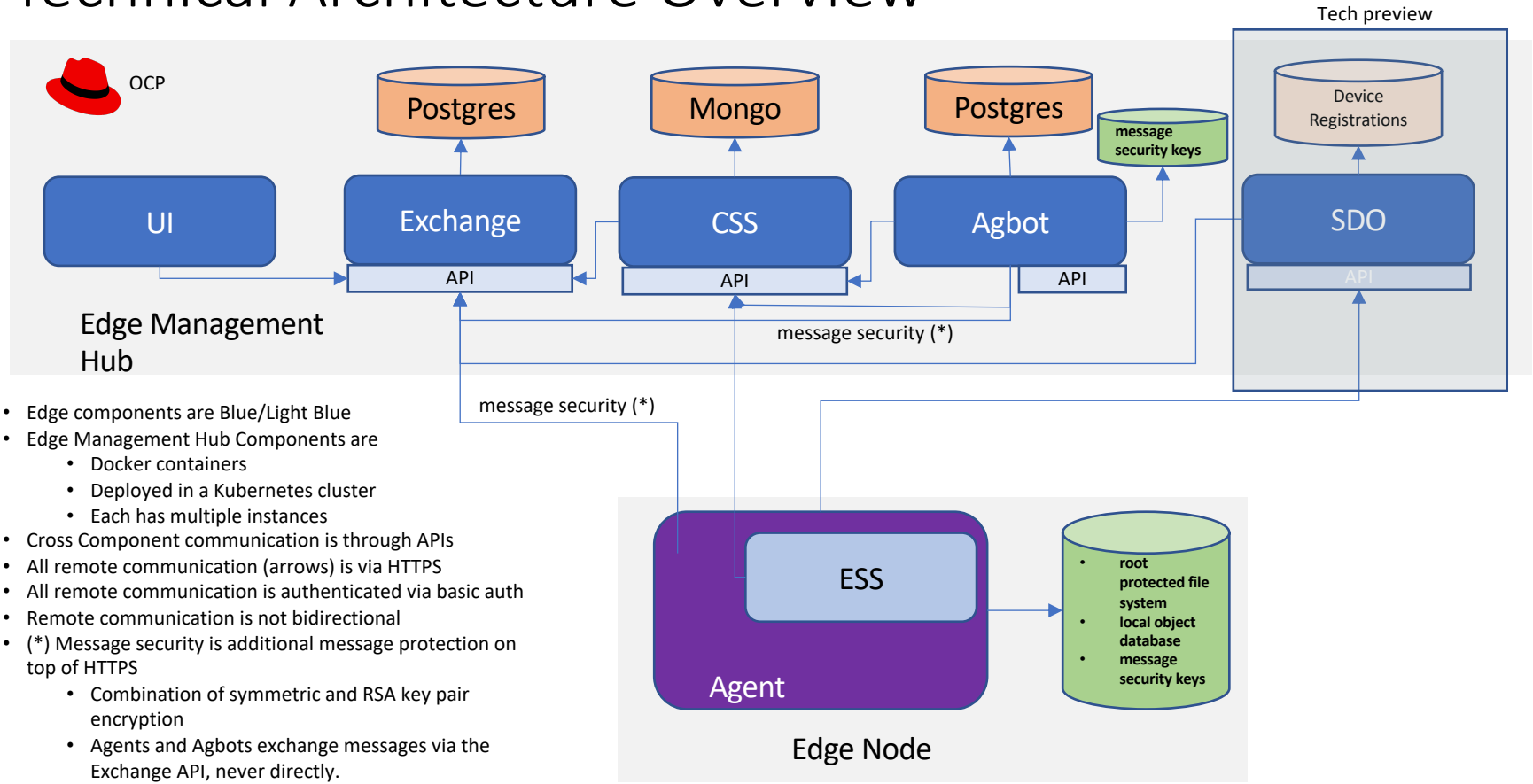
IBM Edge Application Manager tutorial with ML Model Updates

