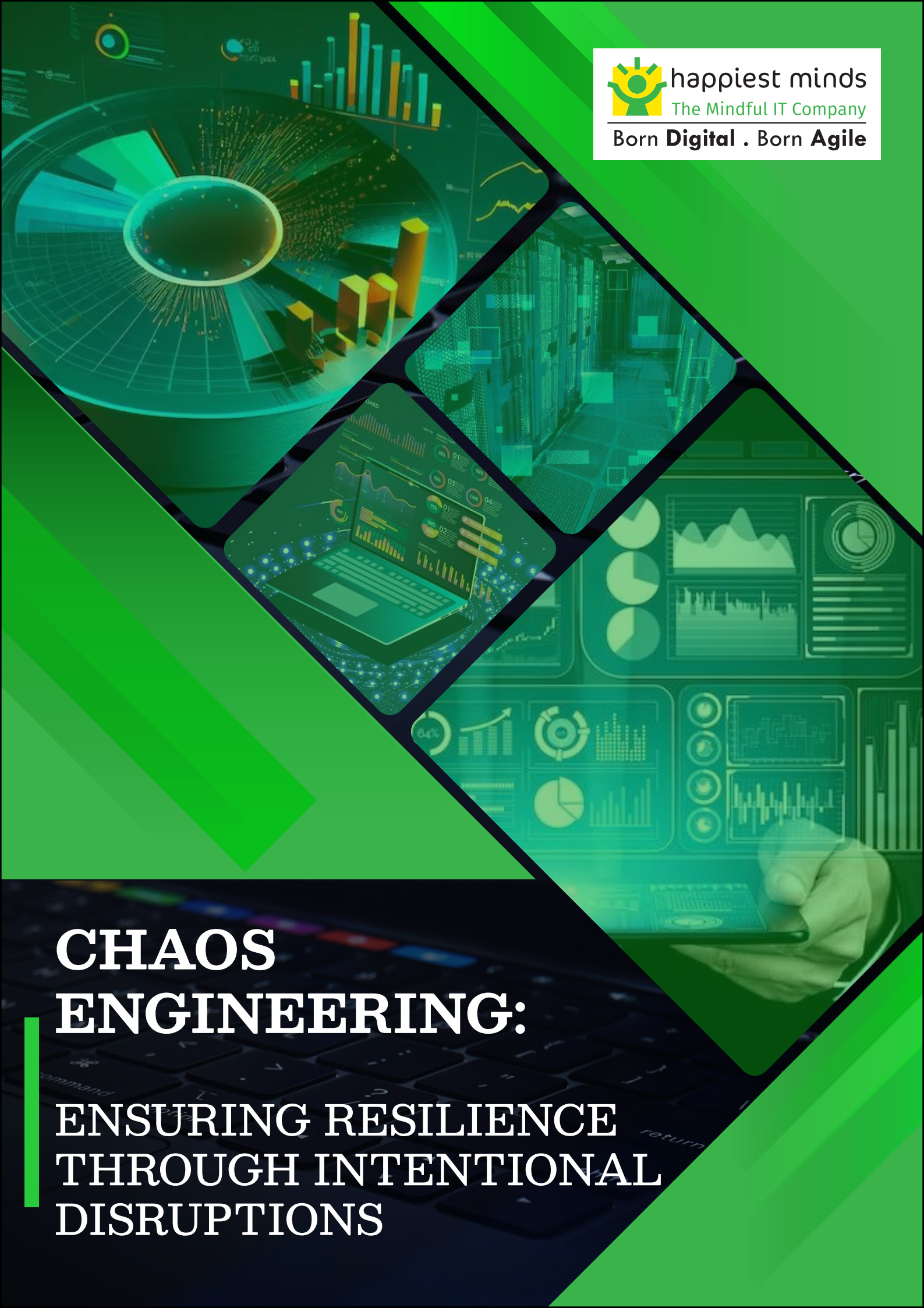


happiest minds
The Mindful IT Company
Born Digital . Born Agile

CHAOS ENGINEERING:

ENSURING RESILIENCE THROUGH INTENTIONAL DISRUPTIONS

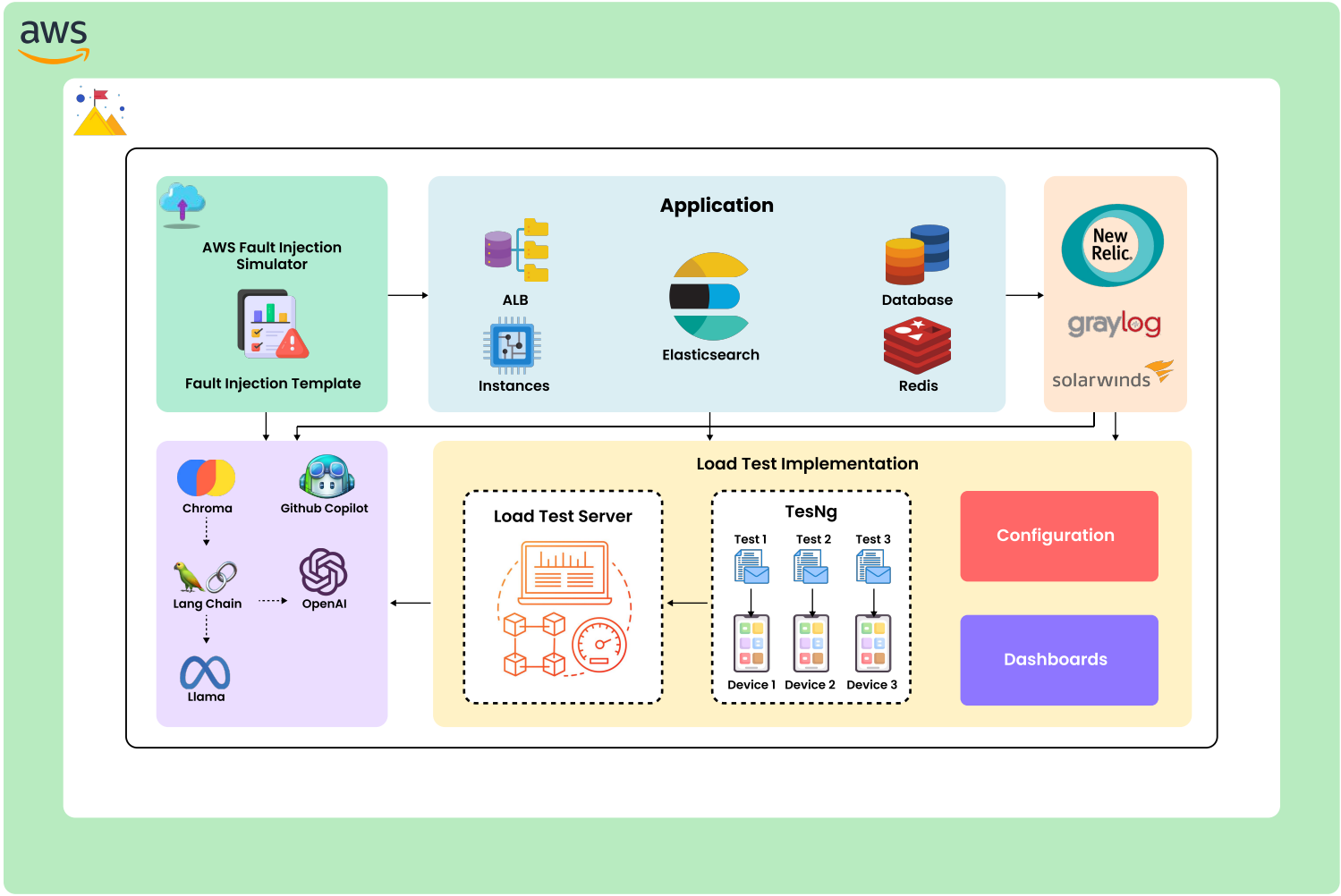
Table of Contents

Problem Statement

Testing the resilience of complex enterprise applications is challenging as several things can change at the same time. Unexpected service outages caused by extreme conditions often affect business continuity

Solution

By designing and executing **Chaos Engineering experiments** we can proactively identify potential failure points and correct them before they cause an actual outage or other disruption. GenAI Model can suggest the experiements to be executed to improve a systems' resilience based on the logs and the impacts caused. Additionally, the quality of Chaos Testing can be improved through reinforcement learning.



Resources should be added with experiments so that while running **Chaos Experiments,** the system can capture the logs and metrics that are associated with the experiments.

