R5 El Alto proposal for policy flows between the new PDP and DCAE component

Requirements on **PDP -** implement the **pub-sub for multiple policies per each component**

1. maintain the database of subscribers with the
    a. list of generic **policy-filters** (each policy-filter is the **resource** from the request json to **/decision/v1** API) TO BE DELETED - refer to Dublin Documentation#3.4PolicyDecisionAPI-GettingPolicyDecisions per component - subscriber
    b. **subscriber_id** should be able to globally and uniquely identify the component instance like "**policies_DCAE_tca_<service-component-name>**".
        i. where **<service-component-name>** uniquely identifies the component in DCAE
    c. **subscriber_topic** for Message Router of DMaaP that
        i. either uniquely identifies the component instance like "**policies_DCAE_tca_<service-component-name>**".
        ii. or can be shared by all the components in DCAE like "**policies_DCAE**". In this case the field ONAPInstance="**<service-component-name>**" can be used to identify the component instance
    d. option to return the policies on subscribe request
    e. option to have the matching /decision/v1 API that sends multiple policies per multiple policy-filters in a single query
2. on policy **push/delete** from PAP and on the **insert-update of the subscriber** record
    a. select **all subscribers** that match to the pushed/deleted policies by **any policy-filter**
    b. for each affected **subscriber** retrieve **all the latest snapshot of policies**
    c. **increment** policy_update_seq value
    d. **notify** each **subscriber_topic** separately with the latest snapshot of policies by sending the message to Message Router of DMaaP with the topic=**subscriber_topic**
3. there might be a need for some logic to identify the *stale subscriptions* by checking with the Message Router of DMaaP on the timestamps of the latest delivered/undelivered message per ach topic PDP has the subscription on

---

Requirements on **Message Router of DMaaP** - no changes to DMaaP seems to be required

1. **persistently** (up to 7 days) deliver all the policy-update notifications per **subscriber_topic**
2. component will listen for the **subscriber_topic** of the policy-update notification from MR of DMaaP with long-collect-polling time like 15 seconds to grab the push notification

---

Alternatives for requirements on **component**

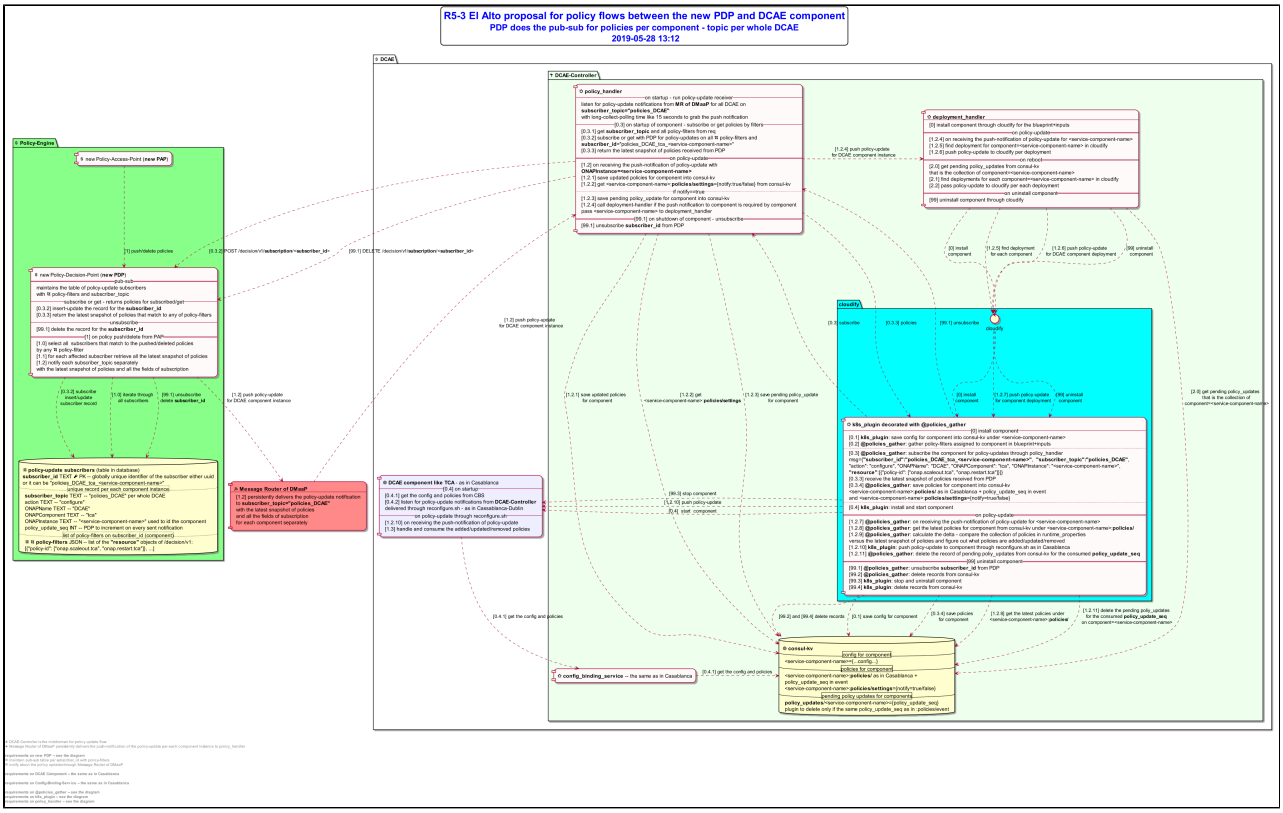
Alternative	impact on component	complexity for DCAE-Controller	comments
5	**highest** - component to do the subscription and listen for policy-updates	**minimal** - only deliver the policy-filters and subscriber-topic for subscription	
5-1	less - component to listen for policy-updates	medium - k8s_plugin+policy-handler to do the subscription for policy-updates with PDP	drawback - no mechanism to bring the policies on the reboot of the component
5-2	less - component to listen for policy-updates	medium-high - config-binding-service+policy-handler to do the subscription for policy-updates with PDP	component can poll config-binding-service to get the latest policies either periodically or on reboot of the component

5-3	**no impact** - component as in Casablanca	**highest** - flow is comparable to Casablanca but is based on DMaaP instead of the web-socket and is per subscription for the component instead of matching policies to policy-filters in policy-handler	

Option - 5-3 - no impact requirements on **component** - the plugin and the policy-handler to do the subscription with PDP for the component and deliver the push notification about the policy-update the same way as in Casablanca

1. single topic shared by all components of DCAE like “policies_DCAE”
2. ONAPInstance value to identify the component instance in DCAE
3. no impact on component - the same behavior as in Casablanca besides the new structure of the policy-body
4. the policy-flow in DCAE-Controller resembles Casablanca with
    a. DMaaP instead of web-socket
    b. less communication chat inside DCAE-Controller
    c. and ONAPInstance matching to component deployment instead of policy-handler doing the rematch of policies to policy-filters /ids and deployment-handler to match the policy-filters/ids to component deployments

