Microsoft

SoCC ACM Symposium on Cloud Computing

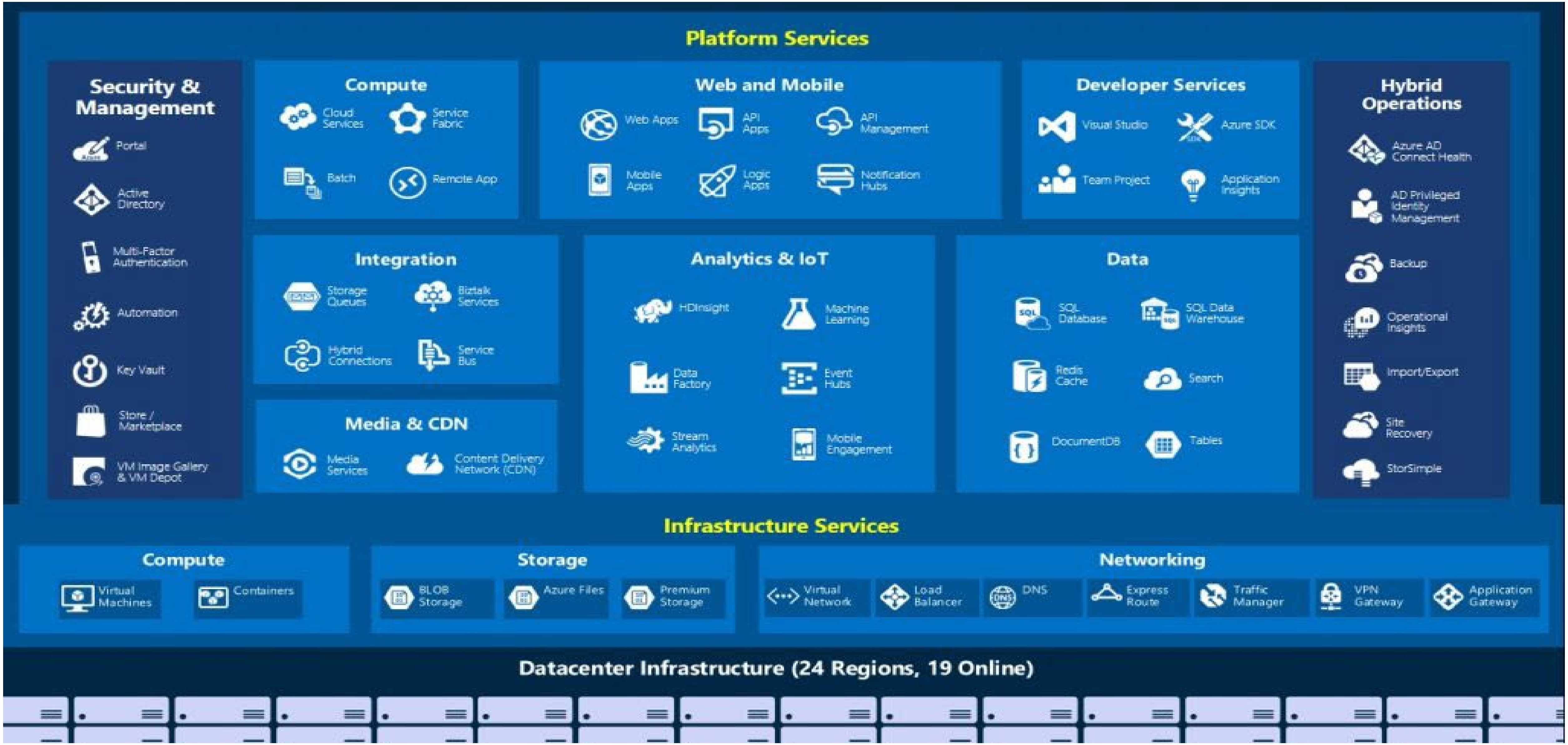
How to Fight Production Incidents? An Empirical Study on a Large-scale Cloud Service

Supriyo Ghosh, Manish Shetty, Chetan Bansal, Suman Nath

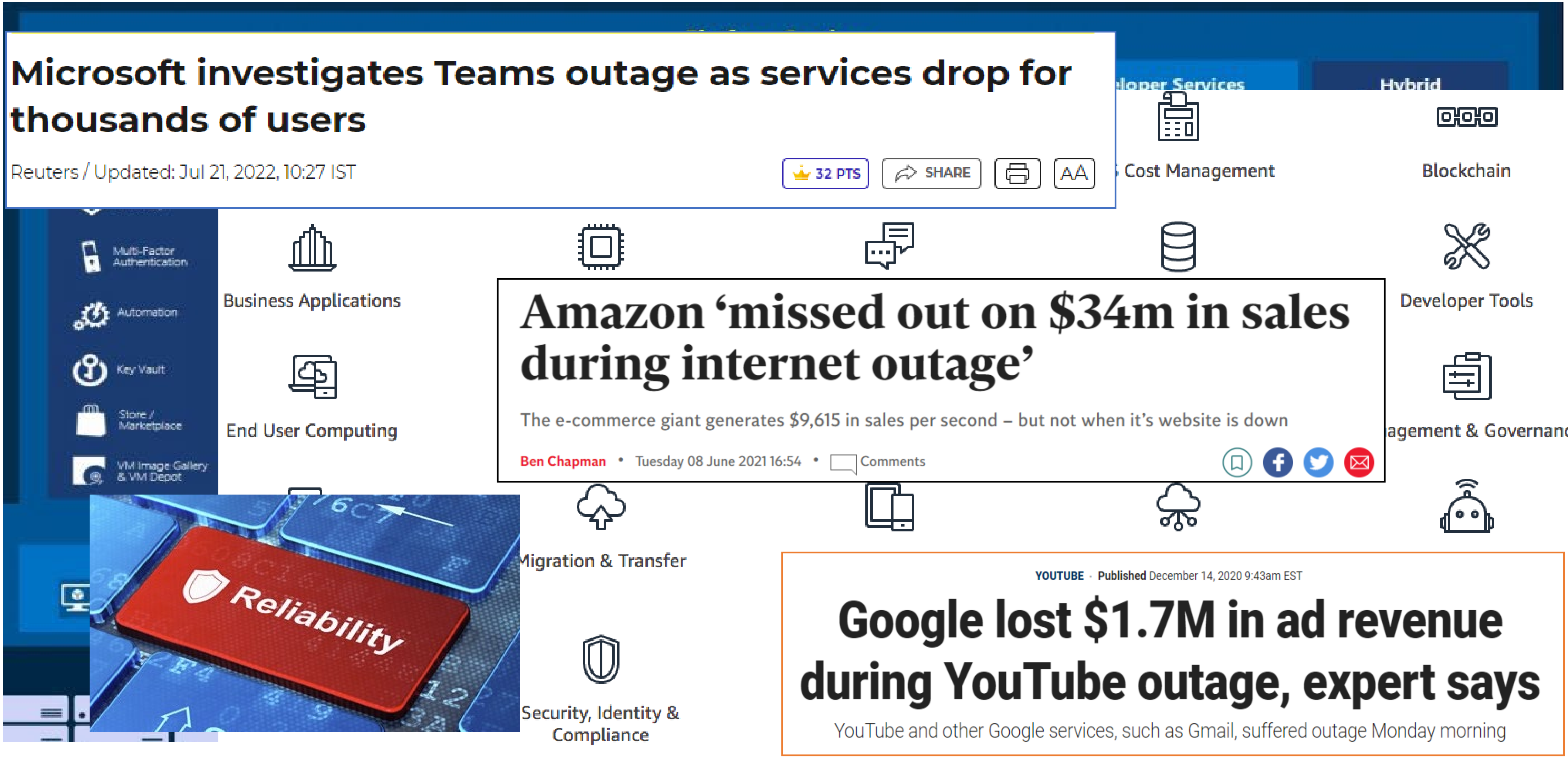
Microsoft

13th Symposium on Cloud Computing (SoCC’22)

Cloud Services in Azure



Cloud Service Incidents are Inevitable and Costly

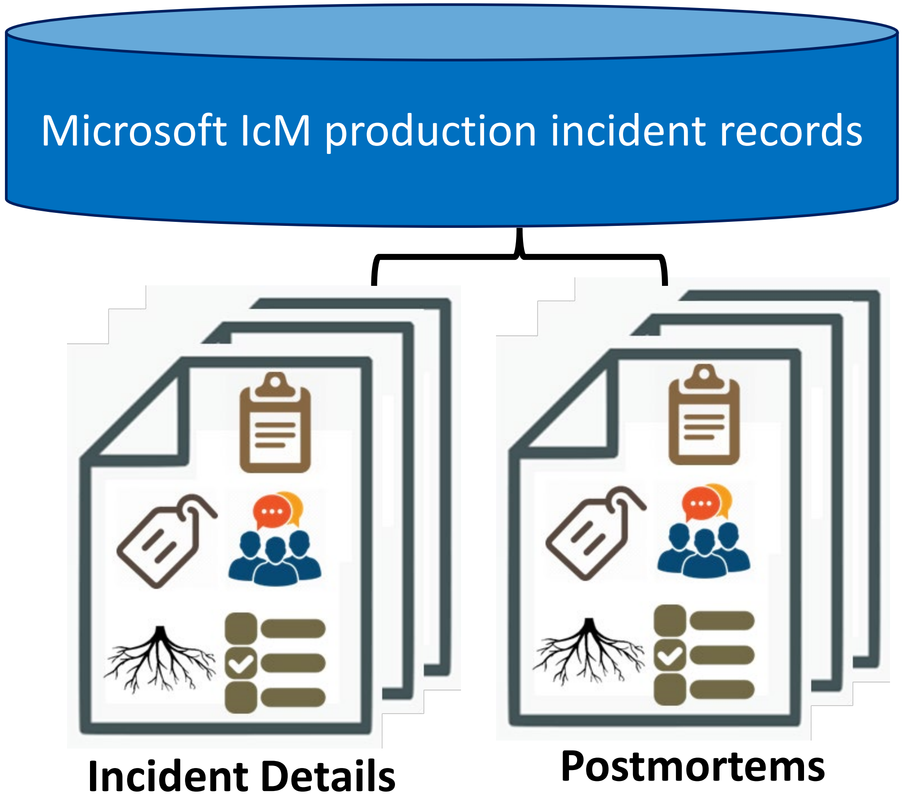


Motivation

- Production incidents adversely affect services.
  - **Financial impact** due to SLA violation.
  - **User dissatisfaction.**
  - **Loss of productivity** of on-call engineers (OCEs).

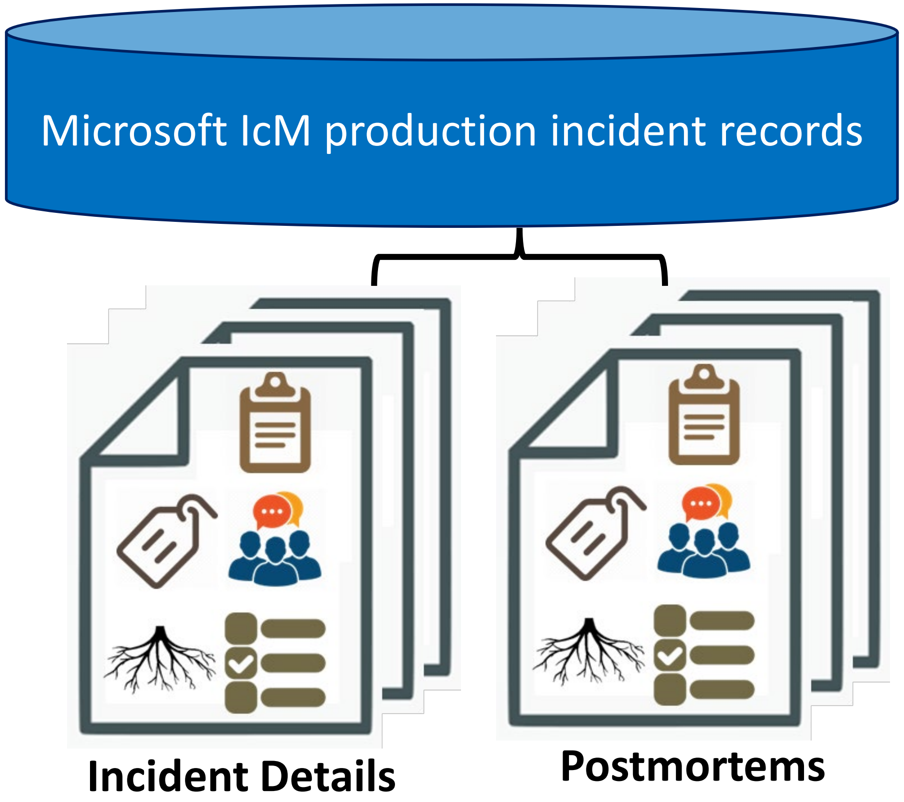
Motivation

- Production incidents adversely affect services.
  - **Financial impact** due to SLA violation.
  - **User dissatisfaction.**
  - **Loss of productivity** of on-call engineers (OCEs).
- Need to study real-world incidents
  - Incident Management tool (IcM) has plethora of rich information for recent high severity production incidents.
  - Post-mortem reports contain useful structured and unstructured information regarding root cause and mitigation.



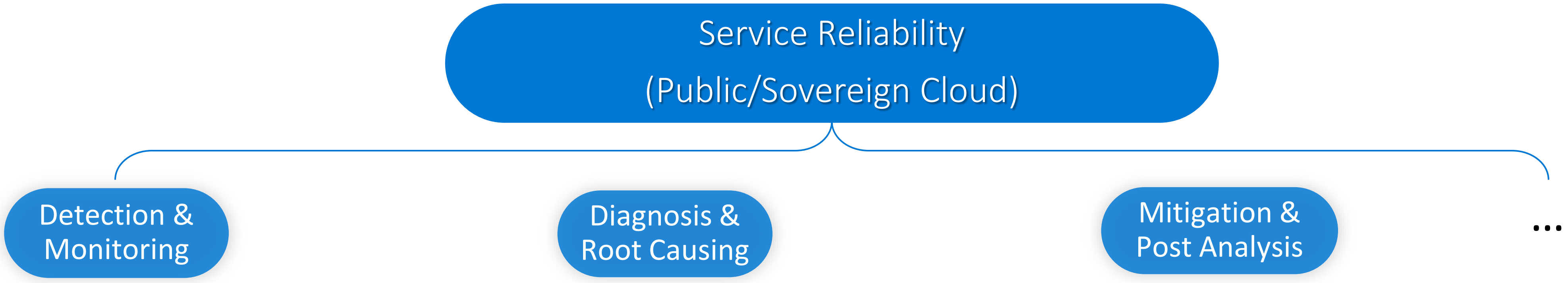
Motivation

- Production incidents adversely affect services.
  - **Financial impact** due to SLA violation.
  - **User dissatisfaction.**
  - **Loss of productivity** of on-call engineers (OCEs).
- Need to study real-world incidents
  - Incident Management tool (IcM) has plethora of rich information for recent high severity production incidents.
  - Post-mortem reports contain useful structured and unstructured information regarding root cause and mitigation.

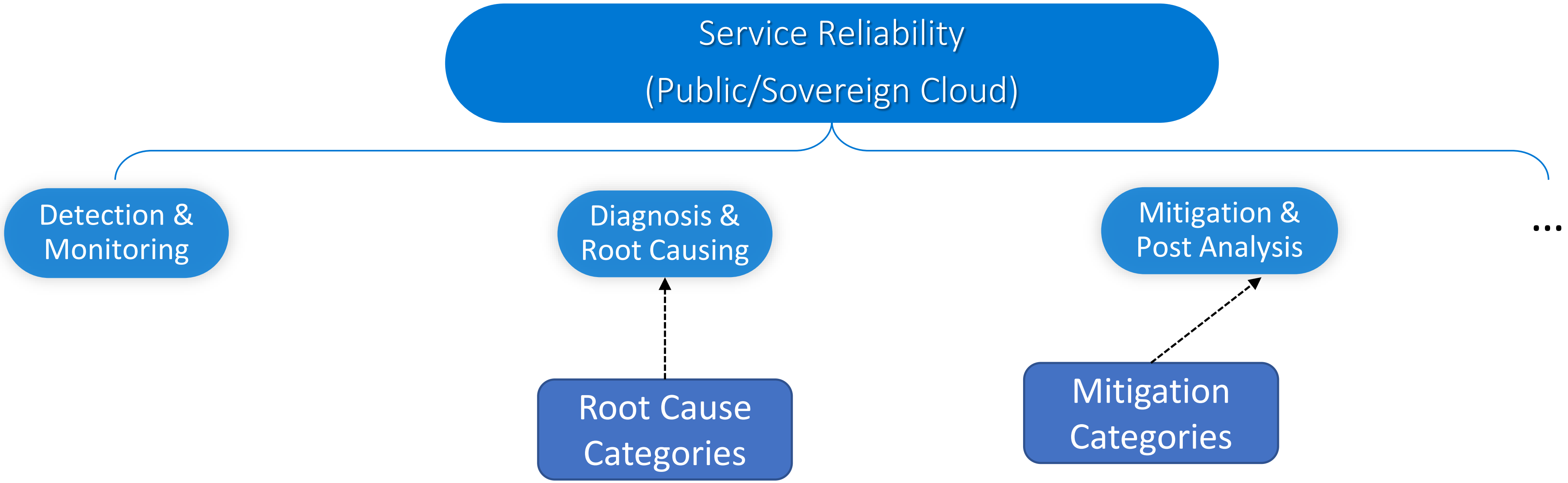


- How to leverage historical incident experiences to improve reliability of services and infrastructure?

