Troubleshooting ESC Issues

This chapter contains the following sections:

Viewing ESC Log Messages

Log messages are created for ESC events throughout the VNF lifecycle. These can be external messages, messages from ESC to other external systems, error messages, warnings, events, failures and so on. The log file can be found at `/var/log/esc/escmanager_tagged.log`.

The log message format is as follows:

```
date=<time-date>] [loglevel=<loglevel>] [tid=<transactionid>] [cl=<classifications>]
[tags=<tags>] [msg=<message>
```

Sample log is as follows:

```
date=15:43:58,46022-Nov-2016]
[loglevel=ERROR ] [tid=0793b5c9-8255-47f3-81e6-fbb59f6571f7] [cl=OS ]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:CSCvd94541,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:test,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0] [msg=sleepingfor5seconds
to allow vm to become ACTIVE instance id:
162344f7-78f9-4e45-9f23-34cf87377fa7
name:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0
```

When a request is received, a RequestDetails object is created which autogenerates a unique transaction id. This value is carried forward across all threads. Classifications and tags are optional. These are prefixes added to the log messages to enhance readability, and help in debugging. With classifications and tags, the log messages can be easily parsed and filtered by the log analysis tools.

The following classifications are supported:

NBI	"com.cisco.esc.rest""com.cisco.esc.filter"(North Bound Interface - Clientinterface)
SBI	"com.cisco.esc.rest"- source is a callback handler or"EventsResource"(South Bound Interface - i.e. between ESC and the VIM)

SM	"com.cisco.esc.statemachines". stands for StateMachine. This classification indicates logs in the StateMachine category.
MONITORING	"com.cisco.esc.monitoring""com.cisco.esc.paadaptor"(MONA related logs)
DYNAMIC_MAPPING	"com.cisco.esc.dynamicmapping""com.cisco.esc.db.dynamicmapping"(MONA related logs)
CONFD	"com.cisco.esc.confd"
CONFD_NOTIFICATION	"com.cisco.esc.confd.notif""com.cisco.esc.confd.ConfdNBIAdapter"
OS	"com.cisco.esc.vim.openstack"
LIBVIRT	"com.cisco.esc.vim.vagrant
VIM	"com.cisco.esc.vim"
REST_EVENT	"ESCManager_Event""com.cisco.esc.util.RestUtils". indicates REST notifications in logs.
WD	"com.cisco.esc.watchdog"
DM	"com.cisco.esc.datamodel""com.cisco.esc.jaxb.parameters"(Datamodel and resource objects)
DB	"com.cisco.esc.db"(Database related logs)
GW	"com.cisco.esc.gateway"
LC	"com.cisco.esc.ESCManager"(Start up related logs)
SEC	"com.cisco.esc.jaas"
MOCONFIG	"com.cisco.esc.moconfig"(MOCONFIG object related logs --this is internal for ESC developers)
POLICY	"com.cisco.esc.policy"(Service/VM Policy related logs)
TP	"com.cisco.esc.threadpool"
ESC	"com.cisco.esc" Any other packages not mentioned above

The following tags are supported:

- **Workflow [wf:]**—Generated using action and resource from RequestDetails object. Example "wf:create_network"
- **Event type [eventType:]**—Event that triggered the current action. Example: "eventType:VM_DEPLOY_EVENT"
- **Resource based**—These values are generated based on the type of parameter used by the event. The hierarchy, that is, the tenant, the vm group and so on is added to the log.

Tenant	[tenant:<tenant name>]

Network	[tenant:<tenant id>, network:<network name>] **Note** The tenant appears only if applicable.
Subnet	[tenant:<tenant name or id>, network:<network name or id>, subnet:<subnet name>] **Note** The tenant appears only if applicable.
User	[tenant:<tenant name>, user:<user name or id>] **Note** The tenant appears only if applicable.
Image	[image:<image name>]
Flavor	[flavor:<flavor name>]
Deployment	[tenant:<tenant name or id>, depName:<deployment name>]
DeploymentDetails	[tenant:<tenant name or id>, depName:<deployment name>, vmGroup:<vm group name>, vmName:<vm name>]
Switch	[tenant:<tenant name or id>, switch:<switch name>]
Volume	[volume:<volume name>]
Service	[svcName:<Service Registration name>]

Further, ESC logs can also be forwarded to an rsyslog server for further analysis and log management.

Filtering Logs Using Confd APIs

You can query and retrieve logs (for example, deployment logs, or error logs) in ESC using log filters introduced in the confd APIs. New filters for Tenant, Deployment Name, and VM Name are introduced. This enables you to query the ESC logs further for most recent error logs using the log filters in Confd APIs. You can also retrieve ESC logs related to the communication between ESC and the OS (by setting the classification tag to "OS").

