Troubleshooting

Introduction to Troubleshooting

Troubleshooting the Element Installer comes down to knowing a little bit about kubernetes and how to check the status of the various resources. This guide will walk you through some of the initial steps that you'll want to take when things are going wrong.

Known issues

Installer fails and asks you to start firewalld

The current installer will check if you have firewalld installed on your system. It does expect to find firewalld started as a systemd service if it is installed. If it is not started, the installer will terminate with a failure that asks you to start it. We noticed some Linux distributions like SLES15P4, RHEL8 and AlmaLinux8 that have firewalld installed as a default package but not enabled, or started.

If you hit this issue, you don't need to enable and start firewalld. The workaround is to uninstall firewalld, if you are not planning on using it.

On SLES

```
zypper remove firewalld -y
```

On RHEL8 :

```
dnf remove firewalld -y
```

Airgapped installation does not start

If you are using element-enterprise-graphical-installer-2023-03.02-gui.bin and element-enterprise-installer-airgapped-2023-03.02-gui.tar.gz. You might run into an error looking like this :

```
Looking in links: ./airgapped/pip

WARNING: Url './airgapped/pip' is ignored. It is either a non-existing path or lacks a
specific scheme.

ERROR: Could not find a version that satisfies the requirement wheel (from versions: none)
```

```
ERROR: No matching distribution found for wheel
```

The workaround for it is to copy the pip folder from the airgapped directory to ~/.element-enterprise-server/installer/airgapped/pip

install.sh problems

Sometimes there will be problems when running the ansible-playbook portion of the installer. When this happens, you can increase the verbosity of ansible logging by editing `.ansible.rc` in the installer directory and setting:

```
export ANSIBLE_DEBUG=true
export ANSIBLE_VERBOSITY=4
```

and re-running the installer. This will generate quite verbose output, but that typically will help pinpoint what the actual problem with the installer is.

Problems post-installation

Checking Pod Status and Getting Logs

- In general, a well-functioning Element stack has at it's minimum the following containers (or pods in kubernetes language) running:

  ```
  [user@element2 ~]$ kubectl get pods -n element-onprem
  kubectl get pods -n element-onprem
  NAME                                                        READY   STATUS
  RESTARTS   AGE
  first-element-deployment-element-web-6cc66f48c5-lvd7w       1/1     Running
  0          4d20h
  first-element-deployment-element-call-c9975d55b-dzjw2       1/1     Running
  0          4d20h
  integrator-postgres-0                                       3/3     Running
  0          4d20h
  synapse-postgres-0                                          3/3     Running
  0          4d20h
  first-element-deployment-integrator-59bcfc67c5-jkbm6        3/3     Running
  0          4d20h
  adminbot-admin-app-element-web-c9d456769-rpk9l              1/1     Running
  0          4d20h
  ```

```
auditbot-admin-app-element-web-5859f54b4f-8lbng              1/1     Running
0          4d20h
first-element-deployment-synapse-redis-68f7bfbdc-wht9m       1/1     Running
0          4d20h
first-element-deployment-synapse-haproxy-7f66f5fdf5-8sfkf    1/1     Running
0          4d20h
adminbot-pipe-0                                              1/1     Running
0          4d20h
auditbot-pipe-0                                              1/1     Running
0          4d20h
first-element-deployment-synapse-admin-ui-564bb5bb9f-87zb4   1/1     Running
0          4d20h
first-element-deployment-groupsync-0                         1/1     Running
0          20h
first-element-deployment-well-known-64d4cfd45f-l9kkr         1/1     Running
0          20h
first-element-deployment-synapse-main-0                      1/1     Running
0          20h
first-element-deployment-synapse-appservice-0                1/1     Running
0          20h
```

The above `kubectl get pods -n element-onprem` is the first place to start. You'll notice in the above, all of the pods are in the `Running` status and this indicates that all should be well. If the state is anything other than "Running" or "Creating", then you'll want to grab logs for those pods. To grab the logs for a pod, run:

```
kubectl logs -n element-onprem <pod name>
```

replacing `<pod name>` with the actual pod name. If we wanted to get the logs from synapse, the specific syntax would be:

```
kubectl logs -n element-onprem first-element-deployment-synapse-main-0
```

and this would generate logs similar to:

```
 2022-05-03 17:46:33,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2887 - Dropped 0 items from caches
2022-05-03 17:46:33,375 - synapse.storage.databases.main.metrics - 471 - INFO -
generate_user_daily_visits-289 - Calling _generate_user_daily_visits
2022-05-03 17:46:58,424 - synapse.metrics._gc - 118 - INFO - sentinel - Collecting gc
1
2022-05-03 17:47:03,334 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2888 - Dropped 0 items from caches
```

```
2022-05-03 17:47:33,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2889 - Dropped 0 items from caches
2022-05-03 17:48:03,333 - synapse.util.caches.lrucache - 154 - INFO -
LruCache._expire_old_entries-2890 - Dropped 0 items from caches
```