Research Questions



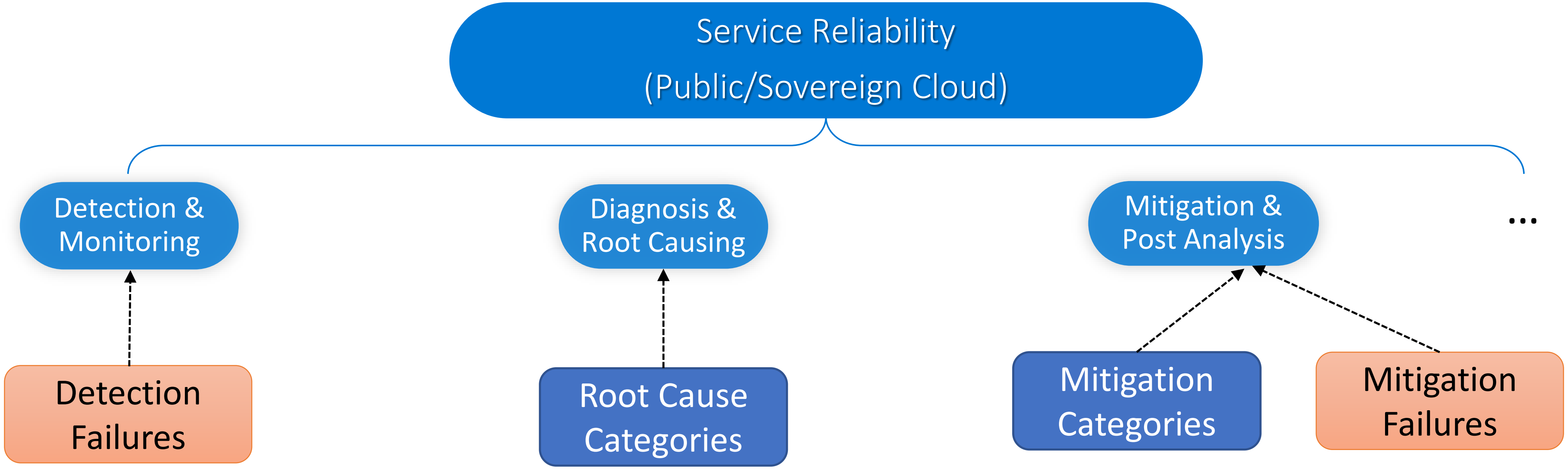
Research Questions



Questions We Aim to Address

1. *Why the incidents occurred and how they were resolved?*

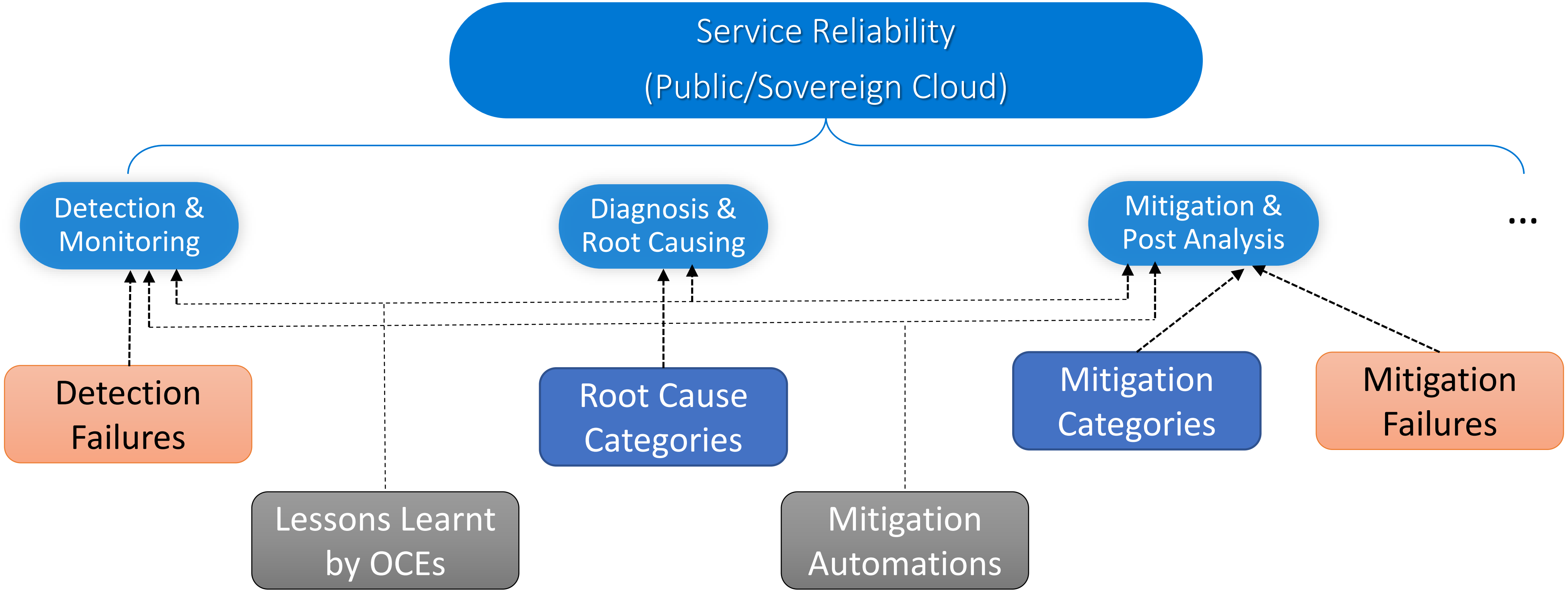
Research Questions



Questions We Aim to Address

1. *Why the incidents occurred and how they were resolved?*
2. *What the gaps were in current processes which caused delayed response?*

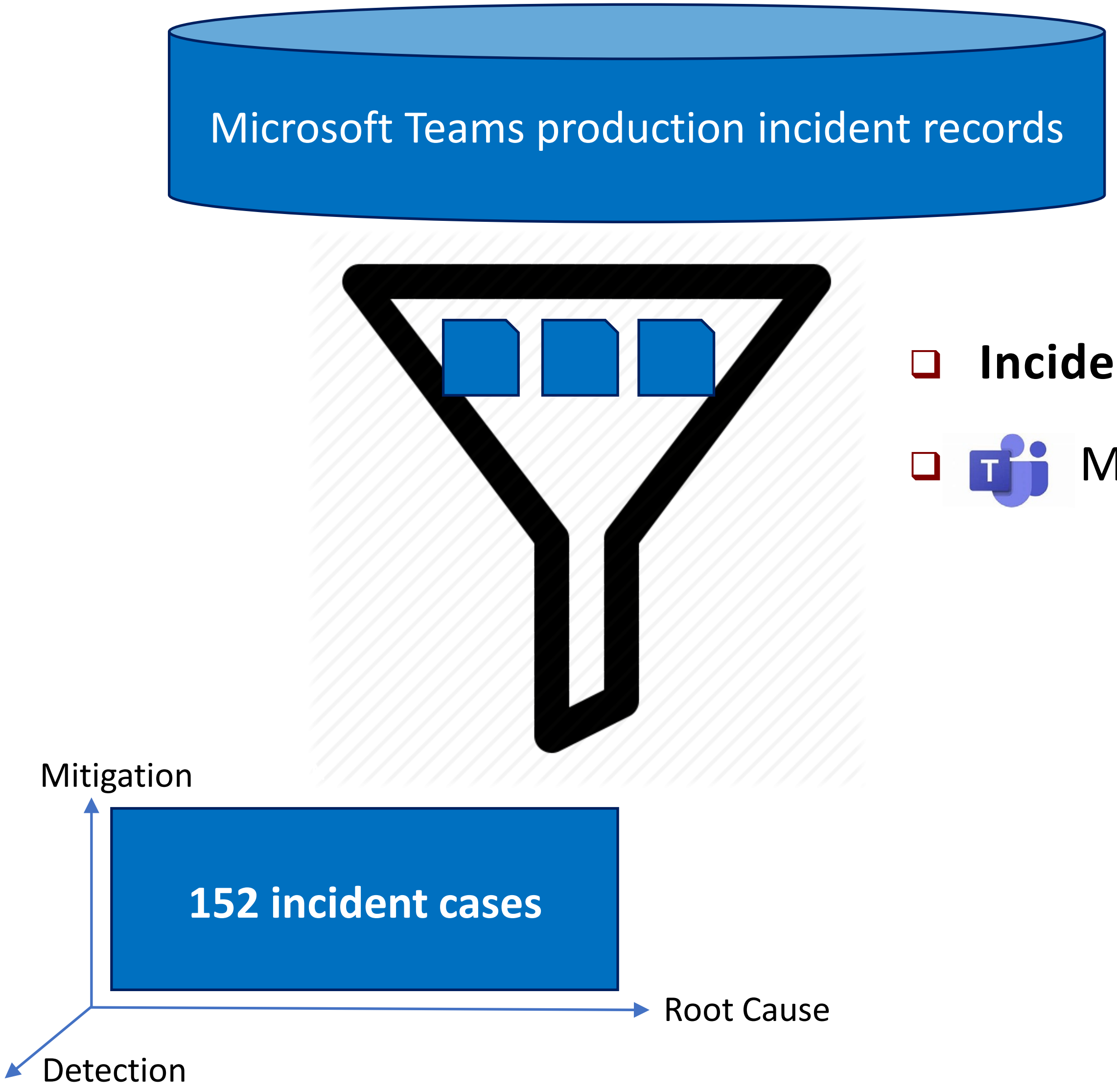
Research Questions



Questions We Aim to Address

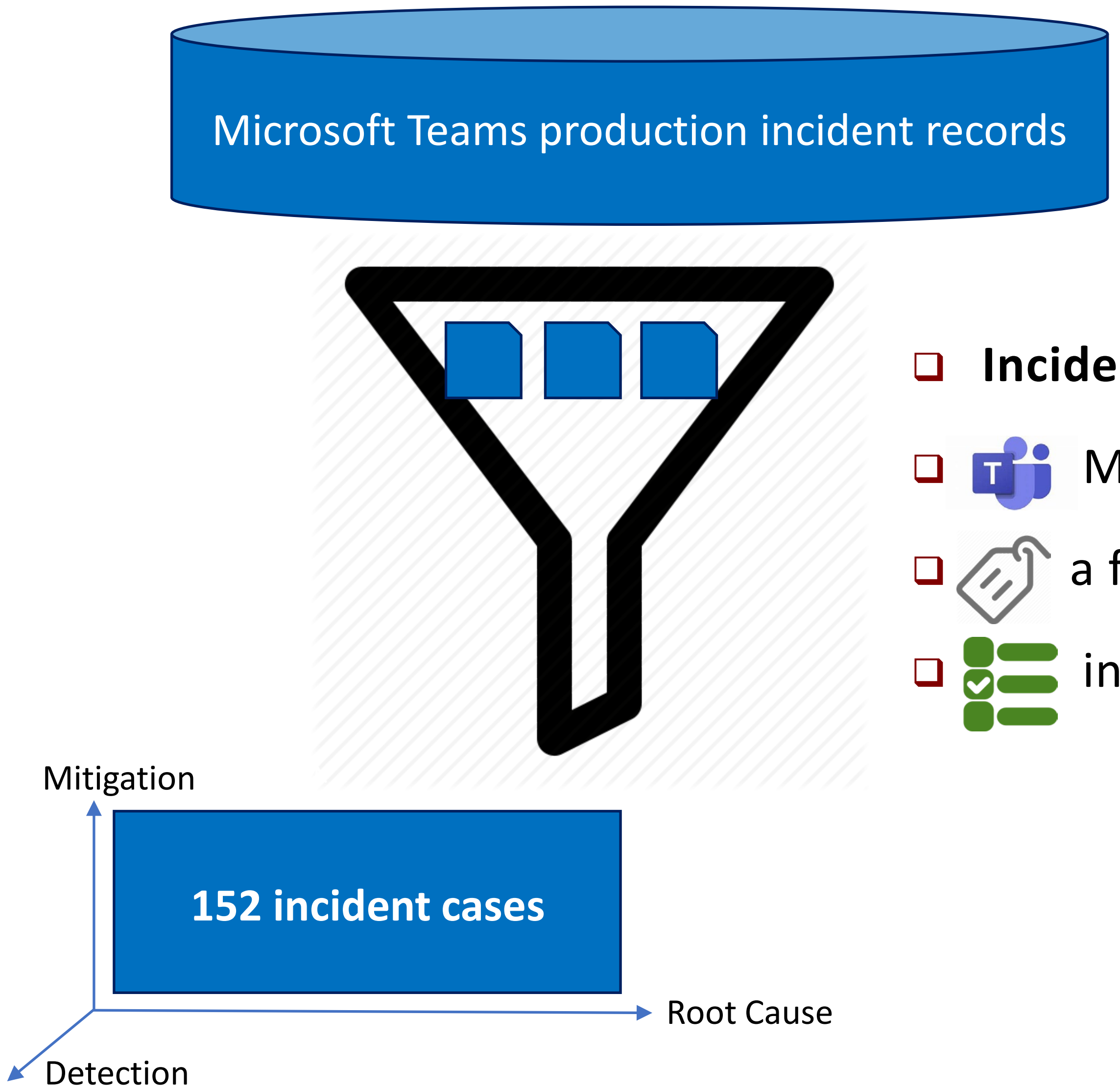
1. *Why the incidents occurred and how they were resolved?*
2. *What the gaps were in current processes which caused delayed response?*
3. *What automation could help make the services resilient?*

Methodology and Dataset



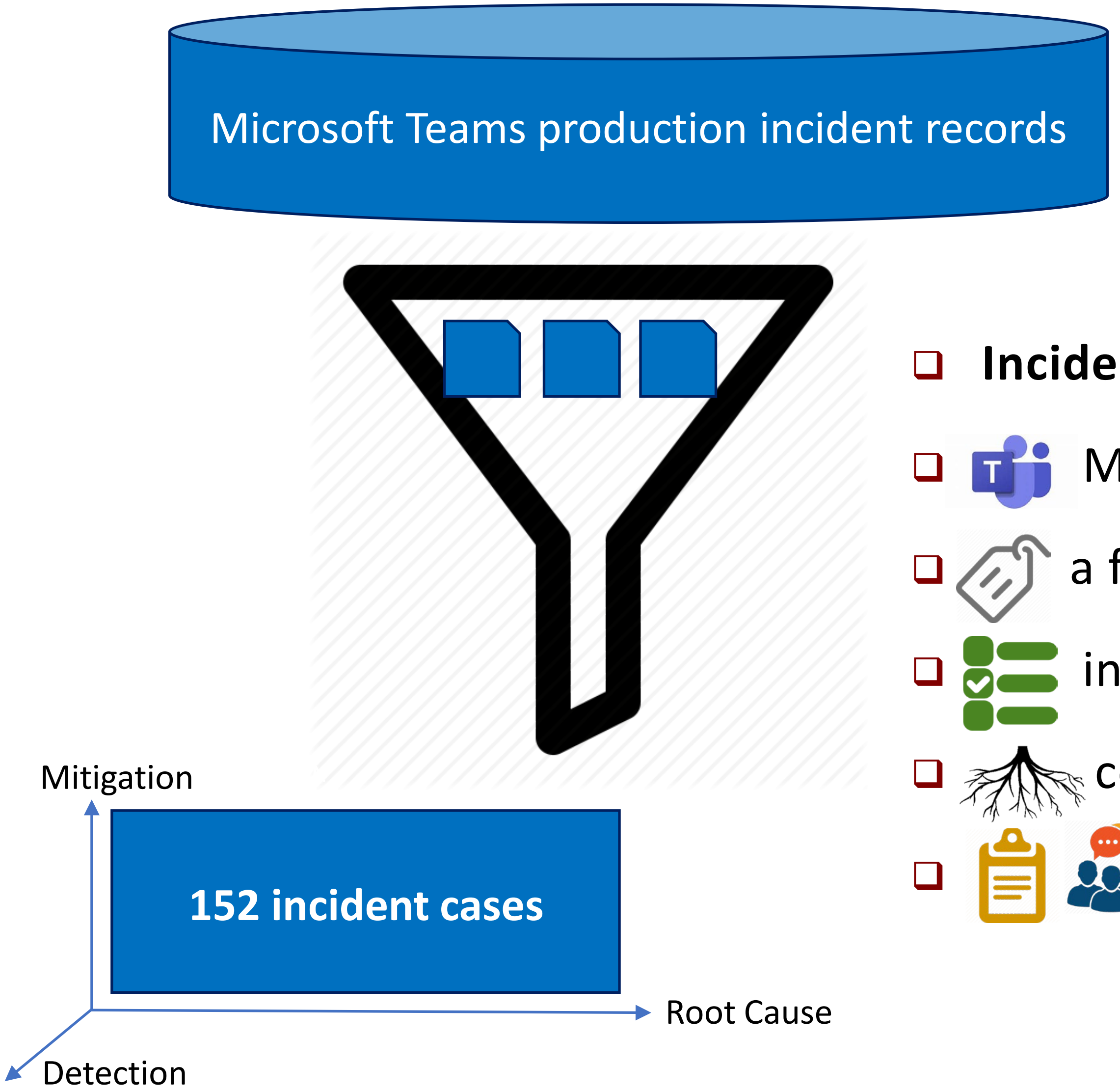
- **Incidents** from one year period (05/15/2021 to 05/15/2022)
- Microsoft Teams service

