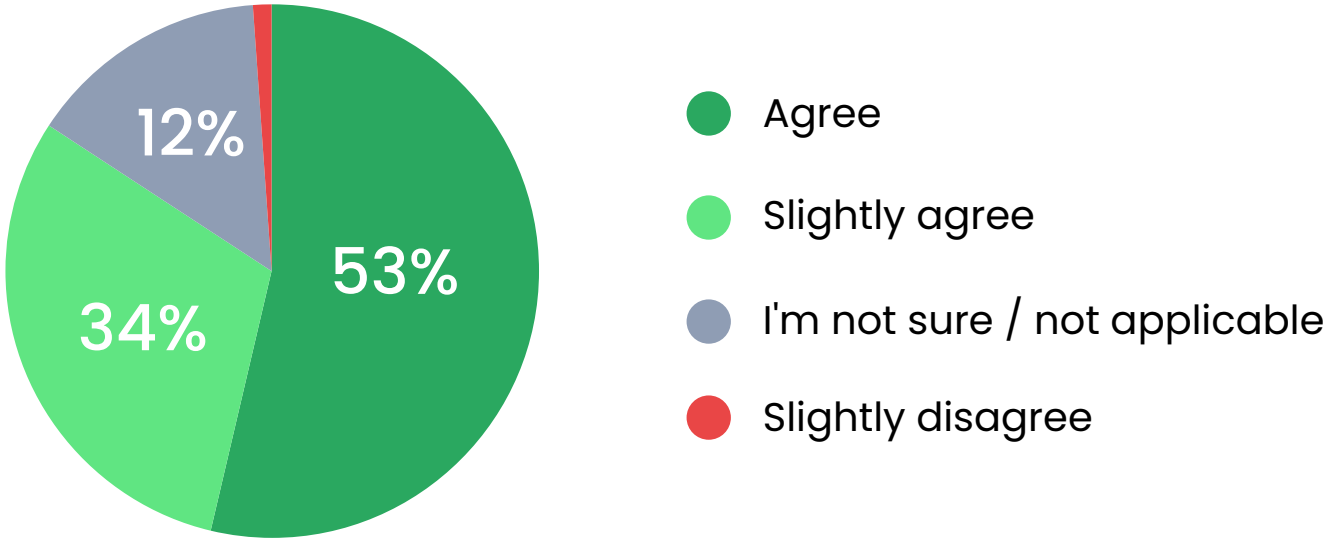
While yielding positive benefits, the continued shift to cloud native environments has also introduced:

- *Exponentially greater complexity*
- *Higher volumes of observability data*
- *More pressure on engineering teams supporting customer-facing services*

Cloud native complexity makes finding and fixing issues harder

87% *of engineers surveyed say cloud native architectures have increased the complexity of discovering and troubleshooting incidents*

Using cloud native architectures has increased the complexity of discovering and troubleshooting incidents



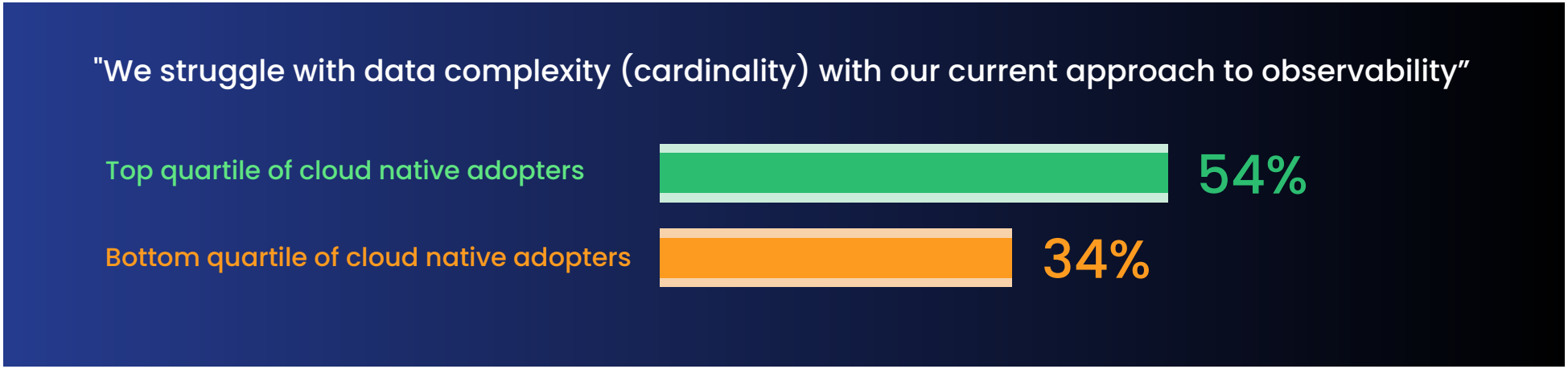
The inability to successfully address cloud native complexity can have significant negative MTTR impacts as well as customer and revenue impacts.

