# Techstrong Research

## PulseMeter Sponsored by ctrlstack

For all their benefits, modern development practices (DevOps, microservices architectures, increased deployment frequency) increase problem diagnosis and resolution complexity. The typical approach to troubleshooting, leveraging centralized observability data, may not catch the root cause of most problems—a code or configuration change. Not quickly pinpointing the root cause of the problem prolongs the issue and may result in temporary fixes, repeat occurrences and longer downtime. The so-called "troubleshooting tax"—the amount of time engineers spend on troubleshooting—remains hard to quantify and a challenge to minimize.

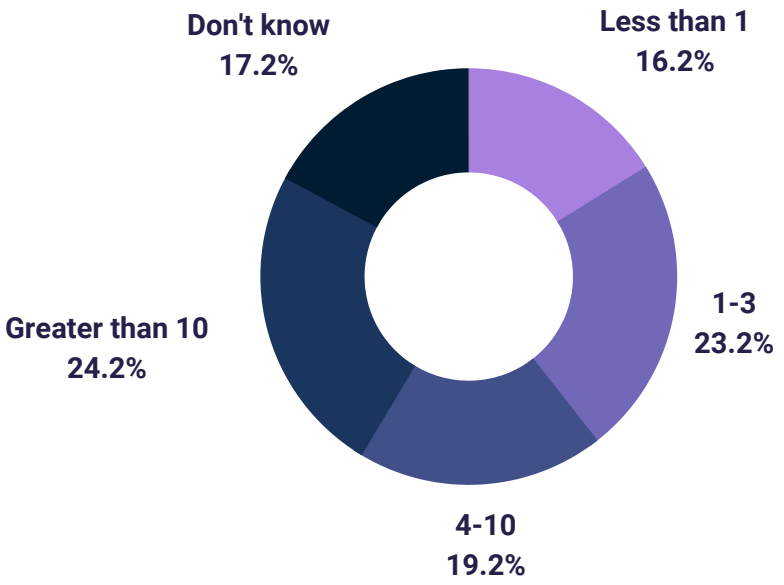
In 2022, Techstrong Research polled our community of DevOps, cloud-native, cybersecurity and digital transformation readers and viewers to take their pulse on tracking changes within their DevOps environment. We found that a greater percentage of the respondents (43% versus 39%) implement more than four changes per day and only 40% have centralized change tracking, although another 45% intend to centralize tracking. Regarding downtime, 51% of respondents have an average MTTR of more than an hour, with 30%-50% of that time spent troubleshooting. We found the troubleshooting tax is material, and organizations should focus efforts on how to reduce it.

---

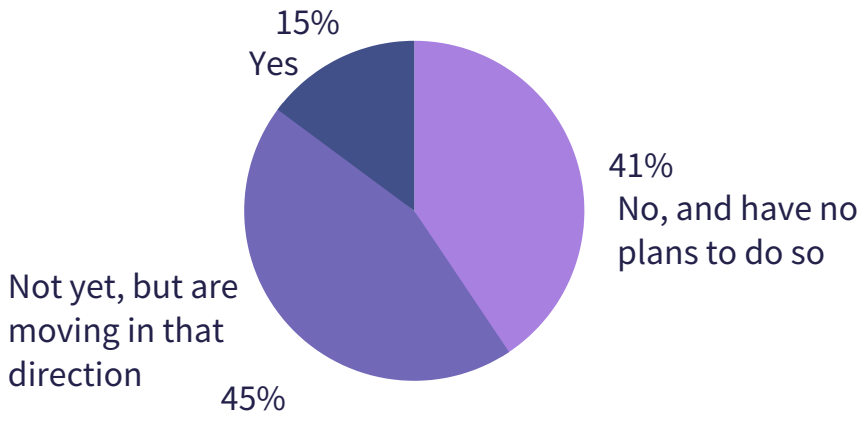
## Frequent code and configuration changes are poised to grow

### How many code or configuration changes happen in your environment daily?

Respondents are split mainly between traditional (< 3 changes) and modern (>4 changes). We expect daily changes to skyrocket as DevOps and cloud-native take root.