Methodology and Dataset



- **Incidents** from one year period (05/15/2021 to 05/15/2022)
- Microsoft Teams service
- a feature-blocker or outage incident (high severity)
- incident has been resolved/mitigated

Methodology and Dataset



- **Incidents** from one year period (05/15/2021 to 05/15/2022)
- Microsoft Teams service
- a feature-blocker or outage incident (high severity)
- incident has been resolved/mitigated
- contains detailed root cause information
- postmortem contains mitigation and discussion

Categorization Strategy

- Dataset split: taxonomy (60 incidents); validation (30 incidents); test set (62 incidents)
- For each of the 6 dimensions
  - Populate summarized text from incident summary and post-mortem reports.
  - Individually labels categories on taxonomy set
  - Identify common taxonomy via discussion

- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Categorization Strategy

- Dataset split: taxonomy (60 incidents); validation (30 incidents); test set (62 incidents)
- For each of the 6 dimensions
  - Populate summarized text from incident summary and post-mortem reports.
  - Individually labels categories on taxonomy set
  - Identify common taxonomy via discussion
  - Individually labels categories on validation set.
  - Finalize taxonomy set via discussion

- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Categorization Strategy

- ❑ Dataset split: taxonomy (60 incidents); validation (30 incidents); test set (62 incidents)
- ❑ For each of the 6 dimensions
  - ❑ Populate summarized text from incident summary and post-mortem reports.
  - ❑ Individually labels categories on taxonomy set
  - ❑ Identify common taxonomy via discussion
  - ❑ Individually labels categories on validation set.
  - ❑ Finalize taxonomy set via discussion
  - ❑ Individually labels categories on test data set
  - ❑ Use **Kohen's kappa** to compute inter-annotator agreement scores (1 is optimal).

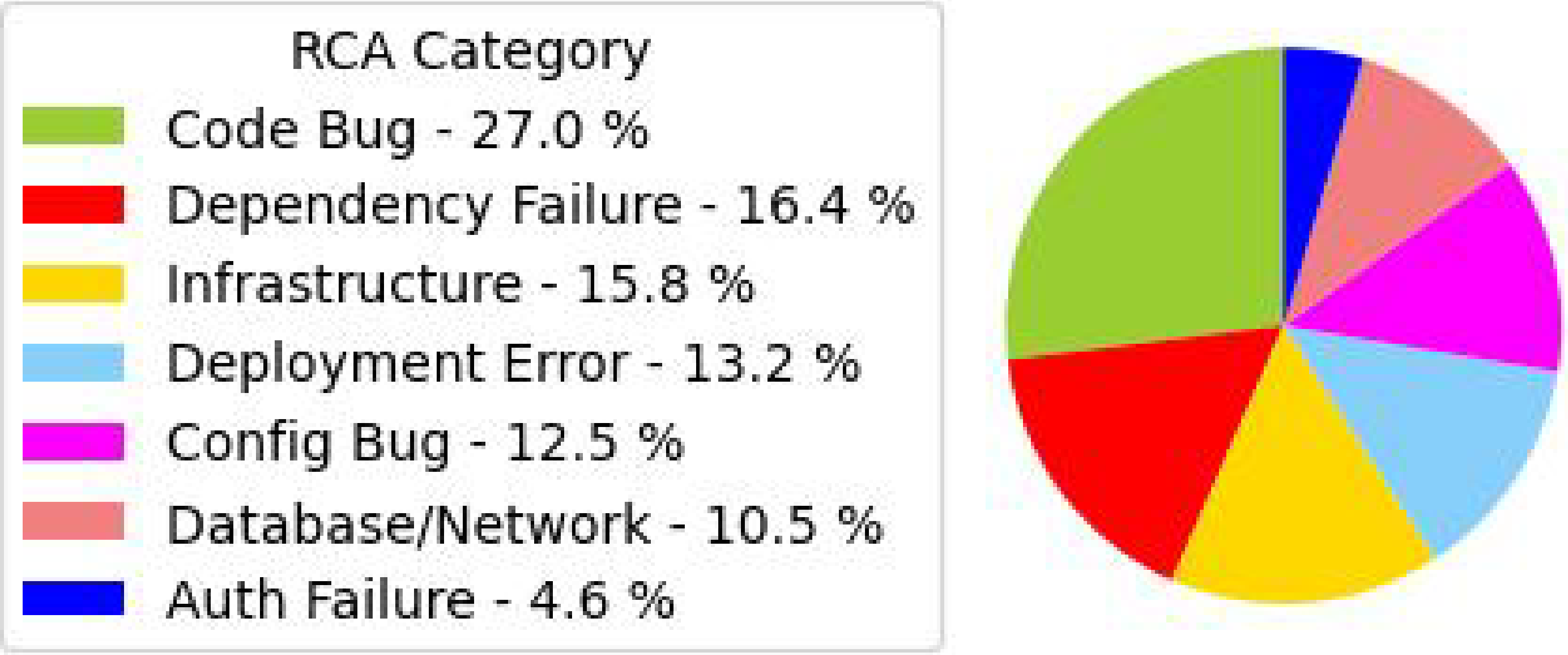
- Root causes **(0.94)**
- Mitigation steps **(0.95)**
- Detection failures **(0.88)**
- Mitigation failures **(0.94)**
- Lessons learnt by OCEs **(0.94)**
- Automation opportunities **(0.98)**

Outline

- Motivation
- Methodology and dataset.
- Root causes and mitigations
- Detection and mitigation failures
- Automation opportunities
- Multi-dimensional correlation
- Summary and future works

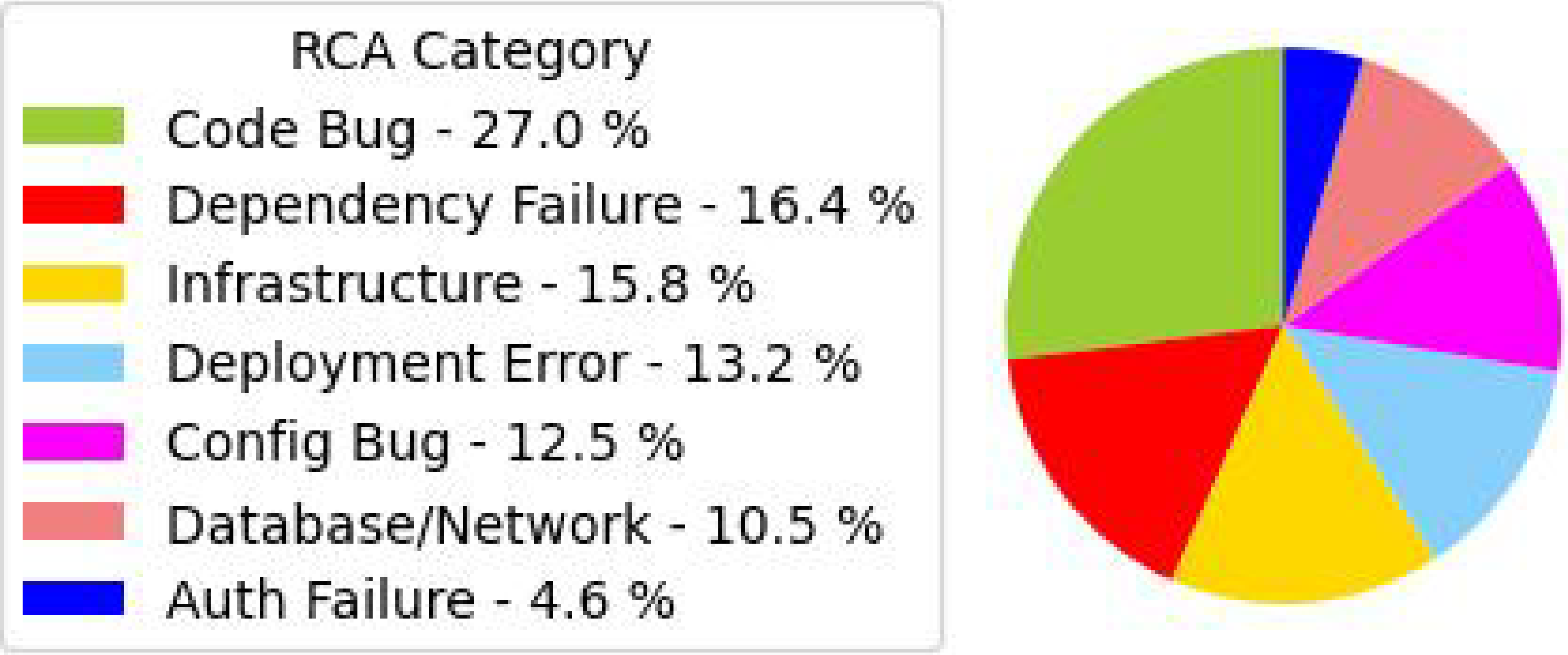
- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Insights from Root Causes



Observation: Majority of incidents (60%) were caused due to non-code/non-config related issues in infrastructure, deployment, and service dependencies.

Insights from Root Causes

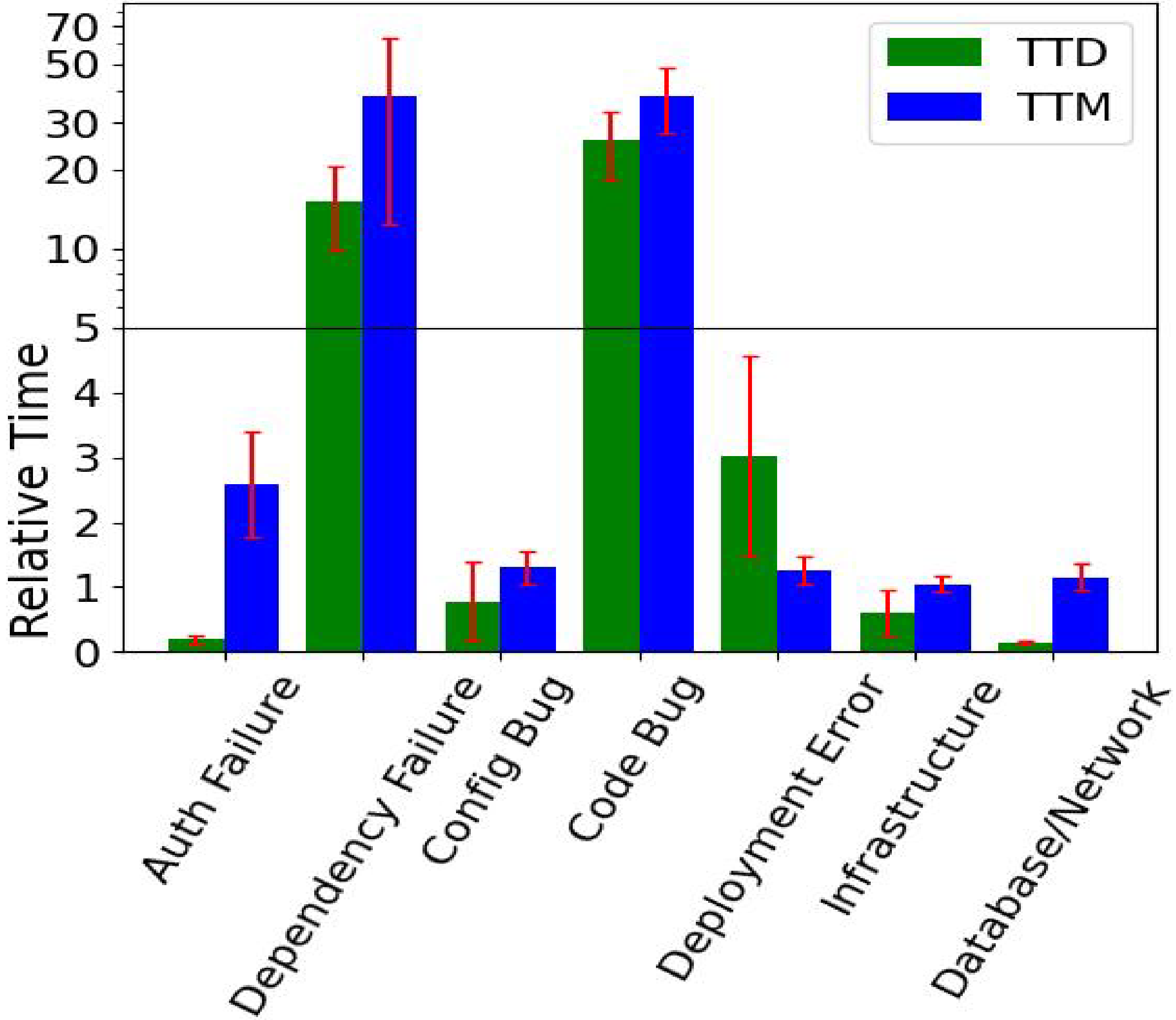


Observation: Majority of incidents (60%) were caused due to non-code/non-config related issues in infrastructure, deployment, and service dependencies.

Implication: Effective techniques need to developed for reliable infra management and safe deployment.

TTD and TTM for Different Root Causes

Observation: The time to detect and mitigate code bugs and dependency failures is significantly higher than other root causes.

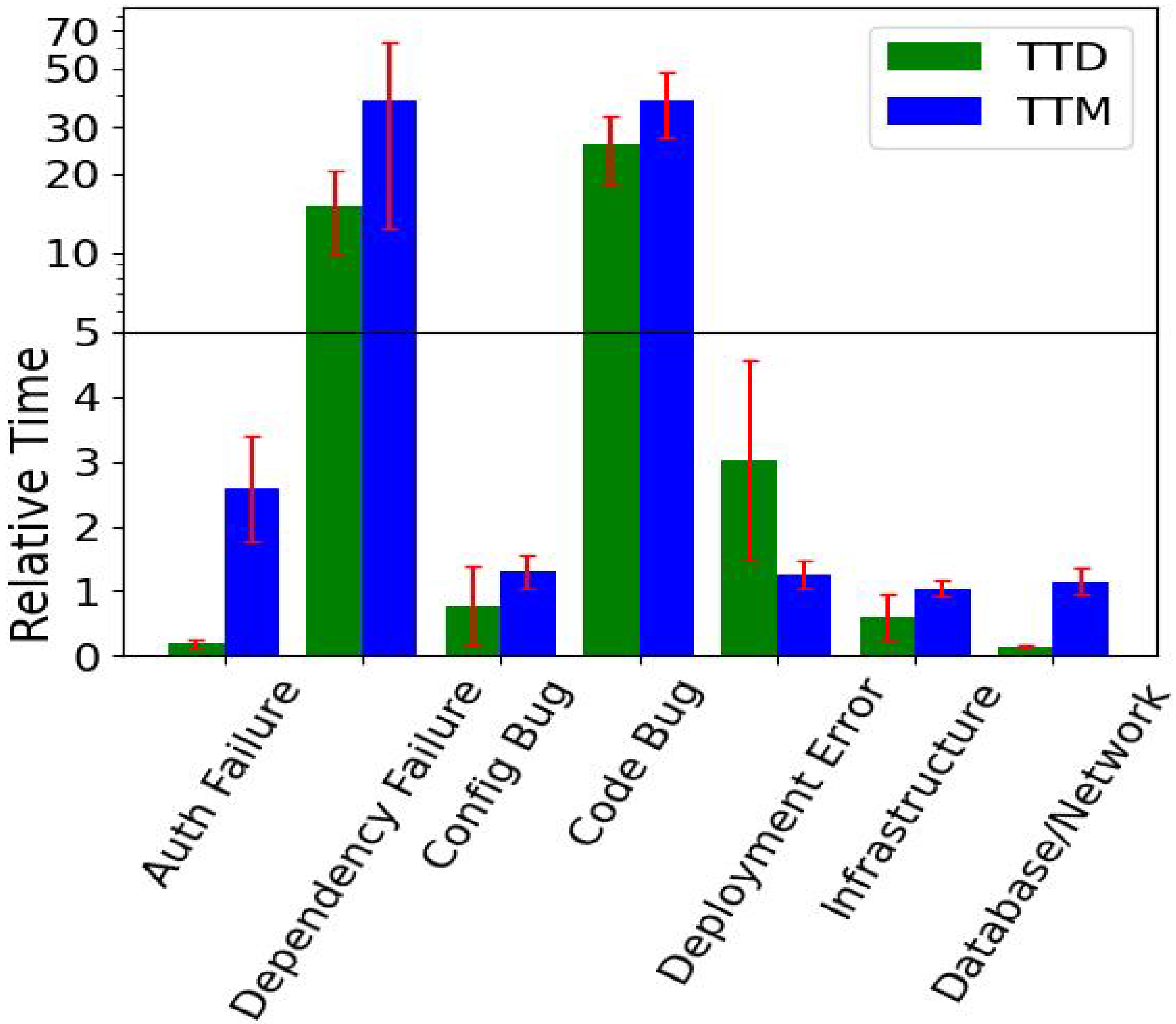


Y-axis shows the normalized time, with the median of time to detect or mitigate of all incidents as 1.

