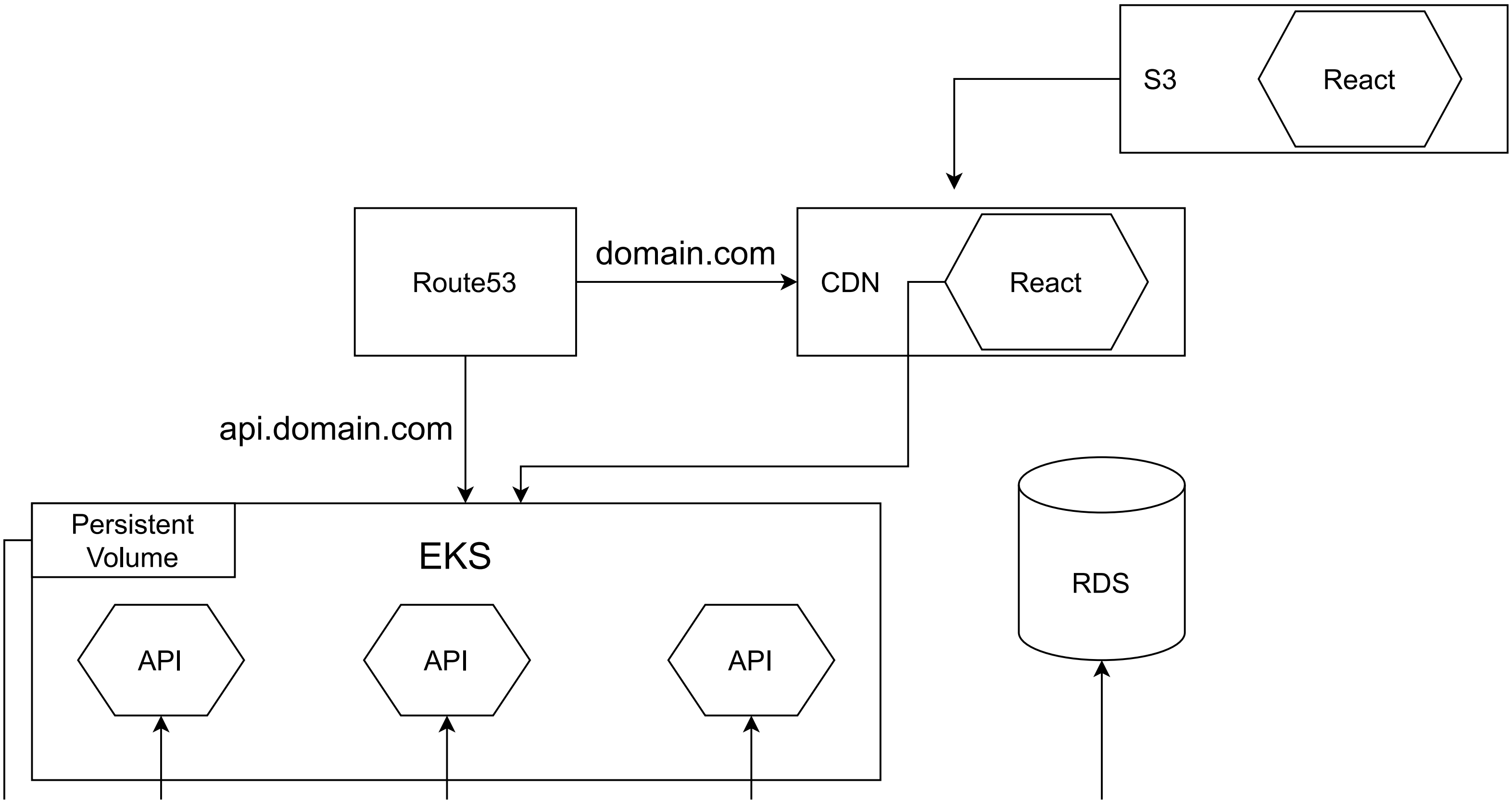
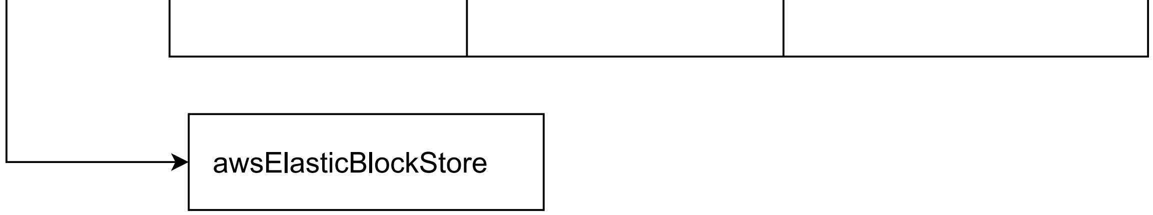
More Simple Solution: Above

