How ACI Faults Are Generated and How to Selectively Prevent Faults from Being Generated

Contents

Introduction

This document explains the high level process of Application Centric Infrastructure (ACI) fault generation and how to prevent a specific fault from being generated. The document demonstrates this with two examples.

How a Fault Gets Generated and How to Selectively Prevent Faults from Being Generated

High Level Mechanism

1. Each fault is a Managed Object (MO) of class faultInst (or faultDelegate). This fault MO is generated by another MO, usually its parent, because some rules are violated.
2. Each MO in the tree that can generate faults has an attribute **monPolDn** which points to another MO which is a monitoring policy object. This object allows the property to be modified and the trigger to generate faults. There are multiple classes of the monitoring policy object, such as: monInfraPol - deals with infra policy (VMM manager, access port policy, physical ports, and so on) - Located in Fabric > Access Policies > Monitoring policiesmonFabricPol - deals with fabric monitoring - located in Fabric > Fabric Policies > Monitoring policiesmonEPGPol - deals with tenants monitoring > located in Tenant > Monitoring Policy menu
3. Usually it will be the default monitoring object. However, by going to the specific area of the object model you can create a specific user defined monitoring policy for any of those monitoring policy classes.
4. You can modify many properties of those monitoring policies. The example will show how you can prevent a given fault from being generated for all objects for which the monitoring policy is applied to. However, you can also modify the fault lifecycle timers (retention time, soaking time, and so on).
5. In order to modify fault severity or prevent a fault from being generated, you need to select the monitoring object that corresponds to the class of the MO that generated this object (for example, parent of the fault).

6. Then under this class, choose the fault code that you want to modify and choose an initial severity of value “squelched”.

This prevents any fault with that code from being generated by the MO that is assigned to this specific monitoring policy.

Example 1 - Fault in a Tenant

Each fault is associated with an object.

```
admin@apic:~> moquery -d "uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]/fault-F0879"
Total Objects shown: 1
# fault.Inst code             : F0879 ack              : no cause            : resolution-failed
changeSet        : childAction      : created          : 2015-01-22T00:05:00.286+01:00
descr            : Failed to form relation to MO uni/tn-RD/ap-App_RD1/epg-EPG_RD11 of class
fvAEPg dn               : uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]/fault-F0879 domain           : infra highestSeverity  : warning lastTransition   :
2015-01-22T00:05:00.286+01:00 lc               : raised modTs            : never
occur            : 1 origSeverity     : warning prevSeverity     : warning rn               :
fault-F0879 rule             : dbgac-rs-to-epg-resolve-fail
```

The previous fault is an MO of class fault.Inst and with code F0879.

The fault is associated with an Endpoint Group (EPG) object as shown next.

This object is the distinguished name (DN) of the parent of the fault. This parent object is of class dbg.RsToEpg.

```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn        : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
rType        : mo
rn           : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state        : missing-target
stateQual    : none
status       :
tCl          : fvAEPg
tType        : mo
uid          : 15374
```

You can see that this EPG object is associated with a monPolDn object. Most objects in the tree are monitored by a monitoring object.