TTD and TTM for Different Root Causes

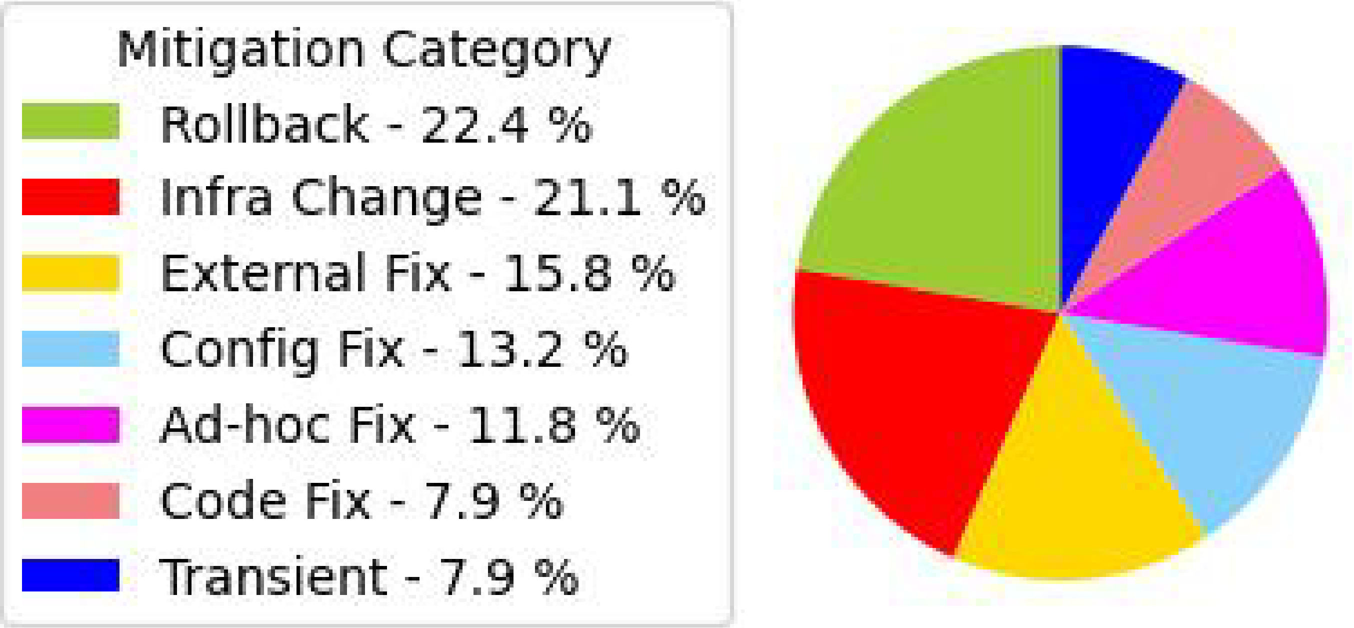
Observation: The time to detect and mitigate code bugs and dependency failures is significantly higher than other root causes.

Implication: We need better observability tool across partner services for better coverages.



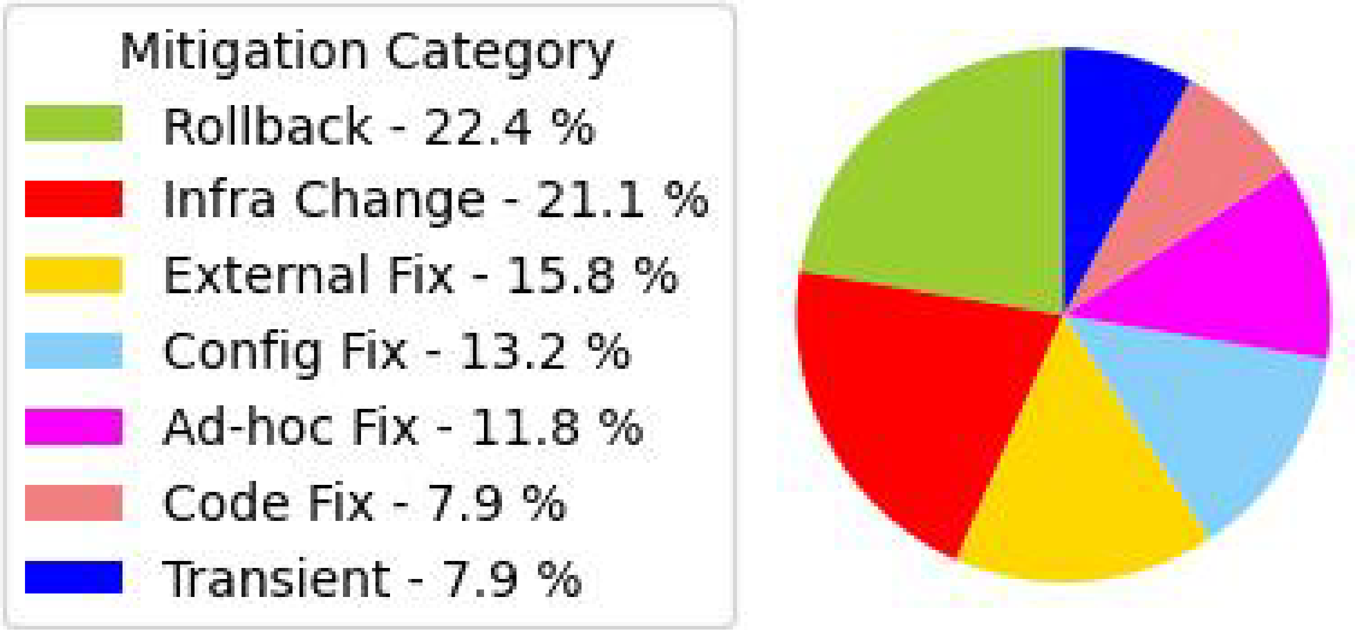
Y-axis shows the normalized time, with the median of time to detect or mitigate of all incidents as 1.

Insights from Mitigation Steps



Observation: Among the 40% incidents that were caused by code/configuration bugs, nearly 80% of incidents were mitigated *without* a code or configuration fix.

Insights from Mitigation Steps

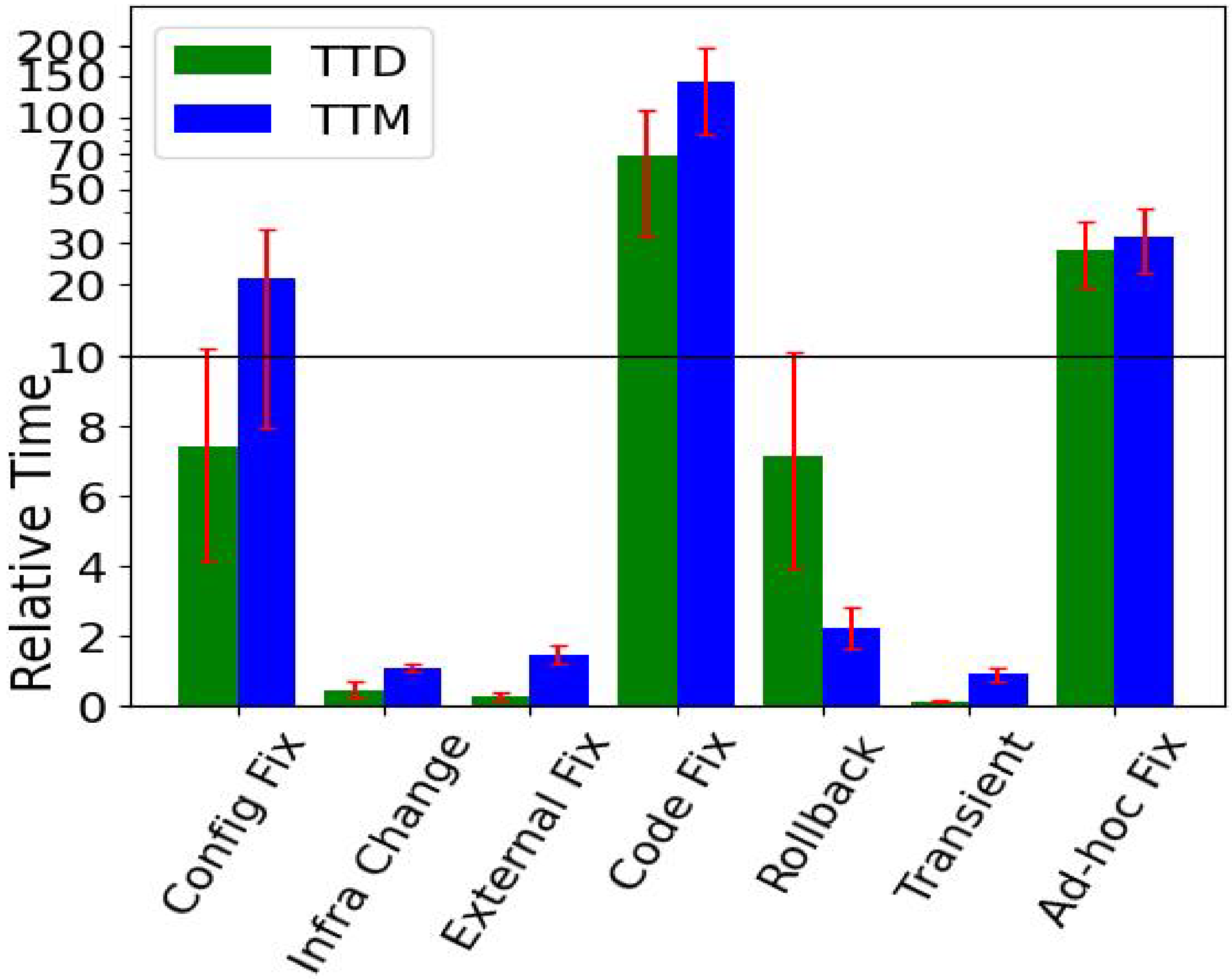


Observation: Among the 40% incidents that were caused by code/configuration bugs, nearly 80% of incidents were mitigated *without* a code or configuration fix.

Implication: We need more effective automation such as auto scaling and auto traffic failover that can mitigate 40% of code/config bugs.

TTD and TTM for Different Mitigation Steps

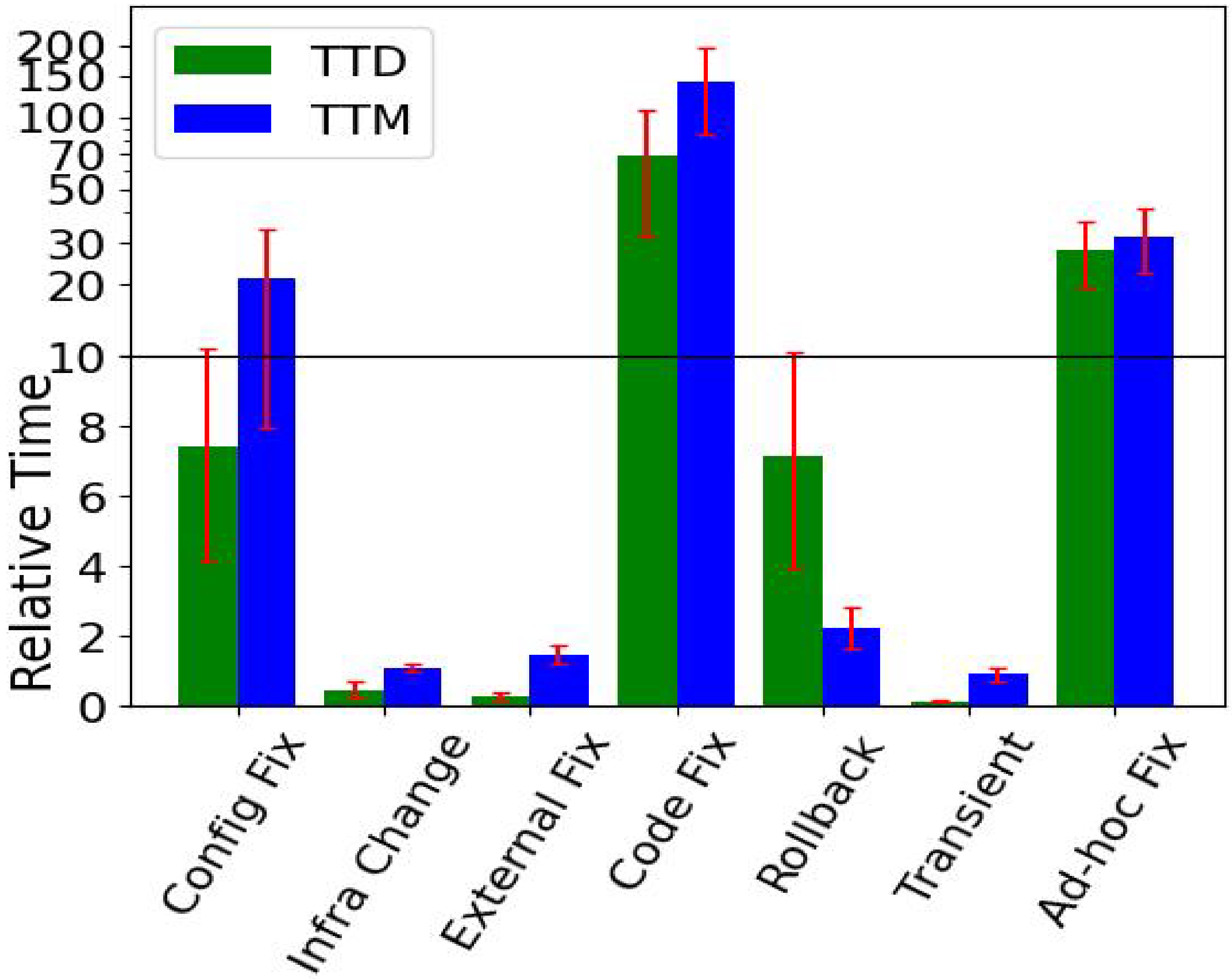
Observation: 30% of the mitigation delay is caused due to manual mitigation steps



TTD and TTM for Different Mitigation Steps

Observation: 30% of the mitigation delay is caused due to manual mitigation steps

Implication: We need automation tools to reduce human involvement.