See inside the diagram for more details, data structures, and flow steps



r5-3_proposed_policy_update_flow

```
@startuml r5-3_proposed_policy_update_flow
allowmixing
scale 4096*4096

skinparam title {
    FontSize 24
    FontColor Blue
    FontStyle Bold
    BorderRoundCorner 15
    BorderThickness 2
}
```

```
skinparam roundcorner 20

skinparam component {
    BackgroundColor Snow
}

skinparam note {
    FontColor Black
    BackgroundColor azure
}

title = R5-3 El Alto proposal for policy flows between the new PDP and DCAE component \n PDP does the pub-sub for policies per component - topic per whole DCAE \n %date[yyyy-MM-dd HH:mm]%

package "<&dollar> Policy-Engine" as policy_engine #88ff88 {
    component "<&dollar> new Policy-Access-Point (<b>new PAP</b>)" as PAP
    component "<&dollar> new Policy-Decision-Point (<b>new PDP</b>)\n==pub-sub==\n maintains the table of policy-update subscribers \nwith <&crop> policy-filters and subscriber_topic\n--subscribe or get - returns policies for subscribed/get--\n[0.3.2] insert-update the record for the <b>subscriber_id</b>\n[0.3.3] return the latest snapshot of policies that match to any of policy-filters\n--unsubscribe--\n[99.1] delete the record for the <b>subscriber_id</b>\n==[1] on policy push/delete from PAP==\n[1.0] select all  subscribers that match to the pushed/deleted policies \n by any <&crop> policy-filter\n[1.1] for each affected subscriber retrieve all the latest snapshot of policies \n[1.2] notify each subscriber_topic separately \nwith the latest snapshot of policies and all the fields of subscription" as PDP

    database "<&list> <b>policy-update subscribers</b> (table in database)\n<b>subscriber_id</b> TEXT <&key> PK -- globally unique identifier of the subscriber either uuid \n or it can be "policies_DCAE_tca_<service-component-name>"\n..unique record per each component instance..\n<b>subscriber_topic</b> TEXT -- "policies_DCAE" per whole DCAE\n action TEXT -- "configure"\nONAPName TEXT -- "DCAE"\nONAPComponent TEXT -- "tca"\nONAPInstance TEXT -- "<service-component-name>" used to id the component\npolicy_update_seq INT -- PDP to increment on every sent notification\n..list of policy-filters on subscriber_id (component).. \n<&list> <&crop> <b>policy-filters</b> JSON -- list of the <b>"resource"</b> objects of /decision/v1: \n [{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}, ...]" as policy_update_subscribers
}

package "<&dollar> DCAE" as DCAE {
    package "<&signpost> DCAE-Controller" as DCAE_Controller #eeffee {
        component "<&aperture> <b>deployment_handler</b>\n--\n[0] install component through cloudify for the blueprint+inputs\n==on policy-update==\n[1.2.4] on receiving the push-notification of policy-update for <service-component-name>\n[1.2.5] find deployment for component=<service-component-name> in cloudify\n [1.2.6] push policy-update to cloudify per deployment\n==on reboot==\n[2.0] get pending policy_updates from consul-kv \n that is the collection of component=<service-component-name>\n[2.1] find deployments for each component=<service-component-name> in cloudify\n[2.2] pass policy-update to cloudify per each deployment\n==on uninstall component==\n[99] uninstall component through cloudify" as deployment_handler

        package "<b>cloudify</b>" as cloudify_server #00ffff {
            control cloudify

            component "<&aperture> <b>k8s_plugin decorated with @policies_gather</b>\n--[0] install component--\n[0.1] <b>k8s_plugin</b>: save config for component into consul-kv under <service-component-name>\n[0.2] <b>@policies_gather</b>: gather policy-filters assigned to component in blueprint+inputs\n..\n [0.3] <b>@policies_gather</b>: subscribe the component for policy-updates through policy_handler \n msg={<b>"subscriber_id":"policies_DCAE_tca_<service-component-name>"</b>,  <b>"subscriber_topic":"policies_DCAE"</b>, \n"action": "configure", "ONAPName": "DCAE", "ONAPComponent": "tca", "ONAPInstance": "<service-component-name>", \n<b>"resource"</b>:[{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}]}\n[0.3.3] receive the latest snapshot of policies received from PDP\n[0.3.4] <b>@policies_gather</b>: save policies for component into consul-kv \n<service-component-name>:<b>policies/</b> as in Casablanca + policy_update_seq in event\n and <service-component-name>:<b>policies/settings</b>={notify=true/false}\n..\n[0.4] <b>k8s_plugin</b>: install and start component\n==on policy-update==\n[1.2.7] <b>@policies_gather</b>: on receiving the push-notification of policy-update for <service-component-name>\n[1.2.8] <b>@policies_gather</b>: get the latest policies for component from consul-kv under <service-component-name>:<b>policies/</b>\n[1.2.9] <b>@policies_gather</b>: calculate the delta - compare the collection of policies in runtime_properties \n versus the latest snapshot of policies and figure out what policies are added/updated/removed\n[1.2.10] <b>k8s_plugin</b>: push policy-update to component through reconfigure.sh as in Casablanca\n[1.2.11] <b>@policies_gather</b>: delete the record of pending poliy_updates from consul-kv for the consumed <b>policy_update_seq</b>\n==[99] uninstall component==\n[99.1] <b>@policies_gather</b>: unsubscribe <b>subscriber_id</b> from PDP\n[99.2] <b>@policies_gather</b>: delete records from consul-kv\n[99.3] <b>k8s_plugin</b>: stop and uninstall component\n[99.4] <b>k8s_plugin</b>: delete records from consul-kv" as k8s_plugin
        }
```

```
        component "<&aperture> <b>config_binding_service</b> -- the same as in Casablanca" as
config_binding_service

        database "<&target> <b>consul-kv</b>\n--config for component--\n<service-component-name>={...
config...}\n--policies for component--\n <service-component-name>:<b>policies/</b> as in Casablanca +
\npolicy_update_seq in event\n <service-component-name>:<b>policies/settings</b>={notify=true/false}\n--
pending policy updates for components--\n<b>policy_updates</b>/<service-component-name>={policy_update_seq}
\n plugin to delete only if the same policy_update_seq as in :policies/event" as consul_kv

        component "<&aperture> <b>policy_handler</b>\n--on startup - run policy-update receiver--\nlisten
for policy-update notifications from <b>MR of DMaaP</b> for all DCAE on \n<b>subscriber_topic="policies_DCAE"
</b>\n with long-collect-polling time like 15 seconds to grab the push notification\n--[0.3] on startup of
component - subscribe or get policies by filters--\n[0.3.1] get <b>subscriber_topic</b> and all policy-
filters from req\n[0.3.2] subscribe or get with PDP for policy-updates on all <&crop> policy-filters and
\n<b>subscriber_id</b>="policies_DCAE_tca_<service-component-name>"\n[0.3.3] return the latest snapshot of
policies received from PDP\n==on policy-update==\n[1.2] on receiving the push-notification of policy-update
with \n <b>ONAPInstance=<service-component-name></b>\n[1.2.1] save updated policies for component into
consul-kv\n[1.2.2] get <service-component-name>:<b>policies/settings</b>={notify:true/false} from consul-
kv\n..if notify==true..\n[1.2.3] save pending policy_update for component into consul-kv\n[1.2.4] call
deployment-handler if the push notification to component is required by component \n pass <service-component-
name> to deployment_handler \n==[99.1] on shutdown of component - unsubscribe==\n[99.1] unsubscribe
<b>subscriber_id</b> from PDP" as policy_handler
    }
    component "<&target> <b>DCAE component like TCA</b> - as in Casablanca\n--[0.4] on startup--\n[0.4.1]
get the config and policies from CBS\n[0.4.2] listen for policy-update notifications from <b>DCAE-Controller<
/b> \n delivered through reconfigure.sh - as in Cassablanca-Dublin\n--on policy-update through reconfigure.
sh--\n[1.2.10] on receiving the push-notification of policy-update\n[1.3] handle and consume the added
/updated/removed policies" as dcae_component #eeeeff
}

component "<&rss> <b>Message Router of DMaaP</b>\n--\n[1.2] persistently delivers the policy-update
notification \nto <b>subscriber_topic="policies_DCAE"</b> \nwith the latest snapshot of policies \n and all
the fields of subscription \n for each component separately" as DMaaP #ff8888

PAP .down.> PDP : [1] push/delete policies
PDP .down.> policy_update_subscribers : [0.3.2] subscribe\n insert/update \nsubscriber record
PDP .down.> policy_update_subscribers : [1.0] iterate through \nall subscribers
PDP ..> policy_update_subscribers : [99.1] unsubscribe \n delete <b>subscriber_id</b>

PDP .down.> DMaaP : [1.2] push policy-update \n for DCAE component instance
DMaaP ..> policy_handler : [1.2] push policy-update \n for DCAE component instance

policy_handler .> deployment_handler : [1.2.4] push policy-update \n for DCAE component instance

dcae_component ..> config_binding_service : [0.4.1] get the config and policies
config_binding_service .right.> consul_kv : [0.4.1] get the config and policies

policy_handler ..> PDP :         [0.3.2] POST /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>

policy_handler ..> PDP : [99.1] DELETE /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>

deployment_handler .down.> cloudify : [0] install \n component
deployment_handler .down.> cloudify : [1.2.5] find deployment \nfor each component
deployment_handler .down.> cloudify : [1.2.6] push policy-update \n for DCAE component deployment
deployment_handler .down.> cloudify : [99] uninstall \n component


cloudify .down.> k8s_plugin : [0] install \ncomponent
cloudify .down.> k8s_plugin : [1.2.7] push policy-update \n for component deployment
cloudify .down.> k8s_plugin : [99] uninstall \ncomponent

k8s_plugin .down.> consul_kv : [0.1] save config for component
k8s_plugin .down.> consul_kv : [0.3.4] save policies \n for component
deployment_handler ..> consul_kv : [2.0] get pending policy_updates \n that is the collection of \n
component=<service-component-name>
policy_handler ..> consul_kv : [1.2.1] save updated policies \n for component
policy_handler ..> consul_kv : [1.2.2] get \n <service-component-name>:<b>policies/settings</b>
policy_handler ..> consul_kv : [1.2.3] save pending policy_update \n for component
k8s_plugin .down.> consul_kv : [1.2.8] get the latest policies under \n <service-component-name>:<b>policies
/</b>
```

```
k8s_plugin .down.> consul_kv : [1.2.11] delete the pending poliy_updates\n for the consumed
<b>policy_update_seq</b> \n on component=<service-component-name>
k8s_plugin .down.> consul_kv : [99.2] and [99.4] delete records

k8s_plugin .left.> dcae_component :    [0.4]  start  component
k8s_plugin .left.> dcae_component : [1.2.10] push policy-update
k8s_plugin .left.> dcae_component : [99.3] stop component

k8s_plugin .up.> policy_handler : [0.3] subscribe
policy_handler ..> k8s_plugin  : [0.3.3] policies
k8s_plugin .up.> policy_handler : [99.1] unsubscribe



left footer
    <&thumb-up> DCAE-Controller is the middleman for policy update flow
    <&thumb-up> Message Router of DMaaP persistently delivers the push-notification of the policy-update per
each component instance to policy_handler

    <b>requirements on new PDP -- see the diagram</b>
    <&task> maintain pub-sub table per subscriber_id with policy-filters
    <&task> notify about the policy updates through Message Router of DMaaP

    <b>requirements on DCAE Component -- the same as in Casablanca

    <b>requirements on Config-Binding-Service -- the same as in Casablanca

    <b>requirements on @policies_gather -- see the diagram</b>
    <b>requirements on k8s_plugin -- see the diagram</b>
    <b>requirements on policy_handler -- see the diagram</b>
endfooter

@enduml
```