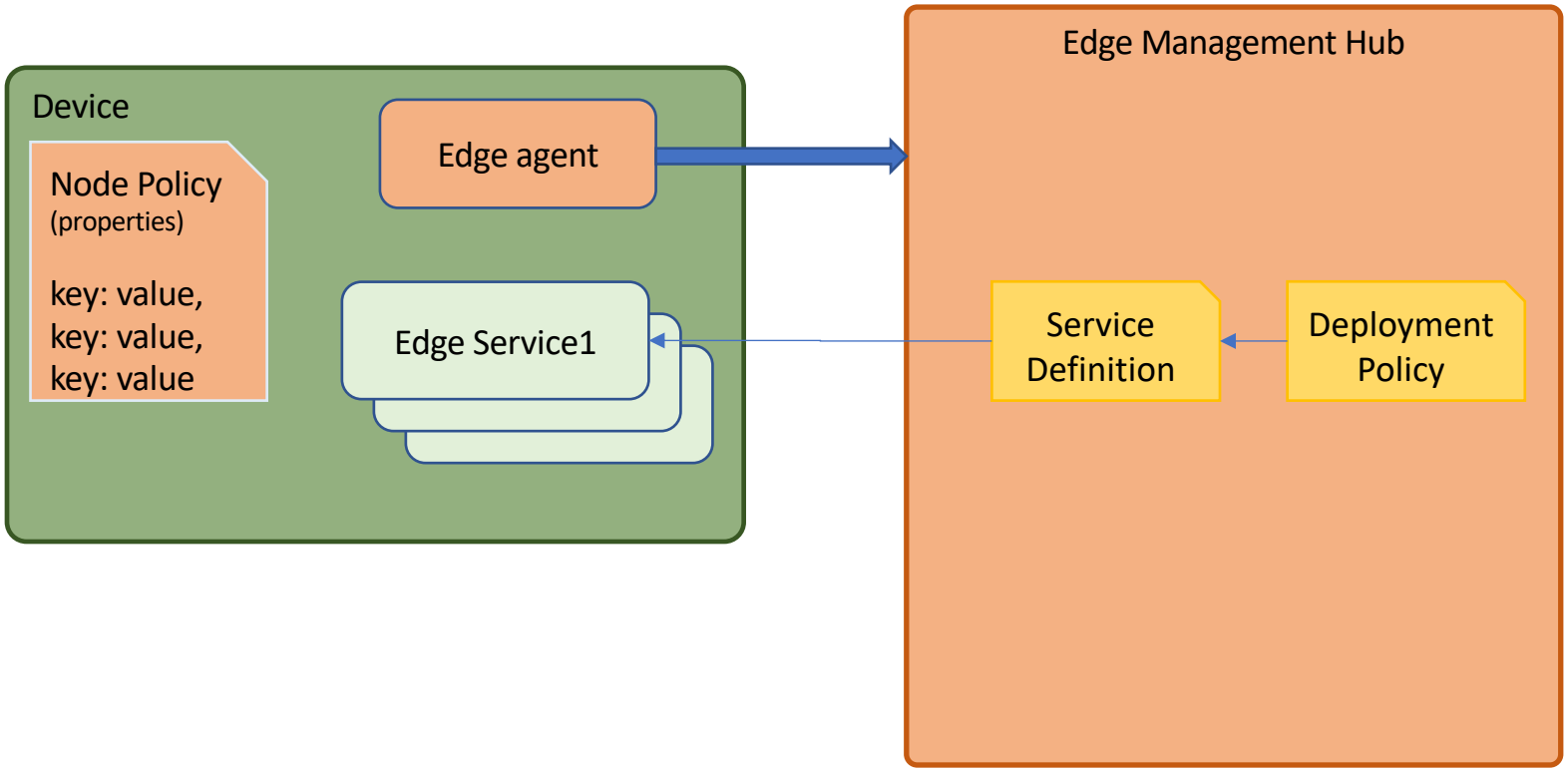
IBM **Cloud and Cognitive Software**

Technical Architecture Overview

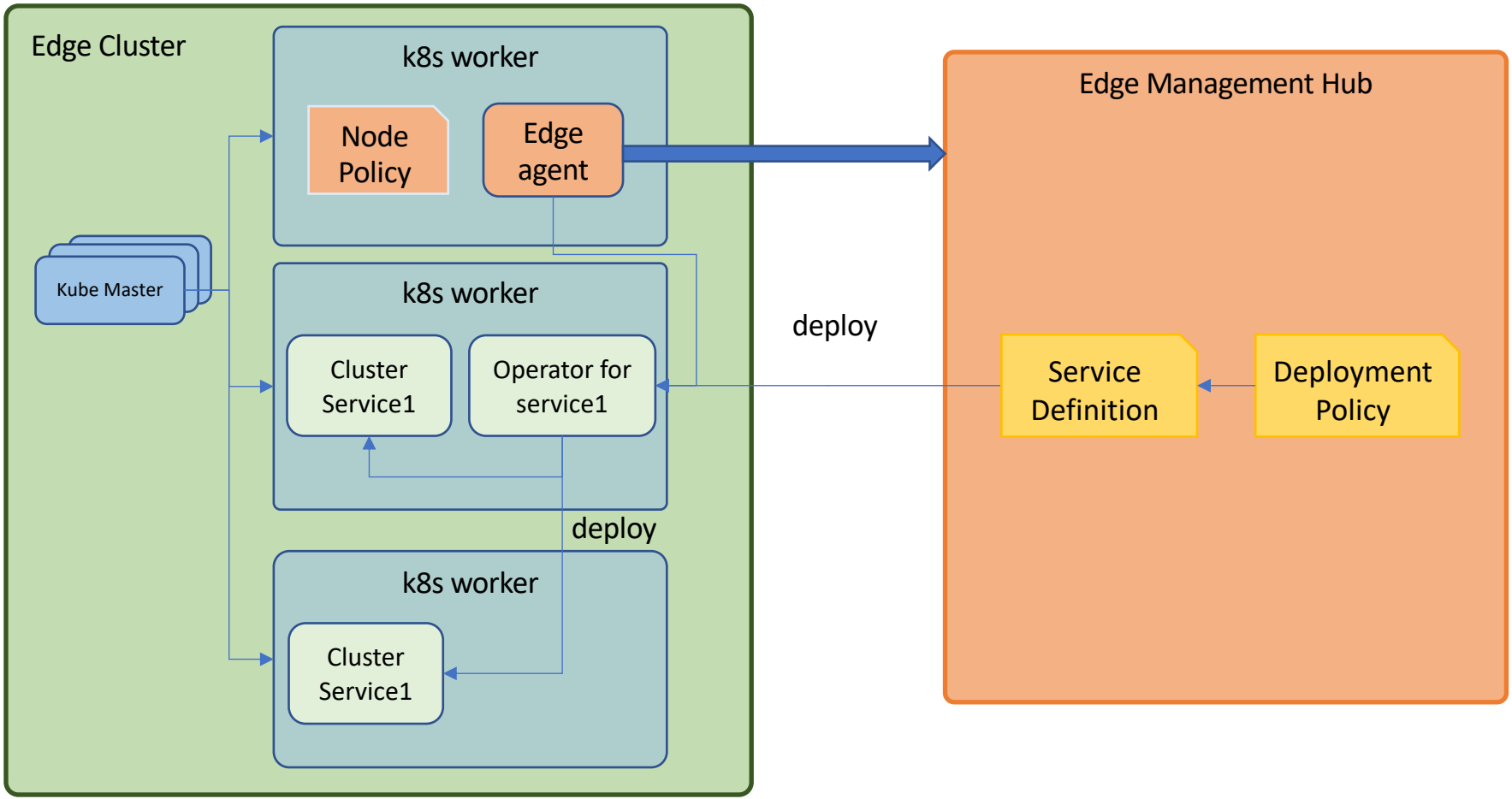


- Edge components are Blue/Light Blue
- Edge Management Hub Components are
  - Docker containers
  - Deployed in a Kubernetes cluster
  - Each has multiple instances
- Cross Component communication is through APIs
- All remote communication (arrows) is via HTTPS
- All remote communication is authenticated via basic auth
- Remote communication is not bidirectional
- (*) Message security is additional message protection on top of HTTPS
  - Combination of symmetric and RSA key pair encryption
  - Agents and Agbots exchange messages via the Exchange API, never directly.
- There are thousands of Edge Nodes

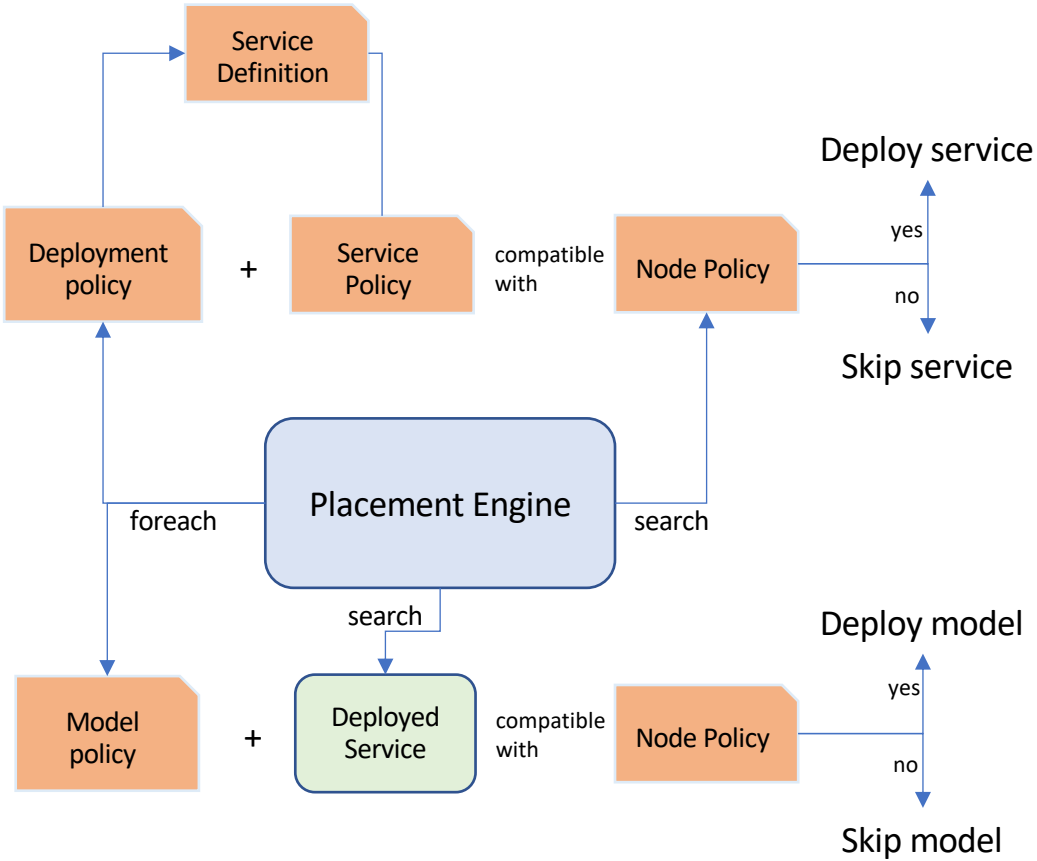
Agent and Service Deployment (Device)



Agent and Service Deployment (Edge Cluster)

