opHA 3 Troubleshooting

Peer cannot be discovered

From the Primary, we can run the discover using the cli tool with debug enabled to get further information:

```
bin/opha-cli.exe act=discover url_base=http://poller username=xxxxxx password=xxxxxx debug=9
```

If there is any connectivity error, you can test the connectivity from the poller to the Primary using curl (This is an example with default credentials):

```
curl -k -d 'username=nmis&password=nm1888' -H 'DNT: 1' -H 'Accept: application/json' http://poller/omk/opCore
/login
```

The response should be something like this:

```
{"message":"Authenticated as user nmis","ok":1}
```

If this is a login error, please review the credentials used are correct in the poller. You can check the file log/auth.log from the poller.

Another common error is a misconfiguration in the poller server. The poller need to be able to generate the registry, a document with discovery information. For this, it needs to have defined both properties opha_url_base and opha_hostname (Or both, can be null if you are not using https). When the registry cannot be created you can see this error in opHA logs:

```
[error] Capability Registration failed: Invalid capability document!
```

You can check if the registry is created in the following url:

```
http://host/en/omk/opHA/api/v1/registry
```

opHA cluster_id already exist!

Check the Primary and all other pollers nmis configuration for the cluster_id property. It can be found in /usr/local/nmis9/conf/Config.nmis. This one should be unique per server, so in case one of the pollers has a repeated one you can delete the property in the configuration file, restart the nmis9d daemon, and nmis will generate a new one.

NOTE: The Opmantek VM ships with the cluster_id preset. In instances where multiple Opmantek VMs are deployed, and connected using opHA, the cluster_id will need to be changed/update on each server. This should be done PRIOR to adding any nodes/devices to the cluster.

Please, notice that in case the server has nodes already, the nodes should be exported and imported again with localised_ids once the cluster_id is changed, as the nodes information won't have the same cluster_id attribute and they will be treated as remote nodes (They cannot be edited, or polled, as an example).

```
localise_ids=true

(default: false), then the cluster id is rewritten to match the local nmis installation
```

After the change, omkd daemon needs to be restarted.

401 Error from the poller

opHA uses user/password to access the registry data from the poller, but once the poller has been discover, it uses a token for authentication. So, we should have enabled the authentication method "token" in the poller.

Check if in <omk_dir>/conf/opCommon.json we have the following (Being X 1, 2 or 3, not matter the order):

```
'auth_method_X' => 'token',
```

Also, the property **auth_token_key** should be set up in the poller configuration.

More information about **authorisation/authentication** in opHA.

Other causes

Another potential cause of a 401 error is the linux system clock on the Poller being too far out of sync with the Primary. In this case, you may see an error similar to below, where the auth.log is showing an info message about the token for the user being expired. Ensuring the clocks are in sync will resolve this issue.

```
2022-04-07 07:50:33.67148] [3842141] [info] 3842141 Token authentication failure, expired token for username=nmis
2022-04-07 07:50:33.67165] [3842141] [info] opmantek_authenticated: user not authenticated, redirecting to login for opCore
```

Configuring a poller over https

From the Primary, we can initiate discovery of a peer using the url https://servername. (using SSL/TLS).

But, we will not be able to query the poller as the poller will report it's url as http: . To force the Primary to use HTTPS to the poller we must have the configuration item opha_url_base set to https otherwise, it won't work.

This can be set in <omk_dir>/conf/opCommon.json in the **poller**:

```
"opha_url_base" : "https://servername.domain.com",
```

If we set the url to https://servername in the discover, the poller is going to send its registry data to the Primary, and the Primary will get the correct url_base for the peer from that information.

If the opha_url_base is blank the Primary will swap the https:// URL for http://

Verify Hostname and URL Base configuration

To verify that the url base and hostnames are correct.

```
grep -E "(opha|opevents|opflow|opcharts|opconfig|opflowsp|opreports|omkd)_(hostname|url_base)" /usr/local/omk
/conf/opCommon.json
```

The results should be that omkd and opha values are set but not others.

```
      "opflowsp_url_base" : "",
      "opflowsp_hostname" : "",
      "opflow_opcharts_url_base" : "",
      "opflow_url_base" : "",
      "opflow_hostname" : "",
      "opcharts_url_base" : "",
      "opcharts_hostname" : "",
      "opconfig_hostname" : "",
      "opconfig_url_base" : "",
      "omkd_url_base" : "",
      "opevents_hostname" : "",
      "opevents_url_base" : "",
      "opreports_url_base" : "",
      "opreports_hostname" : "",
      "opreports_opcharts_url_base" : "http://127.0.0.1:8042",
      "opha_hostname" : "lab-ms-primary",
      "opha_url_base" : "https://lab-ms-primary.opmantek.net",
```