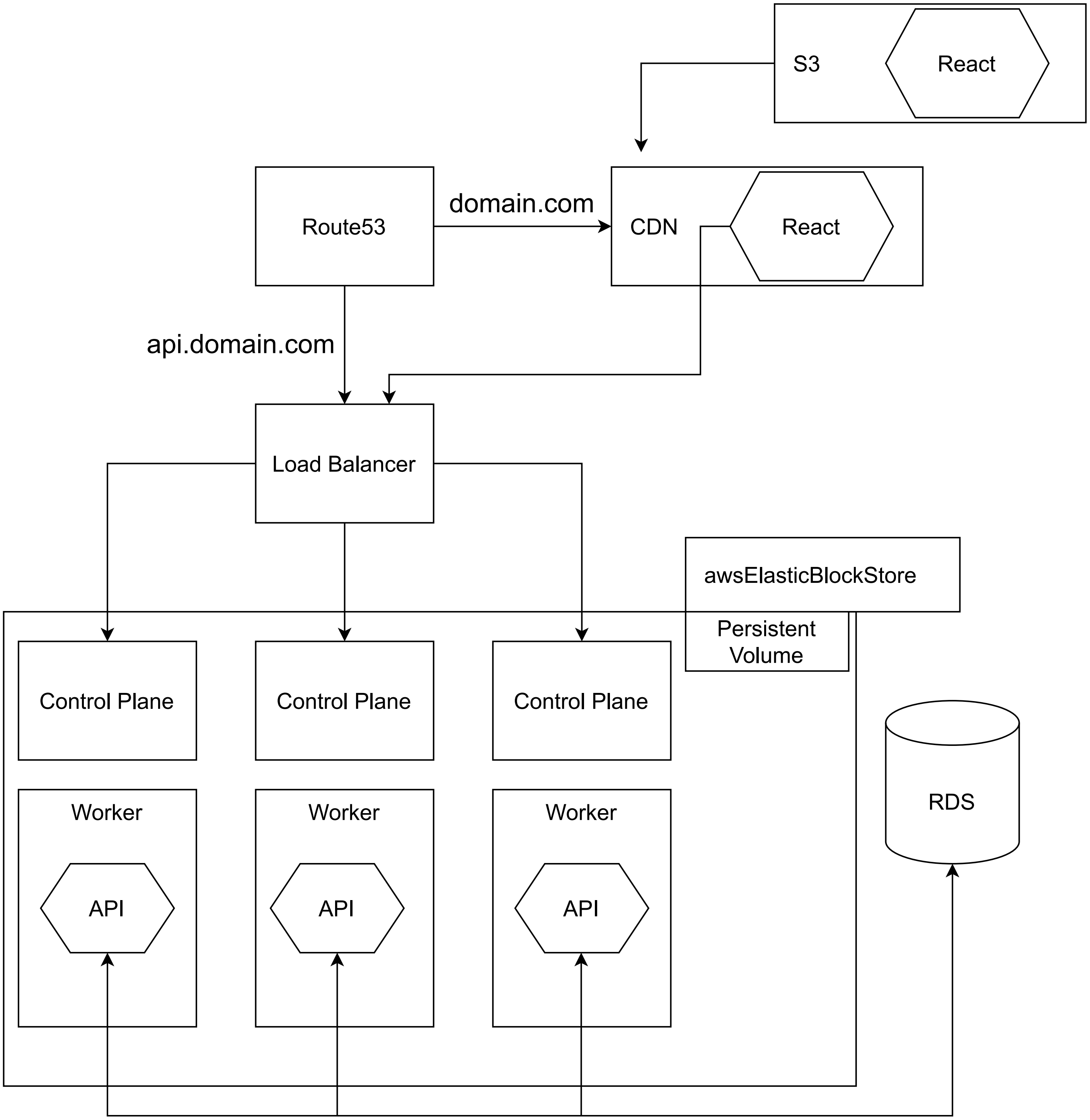
- React App is stored in S3 and then provided to the CDN
- Route53
  - Alias to serve the CDN content `domain.com`
  - "A" Record to an Kubernetes ingress `api.domain.com`
  - Potential to host SSL on the application layer
- Load balancing at the Kubernetes layer
- Database
  - Credentials stored as secrets and used in the deployment
- File Storage
  - k8s persistent volume backed by awsElasticBlockStore
  - Alternatively S3
  - This is all assuming the API is consuming the storage

More Complex Solution: Below

- Mostly the same as the above
- Non-managed Kubernetes
- Load balancer to host SSL
- Potential to host the React application as Kubernetes deployment
  - Example: If the application had to sit on a smaller webserver to proxy API requests or use an API key




awsElasticBlockStore


S3
React
Route53
domain.com
CDN
React
api.domain.com
Load Balancer
awsElasticBlockStore
Persistent Volume
Control Plane
Control Plane
Control Plane
Worker
API
Worker
API
Worker
API
RDS