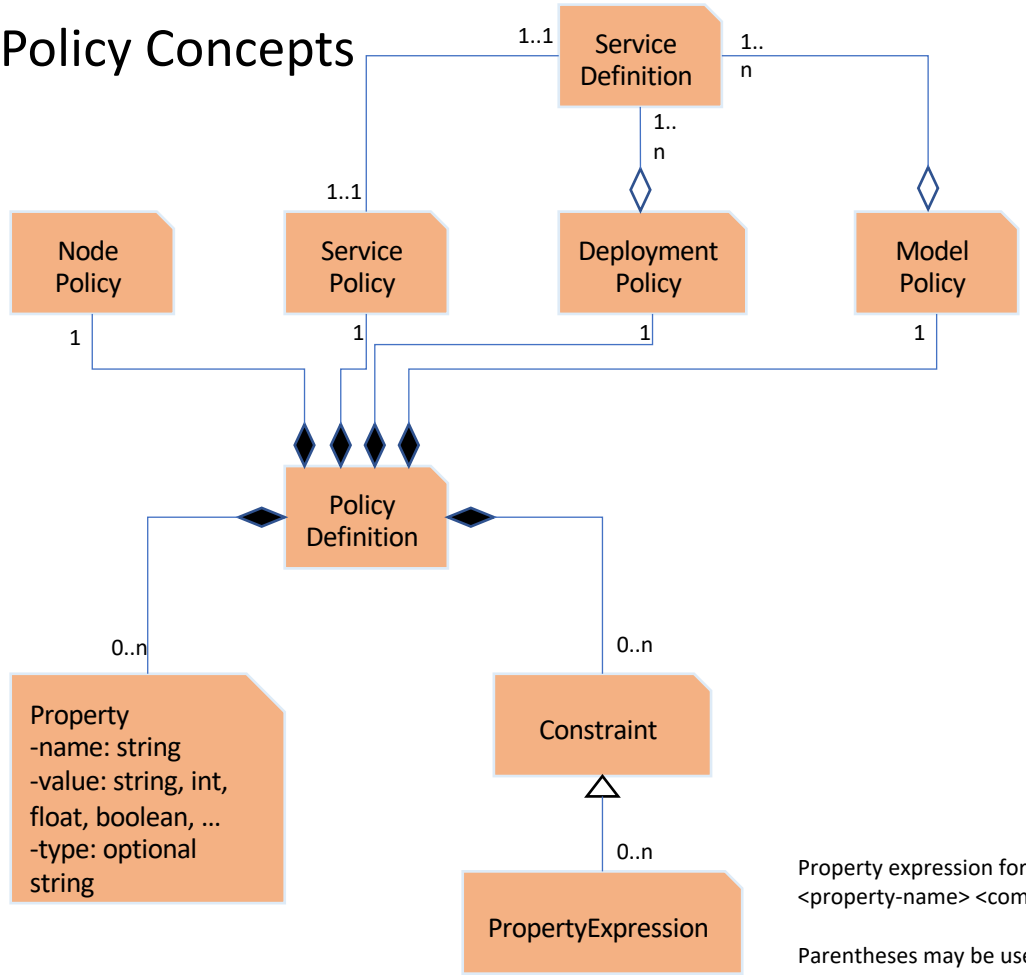


Service Placement



- The placement engine runs in the Agbot.
- Continually searches for nodes that are compatible with existing deployment policies.
- Continually searches for deployed services that have model policy which is compatible with the node where the service is running.
- Deployment policy is logically ‘AND’ed with service policy to form the full set of policy definitions.
- Service policy is optional, specified by the service author.
- Compatibility is bidirectional:
  - the node policy satisfies the combined deployment policy
  - The combined deployment policy satisfies the node policy

Policy Concepts

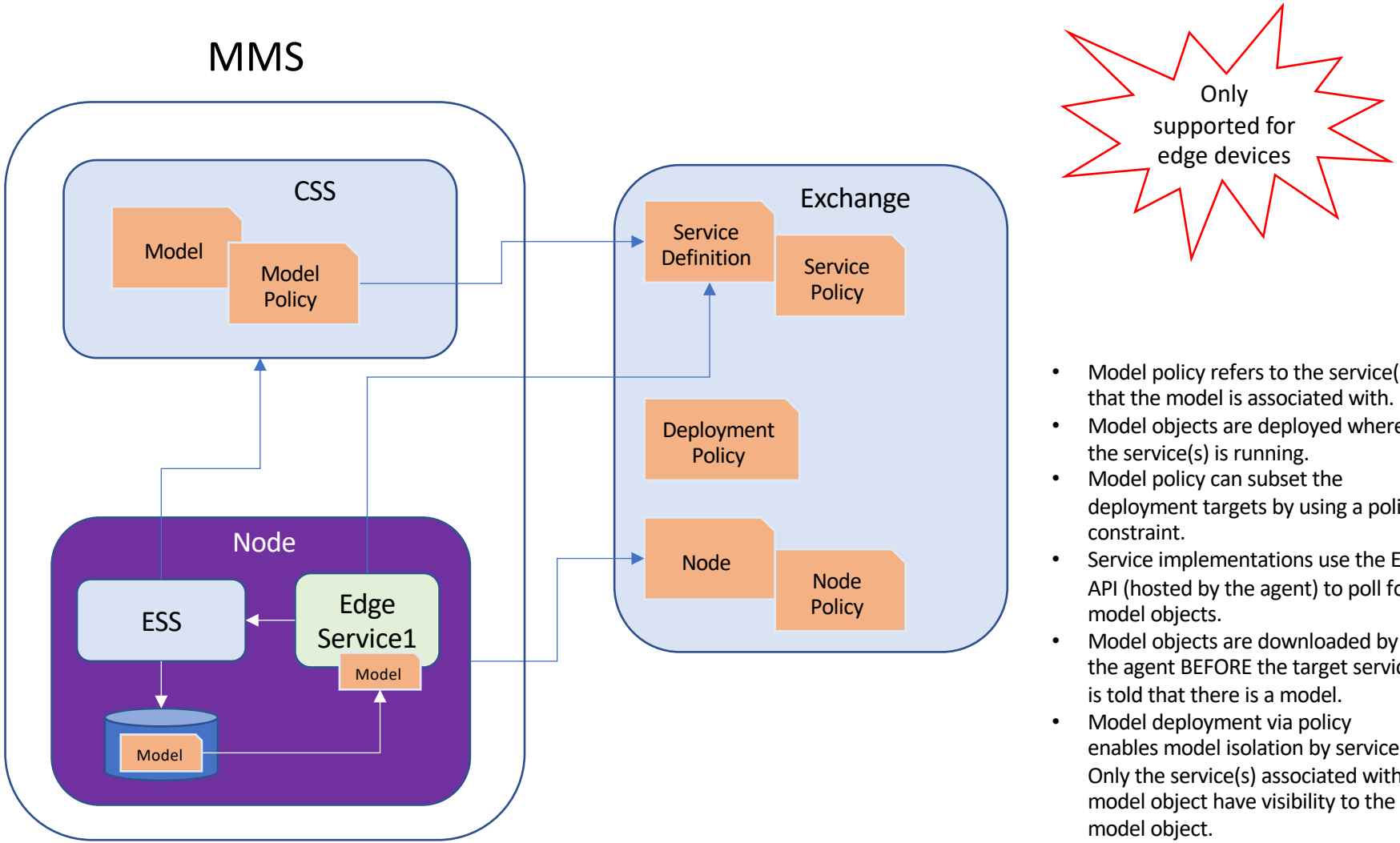


In English:

- A policy definition is composed of 0 or more properties, and 0 or more constraints.
- A Property has name, type (optional), and value.
- A Constraint has zero or more property expressions.
- A node, service, deployment and model policy each have 1 policy definition.
- A service definition and a service policy are independent entities with a 1..1 relationship.
- A deployment policy and a model policy are independent entities that each refer to a service definition.
- A deployment policy may refer to more than 1 version of a service to denote service rollback instructions.
- A model policy may refer to more than 1 service definition, for each service the model is going to be deployed with.