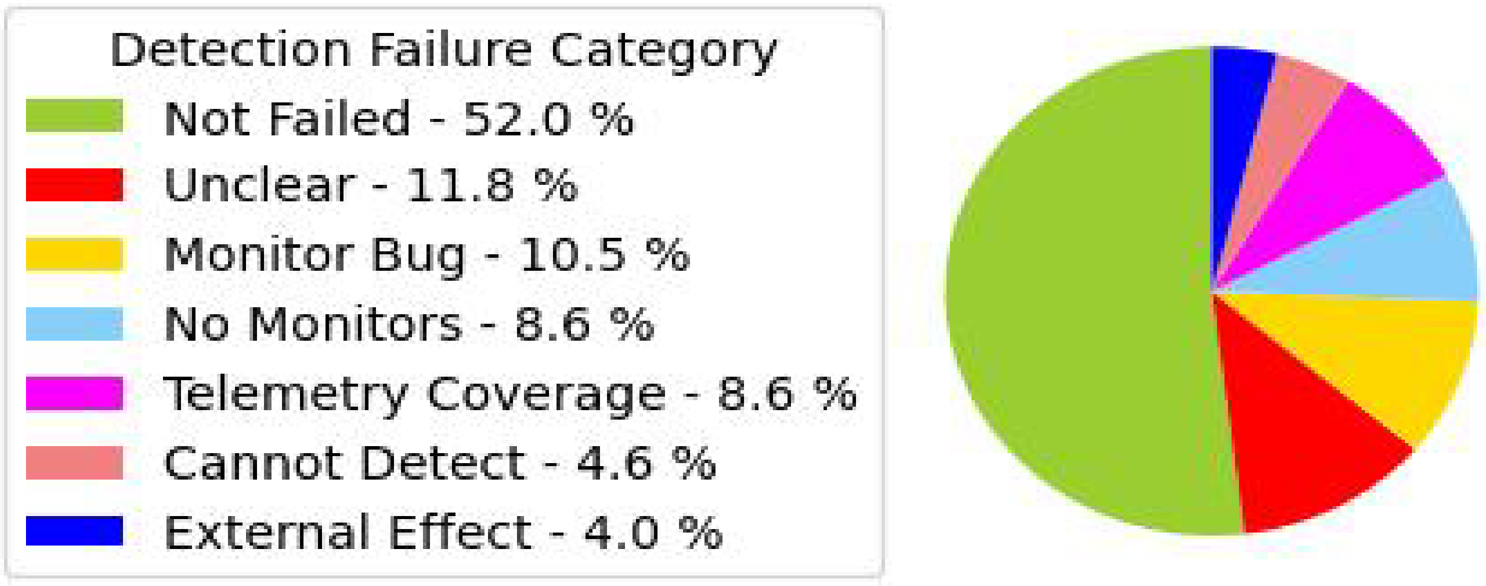


Outline

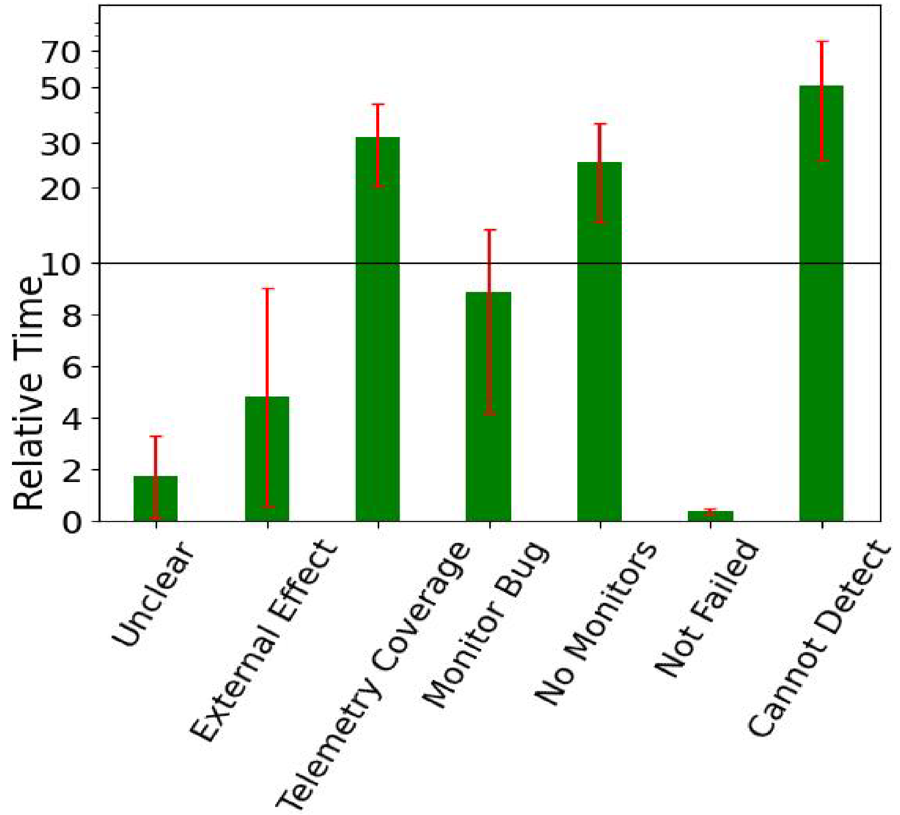
- ❑ Motivation
- ❑ Methodology and dataset.
- ❑ Root causes and mitigations
- ❑ Detection and mitigation failures
- ❑ Automation opportunities
- ❑ Multi-dimensional correlation
- ❑ Summary and future works

- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Insights from Detection Failures

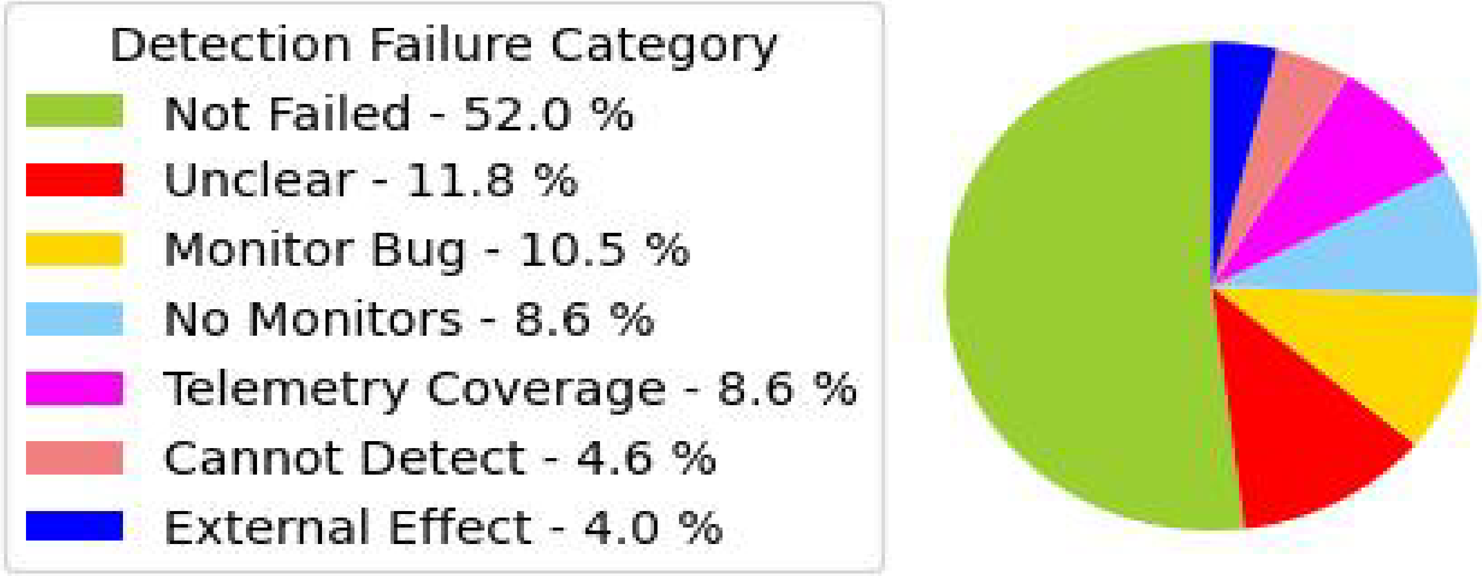


TTD for different detection failures

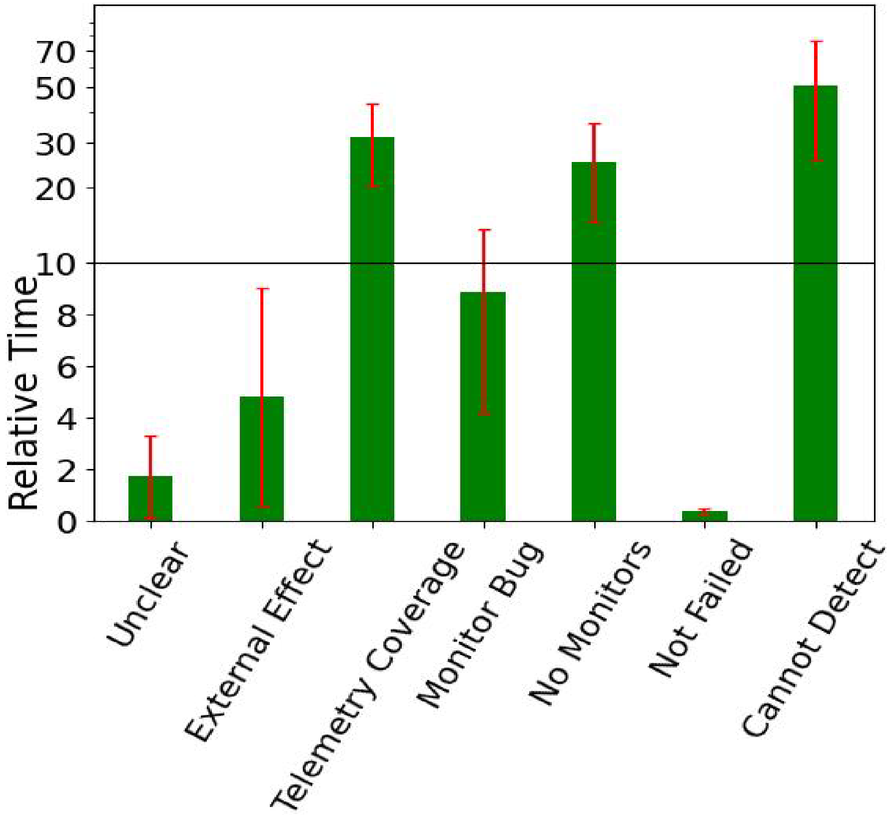


Observation: ≈17% of incidents either **lacked monitors or telemetry coverage**. 10% incidents were not detected **due to bugs**, e.g., high threshold, buggy feature, wrong configuration, etc.

Insights from Detection Failures



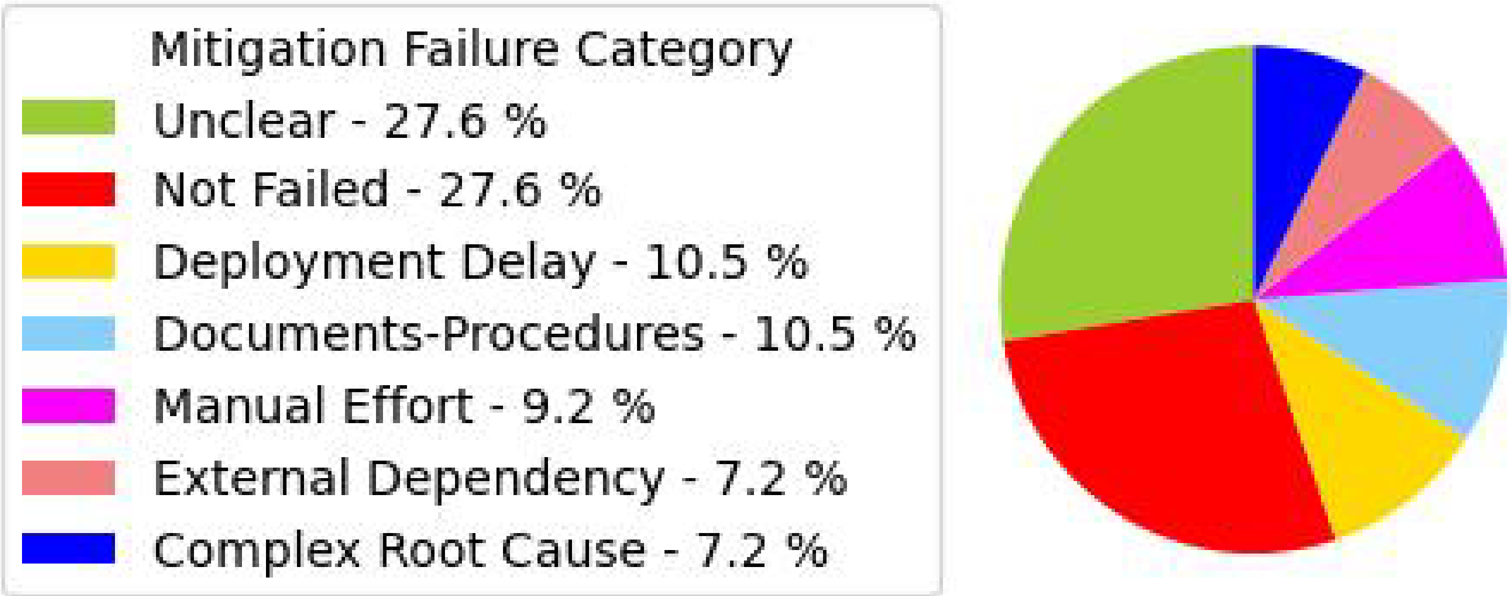
TTD for different detection failures

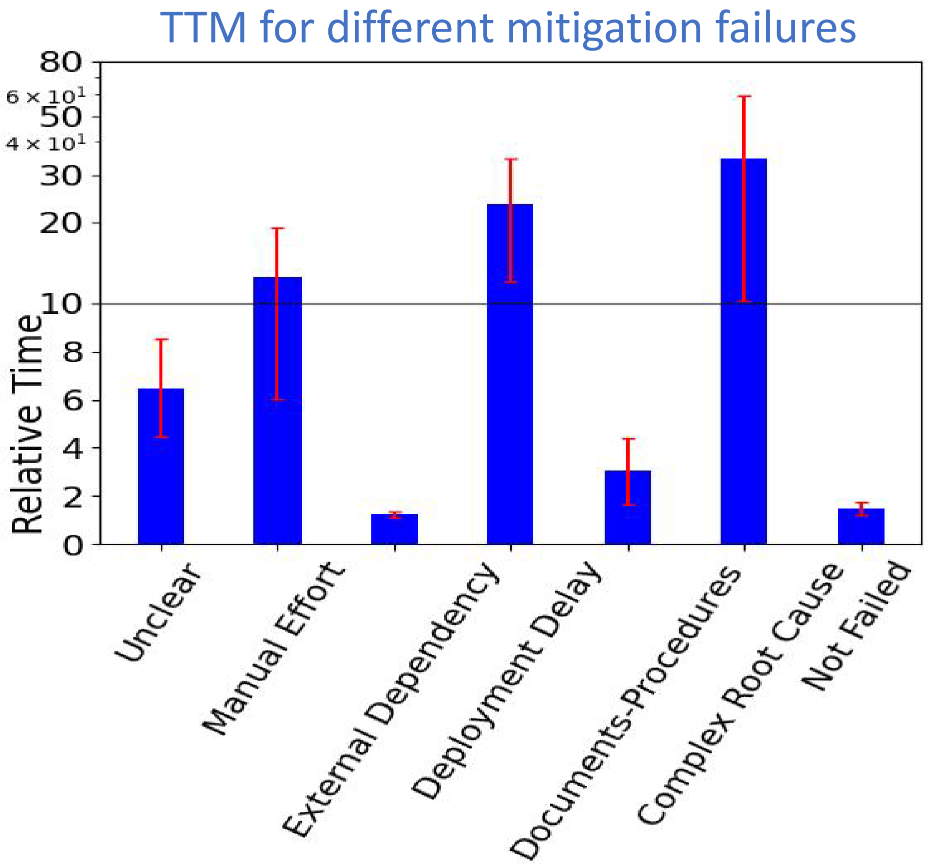


Observation: ≈17% of incidents either **lacked monitors or telemetry coverage**. 10% incidents were not detected **due to bugs**, e.g., high threshold, buggy feature, wrong configuration, etc.

Implication: New watchdogs need be setup with dynamic thresholding mechanism.