The log format to retrieve confd API logs:

```
date=<time-date>] [loglevel=<loglevel>] [tid=<transactionid>] [cl=<classifications>]
[tags=<tags>] [msg=<message>
```

The sample log is as follows:

```
date=15:43:58,46022-Nov-2016] [loglevel=ERROR ] [tid=0793b5c9-8255-47f3-81e6-fbb59f6571f7]
 [cl=OS ]
[tags=wf:create_vm,eventType:VM_DEPLOY_EVENT,tenant:test,depName:test-dep,vmGrpName:test-VNF,
vmName:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0]
[msg=sleepingfor5seconds to allow vm to become ACTIVE instance id:
162344f7-78f9-4e45-9f23-34cf87377fa7 name:test-dep_test_0_dc3f406c-05ca-43b3-af21-0841e3b029a0
```

The parameters for log level, classification and tags are dependent on each other to retrieve the logs. You can successfully retrieve the logs with the following combination.

- log_level=ERROR, classifications=OS, tags=(depName:test-dep)
- log_level=ERROR, classifications=OS, tags=(tenant: test)

The log filter returns a value when all the following conditions are met:

- Log level
- Classifications (if provided)
- Tags (if provided)

Note If there are more than one classification listed, it has to match at least one of the classifications. The same applies to the tags as well.

For example, the following log filter criteria does not return the log sample mentioned earlier:

```
log_level=ERROR, classifications=VIM, tags=(depName:test-dep)
```

It does not return any value though the log level and tags match, the classification VIM does not match.

The data model is as follows:

```
rpc filterLog {
    description "Query and filter escmanager logs using given parameters";
    tailf:actionpoint escrpc;
    input {
      leaf log_level {
        mandatory false;
        description "One of DEBUG / INFO / WARNING / ERROR / TRACE / FATAL. Results will
include all logs at and
                     above the level specified";
        type types:log_level_types;
        default ERROR;
      }
      leaf log_count {
        mandatory false;
        description "Number of logs to return";
        type uint32;
        default 10;
      }
      container classifications {
        leaf-list classification {
         description "Classification values to be used for the log filtering.  For example:
 'OS', 'SM'.
                      Logs containing any of the provided classification values will be
returned.";
          type types:log_classification_types;
        }
      }
      container tags {
        list tag {
          key "name";
          leaf name {
            mandatory true;
            description "Tag name to be used for the log filtering.  For example: 'tenant',
 'depName'.
                         Logs containing any of the provided tag name plus the tag values
will be returned.";
            type types:log_tag_types;
          }
          leaf value {
            mandatory true;
            description "Tag value pairs to be used for the log filtering.  For example:
'adminTenant', 'CSRDeployment'";
            type string;
          }
      }
    }
  }
  output {
    container filterLogResults {
      leaf log_level {
        description "Log level used to filter for the logs.";
        type types:log_level_types;
      }
      list logs {
        container classifications {
          leaf-list classification {
            description "Classifications used to filter for the logs.";
            type types:log_classification_types;
          }
        }
        container tags {
          list tag {
            key "name";
            leaf name {
              mandatory true;
              description "Tag name used to filter for the logs.";
              type types:log_tag_types;
            }
            leaf value {
              mandatory true;
              description "Tag value used to filter for the logs.";
              type string;
            }
          }
        }
        leaf log_date_time {
          description "Timestamp of the log.";
          type string;
        }
        leaf log_message {
          description "The log message.";
          type string;
        }
      }
    }
  }
}
```

You can query for the confd API logs through the netconf console or esc_nc_cli

- Through the netconf-console, run the following query:

  ```
  /opt/cisco/esc/confd/bin/netconf-console --port=830 --host=127.0.0.1 --user=admin
  --privKeyFile=/home/admin/.ssh/confd_id_dsa --privKeyType=dsa --rpc=log.xml
  ```

- Using the esc_nc_cli, run the following query:

  ```
  esc_nc_cli --user <username> --password <password> filter-log log.xml
  ```

  The sample log.xml is as follows:

  ```
  <filterLog xmlns="https://www.cisco.com/esc/esc">
    <log_level>INFO</log_level>
    <log_count>1</log_count>
    <classifications>
      <classification>OS</classification>
      <classification>SM</classification>
    </classifications>
    <tags>
  ```

```
    <tag>
      <name>depName</name>
      <value>CSR_ap1</value>
    </tag>
    <tag>
      <name>tenant</name>
      <value>admin</value>
    </tag>
  </tags>
</filterLog>
```

The response is as follows:

```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <filterLogResults xmlns="https://www.cisco.com/esc/esc">
    <log_level>INFO</log_level>
    <logs>
      <classifications>
        <classification>OS</classification>
        <classification>SM</classification>
      </classifications>
      <tags>
        <tag>
          <name>depName</name>
          <value>CSR_ap1</value>
        </tag>
        <tag>
          <name>tenant</name>
          <value>admin</value>
        </tag>
      </tags>
      <log_date_time>13:06:07,575 31-Oct-2016</log_date_time>
      <log_message> No pending work flow to start.</log_message>
    </logs>
  </filterLogResults>
</rpc-reply>
```

Note The logging API responses are in XML format. If the log messages contain any XML characters, then the characters will be escaped so not to break the XML conformance.