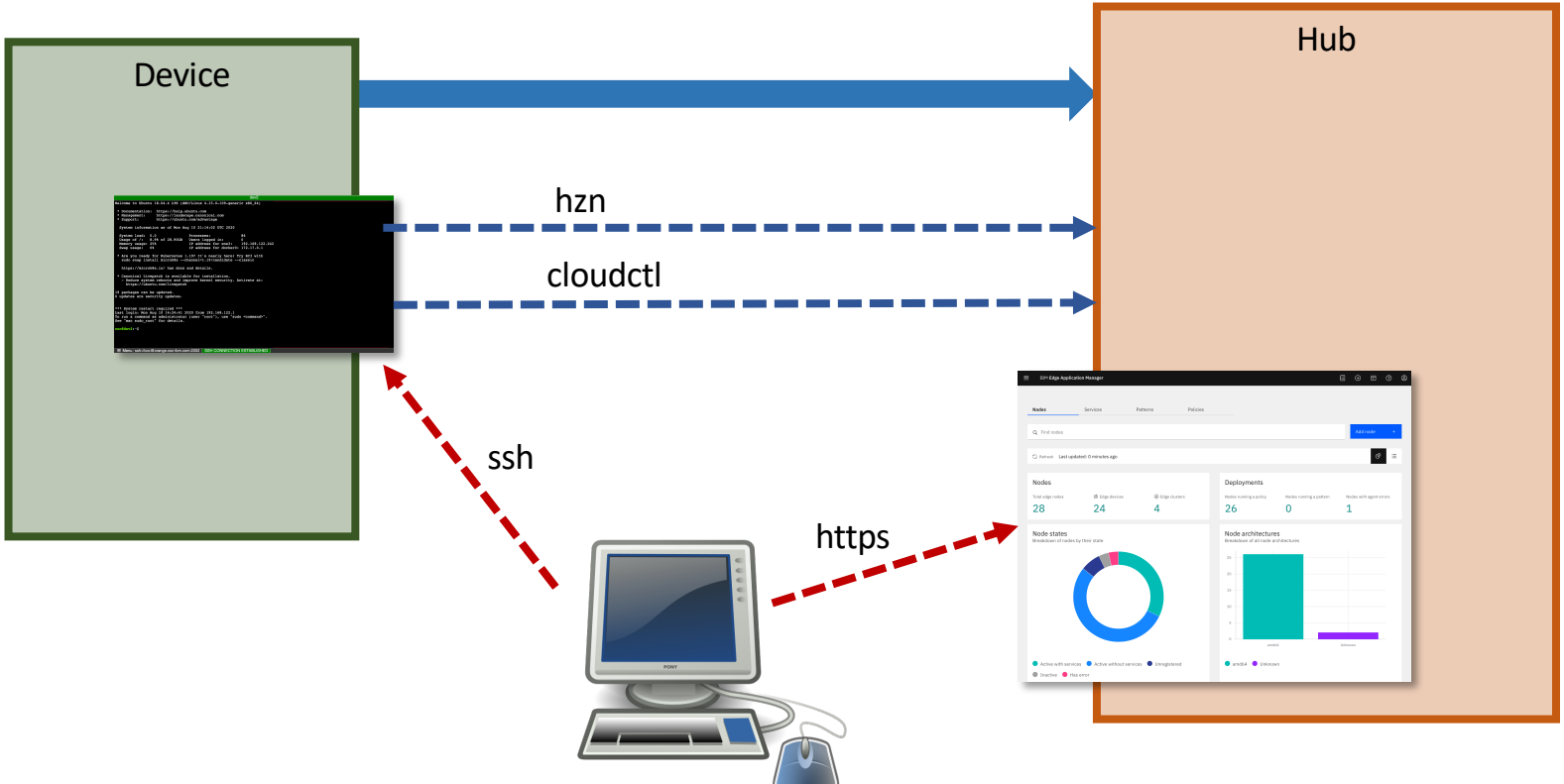
Property expression form:
<property-name> <comparison-operator> <property-value> [<boolean-operator>] …

Parentheses may be used to provide precedence order.
Comparison operator depends on property type.



- Model policy refers to the service(s) that the model is associated with.
- Model objects are deployed where the service(s) is running.
- Model policy can subset the deployment targets by using a policy constraint.
- Service implementations use the ESS API (hosted by the agent) to poll for model objects.
- Model objects are downloaded by the agent BEFORE the target service is told that there is a model.
- Model deployment via policy enables model isolation by service. Only the service(s) associated with a model object have visibility to the model object.

