Moving to CI/CD

The case for Continuous Integration and Continuous Deployment

Continuous Integration (CI)

- CI refers to the practice of several developers merging working copies of code to a shared source with regularity, often several times per day
- Every time code is merged it is run through a rigorous screening process that builds the code, tests the code, and frequently performs analysis of vulnerabilities in code
- Leads to less time spent fixing issues from new code
- Fewer bugs make the transition to production leading to less rework
- Leads to lower testing time and faster delivery
- Catches security vulnerabilities that can be costly

Continuous Deployment (CD)

- CD refers to a software approach where deployments are automated and frequent
- Every process needed to deploy the software fits here such as provisioning infrastructure like servers, moving the necessary files, rolling back code, promoting software to production and verification testing.
- Yields faster deployments through a reduction in configuration errors
- Lowers cost of unused resources through automating infrastructure removal
- Features are released more rapidly as generating more business value
- Fewer engineers involved in deployment leading to lower costs
- Quick to return to working production state when failures occur

Conclusion

- Utilizing both CI/CD includes has the potential to dramatically reduce
  - Infrastructure costs
  - Time to deliver features
  - Time spent fixing bugs
  - Number of release engineers
- It will limit the ability of engineers to manually push to production. Ultimately, this should be a small cost relative to the overwhelming benefits provided