Postmortem: Web Application Outage Incident

Issue Summary: Duration: August 10, 2023, 08:45 AM - August 10, 2023, 10:30 AM (UTC) Impact: The web application experienced a partial outage, rendering the user dashboard inaccessible for approximately 1 hour and 45 minutes. During this period, around 30% of users were unable to access their accounts or perform essential tasks.

Root Cause: The root cause of the outage was identified as a database connection bottleneck. A sudden surge in user activity led to an unforeseen increase in database queries, overwhelming the connection pool capacity and causing a cascading failure.

Timeline:

- 08:45 AM: The issue was detected through a spike in error rate on monitoring dashboards.
- 08:50 AM: Engineering team was alerted automatically via monitoring system.
- 09:00 AM: Initial investigation focused on the application server logs and networking components.
- 09:15 AM: Assumption was made that the issue was related to load balancer misconfiguration.
- 09:30 AM: Load balancer settings were adjusted, but the issue persisted.
- 09:45 AM: Incident was escalated to senior engineers for further investigation.
- 10:00 AM: Database metrics were closely examined, revealing a connection pool bottleneck.
- 10:15 AM: Scaling up the database instance was considered as a temporary fix.
- 10:30 AM: Database connections were reconfigured, resolving the issue and restoring full application functionality.

Misleading Investigation/Debugging Paths: Initial focus on the load balancer configuration proved to be a misleading direction. Scaling up the application server instances did not yield any improvements, as the issue was not on the application server side.

Escalation: The incident was escalated to the Database Operations team and senior software engineers due to its critical nature and persistence.

Resolution: The incident was resolved by identifying the bottleneck in the database connection pool. The connection pool settings were reconfigured to handle a higher number of incoming connections, allowing the application to function without interruptions.

Root Cause and Resolution: The root cause was a database connection pool bottleneck. The surge in user activity caused an unexpected increase in database

queries, saturating the available connection pool and leading to query timeouts and errors. To address this, the connection pool configuration was adjusted, increasing the maximum number of connections and optimizing connection reuse. This change allowed the application to effectively handle the increased load without overwhelming the database.

Corrective and Preventative Measures:

1. **Optimize Database Queries:** Conduct a review of frequently executed queries and optimize them for better performance to reduce the strain on the database.
2. **Load Testing:** Implement regular load testing to identify potential bottlenecks and ensure the system can handle anticipated spikes in user activity.
3. **Auto-scaling Database:** Explore the implementation of auto-scaling for database instances to dynamically adjust resources based on demand.
4. **Enhanced Monitoring:** Set up more granular monitoring on the connection pool and database performance metrics to detect similar issues early.
5. **Runbooks:** Develop comprehensive runbooks outlining steps to diagnose and address common performance bottlenecks to expedite future incident resolution.

Tasks to Address the Issue:

1. **Database Connection Pool Configuration:** Increase the maximum connections allowed in the connection pool to accommodate higher traffic.
2. **Database Scaling Strategy:** Design and document a strategy for scaling the database resources during high traffic periods.
3. **Load Testing Automation:** Implement automated load testing scripts to simulate various traffic scenarios and identify potential vulnerabilities.
4. **Alert Thresholds Review:** Reevaluate monitoring alert thresholds to ensure timely notifications for impending performance issues.
5. **Performance Review Process:** Establish a periodic performance review process to proactively identify and address potential bottlenecks.

By implementing these corrective and preventative measures, we aim to enhance the application's stability and provide a seamless experience to our users even during peak usage periods.