Intelligent Chaos Experiment

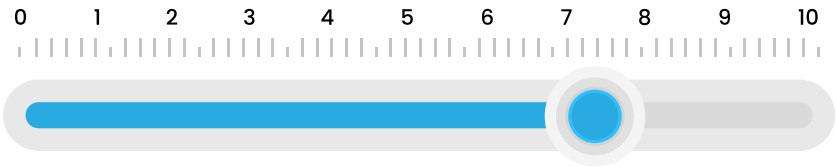
New experiments could be created using GenAI based on the infrastructure, application components, observability metrics and resilience score

Security Testing

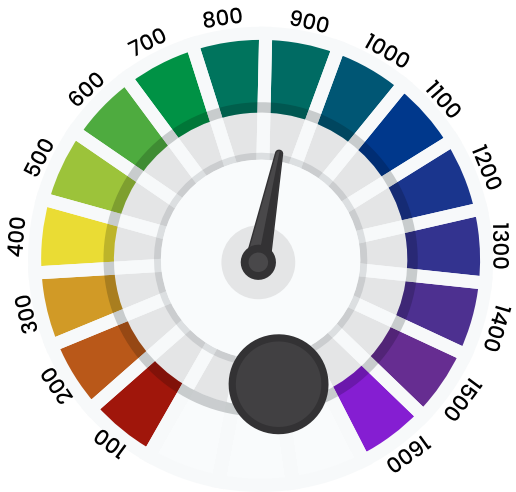
Combining pen testing and automation with Chaos Engineering can help organizations identify and remediate vulnerabilities more effectively, quickly, and efficiently.

Disclaimer: All logos are the property of their respective owners

Resilience Score Calculation



CHAOS EXPERIMENT RESULTS



TEST AUTOMATION RESULTS

LOAD TEST RESULTS

- Rs = Resilience score
- C = Number of Successful Chaos Experiments
- Cmax = Maximum Number of Chaos Experiments
- L = Number of Successful Load Tests
- Lmax = Maximum Number of Load Tests
- A = Number of Successful Automation test cases
- Amax = Maximum Number of Automation tests
- N = Number of testing criteria

RESILIENCE SCORE

$$RS = \frac{\left(\frac{C}{Cmax} \times 100\right) + \left(\frac{L}{Lmax} \times 100\right) + \left(\frac{A}{Amax} \times 100\right)}{N}$$

Chaos Testing Vs Functional Testing

Chaos tests and functional tests are two different approaches with different **objectives and methodologies.** Some of the main differences are listed below.

The main purpose of functional testing is to identify bugs, issues, and defects and fix them. In chaos testing **the main objective is to simulate real-world unexpected events** to test a system's resilience and fault tolerance

Chaos testing does not follow a predefined set of test cases instead it randomly introduces disruptive events to a system to validate its stability and fault tolerance. Regular functional testing follows predefined test cases and **scenarios to make sure a system functions as intended.**

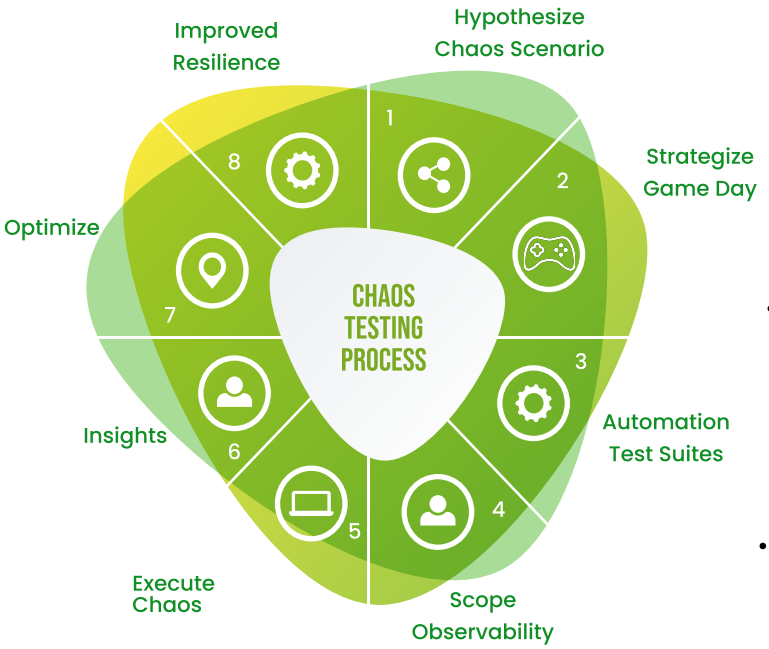
Regular testing is usually done in **controlled and stable environments but in chaos testing** it is conducted in a chaotic environment as the name implies. The environment can be dynamic and unpredictable.

Chaos tests primarily focus on evaluating the capability of a system among unpredicted situations. Regular testing covers functional and non-functional aspects of the software.