# 86% already centralize or plan to centralize production change tracking



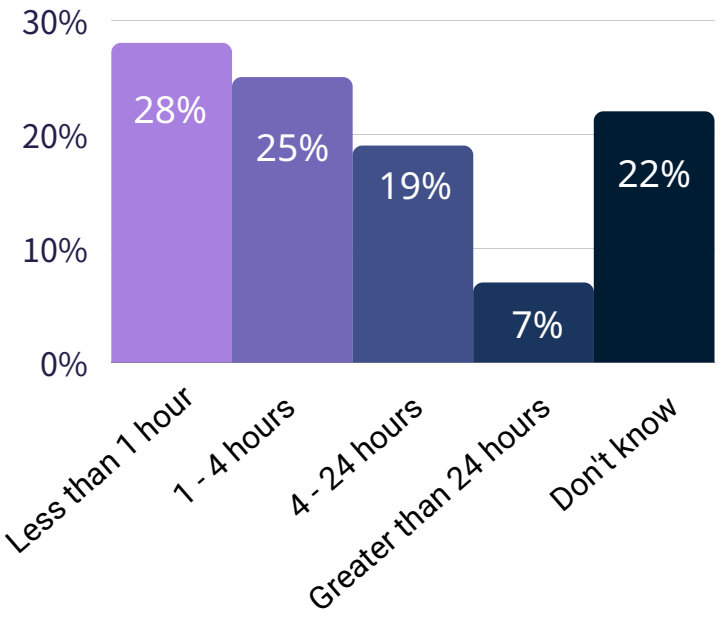

**Is the tracking of every production change centralized across all applications and infrastructure?**

It's just a matter of time before a supermajority of respondents centralize production changes.

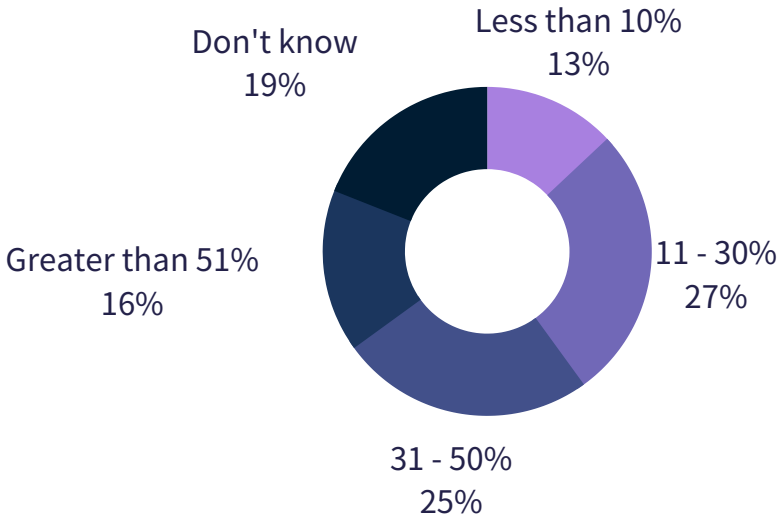
# Average downtime is greater than an hour for over 51% of respondents

**What is MTTR for an outage in your environment?**

Every minute of application downtime costs money, so improving response time should be a corporate imperative.