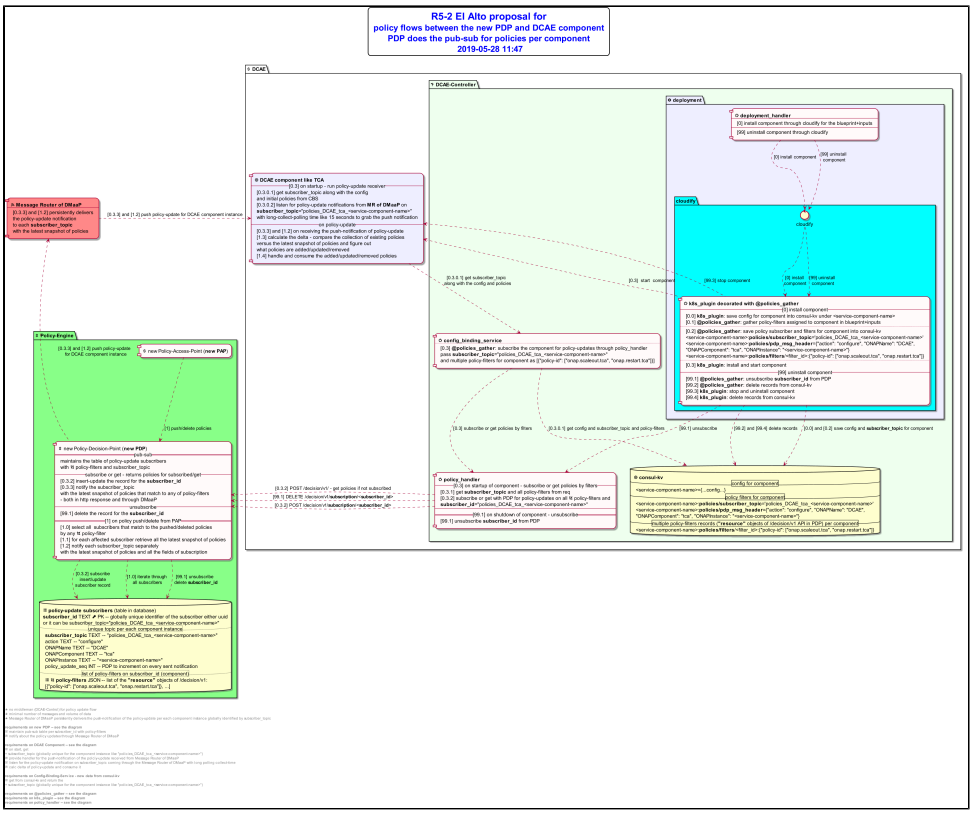
Option - 5-2 - Less requirements on **component** - config-binding-service to call the policy-handler to do the subscription with PDP for the component

1. **on startup** – get the **subscriber_topic="policies_DCAE_tca_<service-component-name>"** and the latest policies from config-binding-service
    a. initialize the policies in the component with policies received from config-binding-service - as it does now.
2. listen for **subscriber_topic** of policy-update notification from MR of DMaaP with long-collect-polling time like 15 seconds to grab the push notification
3. **on receiving the policy-update** pushed notification from DMaaP, handle the policy-update that contains the full snapshot of all the policies that match to the component
    a. calculate the delta between the current set of policies and the received shapshot of policies

