Recreating Tenant Storage Mounts on ECP 5.2.x and 5.3.x

Tenant Storage Mounts give all ECP tenants (EPIC and Kubernetes) posix-based access to their dedicated volumes on ECP's Data Fabric. If ECP's Data Fabric becomes unavailable, or if any ECP hosts lose their mounts to it, you might need to recreate Tenant Storage Mounts after restoring Data Fabric connectivity. These steps apply to Embedded Data Fabric and Data Fabric on Kubernetes ('Picasso').

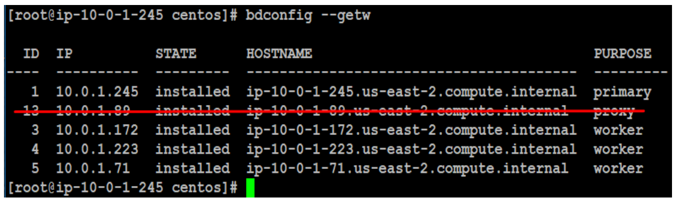
1. Introduction

Ezmeral Container Platform uses Data Fabric as its default Tenant Storage type. You can confirm your platform's Tenant Storage type by checking Settings -> Tenant Storage in the ECP web UI. If the type is set to 'MAPR', then Data Fabric is being used for Tenant Storage, and this article applies to your platform.

When Data Fabric is used for tenant storage, each tenant is automatically provisioned with a tenant storage volume in the Data Fabric cluster, located at *<DF cluster name>/exthcp/tenant-<id>/fsmount*. Additionally, ML Ops tenants have a project repo volume at *<DF cluster name>/exthcp/tenant-<id>/fsmount/repo*.

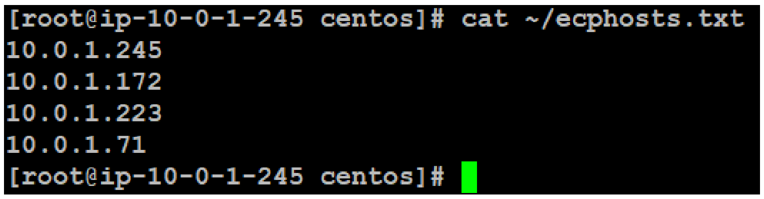
Tenant EPIC virtual nodes and Kubernetes pods are connected to the Data Fabric cluster via an epic-mapr container on each host (excluding gateways) which runs the Mapr Posix Client (basic or platinum). This container mounts the host's */opt/bluedata/mapr/mnt* path to the Data Fabric cluster. In turn, Tenant Storage Mounts connect per-tenant directories under each host's */opt/bluedata/share* path to */opt/bluedata/mapr/mnt*. If users are unable to access tenant storage from inside their EPIC nodes or Kubernetes pods, or if 'Transport endpoint not connected' errors occur when trying to access Tenant Storage directories; or if the EPIC or Kubernetes dashboards show 'Mount Point' or 'PosixClient' services in an error state (red dot), then follow these steps to remount hosts to Data Fabric and recreate the Tenant Storage mounts.

Some of the steps in this article require CLI commands to be run on all platform hosts. We recommend that, on the primary controller, you produce 2 text files – *'ecphosts.txt'* (containing controller hosts and EPIC workers) and *'k8shosts.txt'* (containing all Kubernetes masters and workers) – with one host IP address on each line; and that you configure passwordless SSH (e.g using ssh-copy-id) from the controller to each host. This will allow you to run commands on all hosts using a single bash loop from the primary controller. Use `bdconfig --getw` and `bdconfig –getk8sh` to get the lists of ECP and K8S hosts. If you have a preferred way to run commands on multiple platform hosts e.g clush, feel free to use that instead. Don't include proxy (gateway) hosts in the steps covered by this article – they have no epic-mapr container or tenant storage mounts.



```
[root@ip-10-0-1-245 centos]# bdconfig --getw

  ID  IP          STATE      HOSTNAME                                 PURPOSE
----  ----------  ---------  ---------------------------------------  ---------
   1  10.0.1.245  installed  ip-10-0-1-245.us-east-2.compute.internal  primary
~~  13  10.0.1.89   installed  ip-10-0-1-89.us-east-2.compute.internal   proxy~~
   3  10.0.1.172  installed  ip-10-0-1-172.us-east-2.compute.internal  worker
   4  10.0.1.223  installed  ip-10-0-1-223.us-east-2.compute.internal  worker
   5  10.0.1.71   installed  ip-10-0-1-71.us-east-2.compute.internal   worker
[root@ip-10-0-1-245 centos]#
```



