perplexity

Blue-Green vs. Canary Deployment

Both are deployment strategies used to release software with minimal downtime and risk, but they differ in **how they manage traffic** and **resources**.[1] [2]

1. Blue-Green Deployment (Instant Switch)

- **Concept:** You maintain **two identical environments**:
  - **Blue:** The current live version (Active).
  - **Green:** The new version (Idle/Staging).
- **Process:** You deploy the new version to **Green**. You test it fully. When ready, you switch the Load Balancer to point **100% of traffic** from Blue to Green instantly.[2] [1]
- **Rollback:** Very fast. If Green breaks, you just switch the router back to Blue.
- **Pros:** Instant cutover, easy rollback, clean state (no version mixing).
- **Cons:** Expensive (requires double the infrastructure), "All-or-Nothing" risk (if it breaks, it breaks for everyone instantly).[2]

2. Canary Deployment (Gradual Rollout)

- **Concept:** You deploy the new version to a **small subset of users** (e.g., 5%) while the rest (95%) stay on the old version.
- **Process:** You monitor the "Canary" (new version). If error rates are low, you gradually increase traffic (10% → 50% → 100%).[1] [2]
- **Rollback:** Fast, but requires stopping the rollout and reverting the canary instances.
- **Pros: Lowest Risk** (only 5% of users are affected by a bug), cheaper (no need for 2x resources), allows testing in production.[2]
- **Cons:** Complexity (traffic routing rules needed), slow rollout, mixed versions running simultaneously (compatibility issues).[3]

Comparison Table

Feature	Blue-Green	Canary
Traffic Switch	**100% Instant** (All at once)	**Incremental** (5%, 10%, 50%...)
Infrastructure	**High Cost** (Double resources)	**Low Cost** (Uses existing resources)
Risk	Medium (Affects everyone instantly)	**Low** (Affects few users first)
Complexity	Simple (Router switch)	Complex (Traffic splitting logic)

Feature	Blue-Green	Canary
Best For	Major releases, Schema changes	Experimental features, Risk minimization

Interview One-Liner:
"Blue-Green is safe for **quick rollbacks** but requires double the hardware (expensive). Canary is safe for **minimizing user impact** (only 1% see bugs) but is more complex to set up."

⁂

1. https://codefresh.io/learn/software-deployment/blue-green-deployment-vs-canary-5-key-differences-and-how-to-choose/
2. https://octopus.com/devops/software-deployments/blue-green-vs-canary-deployments/
3. https://www.harness.io/blog/blue-green-canary-deployment-strategies
4. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/429036f4-3bc3-4a7a-8c6f-90246e73b9b9/image.jpg
5. https://www.wissen.com/blog/the-role-of-blue-green-canary-and-feature-flags
6. https://www.xenonstack.com/blog/canary-vs-blue-green-deployment
7. https://blog.christianposta.com/deploy/blue-green-deployments-a-b-testing-and-canary-releases/
8. https://www.reddit.com/r/aws/comments/1bl0cgl/canary_release_vs_greenblue_deployment/
9. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/24f18c0a-b56e-4862-a619-e2959036a5c2/image.jpg
10. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/97e9ed86-ed11-4035-a18b-1eb05c1bab4e/image.jpg
11. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/87f62423-96f2-4071-9802-8f6699e0ecd8/image.jpg
12. https://circleci.com/blog/canary-vs-blue-green-downtime/
13. https://stackoverflow.com/questions/23746038/canary-release-strategy-vs-blue-green
14. https://www.reddit.com/r/devops/comments/1m777h8/scratching_my_head_trying_to_differentiate/