Config-binding-service to do the following on request from component

1. get policy data from consul-kv
2. request policy-handler to subscribe or get policies based on some flags in the blueprint to decide per component on whether to subscribe it to policy-updates or just bring the policies
    a. policy-handler will ask PDP to add the subscription and return the policies for the component or just get the policies for the component
    b. policy-handler to return the policies to config-binding-service

See inside the diagram for more details, data structures, and flow steps



r5-2_proposed_policy_update_flow

```
@startuml r5-2_proposed_policy_update_flow
allowmixing
scale 4096*4096

skinparam title {
    FontSize 24
    FontColor Blue
    FontStyle Bold
    BorderRoundCorner 15
    BorderThickness 2
}

skinparam roundcorner 20

skinparam component {
    BackgroundColor Snow
}

skinparam note {
    FontColor Black
    BackgroundColor azure
}

title = R5-2 El Alto proposal for \n policy flows between the new PDP and DCAE component \n PDP does the pub-
sub for policies per component \n %date[yyyy-MM-dd HH:mm]%

package "<&dollar> Policy-Engine" as policy_engine #88ff88 {
    component "<&dollar> new Policy-Access-Point (<b>new PAP</b>)" as PAP
    component "<&dollar> new Policy-Decision-Point (<b>new PDP</b>)\n==pub-sub==\n maintains the table of
policy-update subscribers \nwith <&crop> policy-filters and subscriber_topic\n--subscribe or get - returns
policies for subscribed/get--\n[0.3.2] insert-update the record for the <b>subscriber_id</b>\n[0.3.3] notify
the subscriber_topic \nwith the latest snapshot of policies that match to any of policy-filters \n - both in
http response and through DMaaP\n--unsubscribe--\n[99.1] delete the record for the <b>subscriber_id</b>\n==
[1] on policy push/delete from PAP==\n[1.0] select all  subscribers that match to the pushed/deleted
policies \n by any <&crop> policy-filter\n[1.1] for each affected subscriber retrieve all the latest
```

```
snapshot of policies \n[1.2] notify each subscriber_topic separately \nwith the latest snapshot of policies
and all the fields of subscription" as PDP

    database "<&list> <b>policy-update subscribers</b> (table in database)\n<b>subscriber_id</b> TEXT <&key>
PK -- globally unique identifier of the subscriber either uuid \n or it can be subscriber_topic="
policies_DCAE_tca_<service-component-name>"\n..unique topic per each component instance..
\n<b>subscriber_topic</b> TEXT -- "policies_DCAE_tca_<service-component-name>"\n action TEXT -- "configure"
\nONAPName TEXT -- "DCAE"\nONAPComponent TEXT -- "tca"\nONAPInstance TEXT -- "<service-component-name>"
\npolicy_update_seq INT -- PDP to increment on every sent notification\n..list of policy-filters on
subscriber_id (component)..\n<&list> <&crop> <b>policy-filters</b> JSON -- list of the <b>"resource"</b>
objects of /decision/v1: \n[{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}, ...]" as
policy_update_subscribers
}

package "<&dollar> DCAE" as DCAE {
    package "<&signpost> DCAE-Controller" as DCAE_Controller #eeffee {
        package "<&cog> deployment" #eeeeff {
            component "<&aperture> <b>deployment_handler</b>\n--\n[0] install component through cloudify for
the blueprint+inputs\n--\n[99] uninstall component through cloudify" as deployment_handler

            package "<b>cloudify</b>" as cloudify_server #00ffff {
                control cloudify
                component "<&aperture> <b>k8s_plugin decorated with @policies_gather</b>\n--[0] install
component--\n[0.0] <b>k8s_plugin</b>: save config for component into consul-kv under <service-component-
name>\n[0.1] <b>@policies_gather</b>: gather policy-filters assigned to component in blueprint+inputs\n..\n
[0.2] <b>@policies_gather</b>: save policy subscriber and filters for component into consul-kv \n<service-
component-name>:<b>policies/subscriber_topic</b>='policies_DCAE_tca_<service-component-name>'\n<service-
component-name>:<b>policies/pdp_msg_header</b>={"action": "configure", "ONAPName": "DCAE", \n
"ONAPComponent": "tca", "ONAPInstance": "<service-component-name>"}\n<service-component-name>:<b>policies
/filters</b>/<filter_id>:{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}\n..\n[0.3] <b>k8s_plugin<
/b>: install and start component\n--[99] uninstall component--\n[99.1] <b>@policies_gather</b>: unsubscribe
<b>subscriber_id</b> from PDP\n[99.2] <b>@policies_gather</b>: delete records from consul-kv\n[99.3]
<b>k8s_plugin</b>: stop and uninstall component\n[99.4] <b>k8s_plugin</b>: delete records from consul-kv" as
k8s_plugin
            }
        }
        component "<&aperture> <b>config_binding_service</b>\n..\n[0.3] <b>@policies_gather</b>: subscribe
the component for policy-updates through policy_handler\n pass <b>subscriber_topic</b>="
policies_DCAE_tca_<service-component-name>" \nand multiple policy-filters for component as [{"policy-id":
["onap.scaleout.tca", "onap.restart.tca"]}]" as config_binding_service

        database "<&target> <b>consul-kv</b>\n--config for component--\n<service-component-name>={...
config...}\n--policy filters for component--\n<service-component-name>:<b>policies/subscriber_topic<
/b>='policies_DCAE_tca_<service-component-name>'\n<service-component-name>:<b>policies/pdp_msg_header</b>=
{"action": "configure", "ONAPName": "DCAE", \n "ONAPComponent": "tca", "ONAPInstance": "<service-component-
name>"}\n--multiple policy-filters records (<b>"resource"</b> objects of /decision/v1 API in PDP) per
component--\n<service-component-name>:<b>policies/filters</b>/<filter_id>:{"policy-id": ["onap.scaleout.
tca", "onap.restart.tca"]}" as consul_kv

        component "<&aperture> <b>policy_handler</b>\n--[0.3] on startup of component - subscribe or get
policies by filters--\n[0.3.1] get <b>subscriber_topic</b> and all policy-filters from req\n[0.3.2]
subscribe or get with PDP for policy-updates on all <&crop> policy-filters and \n<b>subscriber_id</b>="
policies_DCAE_tca_<service-component-name>"\n--\n==[99.1] on shutdown of component - unsubscribe==\n[99.1]
unsubscribe <b>subscriber_id</b> from PDP" as policy_handler
    }
    component "<&target> <b>DCAE component like TCA</b>\n--[0.3] on startup - run policy-update receiver--\n
[0.3.0.1] get subscriber_topic along with the config \n and initial policies from CBS\n[0.3.0.2] listen for
policy-update notifications from <b>MR of DMaaP</b> on \n<b>subscriber_topic</b>="policies_DCAE_tca_<service-
component-name>"\n with long-collect-polling time like 15 seconds to grab the push notification\n--on policy-
update--\n[0.3.3] and [1.2] on receiving the push-notification of policy-update\n[1.3] calculate the delta -
compare the collection of existing policies \n versus the latest snapshot of policies and figure out \nwhat
policies are added/updated/removed \n[1.4] handle and consume the added/updated/removed policies" as
dcae_component #eeeeff
}

component "<&rss> <b>Message Router of DMaaP</b>\n--\n [0.3.3] and [1.2] persistently delivers \nthe policy-
update notification \nto each <b>subscriber_topic</b> \nwith the latest snapshot of policies" as DMaaP
#ff8888

PAP .down.> PDP : [1] push/delete policies
PDP .down.> policy_update_subscribers : [0.3.2] subscribe\n insert/update \nsubscriber record
```