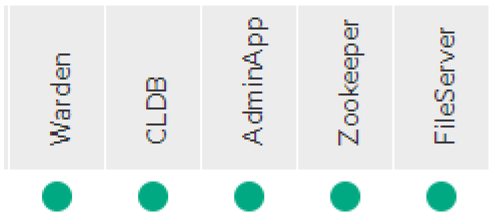
```
[root@ip-10-0-1-245 centos]# cat ~/ecphosts.txt
10.0.1.245
10.0.1.172
10.0.1.223
10.0.1.71
[root@ip-10-0-1-245 centos]#
```

Test this setup with a command such as

```
for host in $(cat ecphosts.txt); do ssh $host "hostname –f;bdconfig --
version";done
```

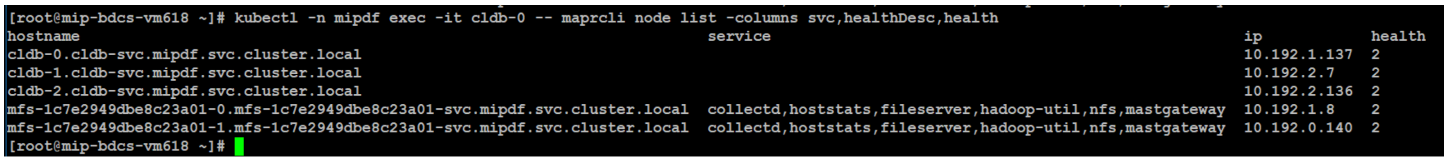
2. Check that Data Fabric is working and accessible

- Embedded Data Fabric: In the ECP web UI, check the EPIC -> Dashboard -> Services tab to make sure that all Data Fabric services are in an 'OK' (green) state:



- Data Fabric on Kubernetes ('Picasso'): From the master of the Data Fabric Kubernetes cluster, determine the name of your DF cluster (*kubectl get po –A | grep cldb*-0 will show you the DF namespace, which has the same name as the cluster), then run:

```
kubectl -n <df cluster name> exec -it cldb-0 -- maprcli node list -columns
svc,healthDesc,health
```

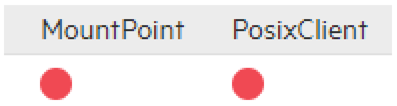


```
[root@mip-bdcs-vm618 ~]# kubectl -n mipdf exec -it cldb-0 -- maprcli node list -columns svc,healthDesc,health
hostname                                                          service                                               ip            health
cldb-0.cldb-svc.mipdf.svc.cluster.local                                                                                 10.192.1.137  2
cldb-1.cldb-svc.mipdf.svc.cluster.local                                                                                 10.192.2.7    2
cldb-2.cldb-svc.mipdf.svc.cluster.local                                                                                 10.192.2.136  2
mfs-1c7e2949dbe8c23a01-0.mfs-1c7e2949dbe8c23a01-svc.mipdf.svc.cluster.local  collectd,hoststats,fileserver,hadoop-util,nfs,mastgateway  10.192.1.8    2
mfs-1c7e2949dbe8c23a01-1.mfs-1c7e2949dbe8c23a01-svc.mipdf.svc.cluster.local  collectd,hoststats,fileserver,hadoop-util,nfs,mastgateway  10.192.0.140  2
[root@mip-bdcs-vm618 ~]#
```

- If the MapR services are not in a healthy state (showing red dots on the service tab for embedded DF, or health status > 2 for Picasso services), troubleshoot the Data Fabric cluster. Data Fabric cluster troubleshooting is not covered by this article, but plenty of resources are available at https://docs.datafabric.hpe.com and https://docs.containerplatform.hpe.com.