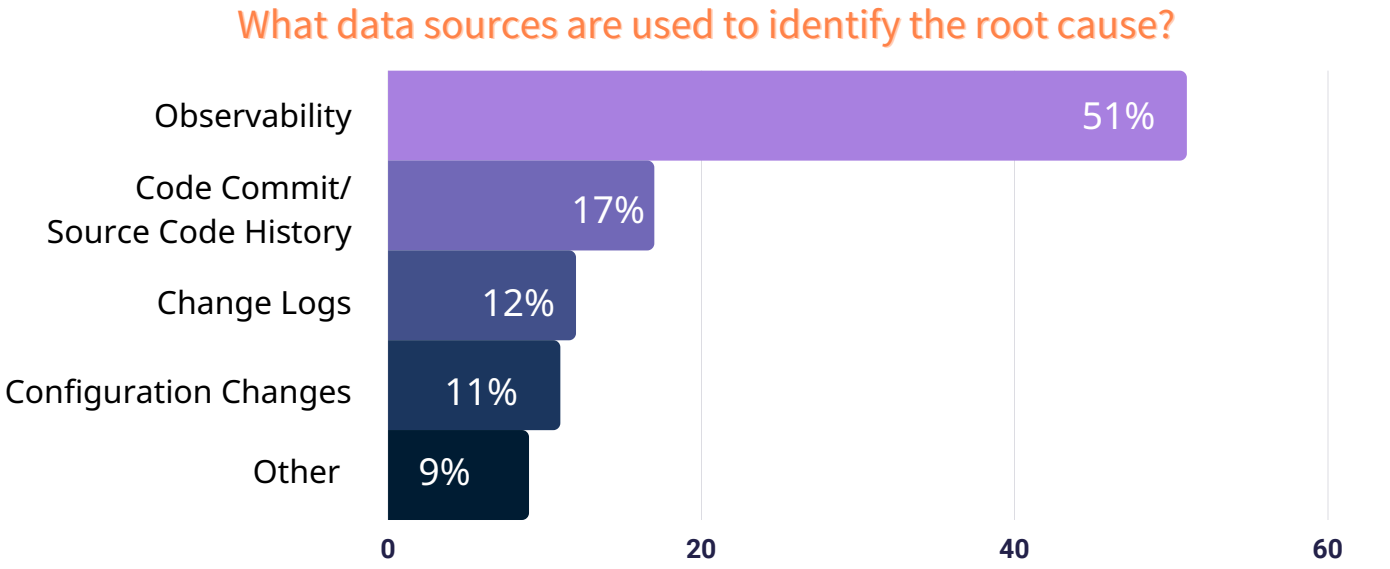

# Identifying root cause faster can reduce the time to restore service by 30% - 50%




**Of that time, how much is spent identifying root cause (troubleshooting/debugging) when issues develop?**

Respondents spent a significant amount of time just figuring out what happened. Streamlining that process restores systems far faster.

# Few respondents looked at change logs and configuration data to identify root cause




Observability data is the go-to for troubleshooters, but are they missing a key piece of the puzzle by not looking at change data, including code commit and configuration changes?

## Techstrong Research Analyst View

Whether it's the opportunity cost of spending time troubleshooting and fixing or lost revenue, downtime incurs both direct and indirect costs. DevOps helps to integrate more robust testing and automation into the integration and deployment processes, which can identify defects before deployment. But issues still happen and defects are still deployed, which means these issues need to be identified and remediated.

DevOps environments are complicated and troubleshooting takes up a significant portion of the response (30%-50%). Streamlining the root cause identification process and reducing that "troubleshooting tax" will reduce the MTTR. But organizations aren't availing themselves of the entirety of the available data to do so.

By only relying on observability data, DevOps practitioners miss a key opportunity to pinpoint issues faster by starting with recent code, infrastructure and configuration changes. What's required is not the lumbering, approval roadblocks of change control boards and processes of the past. Application and infrastructure changes occur at vastly higher velocities, requiring automation of processes throughout DevOps pipelines. Change data, automation and analysis must also be integrated into DevOps workflows to avoid data silos and to keep pace with the delivery demands placed upon software and operations teams.

The bottom line is that broader and contextualized data collection, automation, correlation and analysis improves troubleshooting. In addition to using observability data, DevOps professionals need to broaden their use of data to include operational activities such as code deployment and configuration changes. These events carry relevant context to enable faster and more accurate root cause analysis.