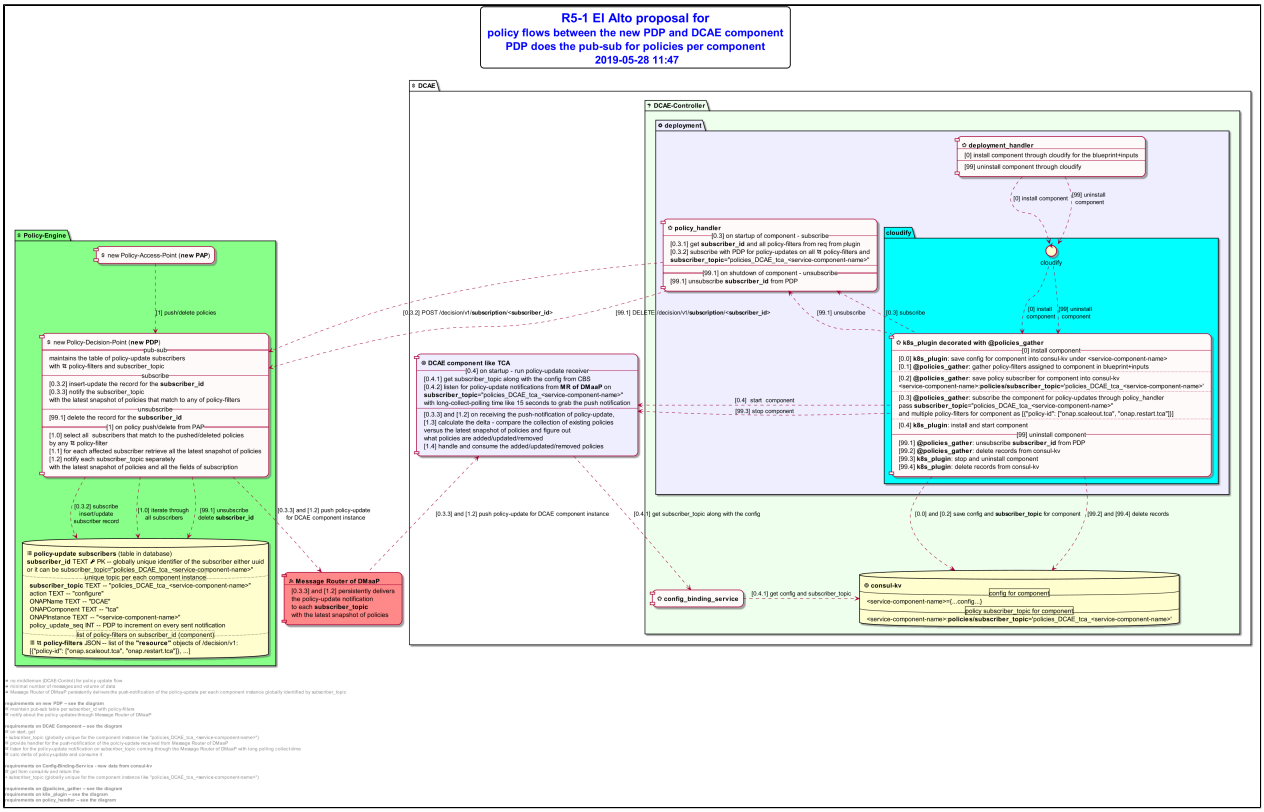
```
PDP .down.> policy_update_subscribers : [1.0] iterate through \nall subscribers
PDP ..> policy_update_subscribers : [99.1] unsubscribe \n delete <b>subscriber_id</b>

PDP .up.> DMaaP : [0.3.3] and [1.2] push policy-update \n for DCAE component instance
DMaaP .> dcae_component : [0.3.3] and [1.2] push policy-update for DCAE component instance

dcae_component ..> config_binding_service : [0.3.0.1] get subscriber_topic \n along with the config and
policies
config_binding_service .right.> consul_kv : [0.3.0.1] get config and subscriber_topic and policy-filters

policy_handler .> PDP :         [0.3.2] POST /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>
policy_handler .> PDP :         [0.3.2] POST /decision/v1/ - get policies if not subscribed
policy_handler .> PDP : [99.1] DELETE /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>

deployment_handler .down.> cloudify : [0] install component
deployment_handler .down.> cloudify : [99] uninstall \ncomponent
cloudify .down.> k8s_plugin : [0] install \ncomponent
cloudify .down.> k8s_plugin : [99] uninstall \ncomponent
k8s_plugin .down.> consul_kv : [0.0] and [0.2] save config and <b>subscriber_topic</b> for component
k8s_plugin .down.> consul_kv : [99.2] and [99.4] delete records

config_binding_service ..> policy_handler : [0.3] subscribe or get policies by filters
k8s_plugin ..> policy_handler : [99.1] unsubscribe

k8s_plugin .left.> dcae_component : [0.3]  start  component
k8s_plugin .left.> dcae_component : [99.3] stop component


left footer
    <&thumb-up> no middleman (DCAE-Control) for policy update flow
    <&thumb-up> minimal number of messages and volume of data
    <&thumb-up> Message Router of DMaaP persistently delivers the push-notification of the policy-update per
each component instance globally identified by subscriber_topic

    <b>requirements on new PDP -- see the diagram</b>
    <&task> maintain pub-sub table per subscriber_id with policy-filters
    <&task> notify about the policy updates through Message Router of DMaaP

    <b>requirements on DCAE Component -- see the diagram</b>
    <&task> on start, get
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")
    <&task> provide handler for the push-notification of the polciy-update received from Message Router of
DMaaP
    <&task> listen for the policy-update notification on subscriber_topic coming through the Message Router
of DMaaP with long polling collect-time
    <&task> calc delta of policy-update and consume it

    <b>requirements on Config-Binding-Service - new data from consul-kv</b>
    <&task> get from consul-kv and return the
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")

    <b>requirements on @policies_gather -- see the diagram</b>
    <b>requirements on k8s_plugin -- see the diagram</b>
    <b>requirements on policy_handler -- see the diagram</b>
endfooter

@enduml
```

Option - 5-1 - Less requirements on **component** - policy-handler to do the subscription with PDP for the component

1. **on startup** – get the **subscriber_topic="policies_DCAE_tca_<service-component-name>"** from config-binding-service
2. listen for **subscriber_topic** of policy-update notification from MR of DMaaP with long-collect-polling time like 15 seconds to grab the push notification
3. **on receiving the policy-update** pushed notification from DMaaP, handle the policy-update that contains the full snapshot of all the policies that match to the component

a. calculate the delta between the current set of policies and the received shapshot of policies