Some data is not updated in the Primary

opHA has a new feature to synchronise only the data that has being added/modified since the last synchronisation. In case some data is not modified, we can perform a force synchronisation, adding some parameters to update only the required data types and nodes:

```
bin/opha-cli.pl act=pull data_types=[nodes|latest_data|...] peers=[nodeNames] force=t
```

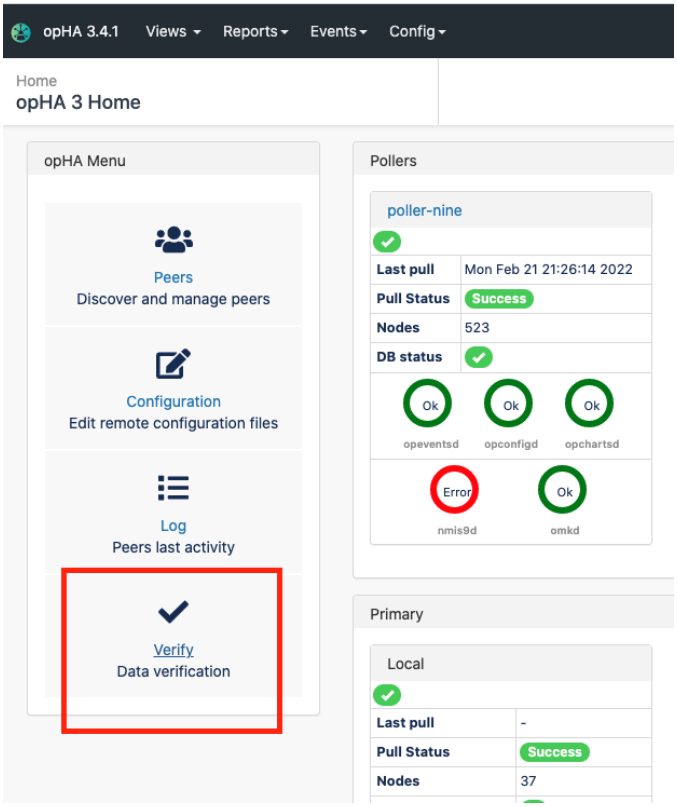
Duplicated nodes

Different situations have been identified causing this issue:

- If the same node name exists in more than one poller, and the configuration item **opevents_auto_create_nodes** is true, a new Local node will be created in the primary server. This is because, the event is just identified by a node name, and the primary cannot choose which of the remote nodes assigned the event.
- If there are **two** Main primary servers: This situation can cause chaos in the environment, as both primaries will change the nodes from the pollers.
- Also, if some catchall data is duplicated in the primary, we would be looking some nodes as duplicates in opCharts.

It is possible to check if there are duplicates:

- In opHA GUI in the Main Page in Verify



- In opHA cli with the command:

```
/usr/local/omk/bin/opha-cli.pl act=data_verify
```

Premature Connection Close

The web server closed the connection before the user agent could receive the whole response or that the user agent got destroyed, which forces all connections to be closed immediately.

We can tune the following values to prevent this error to happen:

In the primary server

If the poller is taking too long to respond, we can increase the value of **omkd_inactivity_timeout** in <omk_dir>/conf/opCommon.json.

Restart omkd is required after the change of this parameter.

In the poller

If the request is taking too long, we can adjust the page size in <omk_dir>/conf/opCommon.json. For each data type, we can adjust the number of elements to send in the following values:

```
      "opha_transfer_chunks" : {
         "status" : 5000,
         "nodes" : 5000,
         "events" : 5000,
         "latest_data" : 5000,
         "inventory" : 5000
...
      },
```

If the request is taking too long, we can decrease the number of elements for each datatype.

If we have an unstable network, we can increase the number to reduce the number of requests.