- Again, for every pod not in the `Running` or `Creating` status, you'll want to use the above procedure to get the logs for Element to look at.
- If you don't have any pods in the `element-onprem` namespace as indicated by running the above command, then you should run:

```
[user@element2 ~]$ kubectl get pods -A
NAMESPACE            NAME                                        READY   STATUS
RESTARTS   AGE
kube-system          calico-node-2lznr                           1/1     Running
0          8d
kube-system          calico-kube-controllers-c548999db-s5cjm     1/1     Running
0          8d
kube-system          coredns-5dbccd956f-glc8f                    1/1     Running
0          8d
kube-system          dashboard-metrics-scraper-6b6f796c8d-8x6p4  1/1     Running
0          8d
ingress              nginx-ingress-microk8s-controller-w8lcn     1/1     Running
0          8d
cert-manager         cert-manager-cainjector-6586bddc69-9xwkj    1/1     Running
0          8d
kube-system          hostpath-provisioner-78cb89d65b-djfq5       1/1     Running
0          8d
kube-system          kubernetes-dashboard-765646474b-5lhxp       1/1     Running
0          8d
cert-manager         cert-manager-5bb9dd7d5d-cg9h8               1/1     Running
0          8d
container-registry   registry-f69889b8c-zkhm5                    1/1     Running
0          8d
cert-manager         cert-manager-webhook-6fc8f4666b-9tmjb       1/1     Running
0          8d
kube-system          metrics-server-5f8f64cb86-f876p             1/1     Running
0          8d
jitsi                sysctl-jvb-vs9mn                            1/1     Running
0          8d
jitsi                shard-0-jicofo-7c5cd9fff5-qrzmk             1/1     Running
0          8d
```

```
jitsi              shard-0-web-fdd565cd6-v49ps              1/1     Running
0          8d
jitsi              shard-0-web-fdd565cd6-wmzpb              1/1     Running
0          8d
jitsi              shard-0-prosody-6d466f5bcb-5qsbb         1/1     Running
0          8d
jitsi              shard-0-jvb-0                            1/2     Running
0          8d
operator-onprem    element-operator-controller-manager-...  2/2     Running
0          4d
updater-onprem     element-updater-controller-manager-...   2/2     Running
0          4d
element-onprem     first-element-deployment-element-web-... 1/1     Running
0          4d
element-onprem     first-element-deployment-element-call-...  1/1     Running
0          4d
element-onprem     integrator-postgres-0                    3/3     Running
0          4d
element-onprem     synapse-postgres-0                       3/3     Running
0          4d
element-onprem     first-element-deployment-integrator-...  3/3     Running
0          4d
element-onprem     adminbot-admin-app-element-web-...       1/1     Running
0          4d
element-onprem     auditbot-admin-app-element-web-...       1/1     Running
0          4d
element-onprem     first-element-deployment-synapse-redis-... 1/1     Running
0          4d
element-onprem     first-element-deployment-synapse-haproxy-.. 1/1     Running
0          4d
element-onprem     adminbot-pipe-0                          1/1     Running
0          4d
element-onprem     auditbot-pipe-0                          1/1     Running
0          4d
element-onprem     first-element-deployment-synapse-admin-ui-. 1/1     Running
0          4d
element-onprem     first-element-deployment-groupsync-0     1/1     Running
0          20h
element-onprem     first-element-deployment-well-known-...  1/1     Running
```

```
0          20h
element-onprem      first-element-deployment-synapse-main-0     1/1     Running
0          20h
element-onprem      first-element-deployment-synapse-appservice-0 1/1   Running
0          20h
```

- This is the output from a healthy system, but if you have any of these pods not in the `Running` or `Creating` state, then please gather logs using the following syntax:

  ```
  kubectl logs -n <namespace> <pod name>
  ```