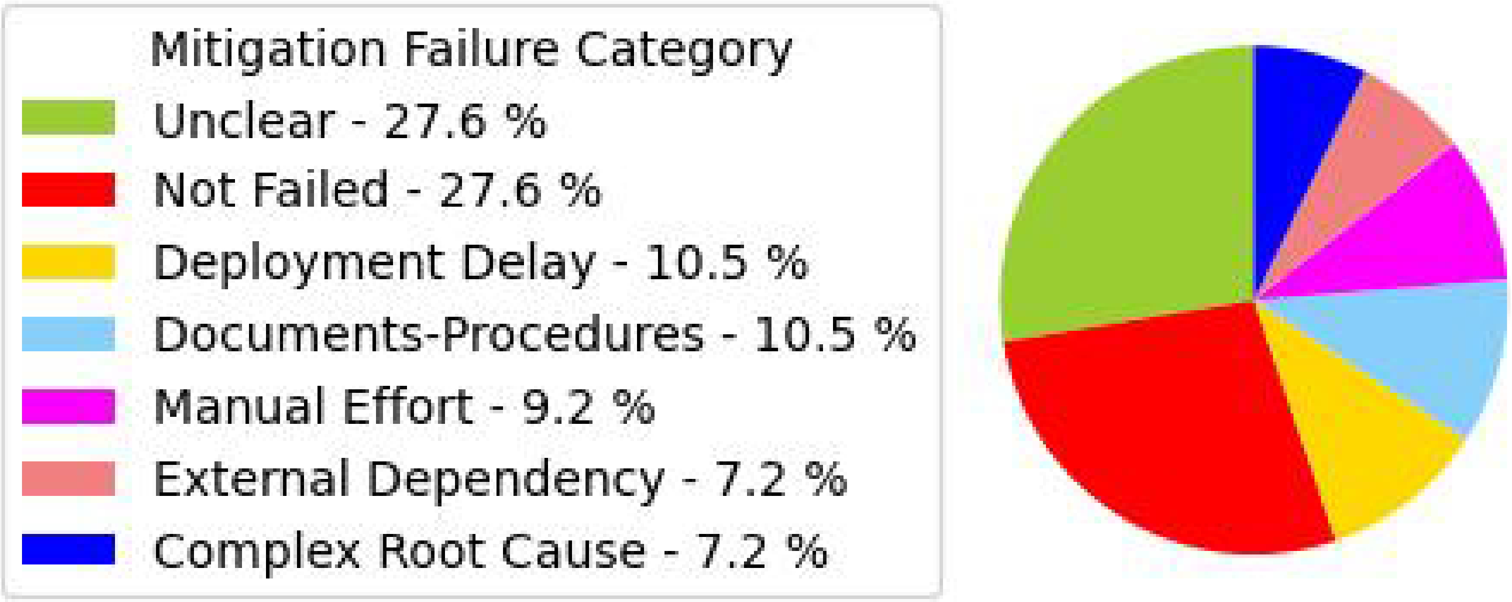
Insights from Mitigation Failures

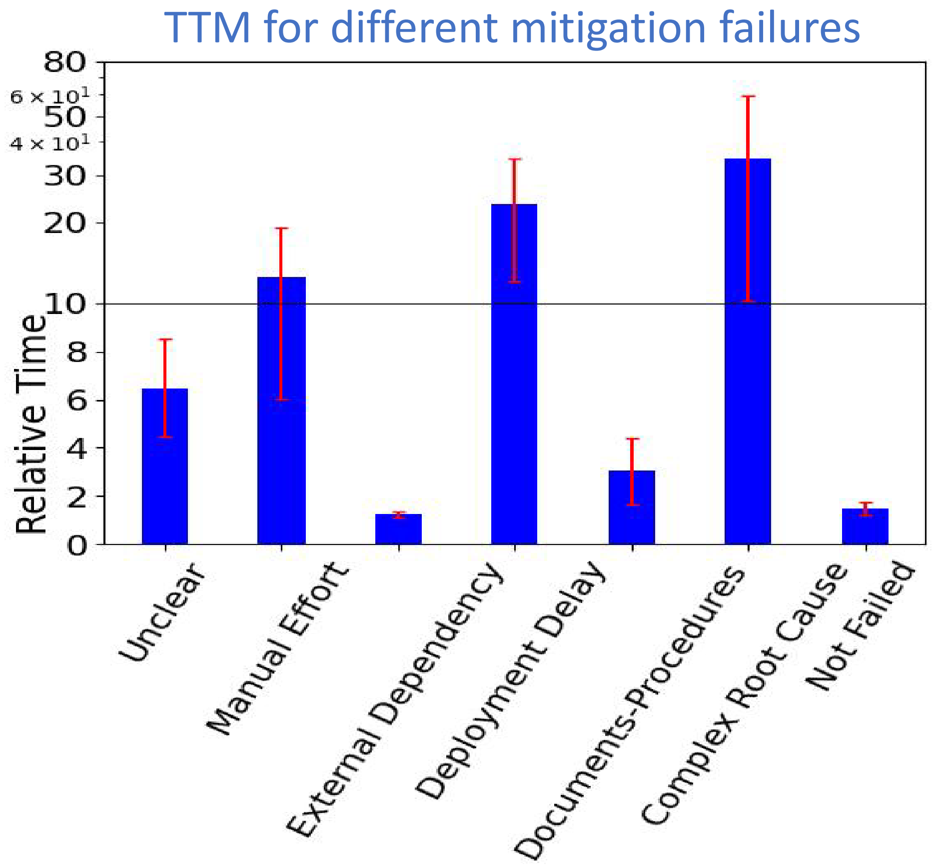




Observation: While 7% mitigation delays are due to complex root causes, 27% of incidents had mitigation delays due to **manual efforts, external dependency and deployment issues.**

Insights from Mitigation Failures





Observation: While 7% mitigation delays are due to complex root causes, 27% of incidents had mitigation delays due to **manual efforts, external dependency and deployment issues.**

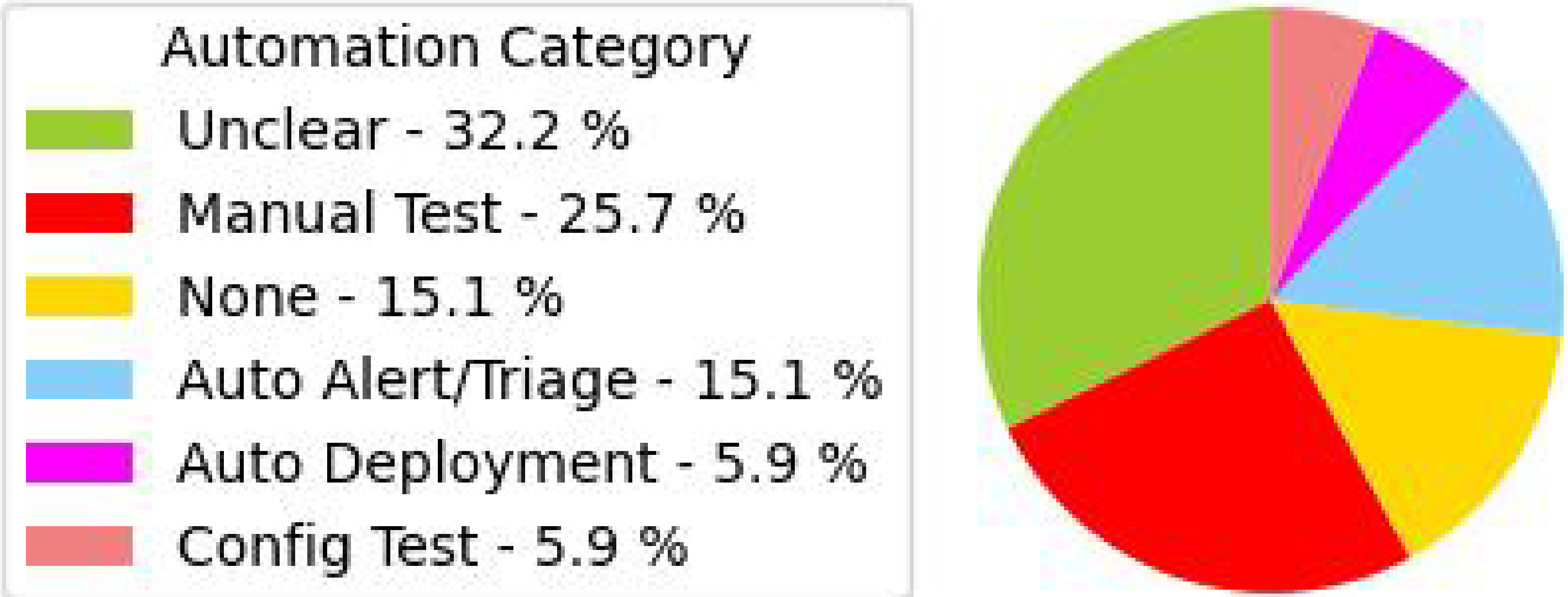
Implication: Reducing human intervention through automation can significantly reduce mitigation delay.

Outline

- Motivation
- Methodology and dataset.
- Root causes and mitigations
- Detection and mitigation failures
- Automation opportunities
- Multi-dimensional correlation
- Summary and future works

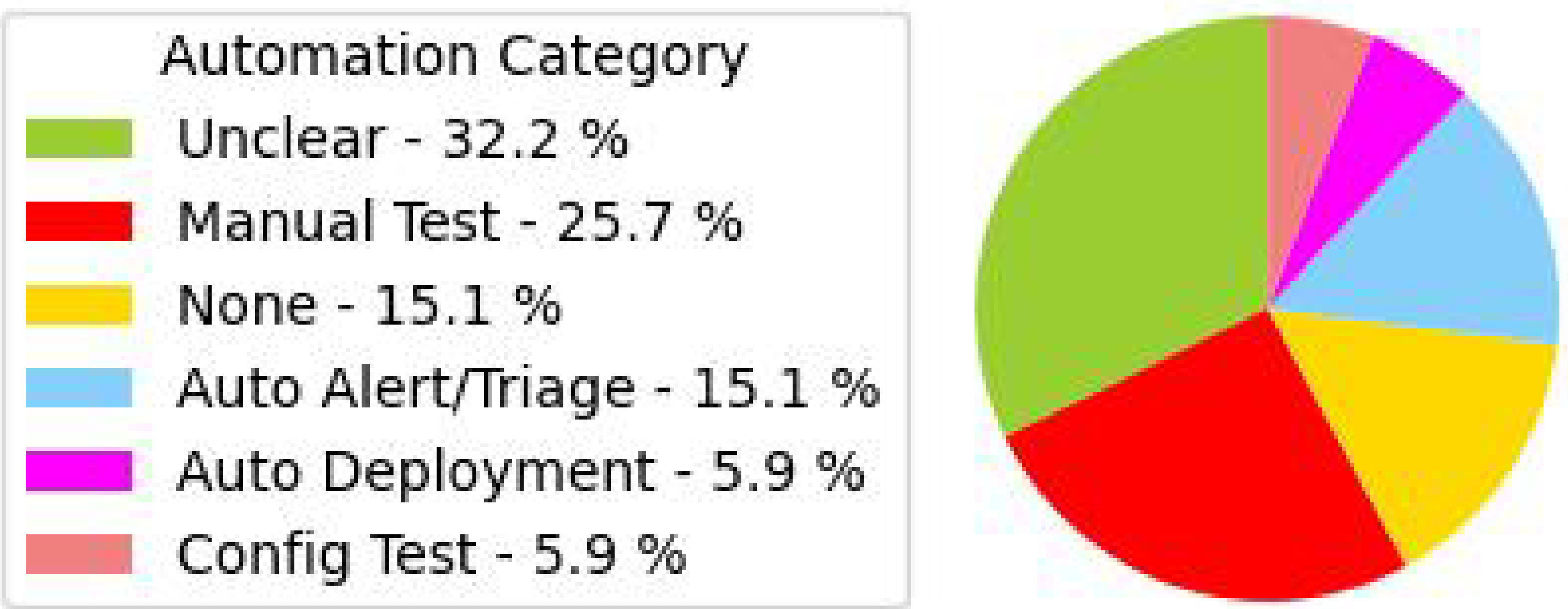
- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Insights from Automation Suggestions by OCEs



Observation: Improving testing was a popular choice for automation opportunities, over monitoring.

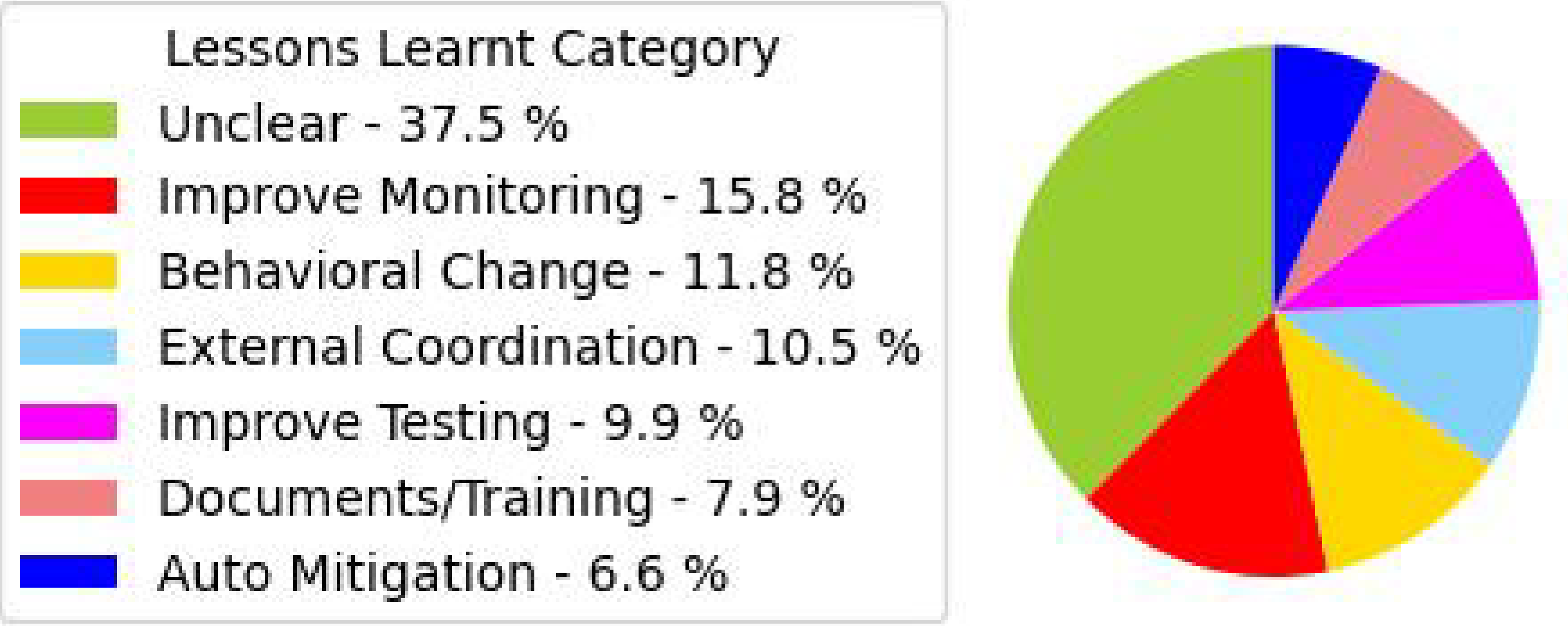
Insights from Automation Suggestions by OCEs



Observation: Improving testing was a popular choice for automation opportunities, over monitoring.

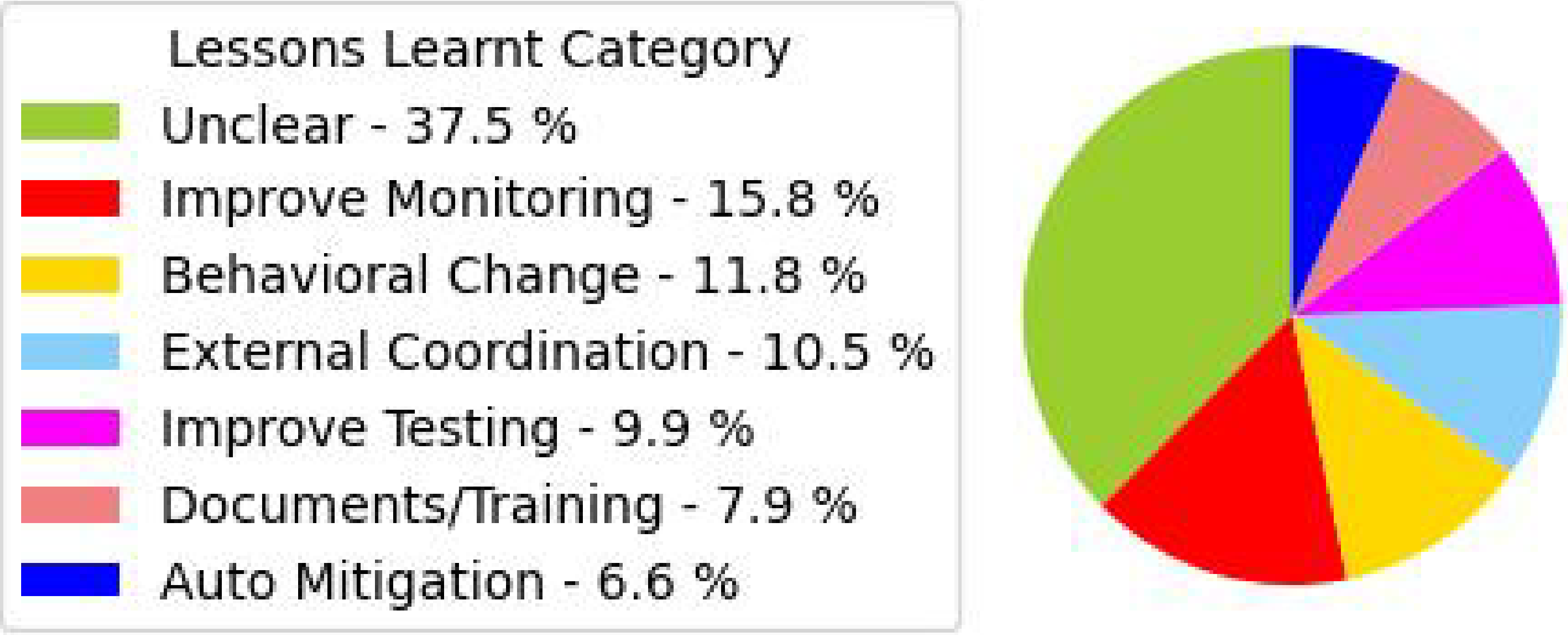
Implication: We need to reduce incidents by identifying issues before they reach production services through automated testing.

Insights from Lessons Learnt by OCEs



Observation: While improving monitoring/testing accounts for majority of the lessons learnt, a significant ≈20% feedback indicated problems with existing documentations.

Insights from Lessons Learnt by OCEs



Observation: While improving monitoring/testing accounts for majority of the lessons learnt, a significant ≈20% feedback indicated problems with existing documentations.