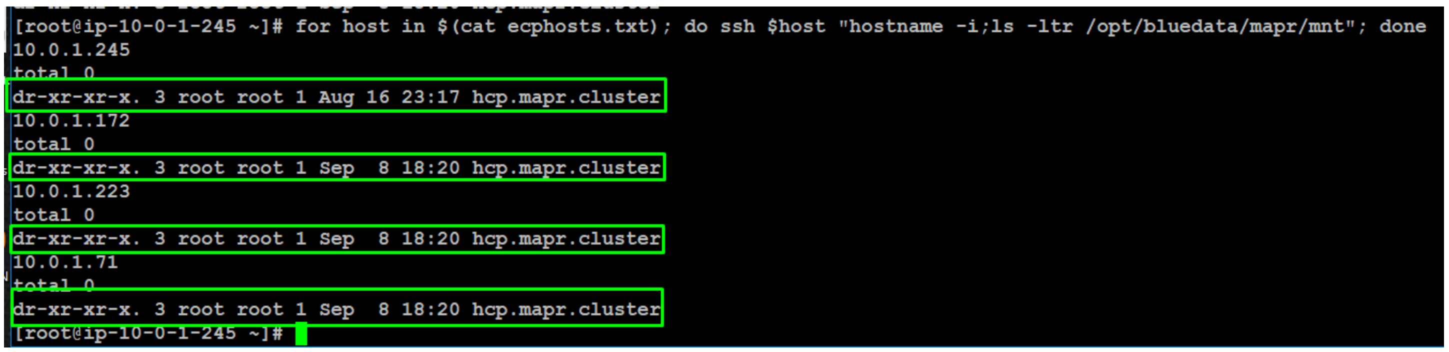
3. Check and fix the MapR Posix Client on each ECP Platform host

Red dots under the Mount Point or Posix Client columns in the EPIC or Kubernetes service status tabs indicate a problem with the mount from that host to the Data Fabric cluster



Even when the MountPoint and PosixClient services statues show green, it is still worth checking that the /opt/bluedata/mapr/mnt mount point is working correctly on each host. From the primary controller, run this command:

```
for host in $(cat ecphosts.txt;cat k8shosts.txt); do ssh $host "hostname
-i;ls -ltr /opt/bluedata/mapr/mnt"; done
```



```
[root@ip-10-0-1-245 ~]# for host in $(cat ecphosts.txt); do ssh $host "hostname -i;ls -ltr /opt/bluedata/mapr/mnt"; done
10.0.1.245
total 0
dr-xr-xr-x. 3 root root 1 Aug 16 23:17 hcp.mapr.cluster
10.0.1.172
total 0
dr-xr-xr-x. 3 root root 1 Sep  8 18:20 hcp.mapr.cluster
10.0.1.223
total 0
dr-xr-xr-x. 3 root root 1 Sep  8 18:20 hcp.mapr.cluster
10.0.1.71
total 0
dr-xr-xr-x. 3 root root 1 Sep  8 18:20 hcp.mapr.cluster
[root@ip-10-0-1-245 ~]#
```

Each host should list the name of the Data Fabric cluster as the contents of its /opt/bluedata/mapr/mnt directory. For embedded Data Fabric, the cluster name is always 'hcp.mapr.cluster'; for Data Fabric on Kubernetes, the cluster name will have been chosen by the user at cluster creation time, and can be determined by checking the namespace name as shown above.

- When the cluster name is listed, the mapr-posix-client-basic systemd service is correctly running inside that host's epic-mapr docker container. If this is correct for all hosts, move on to the next step.
- When the cluster name is not listed, the Mapr Posix Client service is not working properly. For each host in this state, take these steps:
    - SSH into the host
    - Connect into the epic-mapr container: *bdmapr --root bash* (if this fails, check that the container is running with docker ps –a and restart it if necessary: *docker start epic-mapr*)
    - Inside the epic-mapr container, run

```
systemctl stop mapr-posix-client-basic && umount /mapr/mnt &&
systemctl reset-failed mapr-posix-client-basic && systemctl
start mapr-posix-client-basic
```

    - Exit from the container and run *ls –ltr /opt/bluedata/mapr/mnt* on the host to verify that the Data Fabric cluster name is now listed.