See inside the diagram for more details, data structures, and flow steps



R5-1 El Alto proposal for policy flows between the new PDP and DCAE component PDP does the pub-sub for policies per component

```
@startuml r5-1_proposed_policy_update_flow
allowmixing
scale 4096*4096

skinparam title {
    FontSize 24
    FontColor Blue
    FontStyle Bold
    BorderRoundCorner 15
    BorderThickness 2
}

skinparam roundcorner 20

skinparam component {
    BackgroundColor Snow
}

skinparam note {
    FontColor Black
    BackgroundColor azure
}

title = R5-1 El Alto proposal for \n policy flows between the new PDP and DCAE component \n PDP does the pub-sub for policies per component \n %date[yyyy-MM-dd HH:mm]%

package "<&dollar> Policy-Engine" as policy_engine #88ff88 {
    component "<&dollar> new Policy-Access-Point (<b>new PAP</b>)" as PAP
    component "<&dollar> new Policy-Decision-Point (<b>new PDP</b>)\n==pub-sub==\n maintains the table of policy-update subscribers \nwith <&crop> policy-filters and subscriber_topic\n--subscribe--\n[0.3.2] insert-update the record for the <b>subscriber_id</b>\n[0.3.3] notify the subscriber_topic \nwith the latest snapshot of policies that match to any of policy-filters\n--unsubscribe--\n[99.1] delete the record for the
```

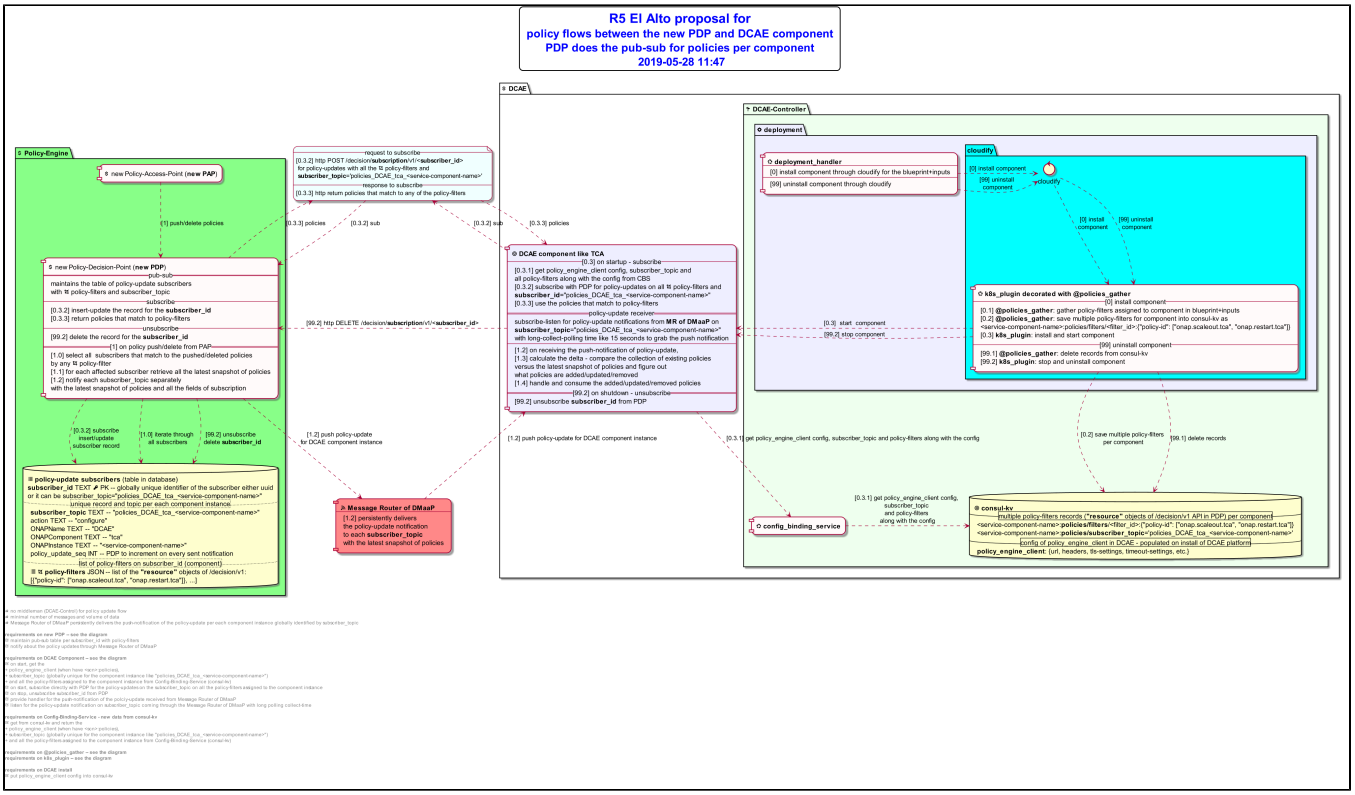
```
<b>subscriber_id</b>\n==[1] on policy push/delete from PAP==\n[1.0] select all  subscribers that match to
the pushed/deleted policies \n by any <&crop> policy-filter\n[1.1] for each affected subscriber retrieve all
the latest snapshot of policies \n[1.2] notify each subscriber_topic separately \nwith the latest snapshot
of policies and all the fields of subscription" as PDP

    database "<&list> <b>policy-update subscribers</b> (table in database)\n<b>subscriber_id</b> TEXT <&key>
PK -- globally unique identifier of the subscriber either uuid \n or it can be subscriber_topic="
policies_DCAE_tca_<service-component-name>"\n..unique topic per each component instance..
\n<b>subscriber_topic</b> TEXT -- "policies_DCAE_tca_<service-component-name>"\n action TEXT -- "configure"
\nONAPName TEXT -- "DCAE"\nONAPComponent TEXT -- "tca"\nONAPInstance TEXT -- "<service-component-name>"
\npolicy_update_seq INT -- PDP to increment on every sent notification\n..list of policy-filters on
subscriber_id (component)..\n<&list> <&crop> <b>policy-filters</b> JSON -- list of the <b>"resource"</b>
objects of /decision/v1: \n[{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}, ...]" as
policy_update_subscribers
}

package "<&dollar> DCAE" as DCAE {
    package "<&signpost> DCAE-Controller" as DCAE_Controller #eeffee {
        package "<&cog> deployment" #eeeeff {
            component "<&aperture> <b>deployment_handler</b>\n--\n[0] install component through cloudify for
the blueprint+inputs\n--\n[99] uninstall component through cloudify" as deployment_handler

            component "<&aperture> <b>policy_handler</b>\n--[0.3] on startup of component - subscribe--\n
[0.3.1] get <b>subscriber_id</b> and all policy-filters from req from plugin\n[0.3.2] subscribe with PDP for
policy-updates on all <&crop> policy-filters and \n<b>subscriber_topic</b>="policies_DCAE_tca_<service-
component-name>"\n--\n==[99.1] on shutdown of component - unsubscribe==\n[99.1] unsubscribe <b>subscriber_id<
/b> from PDP" as policy_handler

            package "<b>cloudify</b>" as cloudify_server #00ffff {
                control cloudify
                component "<&aperture> <b>k8s_plugin decorated with @policies_gather</b>\n--[0] install
component--\n[0.0] <b>k8s_plugin</b>: save config for component into consul-kv under <service-component-
name>\n[0.1] <b>@policies_gather</b>: gather policy-filters assigned to component in blueprint+inputs\n..\n
[0.2] <b>@policies_gather</b>: save policy subscriber for component into consul-kv \n<service-component-
name>:<b>policies/subscriber_topic</b>='policies_DCAE_tca_<service-component-name>'\n..\n[0.3]
<b>@policies_gather</b>: subscribe the component for policy-updates through policy_handler\n pass
<b>subscriber_topic</b>="policies_DCAE_tca_<service-component-name>" \nand multiple policy-filters for
component as [{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}]\n..\n[0.4] <b>k8s_plugin</b>:
install and start component\n--[99] uninstall component--\n[99.1] <b>@policies_gather</b>: unsubscribe