Lab Setup

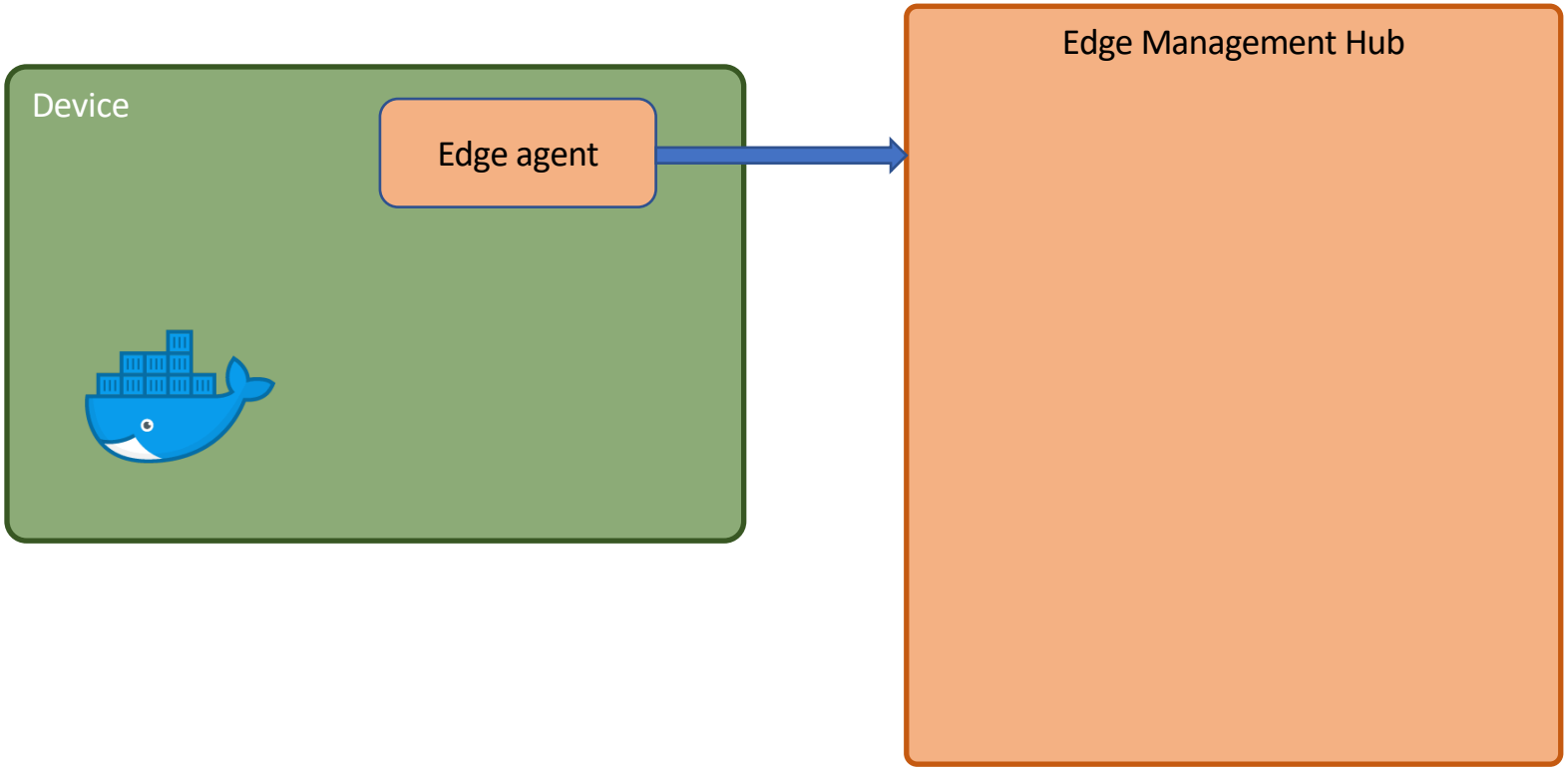


Lab Overview

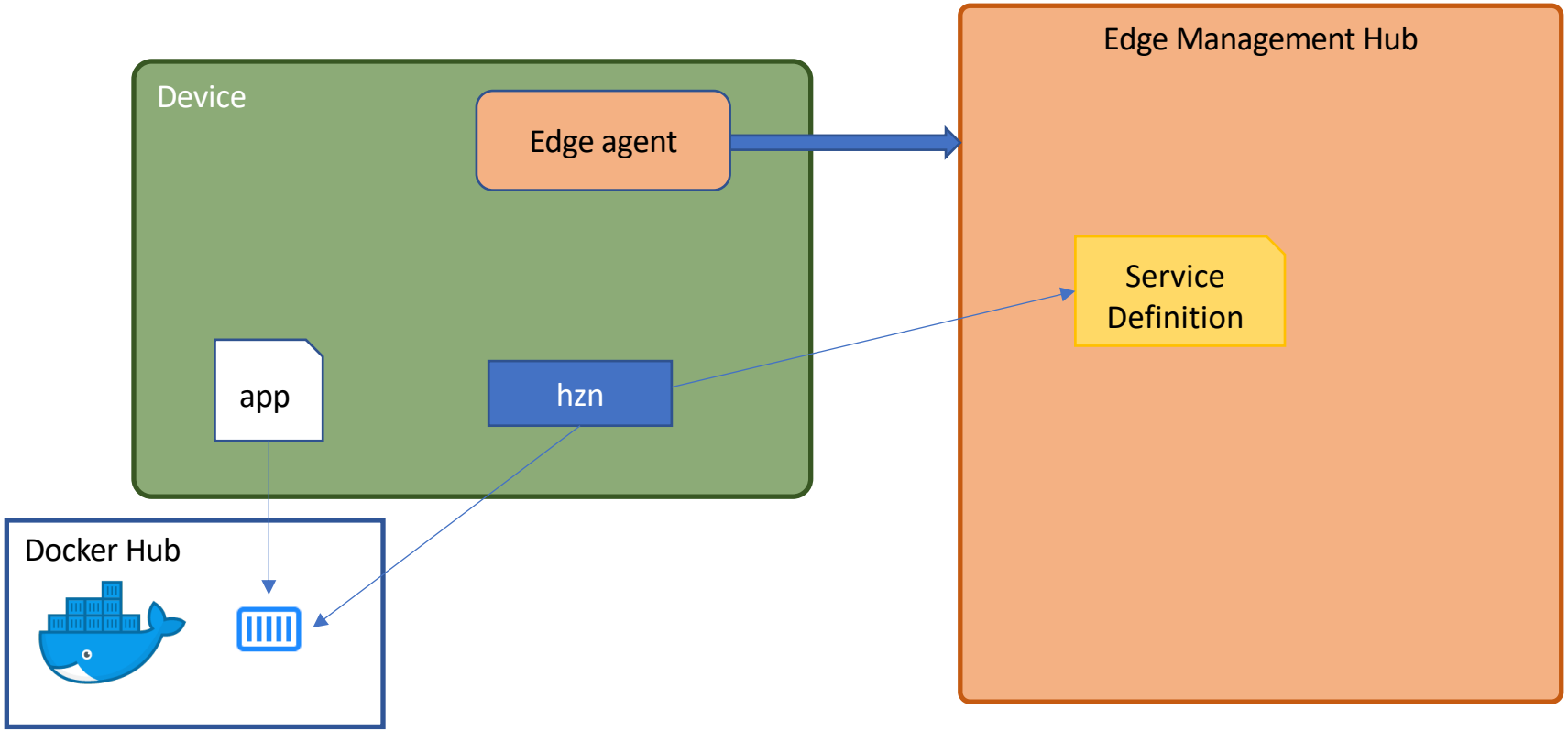
This lab highlights the deployment of a model to an edge device. The sample app is a simple image analysis app using TensorFlow JS. The basic steps in this lab are:

- Install agent on edge device (virtual device in this lab)
- Build and Publish an edge service
- Publish a policy
- Register the edge device
- Validate the deployment of the model to the edge device.
- Update Model
- Verify change in app