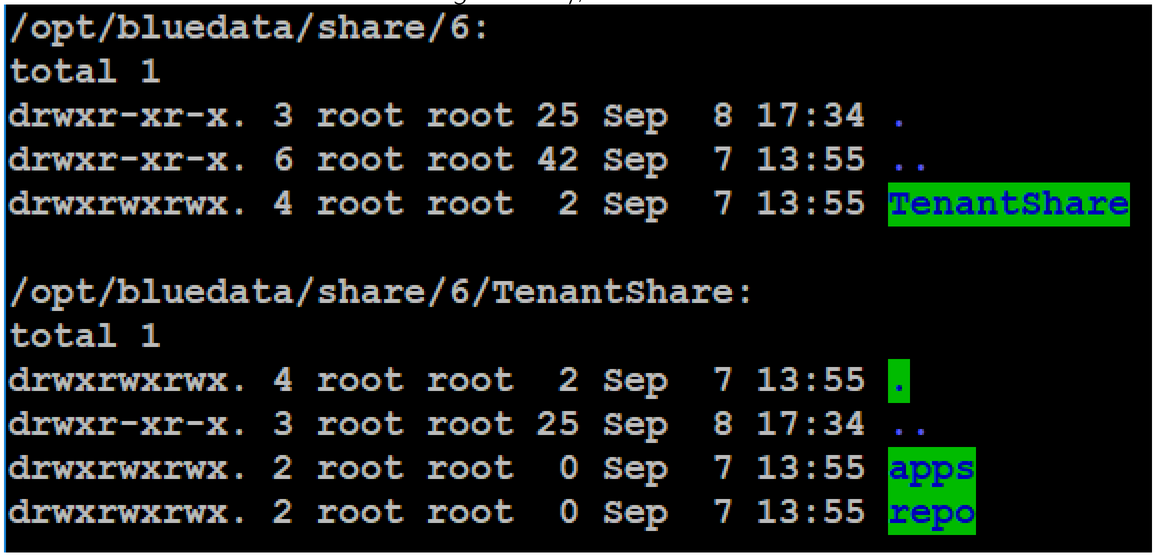
4. Check, clean up and redo the Tenant Storage mount points on EPIC hosts

In this step, we check the mount points under */opt/bluedata/share/<tenant id>* on each EPIC host (that is, controller or EPIC worker). We are looking to see if the mountpoints (a) exist at all; (b) are properly mounted, unmounted or hung; (c) if unmounted, that no local contents exist under them, because any local contents will cause the remounting process to fail.

1. Check to see if any of the /opt/bluedata/share/* mounts are hung on any of the EPIC hosts by looking for 'Transport endpoint not connected' messages:

```
for host in $(cat ecphosts.txt); do ssh $host "hostname -i; ls -laR
/opt/bluedata/share"; done
```

Where the mounts are working correctly, we will see:

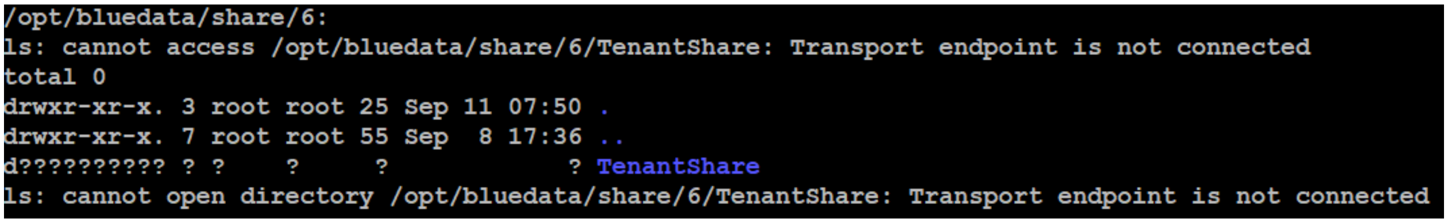


```
/opt/bluedata/share/6:
total 1
drwxr-xr-x. 3 root root 25 Sep  8 17:34 .
drwxr-xr-x. 6 root root 42 Sep  7 13:55 ..
drwxrwxrwx. 4 root root  2 Sep  7 13:55 TenantShare

/opt/bluedata/share/6/TenantShare:
total 1
drwxrwxrwx. 4 root root  2 Sep  7 13:55 .
drwxr-xr-x. 3 root root 25 Sep  8 17:34 ..
drwxrwxrwx. 2 root root  0 Sep  7 13:55 apps
drwxrwxrwx. 2 root root  0 Sep  7 13:55 repo
```

....and possibly significantly more output too, because this ls command was run recursively.