<b>subscriber_id</b> from PDP\n[99.2] <b>@policies_gather</b>: delete records from consul-kv\n[99.3]
<b>k8s_plugin</b>: stop and uninstall component\n[99.4] <b>k8s_plugin</b>: delete records from consul-kv" as
k8s_plugin
            }
        }
        database "<&target> <b>consul-kv</b>\n--config for component--\n<service-component-name>={...
config...}\n--policy subscriber_topic for component--\n<service-component-name>:<b>policies/subscriber_topic<
/b>='policies_DCAE_tca_<service-component-name>'" as consul_kv
        component "<&aperture> <b>config_binding_service</b>" as config_binding_service

    }
    component "<&target> <b>DCAE component like TCA</b>\n--[0.4] on startup - run policy-update receiver--\n
[0.4.1] get subscriber_topic along with the config from CBS\n[0.4.2] listen for policy-update notifications
from <b>MR of DMaaP</b> on \n<b>subscriber_topic</b>="policies_DCAE_tca_<service-component-name>"\n with
long-collect-polling time like 15 seconds to grab the push notification\n--\n[0.3.3] and [1.2] on receiving
the push-notification of policy-update, \n[1.3] calculate the delta - compare the collection of existing
policies \n versus the latest snapshot of policies and figure out \nwhat policies are added/updated/removed
\n[1.4] handle and consume the added/updated/removed policies" as dcae_component #eeeeff
}

component "<&rss> <b>Message Router of DMaaP</b>\n--\n [0.3.3] and [1.2] persistently delivers \nthe policy-
update notification \nto each <b>subscriber_topic</b> \nwith the latest snapshot of policies" as DMaaP
#ff8888

PAP .down.> PDP : [1] push/delete policies
PDP .down.> policy_update_subscribers : [0.3.2] subscribe\n insert/update \nsubscriber record
PDP .down.> policy_update_subscribers : [1.0] iterate through \nall subscribers
PDP ..> policy_update_subscribers : [99.1] unsubscribe \n delete <b>subscriber_id</b>

PDP .down.> DMaaP : [0.3.3] and [1.2] push policy-update \n for DCAE component instance
DMaaP .up.> dcae_component : [0.3.3] and [1.2] push policy-update for DCAE component instance
```

```
dcae_component .right.> config_binding_service : [0.4.1] get subscriber_topic along with the config
config_binding_service .right.> consul_kv : [0.4.1] get config and subscriber_topic

policy_handler ..> PDP :     [0.3.2] POST /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>
policy_handler ..> PDP : [99.1] DELETE /decision/v1/<b>subscription</b>/<<b>subscriber_id</b>>

deployment_handler .down.> cloudify : [0] install component
deployment_handler .down.> cloudify : [99] uninstall \ncomponent
cloudify .down.> k8s_plugin : [0] install \ncomponent
cloudify .down.> k8s_plugin : [99] uninstall \ncomponent
k8s_plugin .down.> consul_kv : [0.0] and [0.2] save config and <b>subscriber_topic</b> for component
k8s_plugin .down.> consul_kv : [99.2] and [99.4] delete records

k8s_plugin .up.> policy_handler : [0.3] subscribe
k8s_plugin .up.> policy_handler : [99.1] unsubscribe

k8s_plugin .left.> dcae_component : [0.4]  start  component
k8s_plugin .left.> dcae_component : [99.3] stop component


left footer
    <&thumb-up> no middleman (DCAE-Control) for policy update flow
    <&thumb-up> minimal number of messages and volume of data
    <&thumb-up> Message Router of DMaaP persistently delivers the push-notification of the policy-update per
each component instance globally identified by subscriber_topic

    <b>requirements on new PDP -- see the diagram</b>
    <&task> maintain pub-sub table per subscriber_id with policy-filters
    <&task> notify about the policy updates through Message Router of DMaaP

    <b>requirements on DCAE Component -- see the diagram</b>
    <&task> on start, get
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")
    <&task> provide handler for the push-notification of the polciy-update received from Message Router of
DMaaP
    <&task> listen for the policy-update notification on subscriber_topic coming through the Message Router
of DMaaP with long polling collect-time
    <&task> calc delta of policy-update and consume it

    <b>requirements on Config-Binding-Service - new data from consul-kv</b>
    <&task> get from consul-kv and return the
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")

    <b>requirements on @policies_gather -- see the diagram</b>
    <b>requirements on k8s_plugin -- see the diagram</b>
    <b>requirements on policy_handler -- see the diagram</b>
endfooter

@enduml
```