Implication: We need better documentations, training, and practices for better incident management and service resiliency.

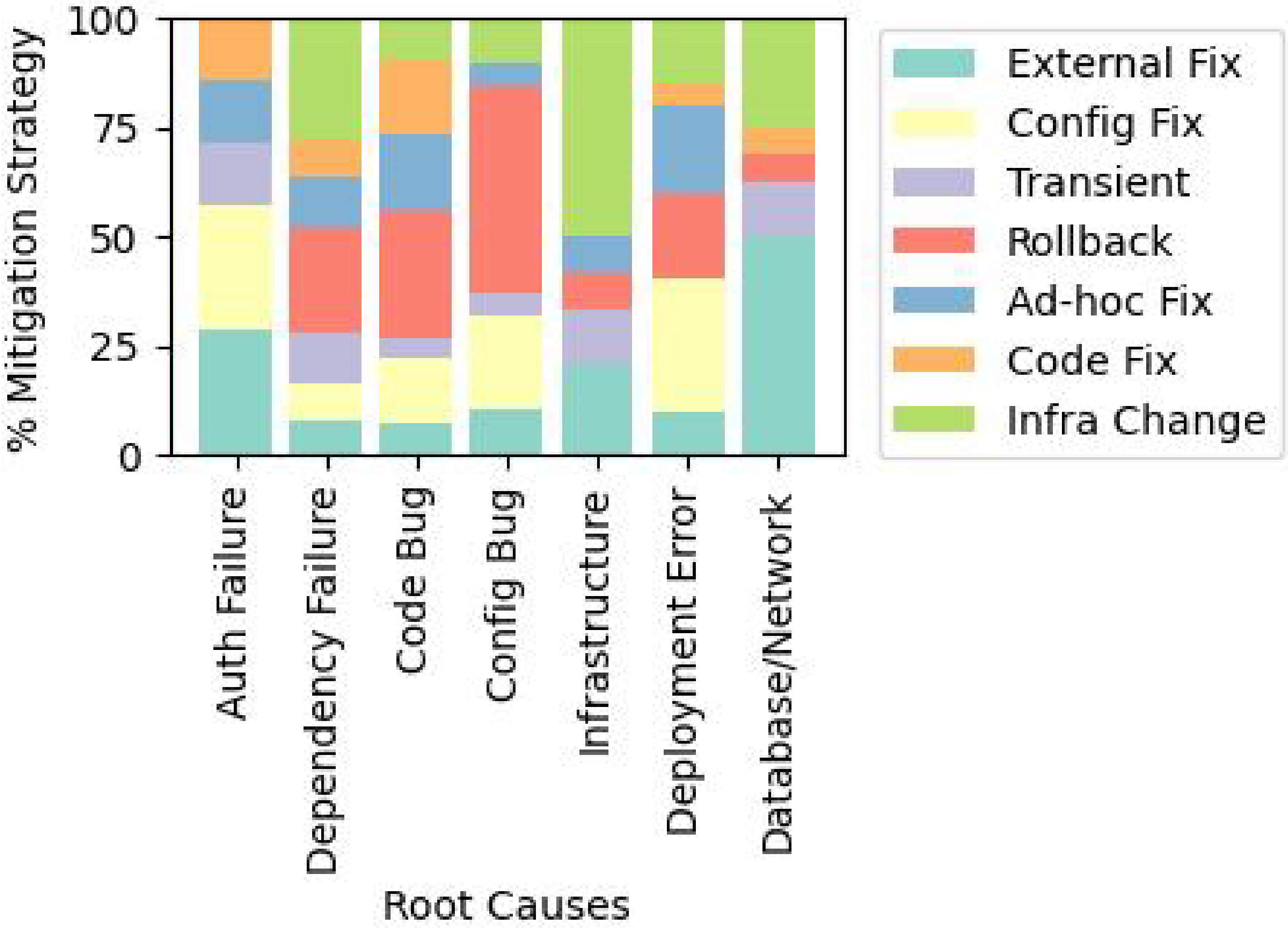
Outline

- ❑ Motivation
- ❑ Methodology and dataset.
- ❑ Root causes and mitigations
- ❑ Detection and mitigation failures
- ❑ Automation opportunities
- ❑ Multi-dimensional correlation
- ❑ Summary and future works

- Root causes
- Mitigation steps
- Detection failures
- Mitigation failures
- Lessons learnt by OCEs
- Automation opportunities

Insights from Root Cause vs. Mitigation Correlation

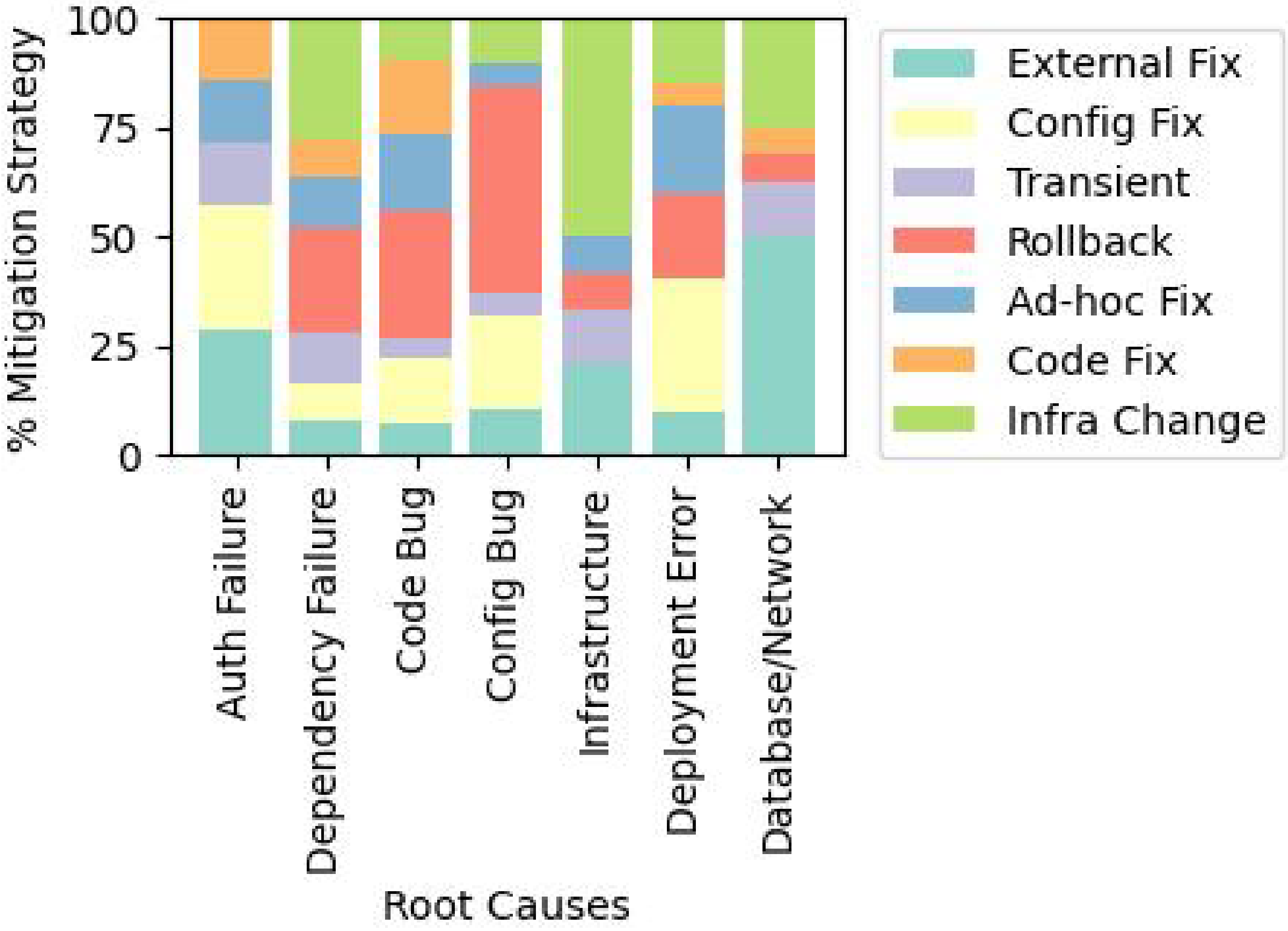
Observation: 47% of configuration bugs mitigated with a rollback compared to only 21% mitigated with a configuration fix, caused due to recent changes.



Insights from Root Cause vs. Mitigation Correlation

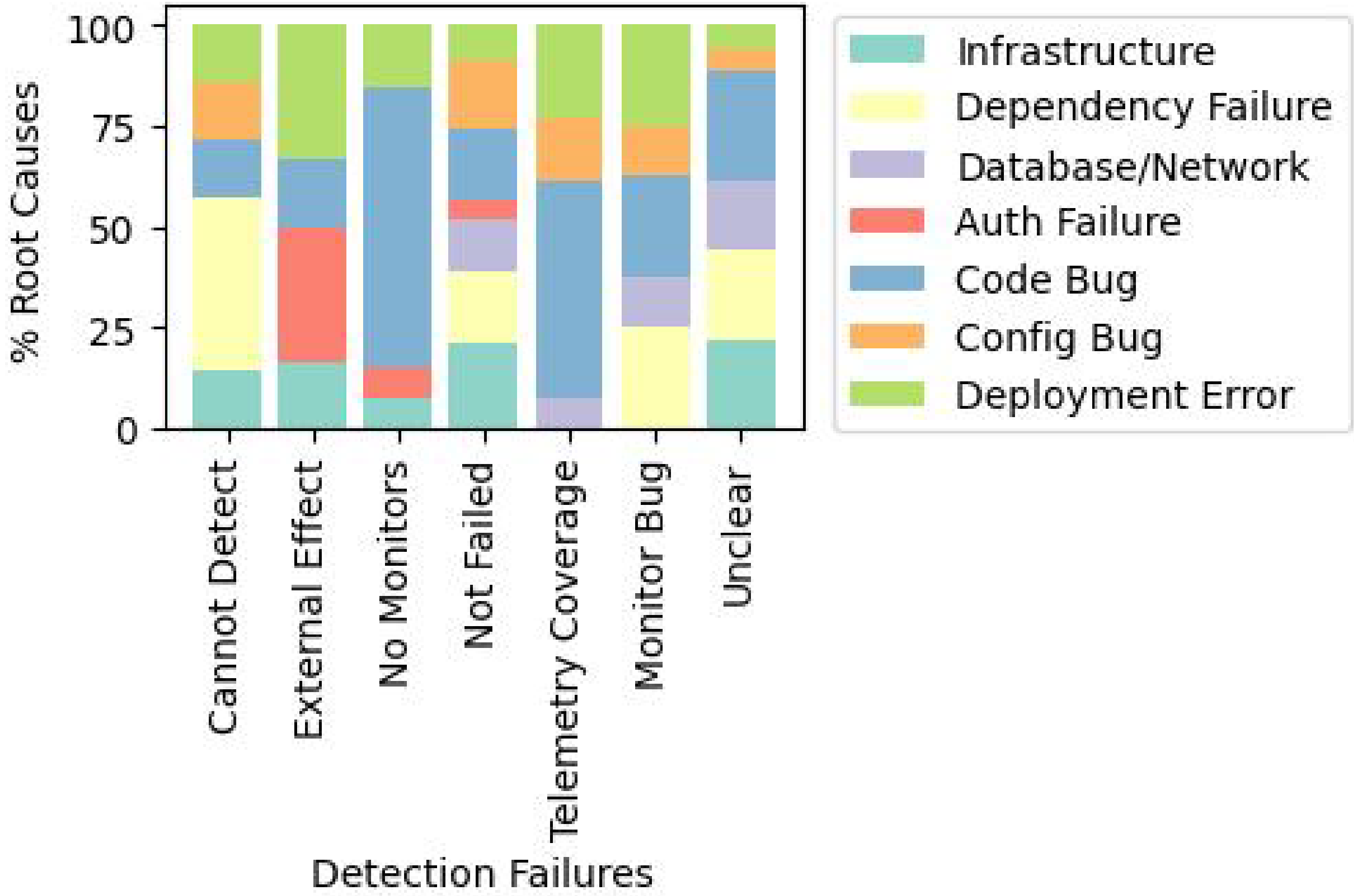
Observation: 47% of configuration bugs mitigated with a rollback compared to only 21% mitigated with a configuration fix, caused due to recent changes.

Implication: These configuration bugs can be identified proactively by rigorous configuration testing.



Insights from Root Cause vs. Detection Failure Correlation

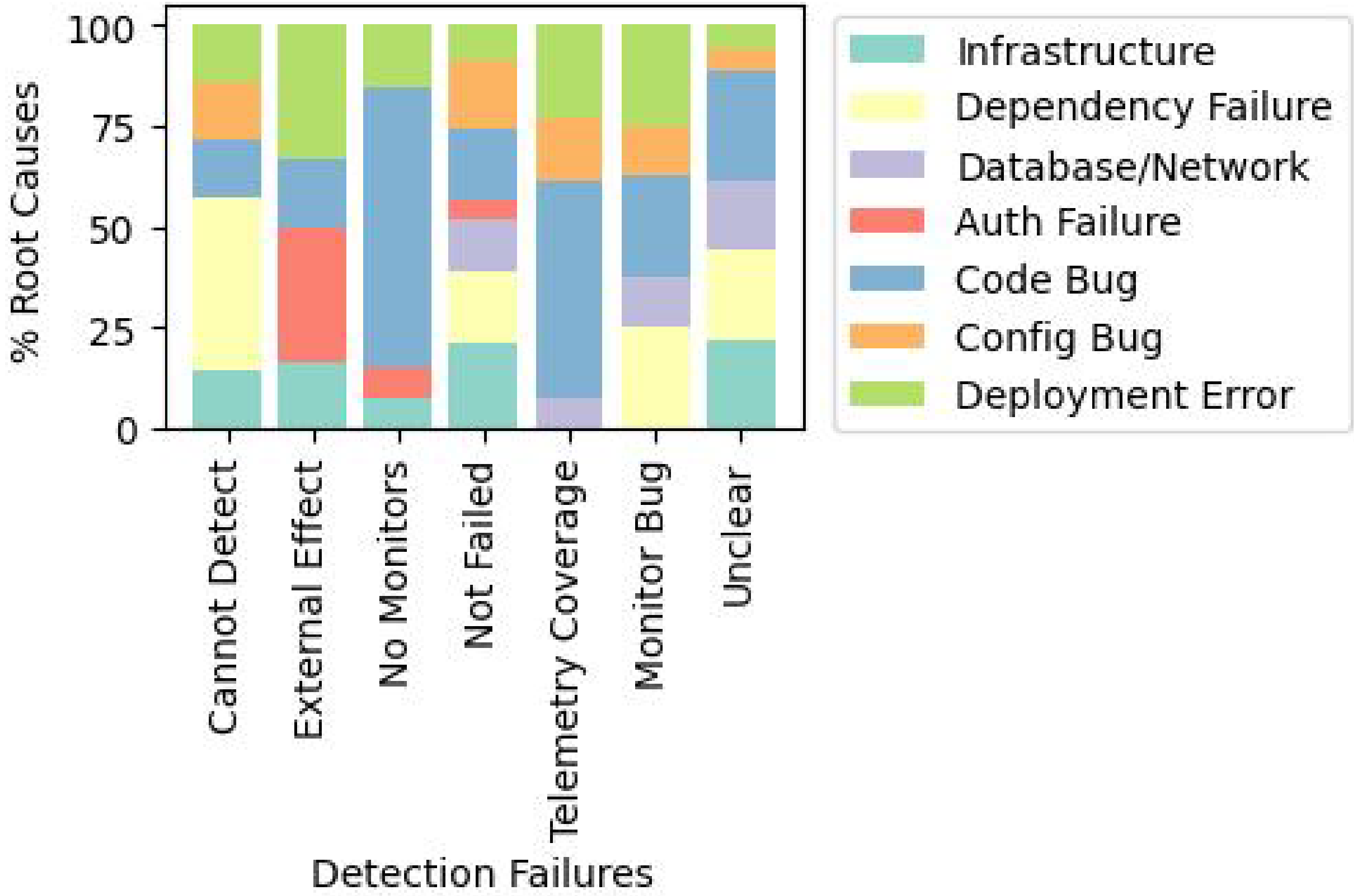
Observation: (1) 70% incident with code bugs does not have monitors. (2) 42% dependency failures are not detectable.