One of the biggest culprits in the increased complexity is due to cardinality.

Cardinality is a math term that describes the unique combinations you can achieve with a single data set. Here's a very simple explanation:

- *If two people each have 20 lollipops, but one person has 5 flavors and 3 sizes, the unique number of combinations they could have is 15.*
- *If the other person has only 5 flavors all one size, the number of unique combinations is 5.*
- *Each person has the same volume of lollipops, but one has a lot more complexity.*

In the observability world, instead of flavors and size combinations, it's time streams. For example, if one engineer decides to add UUID to their metrics, suddenly your cardinality explodes, catastrophically slowing down the systems and driving up costs.

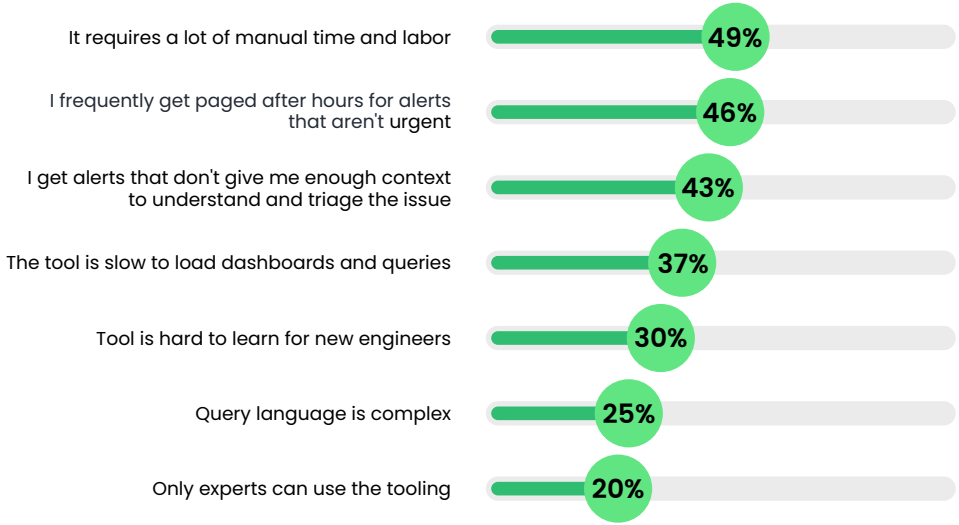


3. Observability tools are failing engineers

The **2023 Cloud Native Observability Report** also reveals the observability tools that companies put in place before cloud native growth — which should be helping engineers achieve MTTR goals — aren't working as promised.

Individual contributor engineers surveyed identified a host of shortcomings with their current observability solutions, ranging from too much manual time and labor to after-hours pages for non-urgent issues to insufficient context to understand and triage an issue.

Top 7 challenges with observability tools



Alerts aren’t working

When alerts show context linking them to services or customers affected by an incident and also give engineers actionable dashboards, companies can significantly shrink time to remediation.