Here is a user defined monitoring object of **class monEPGPol** with dn**.**

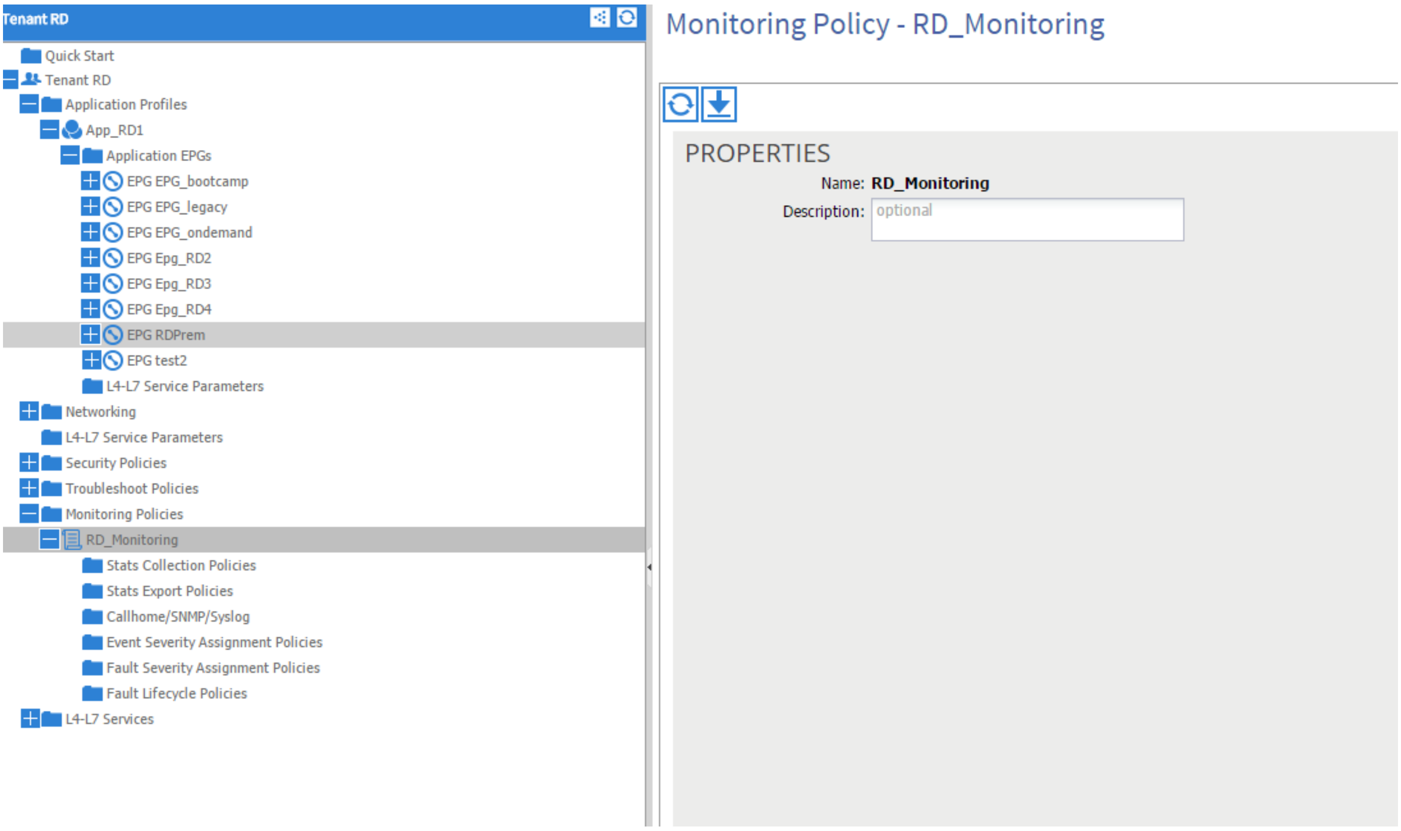
```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn        : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
rType        : mo
```

```
rn            : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state         : missing-target
stateQual     : none
status        :
tCl           : fvAEPg
tType         : mo
uid           : 15374
```

Here is the complete object used for monitoring.

```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn         : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
rType         : mo
rn            : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state         : missing-target
stateQual     : none
status        :
tCl           : fvAEPg
tType         : mo
uid           : 15374
```

The monEPGPol object is configured under the tenant Monitoring Policy, where you can either create a new policy or modify the default one. Here is an example of the monEPGPol name RD_Monitoring.