Insights from Root Cause vs. Detection Failure Correlation

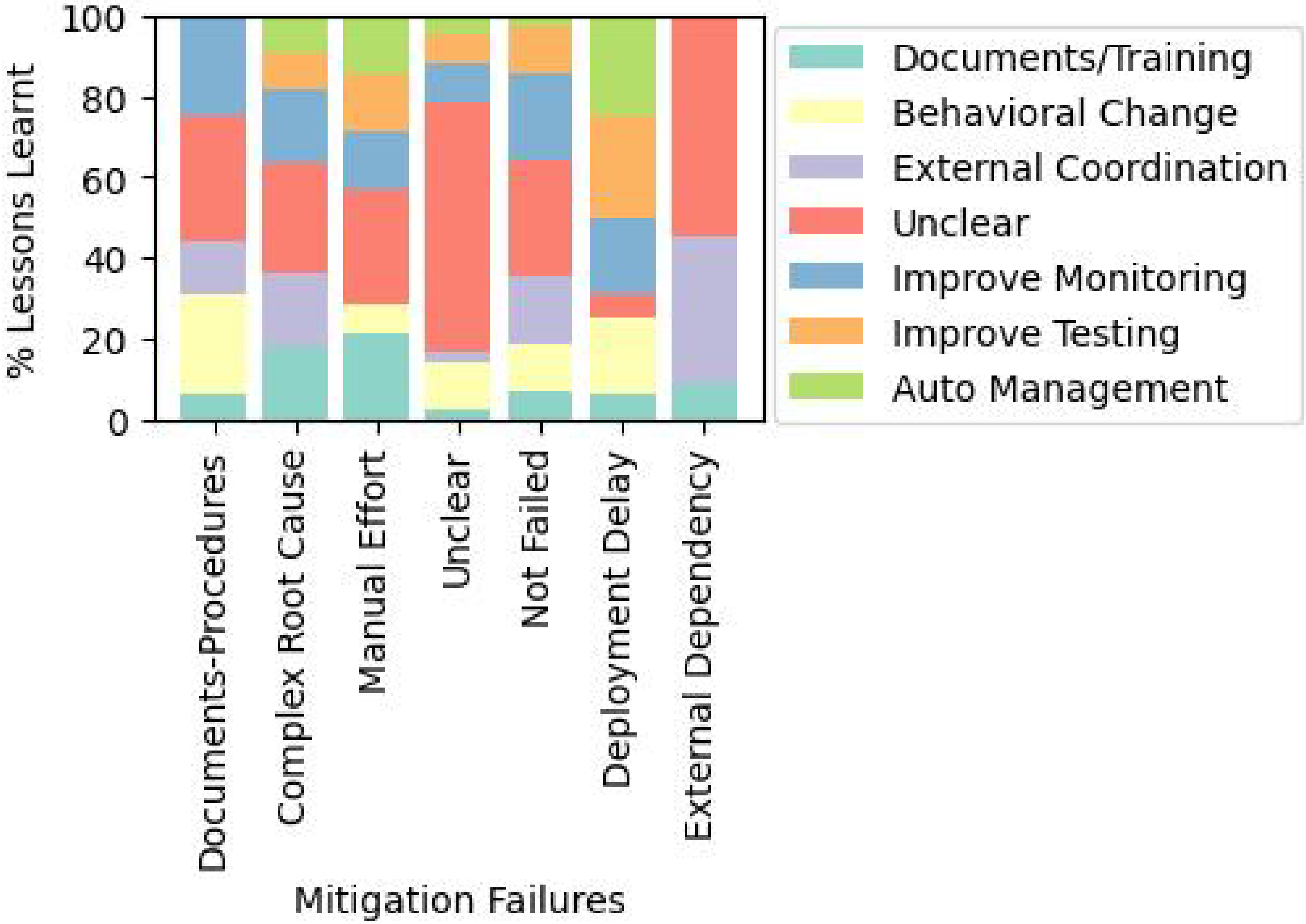
Observation: (1) 70% incident with code bugs does not have monitors.
(2) 42% dependency failures are not detectable.

Implication: (1) We need to invest in monitoring and staged rollout of code changes.
(2) Monitoring coverage needs to be increased across related partner services.



Insights from Mitigation Failure vs. Lessons Learnt Correlation

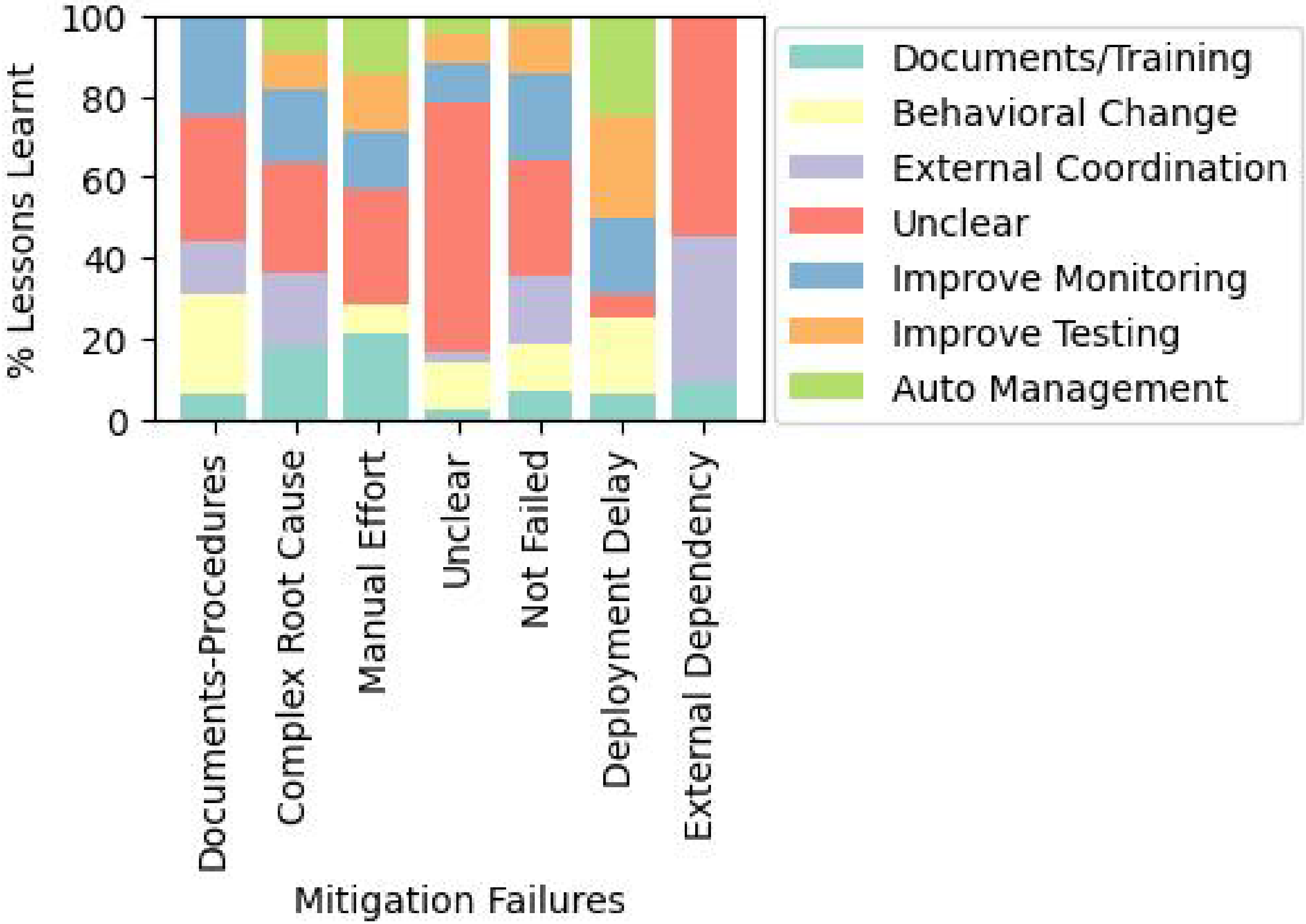
Observation: 21% of incidents where manual effort delayed mitigation, expected improvements in documentation and training.



Insights from Mitigation Failure vs. Lessons Learnt Correlation

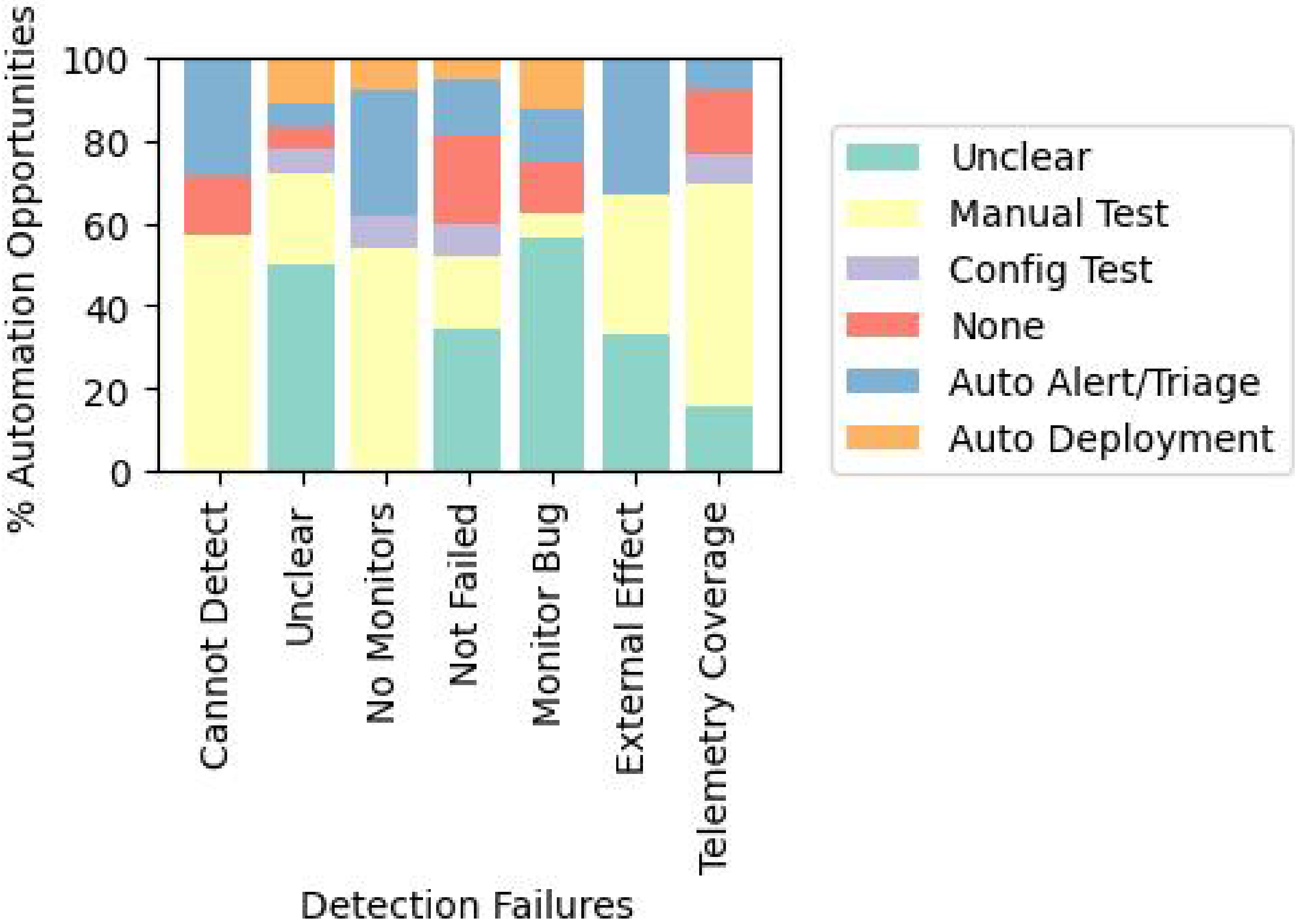
Observation: 21% of incidents where manual effort delayed mitigation, expected improvements in documentation and training.

Implication: Just like with source code, we need to design new metrics and methods to monitor documentation quality.



Insights from Automation vs. Detection Failure Correlation

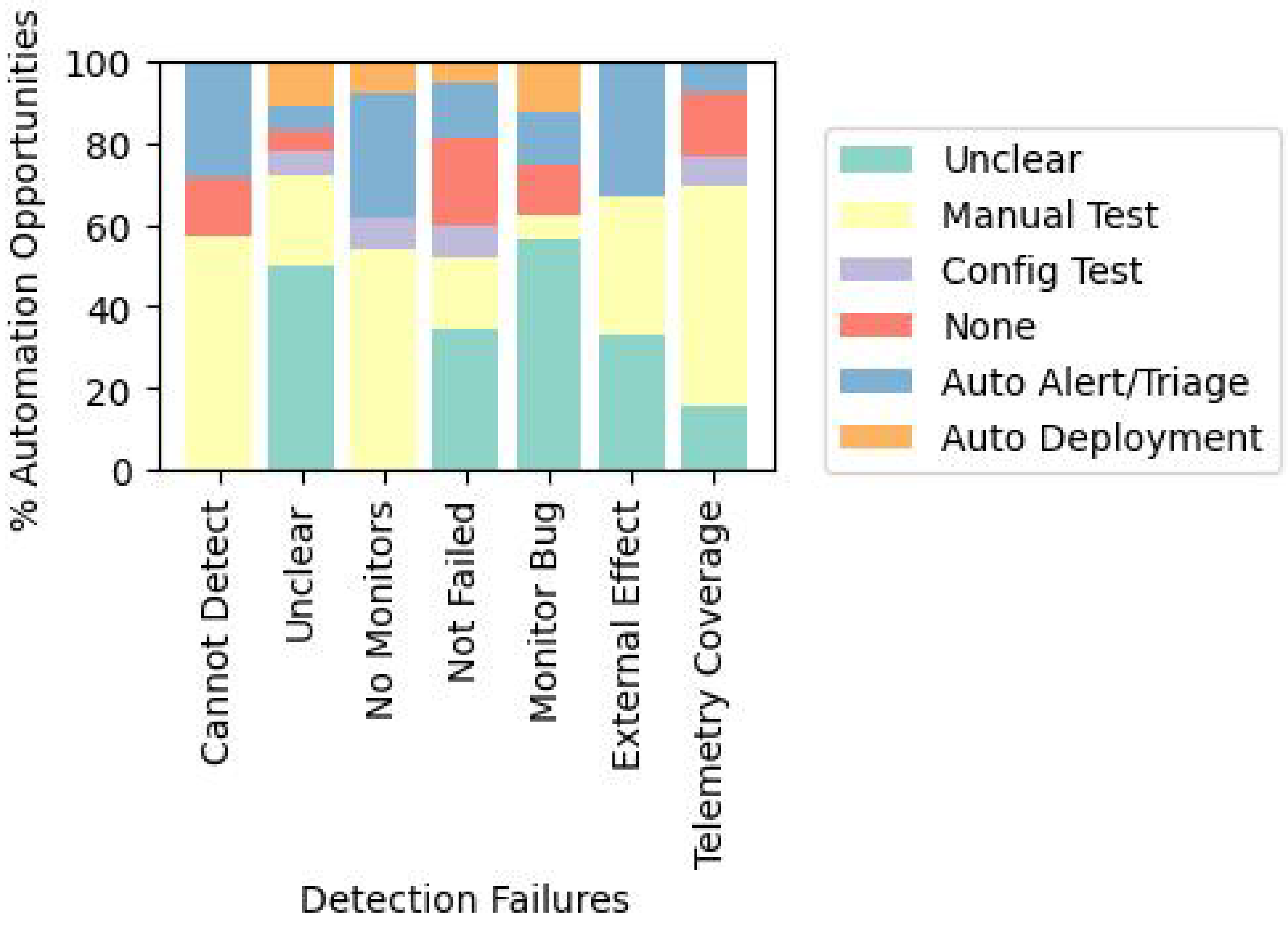
Observations: In more than 50% of incidents that monitors could not detect, OCEs expected an improvement in manual testing over automated alerts (23%) .



Insights from Automation vs. Detection Failure Correlation

Observations: In more than 50% of incidents that monitors could not detect, OCEs expected an improvement in manual testing over automated alerts (23%) .

Implication: Strongly enforcing a "Shift Left" practice with automated tools to aid testing.



Conclusion and Future Directions

Contributions and novelty:

- We analyzed 152 high-severity production incidents from Microsoft Teams to characterize the gaps and opportunities in different stages of the incident lifecycle.
- Our analysis spans both software and non-software related incidents.
- Our novel multi-dimensional correlation study uncovers important insights for improving service reliability.

Conclusion and Future Directions

Contributions and novelty:

- We analyzed 152 high-severity production incidents from Microsoft Teams to characterize the gaps and opportunities in different stages of the incident lifecycle.
- Our analysis spans both software and non-software related incidents.
- Our novel multi-dimensional correlation study uncovers important insights for improving service reliability.

Future Research Directions:

- **Safe deployment**
  - Invest more in proactive detection of code and config bugs by staged rollout of changes.
- **Improvement in monitoring**
  - Leveraging statistical multi-dimensional anomaly detection methods to tackle dynamic traffic.
- **Automation of mitigation steps**
  - Majority of mitigation steps (such as scaling up, failover) can be automated using ML methods.
- **Documentation quality**
  - Just like source code, we need to measure and improve the quality of documentations.