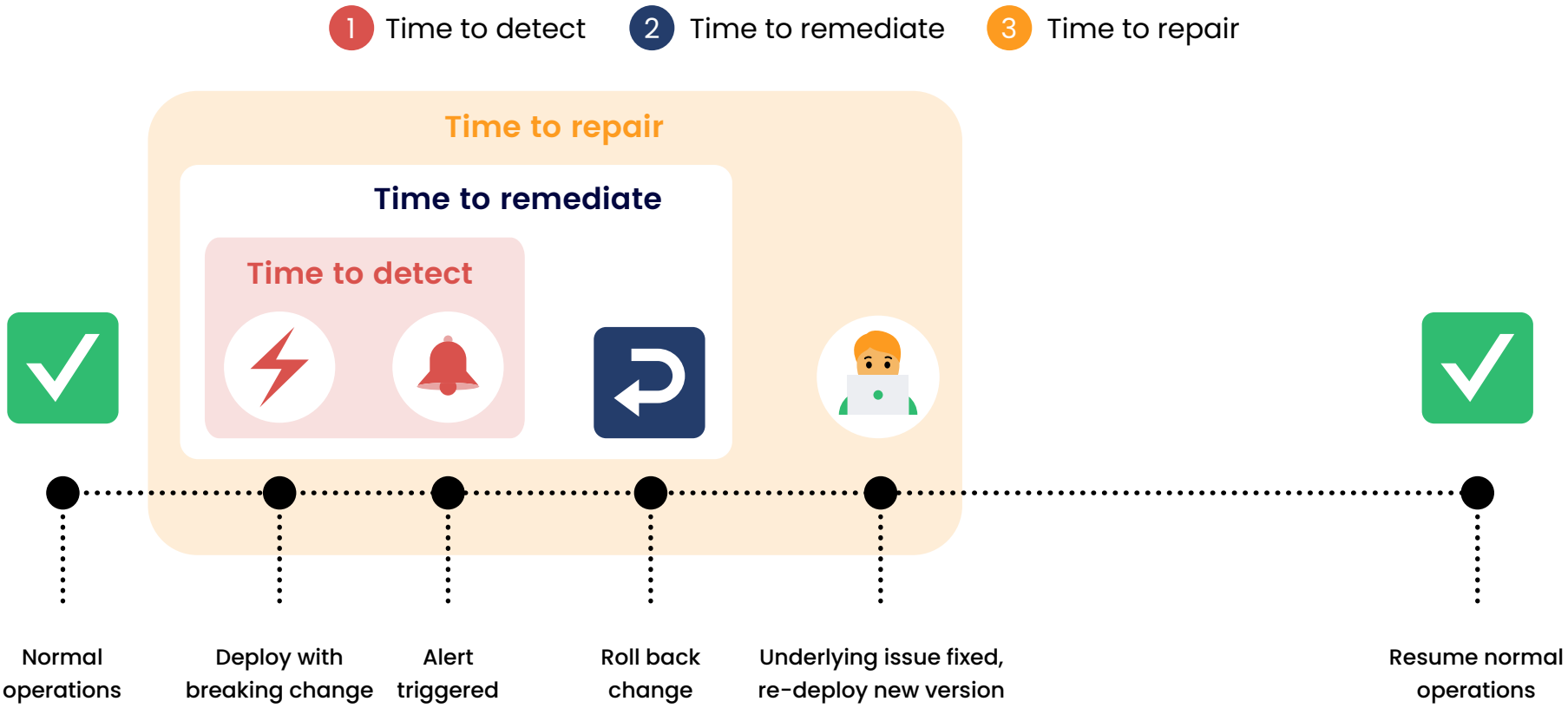
59% *of engineers say half of the incident alerts they receive from current observability solutions aren’t helpful or usable*

If you have existing observability capabilities — alerting, exploration, triage, root-cause analysis — failing your engineers during critical incidents, it’s highly unlikely your teams will meet their MTTR targets.

Implement fixes faster

To get better at achieving your MTTR goal, your organization needs to find ways to shorten the three key steps of the MTTR equation.

Improve MTTR by shortening 3 key steps



Time to detect

mean time to detect (MTTD)

To reduce your MTTD, collect data as frequently as possible so your observability solution can ingest and generate alerts rapidly. This typically includes setting a low scrape interval.

Time to remediate

To reduce the amount of time it takes to stop a customer's pain and put a fix in place (even temporarily) after shrinking your MTTD, close the gap between detection and remediation by providing more context and clear actionable data in the alerts.

Time to repair

Once you've optimized your detection and remediation times, focus on fixing the underlying issue. Even though customers are not typically affected, long repairs force engineers into endless troubleshooting loops, taking valuable time away from innovation.

The faster the engineering teams know there is a problem, the faster they can start to remediate it. Take Robinhood, which operates in the highly-regulated FinTech space, as a case in point. The company had a 4-minute gap between an incident happening and an alert firing. By upgrading its observability platform and shrinking data collection intervals, Robinhood reduced MTTD to 30 seconds or less.

Access more insights from the 2023 Cloud Native Observability Report and learn how you can boost your MTTR success rate at www.Chronosphere.io

Download the Report