Where the mounts are not connected, we will see:



```
/opt/bluedata/share/6:
ls: cannot access /opt/bluedata/share/6/TenantShare: Transport endpoint is not connected
total 0
drwxr-xr-x. 3 root root 25 Sep 11 07:50 .
drwxr-xr-x. 7 root root 55 Sep  8 17:36 ..
d?????????? ? ?    ?     ?            ? TenantShare
ls: cannot open directory /opt/bluedata/share/6/TenantShare: Transport endpoint is not connected
```

For any locations that show 'Transport endpoint not connected', SSH into that host and `umount -l <location where transport endpoint is not connected>`. For example:

```
# umount -l /opt/bluedata/share/15/TenantShare
```

Alternatively, unmount all the mount points – there is no harm in doing this because you will remount them all again in a later step:

```
for host in $(cat ecphosts.txt); do ssh $host "hostname -i;umount -l
/opt/bluedata/share/*"; done
```

2. Check the contents of */opt/bluedata/share/<tenant id>/** on each host and delete any contents which exist locally (i.e not as a result of an existing mount point of Data Fabric). To do this, first run the command:

```
for host in $(cat ecphosts.txt); do ssh $host "hostname -i;ls -ltr
/opt/bluedata/share/*/*; cat /proc/mounts | grep
'TenantShare\|project_repo'"; done
```

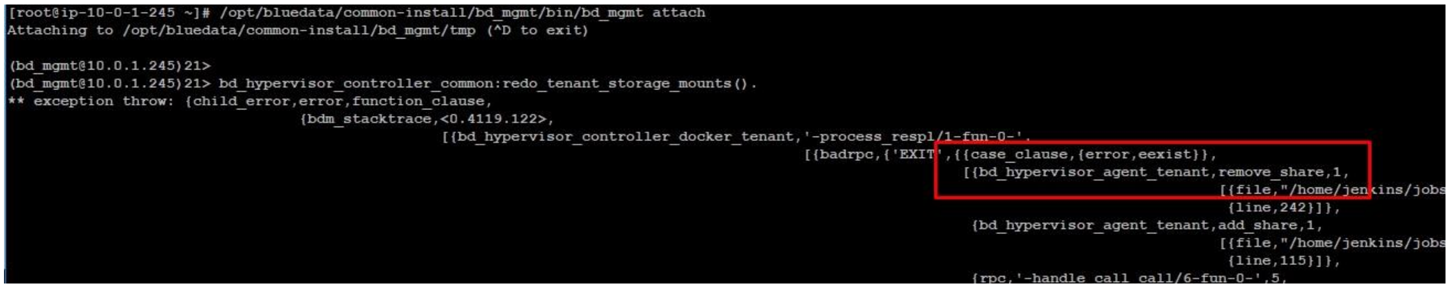
If any of the TenantShare and project_repo directories listed by the above step have contents but are *unmounted*, (i.e the TenantShare and project_repo directories for that tenant do *not* show up in the output of the cat /proc/mounts command), those contents are stored on the local file system. This will prevent the tenant storage mounts from being successfully recreated. Delete any such local contents, or move them to a backup location out of the */opt/bluedata* path. This situation is most likely to happen on the primary or shadow controller host, because a common reason for locally-stored contents being present under /opt/bluedata/share is that users have uploaded files to Tenant Storage via the ECP web UI during a period when the Data Fabric cluster is unmounted from the ECP controller host.

To summarize the purpose of this step: if you find that any TenantShare or project_repo directories are unmounted, but are not empty, empty the directories before continuing. Don't delete any contents from directories that are correctly mounted. No action is needed for directories that are unmounted and empty.

3. Recreate the Tenant Storage mounts on EPIC hosts by running these 5 commands from the primary controller (as the ECP install user, which you can confirm by running the command *cat /etc/bluedata/bluedata.conf | grep user*). You can copy/paste them all as a single unit. Ensure that all quotes and backticks are preserved exactly as shown.

```
ERTS_PATH=/opt/bluedata/common-install/bd_mgmt/erts-*/bin
NODETOOL=/opt/bluedata/common-install/bd_mgmt/bin/nodetool
NAME_ARG=`egrep '^-s?name' $ERTS_PATH/../../releases/1/vm.args`
RPCCMD="$ERTS_PATH/escript $NODETOOL $NAME_ARG rpcterms"
$RPCCMD bd_hypervisor_controller_common redo_tenant_storage_mounts
```