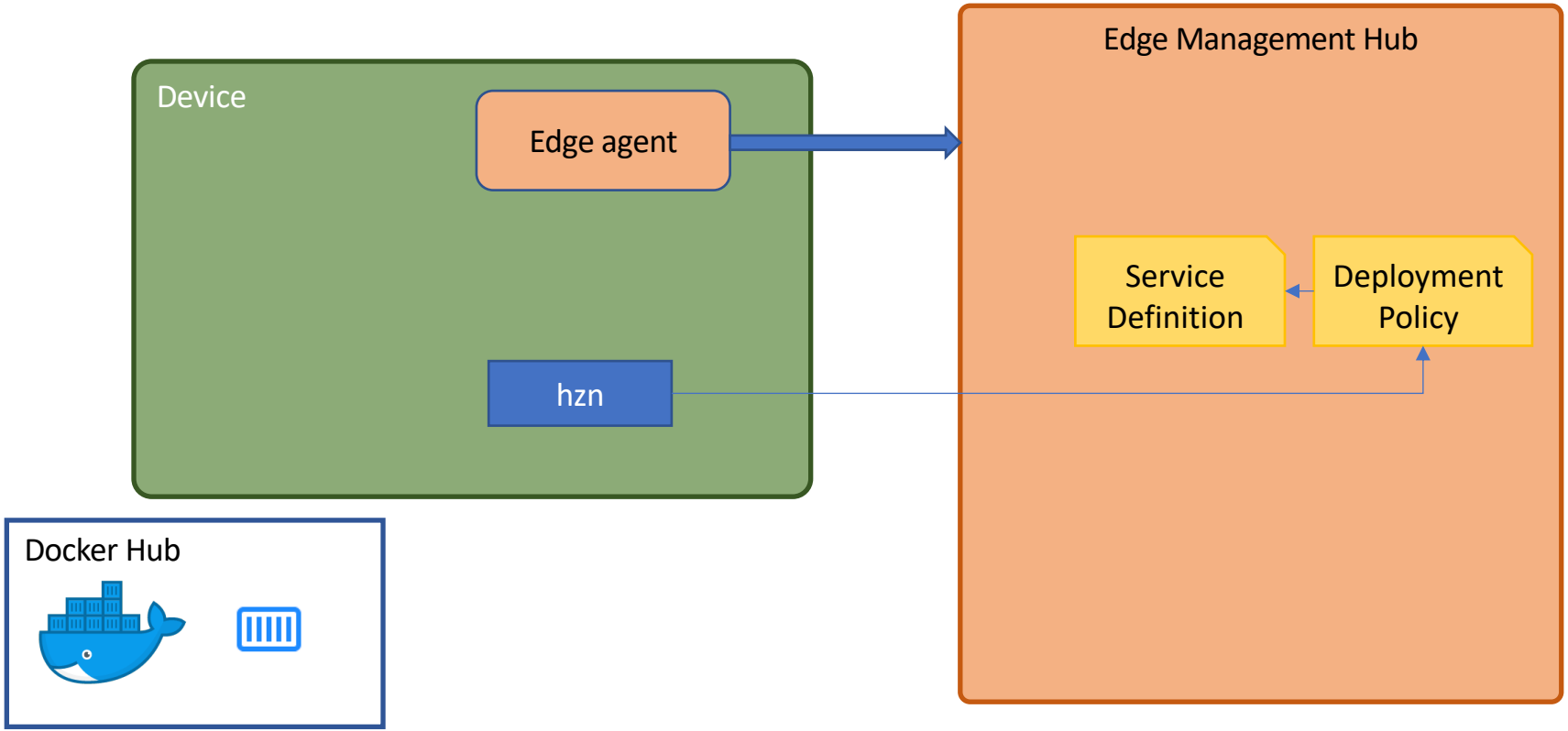
1. Install Agent



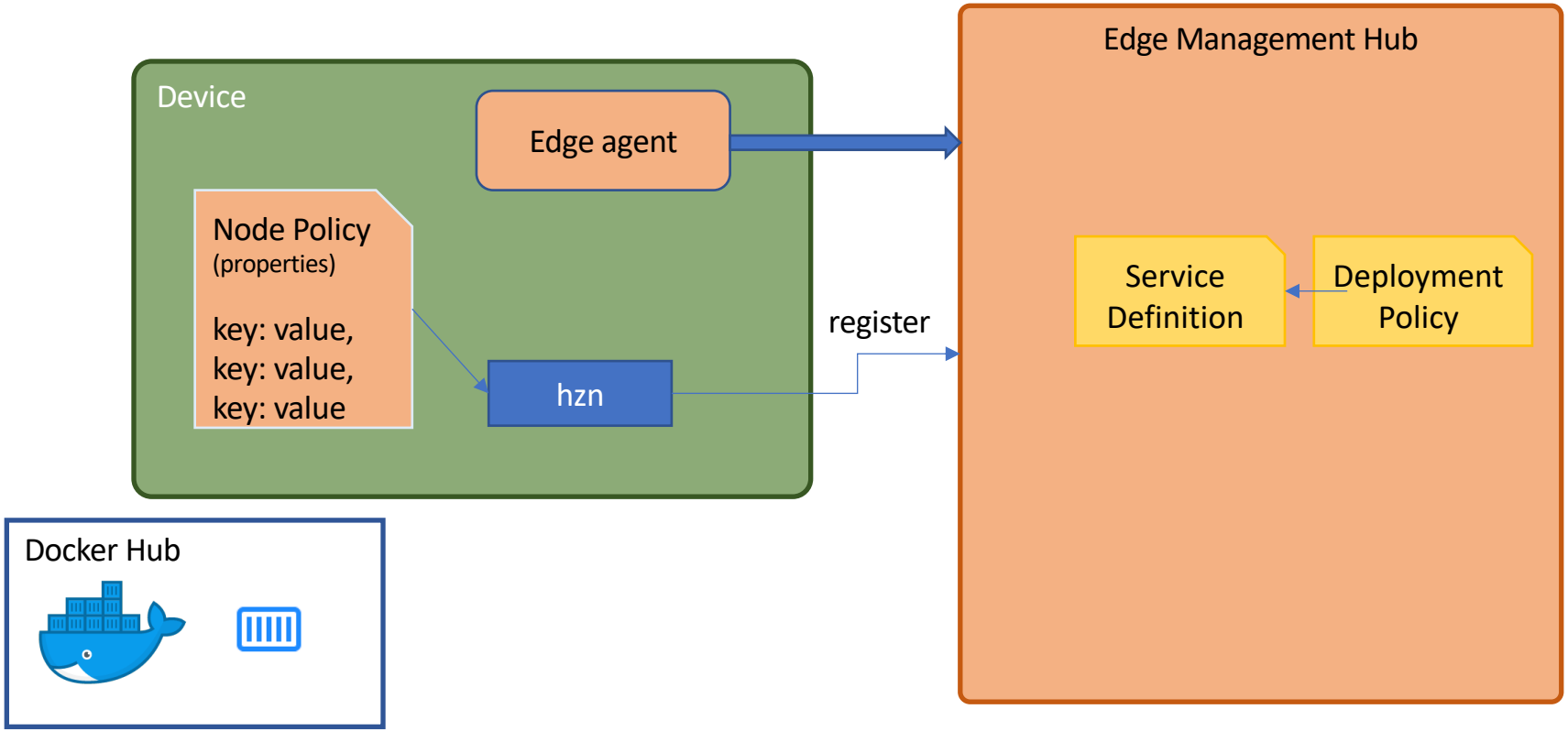
2. Build, Publish Service



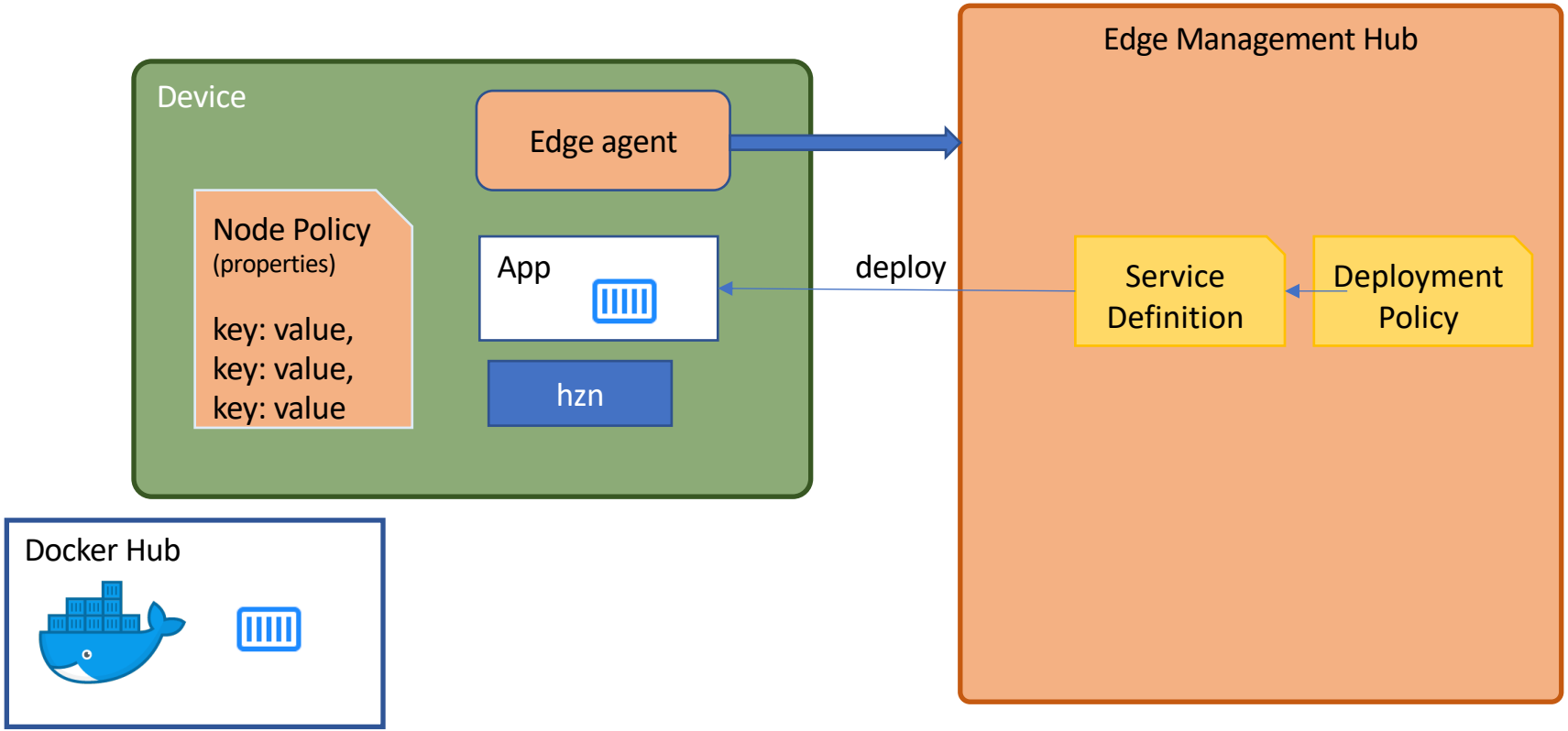
