ROOT CAUSE ANALYSIS

DB Ownership Change

The following is a detailed accounting of the service interruption that Rally users experienced on July 28, 2015 at 2:53 pm.

Root Cause Analysis Summary:

Event Date	July 28, 2015
Event Start	2:53 pm
Downtime Start	2:53 pm
Time Detected	2:53 pm
Time Resolved	4:15 pm
Downtime End Time	3:33
Event End Time	4:15 pm
Root Cause	Initial: User error: misplaced `chown -R` Continuation: R/O pools were reduced to 1, but not disabled Continued impairment (speculative): High concurrency conflict exception rates correlated with a misconfigured TNS listener on the R/O standby post-recovery resulting from the mistaken chown, correcting permissions resolved the concurrency conflict issue.
Customer Impact	Full downtime from 2:53 - 3:33 pm. Likely impaired in some capacity from 15:33 to 4:15 pm pm (concurrency conflict exceptions). Data loss did not occur during this event, since we had closed the app front door prior to failover.
Duration	Total Downtime: 40 minutes Total Impaired Availability: 42 minutes (3:33 - 4:15pm) Time to Detect: 0 minutes Time to Resolve: 1 hour 22 minutes

Future Preventative Measures:

Actions that should be taken to prevent this Event in the future.

Actions	**Description**
Fix user shuffling at converge time	Converge (by some mechanism) seems to shuffle UIDs for local user accounts on prod hosts. This causes ownership of home directories to be changed, and in this particular case caused user to be logged in at / instead of $HOME

©2015 Rally Software Development Corp. All rights reserved. rallydev.com

Spike ‘evasive maneuvers’ in default bash profiles	See if there’s a cheap and easy way to cause destructive actions against / to prompt warnings and explicit confirmation of intention
Get additional R/O DBs online	Accidental misconfiguration of R/O mechanisms may have been avoided by having a second and third R/O DB online and ready for use - (purchase order has been created, stories to be added)
Rebuild db-02	Ensure that the DB machine is in the desired state.
Reproduce Error state in safe environment to determine how the connections were not released.	Understanding the concurrency errors to ensure we are fixing the correct problem.
Produce documentation for R/O transition	Lack of clarity on all of the steps to move the R/O connections.
Correct application healthcheck	We had a false picture of when the application was healthy, and there was a lot of work required after we gave the initial all-clear. The current healthcheck does not involve a DB call to ensure the application is actually serving requests.
SSO Error on ‘unavailable.rallydev.com’	“SSO login causes a S3 error on unavailable.rallydev.com (no POST allowed)”
Understand impact to Flowdock	Flowdock was not responding during this incident, we need to understand if there was a correlation, and if so, why.
ALM exposed EL exceptions	When qd-db-02 became uncommittable, ALM starting showing customers full eclipselink exception content. This should be hidden with extreme prejudice
No Nagios check for ALM 500 errors	We should be alerting when ALM is throwing 500 errors (assumed that this would need to be a threshold/rate to monitor on, plus determine course of action of monitoring these across all the appservers)
Define what it means for ALM to be ‘available’ after an outage	The current healthchecks were too shallow to truly determine the state of the system. Has to be done manually, this should be automated to provide an objective measure, not a request to any individual.

©2015 Rally Software Development Corp. All rights reserved. rallydev.com