If successful, the output of the final command will be a single **'ok'**. You might also see a single-line error message related to Kubernetes, for example *'{error,[{"kubernetes message","Conflict"}]}'*. At this stage, that is not a problem – we have not started checking the Kubernetes hosts yet. However, if you see a stack trace as shown in the screen shot below, it means that some locally-stored files or directories are still present under the TenantShare and/or project_repo directories on one or more ECP/EPIC hosts. In this case, repeat the previous step to find and delete those contents, then repeat this step until the stack trace is no longer generated when you run the *redo_tenant_storage_mounts* command.



```
[root@ip-10-0-1-245 ~]# /opt/bluedata/common-install/bd_mgmt/bin/bd_mgmt attach
Attaching to /opt/bluedata/common-install/bd_mgmt/tmp (^D to exit)

(bd_mgmt@10.0.1.245)21>
(bd_mgmt@10.0.1.245)21> bd_hypervisor_controller_common:redo_tenant_storage_mounts().
** exception throw: {child_error,error,function_clause,
                      {bdm_stacktrace,<0.4119.122>,
                           [{bd_hypervisor_controller_docker_tenant,'-process_respl/1-fun-0-',
                                               [{badrpc,{'EXIT',{{case_clause,{error,eexist}},
                                                    [{bd_hypervisor_agent_tenant,remove_share,1,
                                                                              [{file,"/home/jenkins/jobs
                                                                               {line,242}]},
                                                     {bd_hypervisor_agent_tenant,add_share,1,
                                                                              [{file,"/home/jenkins/jobs
                                                                               {line,115}]},
                                                     {rpc,'-handle_call_call/6-fun-0-',5,
```

4. At this point, the tenant storage mounts should have been recreated successfully on all ECP/EPIC hosts (that is, platform controllers and EPIC workers). A */cat/proc/mounts | grep posix* command on each of these hosts should return a mount for TenantShare in each tenant ID, and project_repo for ML Ops tenants; and you should be able to access those directories without seeing any 'transport endpoint not connected' errors.

5. Phase 2: Check the Kubernetes hosts

Note: On Kubernetes hosts, the directories under /opt/bluedata/share are named for tenant names, not tenant ID numbers. This is different to the directory naming convention used on EPIC hosts. Occasionally, you will see tenant ID numbers at /opt/bluedata/share on Kubernetes hosts; these are generated redundantly by the redo_tenant_storage_mounts procedure in Phase 1, and can be manually deleted or ignored (no posix mounts will be created to these directories).

Before starting this step, ensure that the MapR Posix Client is working correctly on all Kubernetes hosts. The procedure for checking and fixing this was given in step 3. Next, check the Kubernetes hosts for 'transport endpoint not connected' errors:

```
for host in $(cat k8shosts.txt); do ssh $host "hostname -i;ls -laR
/opt/bluedata/share"; done
```

If any 'Transport endpoint not connected' errors show up, unmount the affected mount points using a umount command. In fact, it is no problem to unmount all the Tenant Storage locations on the Kubernetes hosts, because the hpecp-fsmount pods will automatically recreate the all correct mount points (discussed in the next step).

```
for host in $(cat k8shosts.txt); do ssh $host "hostname -i;umount -l
/opt/bluedata/share/*/*"; done
```

Next, connect to the Kubernetes master (or any host capable of running kubectl commands as the admin of your clister) and restart the hpecp fsmount pods on all the hosts:

```
kubectl rollout restart daemonset/hpecp-fsmount -n hpecp
```

You can check that the new hpecp fsmount pods come to Running state with a *kubectl get po –n hpecp command*, and you can also see the attempts to mount Tenant Storage for each tenant by describing the hpecpfsmount objects for each tenant (*kubectl get hpecpfsmounts –A* and then *kubectl describe <some hpecpfsmount> -n <some tenant>*). If any of the underlying tenant storage volumes on the Data Fabric has been deleted, the hpecpfsmount events will show that, and you can fix the problem by manually recreating the volumes at the correct location in the DF. When the hpecp fsmount pods are running, re-check that the expected mounts have been created and that they are accessible from the /opt/bluedata/share directories:

```
for host in $(cat k8shosts.txt); do ssh $host "hostname -i;ls -ltr
/opt/bluedata/share/*/*; cat /proc/mounts | grep
'TenantShare\|project_repo'"; done
```

6. Conclusion

By this stage, you have checked that the MapR Posix Client is running correctly inside the epic-mapr container on all ECP Platform hosts. On EPIC hosts (including platform controllers) you unmounted any 'stuck' Tenant Storage mount points, remove any local files that would have blocked the recreation of the Tenant Storage mounts, and then recreated the tenant storage mounts by running a Bluedata management command. On Kubernetes hosts, you performed the same checks and then forced Tenant Storage mounts to be recreated by restarting the hpecp fsmounts daemon set.