General Installation Errors

This section cover some of the common installation problems and how to troubleshoot them.

Problem/Error	Possible Reason	User Action

Issues encountered while verifying ESC services status at the installation time using `sudo escadm status`	Some services take time to start or have trouble starting.	**1.** Identify the issues using one of the following method:
		• Check the log /var/log/esc/escadm.log
		`$ cat /var/log/esc/escadm.log`
		• Add '-v' to escadm status to show the verbose output of the ESC services and see the services are up and running.
		2. Confirm the status of the identified services that has issues and manually start these services.
		`$ sudo escadm <<service>> status//`
		`If the status is stopped or dead, manually`
		`start the services`
		`using the next command.`
		`$ sudo escadm <<service>> start --v`

Authentication related error during the ESC installation	OpenStack credential-related arguments are missing.	Source your OpenStack RC file and verify the OpenStack clients work properly
ESC HA-related (Active/Standby) Issues		
Network Problem		Check for the following: • The static IP addresses for both ESC nodes used are based on the OpenStack configuration. • The gateway for each network interface is accessible.

Problem/Error	Possible Reason	User Action
ESC active node stays in the "switching-to-active" state	This could be because of the following issues: • ESC HA Active/Standby node can not reach its peer during the initial installation. • ESC service (tomcat) can't start properly due to database problems (etc. database migration, database file corruption). • Confd can't start due to the CDB file corruption. • Postgresql can't start or init due to the file system issues. • The connection between ESC nodes is too slow.	Check for the: • Connectivity between ESC active node and standby node. For initial installation, ESC active service won't be up if it can't reach the standby node. You need to make sure you have both ESC nodes successfully deployed and they can reach each other. • ESC logs at /var/log/esc/esc_haagent.log (ESC 2.X) or /var/log/esc/escadm.log (ESC 3.X) to identify the services that has issues. • Log at /var/log/esc/escmanager.log for esc_service and postgresql issues.
MTU issues for ESC HA Active/Standby		For the ESC VMs, reduce the MTU for network interfaces from 1500 to 1450. Follow the below steps to reduce the MTU value: **1.** Identify the interface you would like to change. Access the interface from the location, /etc/sysconfig/network-scripts/ifcfg-ethX, where X represent the interface number you want to change. **2.** Use a text editor like VIM to add or edit the MTU items for the interface, for instance, set MTU = 1450 **3.** Restart the interface. `network service restart`

Problem/Error	Possible Reason	User Action
Other Issues		There are several useful logs to check for ESC HA Active/Standby troubleshooting. • ESC manager log is located at /var/log/esc/escmanager.log • ESC HA Active/Standby log about esc service startup/stop is located at /var/log/esc/esc_haagent.log for ESC 2.X and /var/log/esc/escadm.log for ESC 3.X • For Keepalived configuration: • Check the configuration file at /etc/keepalived/keepalived.conf • Keepalived log is located at /var/log/messages by grep keepalived or vrrp • For DRBD configuration: • DRBD configuration can be verified by checking the file at /etc/drbd.d/esc.res • The DRBD log is located at /var/log/messages by grep drbd
ESC Services Startup Issues		

Service status reports ETSI service is dead at installation time using sudo escadm status --v	If the host has multiple IP addresses the ETSI service is unable to determine which IP address to use when generating callback URLs.
	This cause can be verified by examining the log file `/var/log/esc/etsi-vnfm/etsi-vnfm.log` for an exception including:
	Configure `server.host` property in configuration file.
		1. Set the server.host property:
		`sudo escadm etsi set –server_host <IP>`
		2. Start the ETSI service:
		`sudo escadm etsi start`

ESC Failover Scenarios

- When customer deploys many VNFs

- When customer does a DB backup
- One or more of the VMs in the VNFs is recovered via redeploy – new VM ID
- ESC Fails – It is a permanent failure
- Customer Restores DB – When there is a mismatch between 1 or more VM IDs
- Same VNF fails, but when ESC tries to recover, it fails because it is not found on the VIM by VM ID and redeploy fails because the ports are in use.
- The workaround is to delete the VM out of band and recover – if the VM is deleted it is not found by ESC and the port is available for redeployment.