How to Upgrade a Fleet of Server

1. Functional Requirements
    a. No downtime
    b. Ability to rollback if needed
    c. Ability to rollout to thousands of servers
2. Case 1) Deploy new binary
    a. Set up a CI pipeline where artifacts are machine images containing the updated binary
    b. Machine image stored in repository reachable by service turning up the new servers
    c. Canary deploy
        i. Divert traffic from 5% of prod servers
            1. Allow existing connections to gracefully terminate
            2. Note: this will cause a 5% increase in traffic to remaining servers
        ii. Once all connections are removed, spin down those servers
        iii. Spin them back up using the new machine image
        iv. Allow traffic to hit those servers
            1. Note: subset of customers will see the new layout (byproduct of canary)
        v. Verify live traffic on new servers behave as expected e.g., automated testing, monitoring
            1. If new servers fail testing then rollback by:
                a. Diverting traffic away from the new servers
                b. Spinning them back down
                c. Redeploy with previous machine image
                d. Allow traffic back on those servers
        vi. Repeat above with increasing increments until 100% of the servers are updated
    d. Blue-green deploy (instead of Canary)
        i. Upgrade servers in secondary environment with new machine image
        ii. Divert traffic from prod servers to the new servers
            1. Note: we can do this gradually or all at once depending on risk tolerance
        iii. Verify live traffic on new servers behave as expected
            1. If new servers fail testing then rollback by diverting traffic back to old prod environment
3. Case 2) Update a configuration file
    a. Store new update file in a central repository e.g., Github or object store e.g., S3
    b. Using a configuration management tool e.g., Ansible we can run a set of commands from one local server to all remote servers to pull the updated file and restart the app e.g., systemctl restart
    c. Verify live traffic on new servers are behaving as expected e.g., automated testing, monitoring
        i. If new servers fail testing then rollback by:
            1. Run set of commands via configuration management tool to pull the last version and restart the app
    d. Note that we can also bake the new config into a new image and follow Case 1. The strategy highlighted above may be more suitable as a hotfix.
4. Case 3) Update infrastructure
    a. Infrastructure should be maintained and stood up as code e.g., Terraform, Cloudformation
    b. Infra changes that require redeploy of servers e.g., scaling up of server type should be done via canary or blue-green as described in Case 1
    c. Infra changes that do not require redeploy of servers e.g., adding more nodes to a deployment can be done without a rollout strategy
    d. In either case (b or c), live traffic should be verified
        i. If new configuration fails then rollback by re-deploying previous version of infra-as-code