Monitoring Policy - RD_Monitoring

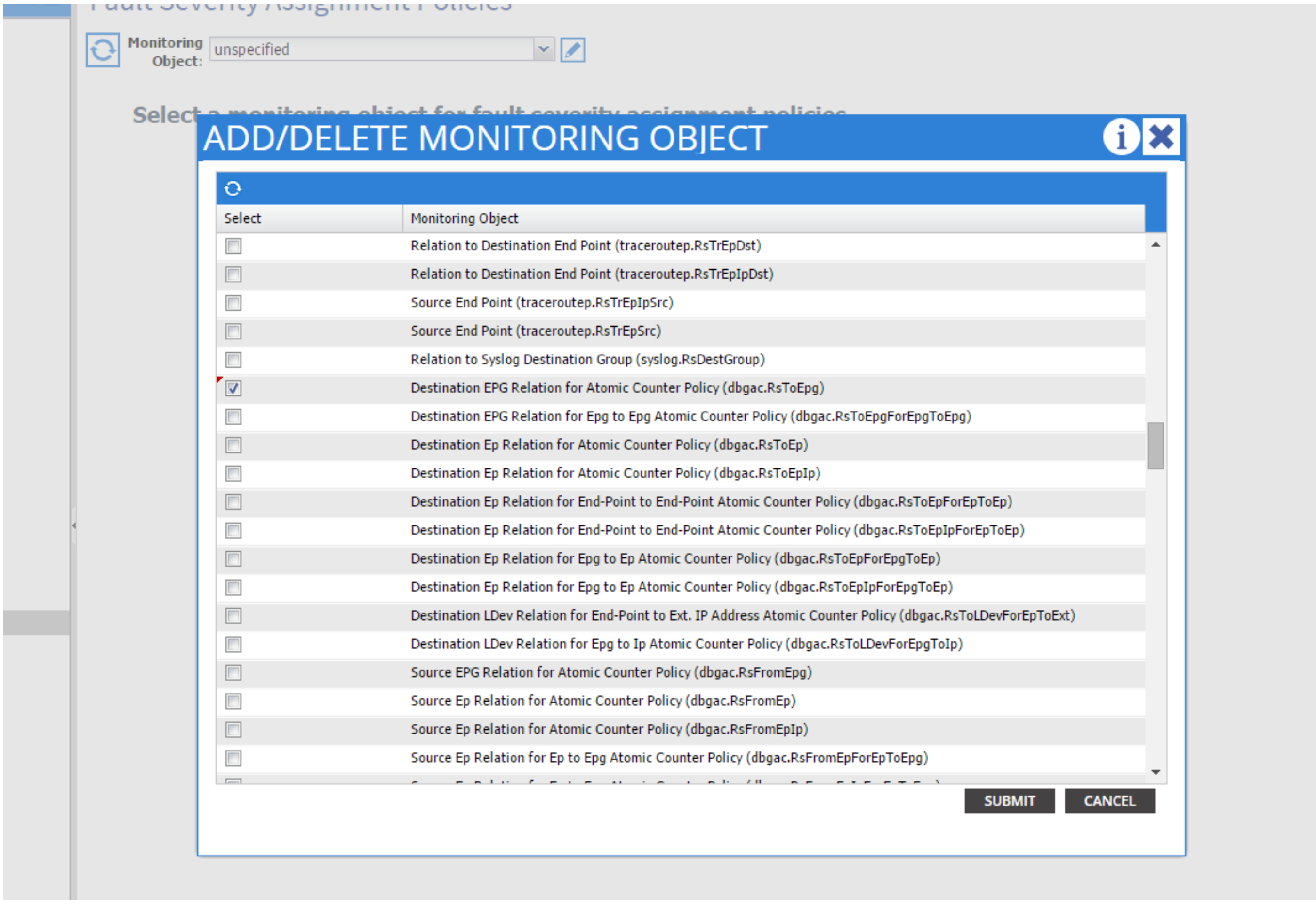
PROPERTIES

Name: RD_Monitoring

Description: optional

You can choose the Fault Severity assignment policies and click the pencil (next to the Monitoring object).

Then, if you choose in the monitoring object list of that monitoring policy, the class for which the fault was created (here **dbgac.RsToEpg**).