3. Publish Policy



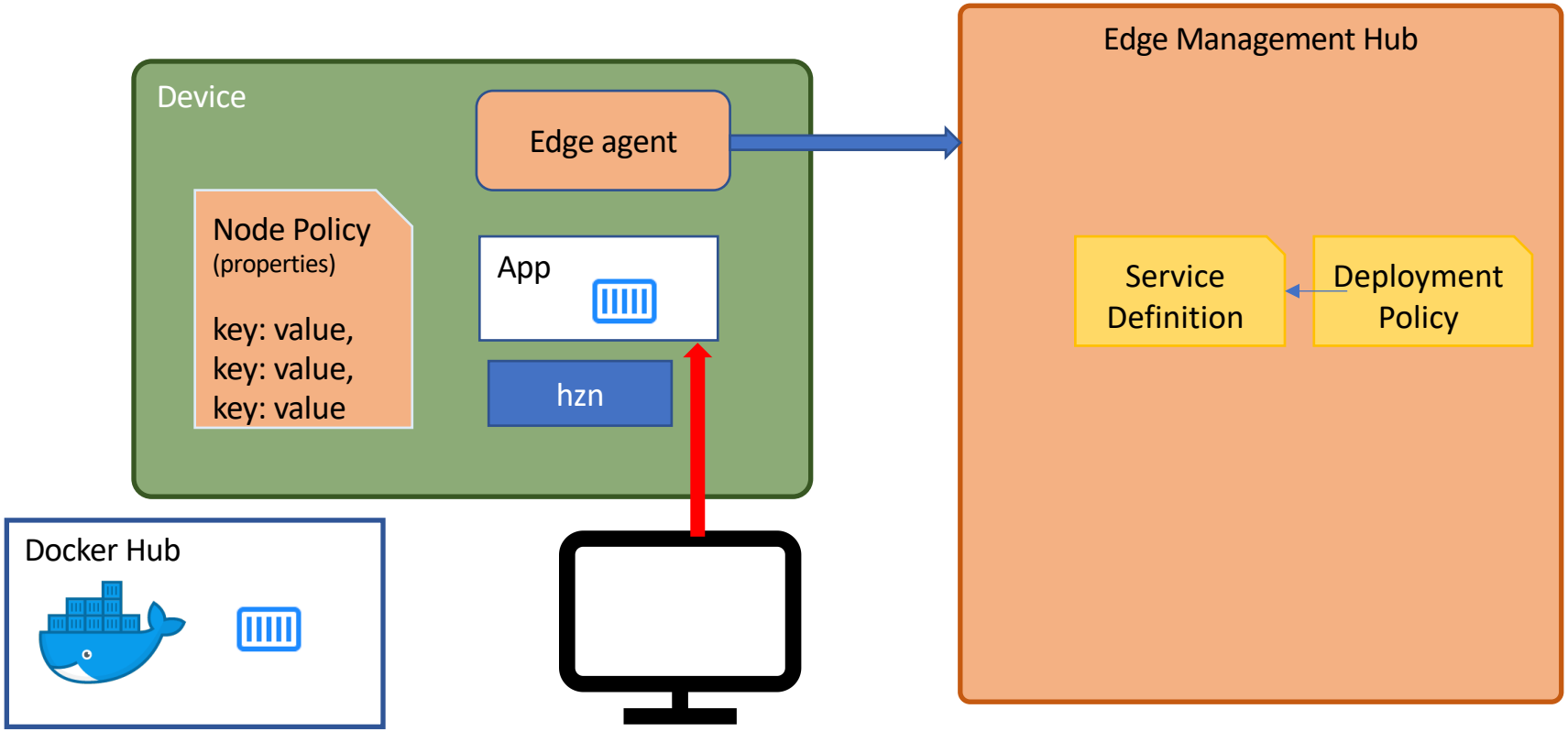
4. Register Device



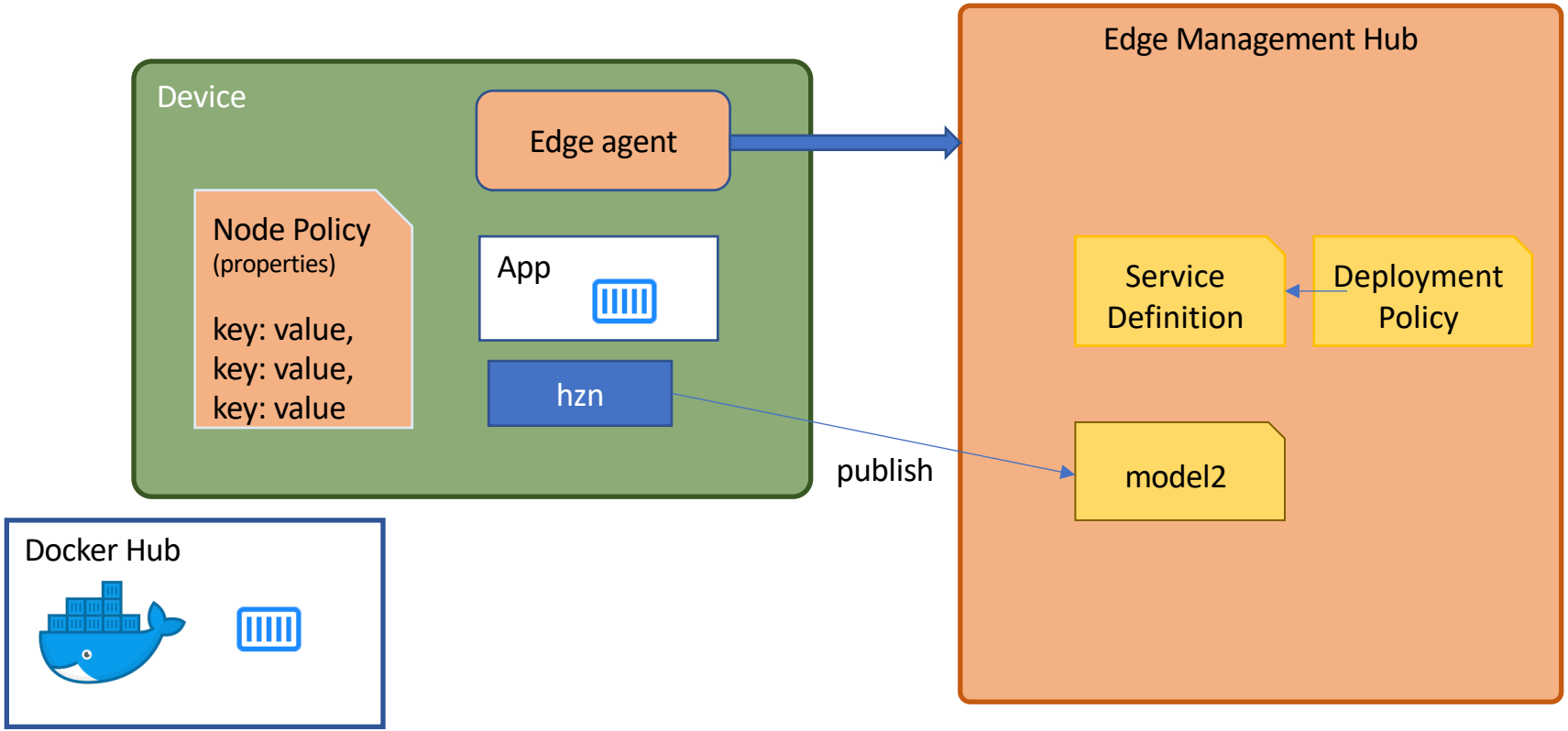
Service (App) gets deployed to device