Best Practices Of Chaos Testing

- Defining the software system's stable and normal behavior when no chaos is introduced.
- Defining the goals and objectives of the Chaos Tests clearly at the beginning to ensure that the system is tested as you prefer.
- Simulate scenarios as close as possible to real-world events. This ensures the quality of the system is upto a certain standard.
- Start with controlled regression tests. This helps in identifying and assessing the impact on the individual components of the system.
- Develop a hypothesis and experiment until the hypothesis is proven.
- Follow the Chaos Test pyramid instead of testing all at once. This helps in identifying bottlenecks and gathering more information for improvement.
- Document all data during an experiment to analyze and learn about the system's behavior under different circumstances.

Chaos Experiment Process



- Define Chaos Experiements or leverage GenAI to generate experiements to mimic outages caused by spikes in terms of memory, CPU, network or even total region outage.
- Created Chaos can be executed to test environment for resilence
- Test Automation combined with Chaos experiment will bring out weak links
- Based on the impacts caused, applications or infrastructure need to be optimized to handle disasters
- Intelligence can be brought in by predictive and prescriptive analytics, and re-inforcement learning

Chaos Testing Metrics

Collecting baseline metrics for Chaos Engineering involves understanding the normal operating conditions of your system before introducing controlled chaos experiments. Here's a step-by-step approach to collecting these baseline metrics:

1. **Identify Key Performance Metrics:**
   - Performance Metrics: Response time, throughput, error rates.
   - Resource Utilization Metrics: CPU usage, memory usage, disk I/O, network I/O.
   - Application-specific Metrics: User transactions, API response times, database query performance.
2. **Identify Key Observability Metrics:**
   - Utilize monitoring and observability tools such as Prometheus, Grafana, Datadog, New Relic, or CloudWatch.
   - Ensure you have comprehensive dashboards to visualize these metrics.
3. **Define Normal Operating Conditions:**
   - Establish what constitutes normal behavior under typical load conditions.
   - Define thresholds for acceptable performance and resource utilization.
4. **Collect Historical Data:**
   - Gather historical data over time to account for variations in load and usage patterns.
   - Use this data to understand baseline trends and identify anomalies.
5. **Perform Load Testing:**
   - Conduct load testing using tools like Apache JMeter, Locust, or Gatling to simulate different levels of traffic.
   - Record metrics under various load conditions to see how the system behaves.
6. **Document Metrics and Baselines:**
   - Create a detailed report documenting the baseline metrics.
   - Include graphs and charts to visualize the normal operating conditions.
7. **Set Up Alerts:**
   - Configure alerts for deviations from baseline metrics.
   - Define thresholds and alerting rules to quickly detect and respond to anomalies during Chaos experiments.
8. **Establish a Control Group:**
   - If possible, set up a control environment where no Chaos experiments are conducted.
   - Compare metrics from the control group with those from the experimental group to assess the impact.

By thoroughly understanding and documenting your baseline metrics, you'll be better prepared to identify the impact of chaos experiments and ensure your system's resilience.

High Severity Incident (SEV) Metrics

High Severity Incident (SEV) metrics are critical for understanding the impact of serious incidents on your system and for improving incident response and resolution processes. Here are key metrics to track for High Severity Incidents:

1. **Incident Frequency:**
- **Number of Incidents:** Total number of high severity incidents over a specific period.
- **Incident Rate:** Incidents per unit time (e.g., per week, per month).

2. **Detection and Response:**
- **Mean Time to Detect (MTTD):** Average time taken to detect an incident from the moment it occurs.
- **Mean Time to Acknowledge (MTTA):** Average time taken to acknowledge the incident after detection.
- **Mean Time to Escalate (MTTE):** Average time taken to escalate the incident to the appropriate team or individual.

3. **Resolution Metrics:**
- **Mean Time to Resolve (MTTR):** Average time taken to fully resolve the incident from the moment it was detected.
- **Mean Time to Recover (MTTR):** Average time taken to recover services to a normal state after an incident.
- **First Time Fix Rate (FTFR):** Percentage of incidents resolved on the first attempt without recurrence.

4. **Impact Metrics:**
- **Affected Users:** Number of users impacted by the incident.
- **Downtime Duration:** Total duration of system unavailability due to the incident.
- **Service Level Agreement (SLA) Breaches:** Number of SLA breaches caused by the incident.
- **Financial Impact:** Estimated financial cost of the incident, including lost revenue and recovery costs.