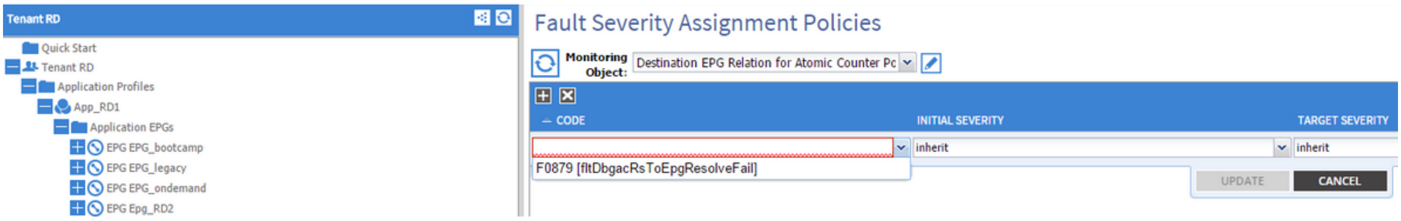
Monitoring Object: unspecified

Select a monitoring object for fault severity assignment policies

ADD/DELETE MONITORING OBJECT

Select	Monitoring Object
□	Relation to Destination End Point (traceroutep.RsTrEpDst)
□	Relation to Destination End Point (traceroutep.RsTrEpIpDst)
□	Source End Point (traceroutep.RsTrEpIpSrc)
□	Source End Point (traceroutep.RsTrEpSrc)
□	Relation to Syslog Destination Group (syslog.RsDestGroup)
☑	Destination EPG Relation for Atomic Counter Policy (dbgac.RsToEpg)
□	Destination EPG Relation for Epg to Epg Atomic Counter Policy (dbgac.RsToEpgForEpgToEpg)
□	Destination Ep Relation for Atomic Counter Policy (dbgac.RsToEp)
□	Destination Ep Relation for Atomic Counter Policy (dbgac.RsToEpIp)
□	Destination Ep Relation for End-Point to End-Point Atomic Counter Policy (dbgac.RsToEpForEpToEp)
□	Destination Ep Relation for End-Point to End-Point Atomic Counter Policy (dbgac.RsToEpIpForEpToEp)
□	Destination Ep Relation for Epg to Ep Atomic Counter Policy (dbgac.RsToEpForEpgToEp)
□	Destination Ep Relation for Epg to Ep Atomic Counter Policy (dbgac.RsToEpIpForEpgToEp)
□	Destination LDev Relation for End-Point to Ext. IP Address Atomic Counter Policy (dbgac.RsToLDevForEpToExt)
□	Destination LDev Relation for Epg to Ip Atomic Counter Policy (dbgac.RsToLDevForEpgToIp)
□	Source EPG Relation for Atomic Counter Policy (dbgac.RsFromEpg)
□	Source Ep Relation for Atomic Counter Policy (dbgac.RsFromEp)
□	Source Ep Relation for Atomic Counter Policy (dbgac.RsFromEpIp)
□	Source Ep Relation for Ep to Epg Atomic Counter Policy (dbgac.RsFromEpForEpToEpg)

You can see all faults associated to that specific class (the only one shown here is F0789).



Fault Severity Assignment Policies

Monitoring Object: Destination EPG Relation for Atomic Counter Pc

CODE	INITIAL SEVERITY	TARGET SEVERITY
F0879 [fltDbgacRsToEpgResolveFail]	inherit	inherit

The fault F0789 is the code of the fault show at the beginning of the example.

You can choose this fault and if you **set an initial Severity to squelched** (you can leave Target Severity to inherit), it prevents such fault from being generated in the future with the presumption that they are generated by an object that has a link to the monitoring policy you just modified.

However it will not clear existing faults, but only new faults.

Example 2 - Physical Fault

In this example, the fault is generated because port 1/25 on leaf is admin up but with no SFP in it.

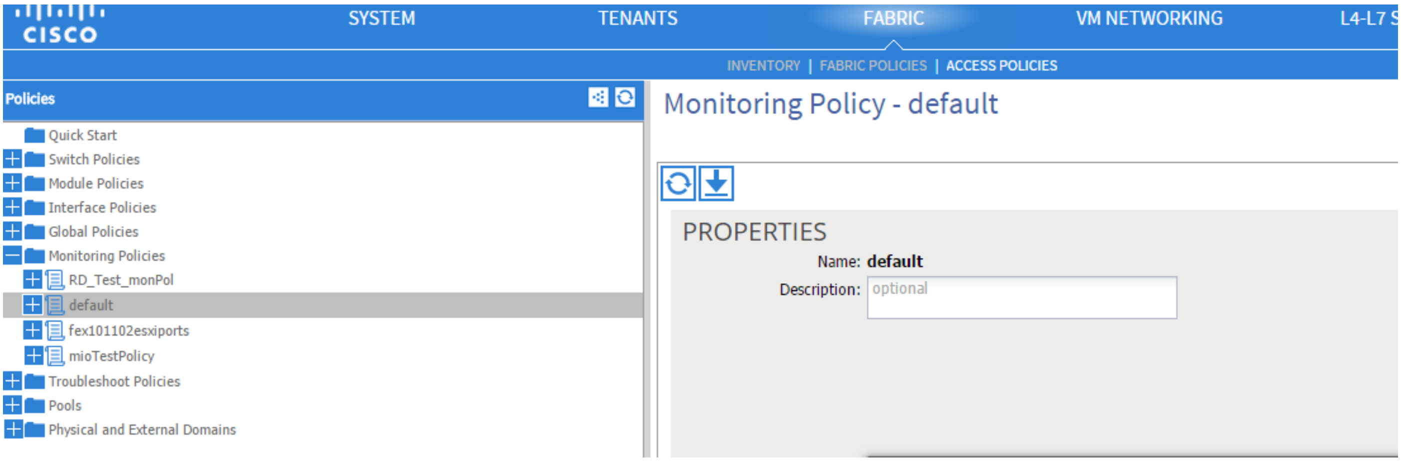
```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn         : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
```

```
rType         : mo
rn            : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state         : missing-target
stateQual     : none
status        :
tCl           : fvAEPg
tType         : mo
uid           : 15374
```