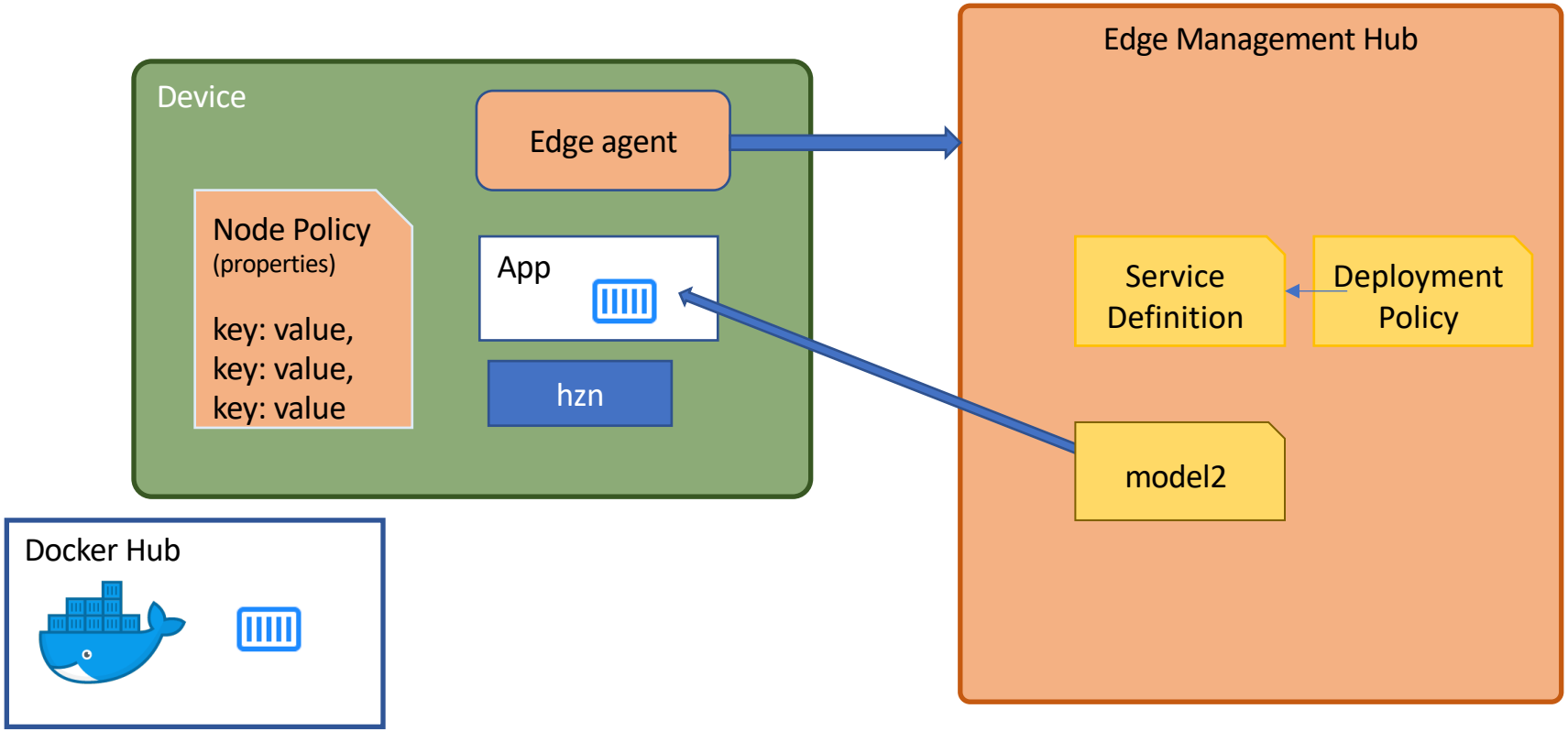
5. Validate App



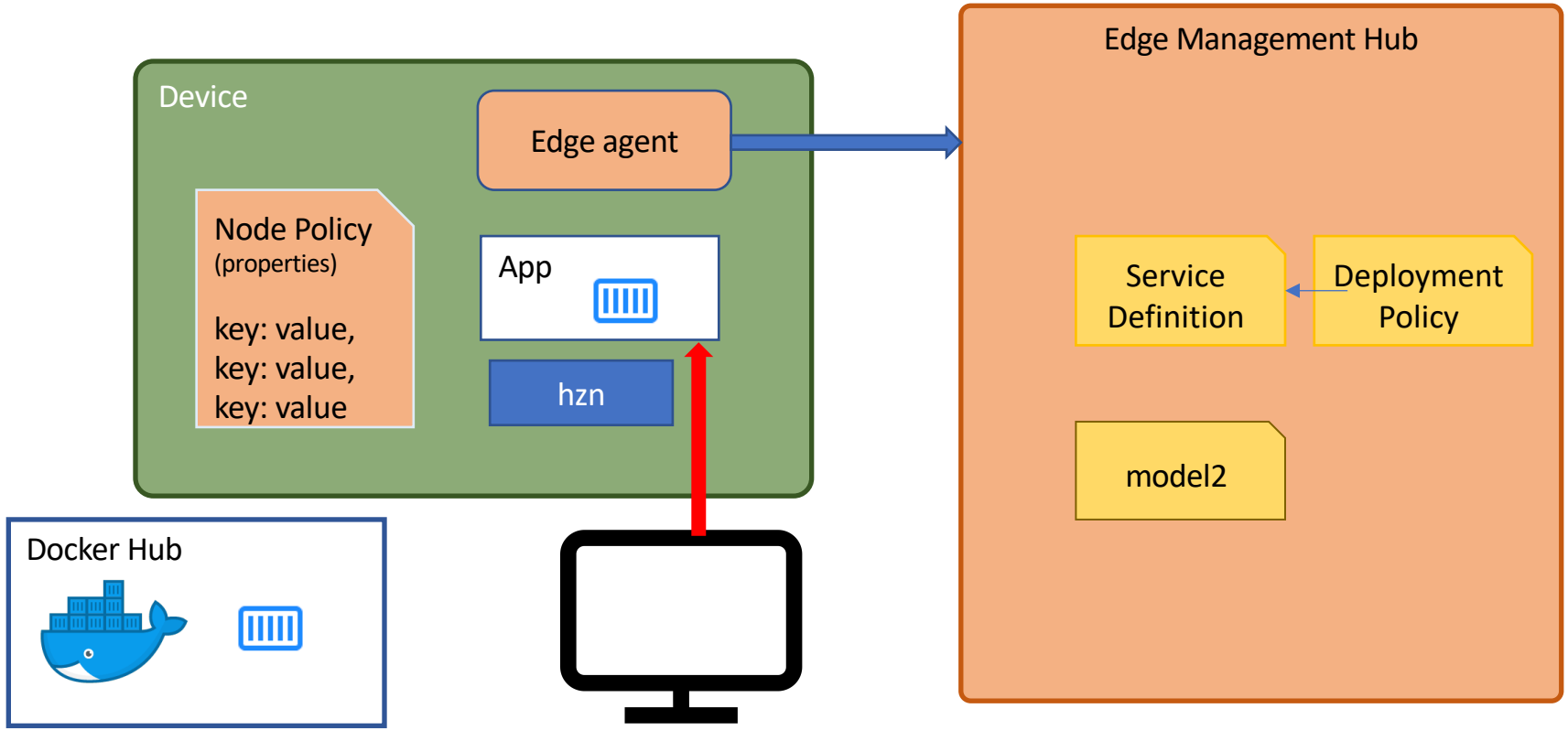
6. Update Model



Application sees new model and pulls it



5. Verify updated model



Thank You