Option - 5 - More requirements on component (initial proposal) - too much to ask the components

1. **on startup** – subscribe to PDP with the policy-filter the component is interested in and the **subscriber_topic= "policies_DCAE_tca_<service-component-name>"** that that uniquely identifies the component instance
2. listen for **subscriber_topic** of policy-update notification from MR of DMaaP with long-collect-polling time like 15 seconds to grab the push notification
3. **on receiving the policy-update** pushed notification from DMaaP, handle the policy-update that contains the full snapshot of all the policies that match to the component
    a. calculate the delta between the current set of policies and the received shapshot of policies
4. **n stop**, unsubscribe **subscriber_topic** from PDP

See inside the diagram for more details, data structures, and flow steps



R5 El Alto proposal for policy flows between the new PDP and DCAE component PDP does the pub-sub for policies per component

```
@startuml r5_proposed_policy_update_flow
allowmixing
scale 4096*4096

skinparam title {
    FontSize 24
    FontColor Blue
    FontStyle Bold
    BorderRoundCorner 15
    BorderThickness 2
}

skinparam roundcorner 20

skinparam component {
    BackgroundColor Snow
}

skinparam note {
    FontColor Black
    BackgroundColor azure
}

title = R5 El Alto proposal for \n policy flows between the new PDP and DCAE component \n PDP does the pub-sub for policies per component \n %date[yyyy-MM-dd HH:mm]%

package "<&dollar> Policy-Engine" as policy_engine #88ff88 {
    component "<&dollar> new Policy-Access-Point (<b>new PAP</b>)" as PAP
    component "<&dollar> new Policy-Decision-Point (<b>new PDP</b>)\n==pub-sub==\n maintains the table of policy-update subscribers \nwith <&crop> policy-filters and subscriber_topic\n--subscribe--\n[0.3.2] insert-update the record for the <b>subscriber_id</b>\n[0.3.3] return policies that match to policy-filters\n--unsubscribe--\n[99.2] delete the record for the <b>subscriber_id</b>\n==[1] on policy push/delete from PAP==\n[1.0] select all  subscribers that match to the pushed/deleted policies \n by any <&crop> policy-filter\n[1.1] for each affected subscriber retrieve all the latest snapshot of policies \n[1.2] notify each subscriber_topic separately \nwith the latest snapshot of policies and all the fields of subscription" as PDP
```

```
    database "<&list> <b>policy-update subscribers</b> (table in database)\n<b>subscriber_id</b> TEXT <&key>
PK -- globally unique identifier of the subscriber either uuid \n or it can be subscriber_topic="
policies_DCAE_tca_<service-component-name>"\n..unique record and topic per each component instance..
\n<b>subscriber_topic</b> TEXT -- "policies_DCAE_tca_<service-component-name>"\n action TEXT -- "configure"
\nONAPName TEXT -- "DCAE"\nONAPComponent TEXT -- "tca"\nONAPInstance TEXT -- "<service-component-name>"
\npolicy_update_seq INT -- PDP to increment on every sent notification\n..list of policy-filters on
subscriber_id (component)..\n<&list> <&crop> <b>policy-filters</b> JSON -- list of the <b>"resource"</b>
objects of /decision/v1: \n[{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}, ...]" as
policy_update_subscribers
}

package "<&dollar> DCAE" as DCAE {
    package "<&signpost> DCAE-Controller" as DCAE_Controller #eeffee {
        package "<&cog> deployment" #eeeeff {
            component "<&aperture> <b>deployment_handler</b>\n--\n[0] install component through cloudify for
the blueprint+inputs\n--\n[99] uninstall component through cloudify" as deployment_handler
            package "<b>cloudify</b>" as cloudify_server #00ffff {
                control cloudify
                component "<&aperture> <b>k8s_plugin decorated with @policies_gather</b>\n--[0] install
component--\n[0.1] <b>@policies_gather</b>: gather policy-filters assigned to component in blueprint+inputs\n
[0.2] <b>@policies_gather</b>: save multiple policy-filters for component into consul-kv as \n<service-
component-name>:policies/filters/<filter_id>:{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}\n[0.3]
<b>k8s_plugin</b>: install and start component\n--[99] uninstall component--\n[99.1] <b>@policies_gather<
/b>: delete records from consul-kv\n[99.2] <b>k8s_plugin</b>: stop and uninstall component" as k8s_plugin
            }
        }
        database "<&target> <b>consul-kv</b>\n--multiple policy-filters records (<b>"resource"</b> objects
of /decision/v1 API in PDP) per component--\n<service-component-name>:<b>policies/filters</b>/<filter_id>:
{"policy-id": ["onap.scaleout.tca", "onap.restart.tca"]}\n<service-component-name>:<b>policies
/subscriber_topic</b>='policies_DCAE_tca_<service-component-name>'\n--config of policy_engine_client in DCAE
- populated on install of DCAE platform--\n<b>policy_engine_client</b>: {url, headers, tls-settings, timeout-
settings, etc.}" as consul_kv
        component "<&aperture> <b>config_binding_service</b>" as config_binding_service
    }
    component "<&target> <b>DCAE component like TCA</b>\n--[0.3] on startup - subscribe--\n[0.3.1] get
policy_engine_client config, subscriber_topic and \nall policy-filters along with the config from CBS\n
[0.3.2] subscribe with PDP for policy-updates on all <&crop> policy-filters and \n<b>subscriber_id</b>="
policies_DCAE_tca_<service-component-name>"\n[0.3.3] use the policies that match to policy-filters\n==policy-
update receiver==\nsubscribe-listen for policy-update notifications from <b>MR of DMaaP</b> on
\n<b>subscriber_topic</b>="policies_DCAE_tca_<service-component-name>"\n with long-collect-polling time like
15 seconds to grab the push notification\n--\n[1.2] on receiving the push-notification of policy-update, \n
[1.3] calculate the delta - compare the collection of existing policies \n versus the latest snapshot of
policies and figure out \nwhat policies are added/updated/removed \n[1.4] handle and consume the added
/updated/removed policies\n==[99.2] on shutdown - unsubscribe==\n[99.2] unsubscribe <b>subscriber_id</b>
from PDP" as dcae_component #eeeeff
}

component "<&rss> <b>Message Router of DMaaP</b>\n--\n [1.2] persistently delivers \nthe policy-update
notification \nto each <b>subscriber_topic</b> \nwith the latest snapshot of policies" as DMaaP #ff8888

note "--request to subscribe--\n[0.3.2] http POST /decision/<b>subscription</b>/v1/<<b>subscriber_id</b>>\n
for policy-updates with all the <&crop> policy-filters and \n <b>subscriber_topic<
/b>='policies_DCAE_tca_<service-component-name>'\n--response to subscribe--\n[0.3.3] http return policies
that match to any of the policy-filters" as message_sub_to_PDP

deployment_handler .right.> cloudify : [0] install component
deployment_handler .> cloudify : [99] uninstall \ncomponent
cloudify .down.> k8s_plugin : [0] install \ncomponent
cloudify .down.> k8s_plugin : [99] uninstall \ncomponent
k8s_plugin .down.> consul_kv : [0.2] save multiple policy-filters \nper component
k8s_plugin .down.> consul_kv : [99.1] delete records

k8s_plugin .> dcae_component : [0.3]  start  component
dcae_component ..> config_binding_service : [0.3.1] get policy_engine_client config, subscriber_topic and
policy-filters along with the config
config_binding_service .> consul_kv : [0.3.1] get policy_engine_client config, \nsubscriber_topic \nand
policy-filters \nalong with the config

PAP .down.> PDP : [1] push/delete policies

dcae_component .up.> message_sub_to_PDP : [0.3.2] sub
```

```
message_sub_to_PDP ..> PDP : [0.3.2] sub
PDP ..> policy_update_subscribers : [0.3.2] subscribe\n insert/update \nsubscriber record

PDP ..> message_sub_to_PDP : [0.3.3] policies
message_sub_to_PDP ..> dcae_component : [0.3.3] policies

PDP .down.> policy_update_subscribers : [1.0] iterate through \nall subscribers

PDP ..> DMaaP : [1.2] push policy-update \n for DCAE component instance
DMaaP ..> dcae_component : [1.2] push policy-update for DCAE component instance

k8s_plugin .> dcae_component : [99.2] stop component

dcae_component .> PDP : [99.2] http DELETE /decision/<b>subscription</b>/v1/<<b>subscriber_id</b>>

PDP ..> policy_update_subscribers : [99.2] unsubscribe \n delete <b>subscriber_id</b>

left footer
    <&thumb-up> no middleman (DCAE-Control) for policy update flow
    <&thumb-up> minimal number of messages and volume of data
    <&thumb-up> Message Router of DMaaP persistently delivers the push-notification of the policy-update per
each component instance globally identified by subscriber_topic

    <b>requirements on new PDP -- see the diagram</b>
    <&task> maintain pub-sub table per subscriber_id with policy-filters
    <&task> notify about the policy updates through Message Router of DMaaP

    <b>requirements on DCAE Component -- see the diagram</b>
    <&task> on start, get the
        + policy_engine_client (when have <scn>:policies),
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")
        + and all the policy-filters assigned to the component instance from Config-Binding-Service (consul-
kv)
    <&task> on start, subscribe directly with PDP for the policy-updates on the subscriber_topic on all the
policy-filters assigned to the component instance
    <&task> on stop, unsubscribe subscriber_id from PDP
    <&task> provide handler for the push-notification of the polciy-update received from Message Router of
DMaaP
    <&task> listen for the policy-update notification on subscriber_topic coming through the Message Router
of DMaaP with long polling collect-time

    <b>requirements on Config-Binding-Service - new data from consul-kv</b>
    <&task> get from consul-kv and return the
        + policy_engine_client (when have <scn>:policies),
        + subscriber_topic (globally unique for the component instance like "policies_DCAE_tca_<service-
component-name>")
        + and all the policy-filters assigned to the component instance from Config-Binding-Service (consul-
kv)

    <b>requirements on @policies_gather -- see the diagram</b>
    <b>requirements on k8s_plugin -- see the diagram</b>

    <b>requirements on DCAE install</b>
    <&task> put policy_engine_client config into consul-kv

endfooter

@enduml
```