This is associated with a physical port. Here is the parent MO that generated that fault.

```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn         : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
rType         : mo
rn            : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state         : missing-target
stateQual     : none
status        :
tCl           : fvAEPg
tType         : mo
uid           : 15374
```

This is associated with the monInfraPol object configured as shown here.



Monitoring Policy - default

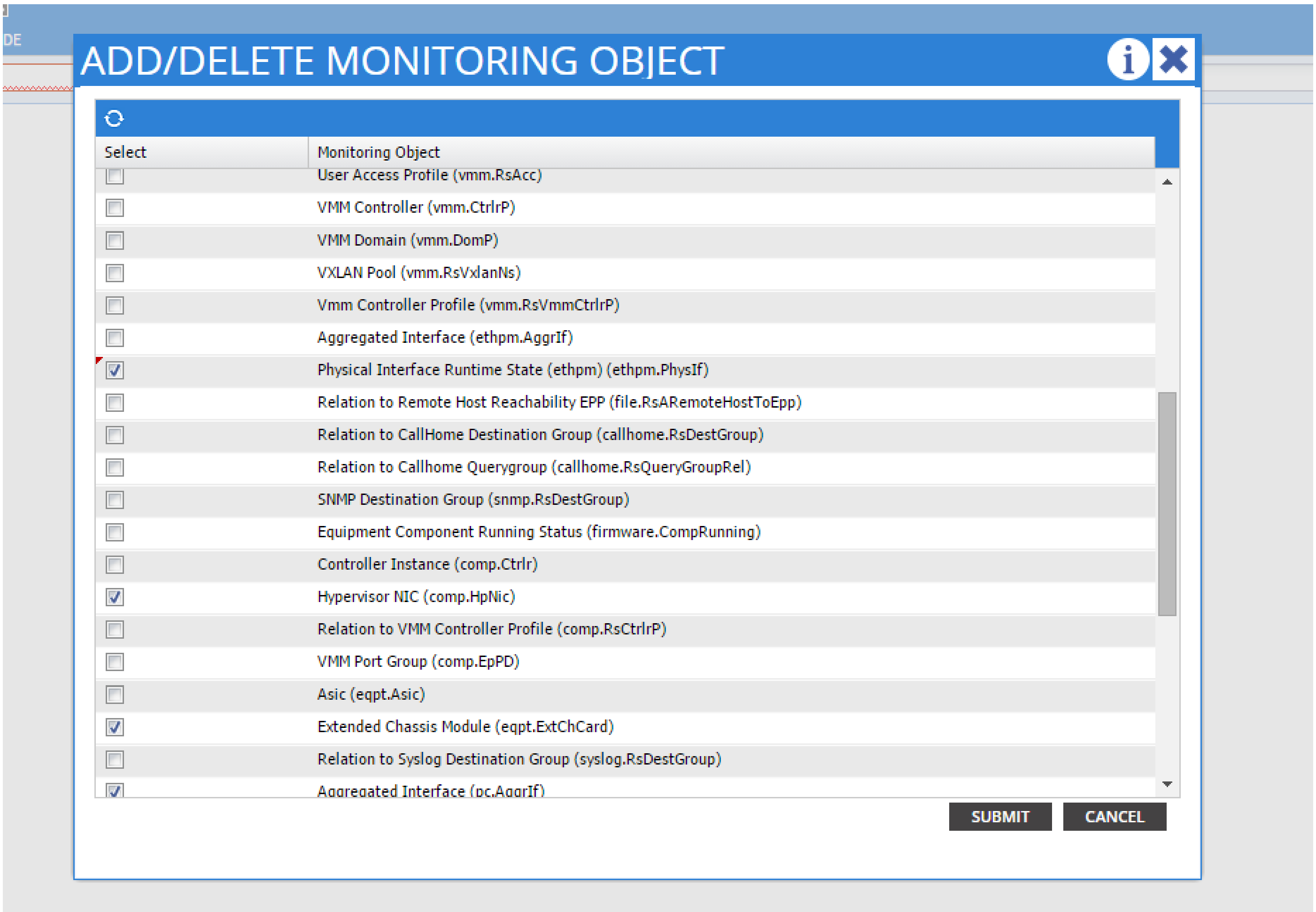
PROPERTIES

Name: default

Description: optional

```
admin@apic:~> moquery -d uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-
EPG_RD11]
Total Objects shown: 1
# dbgac.RsToEpg tDn          : uni/tn-RD/ap-App_RD1/epg-EPG_RD11 childAction  : dn           :
uni/tn-RD/ipToEpg-Ext_10.200.1.101/rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11] forceResolve : no
lcOwn         : local modTs        : 2014-12-05T12:56:29.340+01:00 monPolDn     : uni/tn-
RD/monepg-RD_Monitoring
rType         : mo
rn            : rstoEpg-[uni/tn-RD/ap-App_RD1/epg-EPG_RD11]
state         : missing-target
stateQual     : none
status        :
tCl           : fvAEPg
tType         : mo
uid           : 15374
```

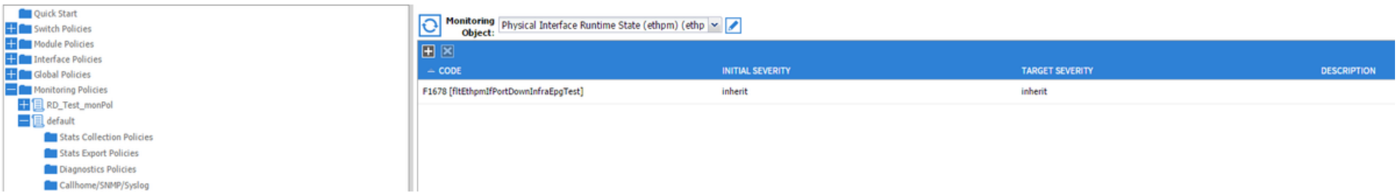
Under the Fault Severity Assignment policy, click the pencil in the work pane, next to the monitoring object drop-down list. Add a class where you modify the monitoring properties. Then choose the class of the object that generated the fault, that is, ethmPhysIf.



Select	Monitoring Object
□	User Access Profile (vmm.RsAcc)
□	VMM Controller (vmm.CtrlrP)