5. **Root Cause Analysis:**
- **Time to Identify Root Cause:** Time taken to determine the root cause of the incident.
- **Root Cause Categories:** Classification of root causes (e.g., hardware failure, software bug, human error).

6. **Communication and Coordination:**
- **Stakeholder Notifications:** Number of notifications sent to stakeholders during the incident.
- **Incident Updates:** Frequency of updates provided to stakeholders and users during the incident.
- **Post-Mortem Completeness:** Percentage of incidents with a completed post-mortem analysis.

7. **Preventive Measures:**
- **Action Items Identified:** Number of action items identified during post-mortem analysis.
- **Action Items Completed:** Percentage of identified action items that have been completed to prevent future incidents.

8. **Customer Satisfaction:**
- **Customer Satisfaction (CSAT):** Customer satisfaction rating after incident resolution.
- **Net Promoter Score (NPS):** Customer loyalty and satisfaction score post-incident.

Tracking these metrics will help your organization understand the impact of high severity incidents, improve incident management processes, and enhance overall system resilience.

Business Outcome

Ensures our enterprise IT system is resilient to disasters

Ensures high availability and reduces number of production incidents

Improves the end user experience

Which in turn have positive impacts on revenue

About Authors

RANJITH KUMAR MURUGASAMY

ASSOCIATE DIRECTOR • PDES-JOSS

Ranjith Kumar is a seasoned Technologist with over 15 years of experience in designing innovative and scalable solutions. He always aims to foster a culture of innovation and collaboration, leveraging expertise in cloud computing, architecture, DevOps, security, data and product engineering. With a strategic vision, he focus on integrating emerging technologies, ensuring scalability, and nurturing talent. Ranjith is passionate about driving technological excellence and committed to leading our organization to the forefront of innovation, delivering exceptional value to our customers and stakeholders.

SIVAKUMARAN M P

SENIOR TECHNICAL LEAD • PDES-JOSS

Sivakumaran is a highly experienced Tech Lead with a strong background in framework development, specializing primarily in Java. With over a decade of expertise, he has a proven track record in designing and implementing sophisticated frameworks tailored for load testing. His proficiency with various load testing tools and techniques has enabled him to optimize performance and ensure the scalability of complex software systems. His commitment and continuous improvement positions him as a pivotal asset in driving technological excellence.

RAMASUBRAMANIAM R

TECHNICAL LEAD • PDES-JOSS

Ramasubramaniam is a highly skilled Tech Lead with a specialization in frameworks development primarily in Java stack. He brings over a decade of experience in creating robust, scalable, and efficient testing solutions. With a deep understanding of programming languages and testing tools, he has successfully led cross-functional teams in driving automation initiatives that reduce manual effort and accelerate release cycles.

SATHISH KUMAR

TECH LEAD • PDES-JOSS

Sathish Kumar is a highly skilled Tech Lead with extensive expertise in framework development, specializing in JavaScript and frontend technologies. With over a decade of experience, Sathish has a proven ability to design and implement cutting-edge frameworks that enhance the efficiency and performance of web applications. His deep knowledge of modern frontend technologies, including React, Angular, and Vue.js, enables him to create seamless and intuitive user experiences. He has always been a valuable asset in driving the success of digital projects.

About Happiest Minds Technologies

Happiest Minds Technologies Limited (NSE: HAPPSTMNDS), a Mindful IT Company, enables digital transformation for enterprises and technology providers by delivering seamless customer experiences, business efficiency and actionable insights. We do this by leveraging a spectrum of disruptive technologies such as: artificial intelligence, blockchain, cloud, digital process automation, internet of things, robotics/drones, security, virtual/ augmented reality, etc. Positioned as 'Born Digital . Born Agile', our capabilities span Product & Digital Engineering Services (PDES), Generative AI Business Services (GBS) and Infrastructure Management & Security Services (IMSS). We deliver these services across industry groups: Banking, Financial Services & Insurance (BFSI), EdTech, Healthcare & Life Sciences, Hi-Tech and Media & Entertainment, Industrial, Manufacturing, Energy & Utilities, and Retail, CPG & Logistics. The company has been recognized for its excellence in Corporate Governance practices by Golden Peacock and ICSI. A Great Place to Work Certified™ company, Happiest Minds is headquartered in Bengaluru, India with operations in the U.S., UK, Canada, Australia, and the Middle East.

For more information, write to us at **business@happiestminds.com**

happiest minds
The Mindful IT Company
Born **Digital** . Born **Agile**

www.happiestminds.com