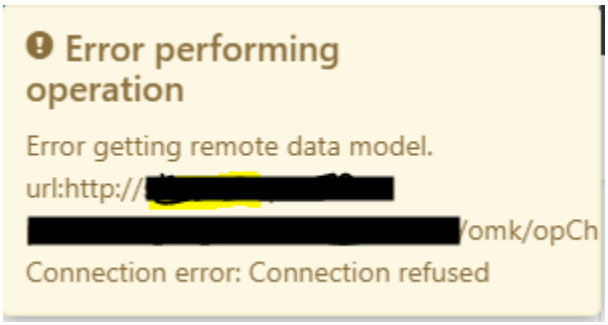
Restart omkd is required after the change of this parameter.

Error performing operation - Error getting remote data model

There are several issues which can result in seeing the error message "Error getting remote data model" in the GUI, these are to do with HTTP/HTTPS connectivity and authorisation settings in the primary and poller servers.

Connection error: Connection Refused.

You might be seeing an error in the GUI as follows:



Ensure that the opHA API user is configured to be the same as the peer setup, the user should exist in the NMIS Users.nmis file and have permissions assigned, by default this is set to omkapiha, check <omk_dir>/conf/opCommon.json

```
"opha_api_user": "omkapiha",
```

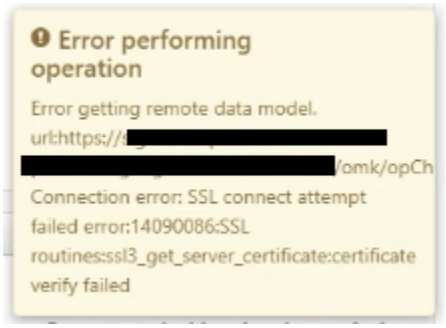
After changing restart the daemon.

```
systemctl restart omkd
```

Note: This error can also occur if you upgrade opHA and do not accept the EULA on the pollers. Double check the status of the pollers from the main opHA dashboard on the primary.

Connection error: SSL connect attempt failed error

The error in the GUI would be as follows:



In this case, the SSL certificate is likely to be a local certificate authority (CA), or you might be using self-signed SSL certificates, in which case you will need to let the applications know it is OK.

On the primary server change the following configuration option to reflect the same as below to <omk_dir>/conf/opCommon.json in the opHA section.

```
"opha_allow_insecure" : "1",
```

You may also need to enable the below as well on the primary server:

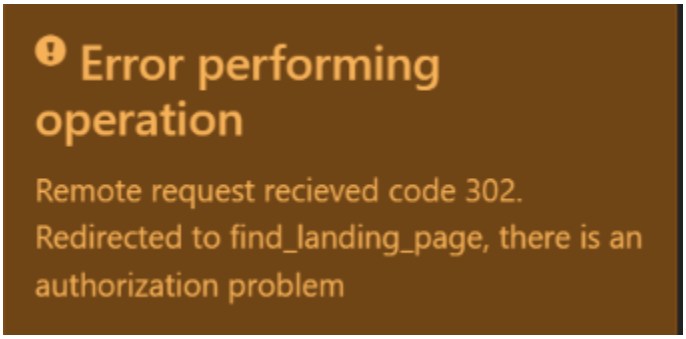
```
"omk_ua_insecure" : "1",
```

After changing restart the daemon.

```
systemctl restart omkd
```

Connection error: there is an authorization problem

In the GUI, you observe the following error:



You should ensure that the opHA API user that is defined in opCommon.json on both the Primary/Main Primary and the poller(s) is the same user, and that this user exists in the Users.nmis table. By default the configured user is "omkapiha".

```
"opha_api_user" : "omkapiha",
```

Teapot error: error saving node to remote

in the GUI if you observe the following error:

blocked URL

Check the /usr/local/omk/log/opDaemon.log and if you see the following lines:

> [debug] current_app_log: bad log, application_key missing
>
> [error] NodeData::update_resource Error creating node in remote. Reason: 418 I'm a teapot

[debug] 418 I'm a teapot (0.127757s, 7.827/s)

Validate that pollers and primary have the same types set in nmis9/conf/Config.nmis for each of the following: 'nodetype_list', 'nettype_list', 'roletype_list'

An easy way to do this is using the patch_config.pl tool:

```
/usr/local/nmis9/admin/patch_config.pl -r /usr/local/nmis9/conf/Config.nmis roletype_list
/usr/local/nmis9/admin/patch_config.pl -r /usr/local/nmis9/conf/Config.nmis nettype_list
/usr/local/nmis9/admin/patch_config.pl -r /usr/local/nmis9/conf/Config.nmis nodetype_list
```

If mismatched then update, restart daemons (nmis9d and omkd) then rediscover poller.