□	VMM Domain (vmm.DomP)
□	VXLAN Pool (vmm.RsVxlanNs)
□	Vmm Controller Profile (vmm.RsVmmCtrlrP)
□	Aggregated Interface (ethpm.AggrIf)
☑	Physical Interface Runtime State (ethpm) (ethpm.PhysIf)
□	Relation to Remote Host Reachability EPP (file.RsARemoteHostToEpp)
□	Relation to CallHome Destination Group (callhome.RsDestGroup)
□	Relation to Callhome Querygroup (callhome.RsQueryGroupRel)
□	SNMP Destination Group (snmp.RsDestGroup)
□	Equipment Component Running Status (firmware.CompRunning)
□	Controller Instance (comp.Ctrlr)
☑	Hypervisor NIC (comp.HpNic)
□	Relation to VMM Controller Profile (comp.RsCtrlrP)
□	VMM Port Group (comp.EpPD)
□	Asic (eqpt.Asic)
☑	Extended Chassis Module (eqpt.ExtChCard)
□	Relation to Syslog Destination Group (syslog.RsDestGroup)
☑	Aggregated Interface (pc.AggrIf)

Choose this class and click the + icon in order to see each fault generated for that object.

In this example, you can see fault F1678 and it its properties can be modified. Choosing Initial severity Squelched and target severity inherit prevents new faults of that code from being generated from the object that has this monitoring policy applied.



Monitoring Object: Physical Interface Runtime State (ethpm) (ethp

CODE	INITIAL SEVERITY	TARGET SEVERITY	DESCRIPTION
F1678 [fltEthpmIfPortDownInfraEpgTest]	inherit	inherit	

After you make the change, if you enable port 1/25 with no SFP in it it will not generate any faults!

> **Note :** In versions earlier than Software Version 2.2: Existing faults (even in Clearing retaining mode) will not be cleared.

> **Note :** In Software Version 2.2 and later: Even existing faults will be affected by the new policy.