ArgoCD Configs - Server Side Apply Option with ArgoCD

Server Side Apply Option with ArgoCD

The "Server-Side Apply" option in ArgoCD changes how resource updates are applied to the Kubernetes cluster. Let's break down what this means in simple terms.

Client-Side Apply vs. Server-Side Apply

Client-Side Apply (OFF)

- **Default Behavior**: This is the standard method used by kubectl apply.
- **How It Works**:
- The kubectl command sends the entire resource configuration from your local machine (client) to the Kubernetes API server.
- The server replaces the current resource configuration with the new one provided by the client.
- **Pros**:
- Simple and straightforward.
- Works well for many standard use cases.
- **Cons**:
- Can lead to conflicts if multiple clients are updating the same resource because it replaces the entire configuration.

Server-Side Apply (ON)

- **Advanced Behavior**: This is a more sophisticated approach that Kubernetes provides.
- **How It Works**:
- The client sends a partial or full resource configuration to the server.
- The server merges this configuration with the existing resource configuration.

- This approach uses a concept called "field ownership," where different clients can own and update different parts of the same resource.
- **Pros**:
- Better handling of concurrent updates from multiple clients.
- More granular control over resource updates, reducing conflicts.
- Useful for declarative resource management, where multiple tools or teams manage different parts of the same resource.
- **Cons**:
- Slightly more complex to understand and implement.
- Requires Kubernetes 1.18 or later.

Real-Life Example

Imagine you have a Deployment managed by multiple teams:

1. **Team A**: Manages the container image and replicas.
2. **Team B**: Manages the resource limits and environment variables.

Scenario with Client-Side Apply (OFF)

- **Team A**: Updates the container image and applies the change.
- Command: kubectl apply -f deployment.yaml
- **Team B**: Updates the resource limits and applies the change.
- Command: kubectl apply -f deployment.yaml
- **Potential Conflict**: If Team B's deployment.yaml doesn't include the latest image changes from Team A, applying it will overwrite Team A's changes.

Scenario with Server-Side Apply (ON)

- **Team A**: Updates the container image and applies the change using server-side apply.

- Command: kubectl apply --server-side -f deployment.yaml
- **Team B**: Updates the resource limits and applies the change using server-side apply.
- Command: kubectl apply --server-side -f deployment.yaml
- **No Conflict**: The server merges the changes, ensuring both the new image from Team A and the new resource limits from Team B are applied.

How to Configure in ArgoCD

You can enable Server-Side Apply in ArgoCD by setting the Server-Side Apply option to ON.

1. yamlCopy codeapiVersion: argoproj.io/v1alpha1
2. kind: Application
3. metadata:
4. name: my-application
5. spec:
6. project: default
7. source:
8. repoURL: https://github.com/my-repo.git
9. path: manifests
10. destination:
11. server: https://kubernetes.default.svc
12. namespace: default
13. syncPolicy:
14. automated:
15. prune: true
16. selfHeal: true
17. syncOptions:

18. - Server-Side Apply=true

Analogy: Updating a Document

Consider a shared document managed by multiple editors:

- **Client-Side Apply (OFF)**:
- Each editor sends their entire copy of the document.
- The latest version completely replaces the previous one.
- Risk of overwriting each other's changes.
- **Server-Side Apply (ON)**:
- Each editor sends their changes or edits.
- The server merges these changes into the existing document.
- Changes from different editors are combined smoothly.

Conclusion

"Server-Side Apply" is a powerful feature for managing Kubernetes resources, especially in environments where multiple clients or teams update the same resources. It helps reduce conflicts and provides better control over resource updates by merging changes on the server. For most basic use cases, client-side apply is sufficient, but for more complex scenarios involving multiple contributors, server-side apply is highly beneficial.