Use a Kubnernetes Based Container as Runtime

[adr-0001:Active:11/2016]

Context and Problem Statement

We would like to have this project be used as a demo to showcase some of the interesting blockchain capabilities. In the real world, consensus-based algorithms are very computationally intensive, and we wanted to simulate creating an architecture that would be suitable and scalable for this type of application. Best practice dictates the use of a cloud native architecture and a container-based runtime orchestrator.

Kubernetes (K8s) was chosen as the container solution because it is very popular, supports development via Minikube, and can be deployed to all major cloud providers (AWS, GCF, Azure)

Considered Options

- Kubernetes - Open Source
- Amazon EKS - Amazon's Managed Kubernetes Service
- Amazon Fargate - Amazon's General Purpose Container Solution
- Amazon's ECS - Amazon Elastic Container Service
- Other cloud based container and Kubernetes hosted services were not considered because of expertise on the Amazon platform

Decision Outcome

Chosen option: "Amazon EKS", because

- Development can be done local with MiniKube
- Deployment to AWS via ECS is simple and can be fully automated
- World class Amazon support and proven scale
- Cost effective

```
@startuml Basic Sample
!includeurl https://raw.githubusercontent.com/RicardoNiepel/C4-
PlantUML/master/C4_Container.puml

Person(admin, "User")
package "Container Runtime" <<boundary>> as c1 {
    Container(web_app, "Blockchain Simulator", "Angular", "Allows users to
understand basic Blockchain Operators")
}

Rel(admin, web_app, "Uses", "HTTPS")
@enduml
```

```
@startuml "Architecure Highlights"
!includeurl https://raw.githubusercontent.com/RicardoNiepel/C4-
PlantUML/master/C4_Container.puml

LAYOUT_TOP_DOWN
'LAYOUT_AS_SKETCH
'LAYOUT_WITH_LEGEND

Person(admin, "User")
package "Container Runtime" <<boundary>> as c1 {
    package "Blockchain Simulator" <<boundary>> as c2 {
        hide stereotype

        Container(wa, "WebApp", "Angular", "Allows users to understand
basic Blockchain Operators")

        Container(wa_miner, "Embeded Miner", "Typescript", "Uses
Observables in the client to mine blocks")

        Container(wa_cli, "Miner Client", "Typescript", "Uses Observables
in the client to mine blocks")

        Container(go_miner, "GoLang Miner", "GoLang", "Uses GinGonic
framework [[https://gin-gonic.github.io/gin/ link]]")

        Container(kotlin_miner, "Kotlin Miner", "Typescript", "Uses
Observables in the client to mine blocks")

        Container(node_miner, "Node Miner", "Typescript", "Uses
Observables in the client to mine blocks")

        Container(scala_miner, "Scala Miner", "Typescript", "Uses
Observables in the client to mine blocks")

        Rel_L(wa, wa_miner,"typescript angular service")
        Rel_R(wa, wa_cli,"typescript angular service")

        Rel(wa_cli,go_miner,"HTTP")
        Rel(wa_cli,kotlin_miner,"HTTP")
        Rel(wa_cli,node_miner,"HTTP")
        Rel(wa_cli,scala_miner,"HTTP")
    }
}

Rel(admin, wa, "Uses", "HTTPS")
@enduml
```