AGENDA

Platform Architecture
Service Component Pre-requisite
Onboarding Flow
Catalog & Service Template
Control Loop Configuration Flow
R3 Focus
DCAE PLATFORM EVOLUTION

Feb’2017: DCAE1.0
- 16 repository
- 5 VM

Nov’2017: DCAEGEN2
- 11 repository
- 14 VM Based

May’2018: DCAEGEN2.X (K8S)
ONAP DCAE Architecture (R1)

DCAE Platform components
- Cloudify (& Plugins)
- Consul
- Config Binding Service (CBS)
- Service change Handler (SCH)
- Policy Handler (PH)
- Deployment Handler (DH)
- Inventory API
- CDAP Broker
- PGaaS
- CDAP Cluster
- Docker Hosts
R2 Platform Updates

- Utilizing OOM Kubernetes cluster
- No Hadoop cluster deployment
- Containerized Cloudify
- Using Helm chart for Bootstrap, Cloudify Manager, Redis
- All platform (and services) deployment through Cloudify (custom K8 plugin)
- PGaaS deployment through OOM and Cloudify
- OOM Deployed Consul will be utilized
- MSB (Api and websocket)
- Policy Flow enablement
Service Components

- Modular
- Well-defined interfaces
- Light weight
- Scale quickly
- Technology freedom
- Deployment Flexibility
Service Component Requirements

- Overall functionality
- Spec creation and validation through dcae_cli utility
- State synchronization
- Utilize ConfigBindingService API’s
- Enforcement of Policy Decisions
Component specification

- Defining input and output structure
- Topic Keys
- Configuration
  - Design-time
  - Deployment-time
  - Policy editable
- Policy Model
- Filesystem mapping
- Port mapping
- Image specification
Onboarding Flow (Target)

This is the flow for onboarding Service components. All components get onboarded and made available to SDC template designer.

Loop (for all DCAE Components):
1. Creates JSON schema specifying the metadata required by this component.
2. Inputs
3. Outputs
4. Configuration

2. Verifies the spec and deployment (if CBS/Consul are setup).
3. Use TOSCA tool to create artifact.
4. TOSCA Artifact created.
5. Imports into SDC to onboard their DCAE component.
6. Saved into catalog.

Component Developer
DCAE_Comp JSON
DCAE_CLI
TOSCA Tool
MODELS/BLUEPRINTS
SDC
SDC Catalog
DCAE(SCH)
CLAMP
Policy
Component spec (Example)

```json
{
  "self": {
    "version": "1.2.1",
    "name": "dcae-ves-collector",
    "description": "Collector for receiving VES events through restful interface",
    "component_type": "docker"
  },
  "streams": {
    "subscribes": [],
    "publishes": [
      {
        "format": "VES_specification",
        "version": "5.20.4",
        "type": "message router",
        "config_key": "ves-fault"
      }
    ],
    "parameters": [
      {
        "name": "collector.service.port",
        "value": -1,
        "description": "standard http port"
      },
      {
        "name": "collector.service.secure.port",
        "value": 8443,
        "description": "secure port",
        "designer_editable": true
      }
    ],
    "auxiliary": {
      "healthcheck": {
        "type": "https",
        "interval": "15s",
        "timeout": "1s",
        "endpoint": "/healthcheck"
      },
      "volumes": [
        {
          "container": {"bind": "/opt/app/dcae-certificate"},
          "host": {
            "path": "/opt/app/dcae-certificate"
          }
        },
        {
          "container": {"bind": "/opt/app/VESCollector/logs"},
          "host": {
            "path": "/opt/logs/DCAE/VESCollector/logs"
          }
        }
      ],
      "ports": [
        "8443:8443"
      ],
      "artifacts": [
        {
          "type": "docker image",
          "url": "nexus.onap.org:10001/onap/org.onap.dcae.gen2.collectors.ves.vescollector:1.2.1"
        }
      ]
    }
  }
}
```
Blueprint (Example)

Tosca_definitions_version: cloudify_dsl_3_3

description:

This blueprint will install the Ves collector and provision the needed message router topics.

imports:
- http://www.getcloudify.org/spec/cloudify/3.4/types.yaml
- https://NEXUS_REPO_HOST:8443/repository/NEXUS_REPO/type_files/docker/2.2.6/node-type.yaml
- https://NEXUS_REPO_HOST:8443/repository/NEXUS_REPO/type_files/relationship/1.0.8/node-type.yaml
- http://NEXUS_REPO_HOST:8081/repository/NEXUS_REPO/type_files/dnmap/dnmap_mr.yaml

inputs:

- service_id:
  description: Unique id used for an instance of this DCAE service. Use deployment id
  default: "foo_bar"
- location_id:
  default: "foo_bar"
- topic00_username:
  default: "atc9s"
- topic00_password:
  default: "secret"
- topic00_client_role:
  default: "cmist.dcm.member"

node_templates:

- topic00:
  type: dcae.nodes.Topic
  properties:
    topic_name: sec-fault-unsecure
- topic01:
  type: dcae.nodes.Topic
  properties:
    topic_name: sec-measurement-unsecure
- topic02:
  type: dcae.nodes.Topic
  properties:
    topic_name: sec-heartbeat-unsecure
- topic03:
  type: dcae.nodes.Topic
  properties:
    topic_name: sec-other-unsecure
- component00:
  type: dcae.nodes.DockerContainerForComponentsUsingDnmap

properties:

  service_component_type:
    "dcae-controller-ves-collector"
  service_id:
    get_input: service_id
  location_id:
    get_input: location_id
  application_config:
    collector.keystore.passwordfile: "/opt/app/dcae-certificate/.password"
    collector.service.security.port: -1
    truststore.maxthreads: "200"
    collector.keystore.file.location: "/opt/app/dcae-certificate/keystore.jks"
   _RETURNTRANSFER: 6
    collector.service.port: 8080
  streams_publishes:
    sec_fault_unsecure:
      naf_password: 
        get_input: topic00_naf_password
      dmap_info: "<topic00>
  services_calls:
    collector.dockerstreamid: fault-sec_fault,route-sec_to-hp|syslog-sec_syslog|heartbeat-sec_heartbeat
    header.authflag: user1|user2,base64enc|base64enc|base64enc|base64enc
  streams_subscribes: ()
  collector.InputQueue.maxPending: 9999
  collector.schema.file: "/etc/CommEventFormat_27.2.json"
  collector.keystore.alias: dynamically generated
  Image:

  NEXUS_REPO_HOST:18443/dcae-dev-raw/dcae-controller-ves-collector:1.1
  docker_config:
    healthcheck:
      type: "http"
      interval: "15s"
      timeout: "1s"
      port: 8443
  volumes:
    - container:
      bind: "/opt/app/dcae-certificate"
      host: "/path"/opt/app/dcae-certificate
    - container:
      bind: "/opt/app/VESCollector/logs"
      host: "/path"/opt/Logs/DCAEVES/VESCollector/logs
  streams_publishes:
    - name: topic00
      location: 
        get_input: topic00_location
        client_role: 
          get_input: topic00_client_role
        type: message_router
    - name: topic01
      location: 
        get_input: topic01_location
        client_role: 
          get_input: topic01_client_role
        type: message_router
    - name: topic02
      location: 
        get_input: topic02_location
        client_role: 
          get_input: topic02_client_role
        type: message_router
    - name: topic03
      location: 
        get_input: topic03_location
        client_role: 
          get_input: topic03_client_role
        type: message_router
    - name: topic04
      location: 
        get_input: topic04_location
        client_role: 
          get_input: topic04_client_role
        type: message_router

  docker_host:
    type: dcae.nodes.SelectedDockerHost
    properties:
      location_id:
Amsterdam Flows

- Virtual Firewall
- Virtual DNS
- vCPE
- VoLTE
Post Beijing Flows

PRH → Holmes → Heartbeat Service → TCA

VNF/PNF → VES Collector → VES Mapper

SNMP Collector
Onboarding, Service Design & Deployment Flow (Target view)
Configuration Flow

1 → Policy Create/update/delete
2 → Policy distribution
3 → Notification on policy change
4 → Get updated policy list
5 → If DCAE scope is updated; get latest policy config and notify deployment handler with deltas
6 → Identify components matching policy received.
7 → Update list of policy into consul KV under SCN folder
8 → Notification service component on policy update
9 → Invoke CBS API (<CBS>/service_component_all/) to fetch latest policy
10 → Fetches config under SCN folder and returns json object
11 → Receives updated policy and enforce new rules.
Policy Model/Configuration (sample)
DCAE platform functions after onboarding

- Deployment of service components based on operation requirement and business need.
- Retrieval of blueprint via SDC distribution and store in DCAE inventory.
- Retrieval of configuration policy and policy updates from Policy and making available in Consul.
- Service component deployment (and undeploy) support through CLAMP.
- Registration (deregistration) of components in Consul and/or Kubernetes platform for health check and application state.
- Service registration (deregistration) with MSB when applicable.
- DMaaP resource provisioning when applicable.
- Provides configuration api and policy configuration via CBS.
- Components scaling.
R3 Focus Areas

- SDC Design Studio integration for onboarding and generation of K8S blueprint
- Expanding Service Component portfolio
- Service component Policy modelling
- Enhanced S3P level support
- AAF Integration
- Integration with newer Analytics Platform (PNDA/Flink)
Questions?