- So to gather logs for the kubernetes ingress, you would run:

  ```
  kubectl logs -n ingress nginx-ingress-microk8s-controller-w8lcn
  ```

  and you would see logs similar to:

  ```
  I0502 14:15:08.467258       6 leaderelection.go:248] attempting to acquire leader
  lease ingress/ingress-controller-leader...
  I0502 14:15:08.467587       6 controller.go:155] "Configuration changes detected,
  backend reload required"
  I0502 14:15:08.481539       6 leaderelection.go:258] successfully acquired lease
  ingress/ingress-controller-leader
  I0502 14:15:08.481656       6 status.go:84] "New leader elected" identity="nginx-
  ingress-microk8s-controller-n6wmk"
  I0502 14:15:08.515623       6 controller.go:172] "Backend successfully reloaded"
  I0502 14:15:08.515681       6 controller.go:183] "Initial sync, sleeping for 1
  second"
  I0502 14:15:08.515705       6 event.go:282] Event(v1.ObjectReference{Kind:"Pod",
  Namespace:"ingress", Name:"nginx-ingress-microk8s-controller-n6wmk", UID:"548d9478-
  094e-4a19-ba61-284b60152b85", APIVersion:"v1", ResourceVersion:"524688",
  FieldPath:""}): type: 'Normal' reason: 'RELOAD' NGINX reload triggered due to a
  change in configuration
  ```

  Again, for all pods not in the `Running` or `Creating` state, please use the above method to get log data to send to Element.

Other Commands of Interest

Some other commands that may yield some interesting data while troubleshooting are:

- **Verify DNS names and IPs in certificates**
  In the `certs` directory under the configuration directory, run:

  ```
  for i in $(ls *crt); do echo $i && openssl x509 -in $i -noout -text | grep DNS; done
  ```

This will give you output similar to:

```
local.crt
              DNS:local, IP Address:192.168.122.118, IP Address:127.0.0.1
synapse2.local.crt
              DNS:synapse2.local, IP Address:192.168.122.118, IP Address:127.0.0.1
```

and this will allow you to verify that you have the right host names and IP addresses in your certificates.

- **Show hostname to IP mappings from within a pod**
  Run:

  ```
  kubectl exec -n element-onprem <pod_name> -- getent hosts
  ```

  and you will see output similar to:

  ```
  127.0.0.1       localhost
  127.0.0.1       localhost ip6-localhost ip6-loopback
  10.1.241.30     instance-hookshot-0.instance-hookshot.element-
  onprem.svc.cluster.local instance-hookshot-0
  192.168.122.5   ems.onprem element.ems.onprem hs.ems.onprem adminbot.ems.onprem
  auditbot.ems.onprem integrator.ems.onprem hookshot.ems.onprem admin.ems.onprem
  eleweb.ems.onprem
  ```

  This will help you troubleshoot host resolution.
- **Show all persistent volumes and persistent volume claims for the `element-onprem` namespace:**

  ```
  kubectl get pv -n element-onprem
  ```

  This will give you output similar to:

  ```
  NAME                                       CAPACITY   ACCESS MODES   RECLAIM
  POLICY   STATUS   CLAIM
  STORAGECLASS        REASON   AGE
  pvc-fc3459f0-eb62-4afa-94ce-7b8f8105c6d1   20Gi       RWX
  Delete           Bound    container-registry/registry-claim
  microk8s-hostpath            8d
  integrator-postgres                        5Gi        RWO
  Recycle          Bound    element-onprem/integrator-postgres
  microk8s-hostpath            8d
  synapse-postgres                           5Gi        RWO
  Recycle          Bound    element-onprem/synapse-postgres
  microk8s-hostpath            8d
  hostpath-synapse-media                     50Gi       RWO
  Recycle          Bound    element-onprem/first-element-deployment-synapse-media
  ```

```
microk8s-hostpath            8d
adminbot-bot-data                          10M        RWO
Recycle          Bound    element-onprem/adminbot-bot-data
microk8s-hostpath            8d
auditbot-bot-data                          10M        RWO
Recycle          Bound    element-onprem/auditbot-bot-data
microk8s-hostpath            8d
```

~~~

- **Show the synapse configuration:**

 For installers prior to 2022-05.06, use:
 ```bash
 kubectl describe cm -n element-onprem first-element-deployment-synapse-shared
~~~

and this will return output similar to:

```
send_federation: True
start_pushers: True
turn_allow_guests: true
turn_shared_secret: n0t4ctuAllymatr1Xd0TorgSshar3d5ecret4obvIousreAsons
turn_uris:
- turns:turn.matrix.org?transport=udp
- turns:turn.matrix.org?transport=tcp
turn_user_lifetime: 86400000
```

For the 2022-05.06 installer and later, use:

```
kubectl -n element-onprem get secret synapse-secrets -o yaml 2>&1 | grep shared.yaml | awk -F
'shared.yaml: ' '{print $2}' - | base64 -d
```

For the 2023-05.05 installer and later, use:

```
kubectl get secrets/first-element-deployment-synapse-secrets -n element-onprem -o yaml | grep
shared.yaml | awk '{ print $2}' | base64 -d
```

and you will get output similar to the above.

- **Show the Element Web configuration:**

```
kubectl describe cm -n element-onprem app-element-web
```

and this will return output similar to:

```
config.json:
----
{
    "default_server_config": {
        "m.homeserver": {
            "base_url": "https://synapse2.local",
            "server_name": "local"
        }
  },
  "dummy_end": "placeholder",
  "integrations_jitsi_widget_url": "https://dimension.element2.local/widgets/jitsi",
  "integrations_rest_url": "https://dimension.element2.local/api/v1/scalar",
  "integrations_ui_url": "https://dimension.element2.local/element",
  "integrations_widgets_urls": [
      "https://dimension.element2.local/widgets"
  ]
}
```

- **Show the nginx configuration for Element Web: (If using nginx as your ingress controller in production or using the PoC installer.)**

```
kubectl describe cm -n element-onprem app-element-web-nginx
```

and this will return output similar to:

```
  server {
      listen       8080;

      add_header X-Frame-Options SAMEORIGIN;
      add_header X-Content-Type-Options nosniff;
      add_header X-XSS-Protection "1; mode=block";
      add_header Content-Security-Policy "frame-ancestors 'self'";
      add_header X-Robots-Tag "noindex, nofollow, noarchive, noimageindex";

      location / {
          root   /usr/share/nginx/html;
          index  index.html index.htm;
```

```
            charset utf-8;
        }
    }
```

- **Check list of active kubernetes events:**

  ```
  kubectl get events -A
  ```

  You will see a list of events or the message `No resources found`.
- Show the state of services in the `element-onprem` namespace:

  ```
  kubectl get services -n element-onprem
  ```

  This should return output similar to:

  ```
  NAME                             TYPE        CLUSTER-IP       EXTERNAL-IP
  PORT(S)                     AGE
  postgres                         ClusterIP   10.152.183.47    <none>
  5432/TCP                    6d23h
  app-element-web                  ClusterIP   10.152.183.60    <none>
  80/TCP                      6d23h
  server-well-known                ClusterIP   10.152.183.185   <none>
  80/TCP                      6d23h
  instance-synapse-main-headless   ClusterIP   None             <none>
  80/TCP                      6d23h
  instance-synapse-main-0          ClusterIP   10.152.183.105   <none>
  80/TCP,9093/TCP,9001/TCP    6d23h
  instance-synapse-haproxy         ClusterIP   10.152.183.78    <none>
  80/TCP                      6d23h
  ```

- **Show the status of the stateful sets in the `element-onprem` namespace:**

  ```
  kubectl get sts -n element-onprem
  ```

  This should return output similar to:

  ```
  NAME                     READY   AGE
  postgres                 1/1     6d23h
  instance-synapse-main    1/1     6d23h
  ```

- **Show deployments in the `element-onprem` namespace:**

  ```
  kubectl get deploy -n element-onprem
  ```

  This will return output similar to:

```
NAME                        READY   UP-TO-DATE   AVAILABLE   AGE
app-element-web             1/1     1            1           6d23h
server-well-known           1/1     1            1           6d23h
instance-synapse-haproxy    1/1     1            1           6d23h
```

- **Show the status of all namespaces:**

  ```
  kubectl get namespaces
  ```

  which will return output similar to:

  ```
  NAME                   STATUS   AGE
  kube-system            Active   20d
  kube-public            Active   20d
  kube-node-lease        Active   20d
  default                Active   20d
  ingress                Active   6d23h
  container-registry     Active   6d23h
  operator-onprem        Active   6d23h
  element-onprem         Active   6d23h
  ```

- **View the MAU Settings in Synapse:**

  ```
  kubectl get  -n element-onprem secrets/synapse-secrets -o yaml | grep -i shared.yaml
  -m 1| awk -F ': ' '{print $2}' - | base64 -d
  ```

  which will return output similar to:

  ```
  # Local custom settings
  mau_stats_only: true

  limit_usage_by_mau: False
  max_mau_value: 1000
  mau_trial_days: 2

  mau_appservice_trial_days:
    chatterbox: 0

  enable_registration_token_3pid_bypass: true
  ```

- **Redeploy the micro8ks setup**
  It is possible to redeploy microk8s by running the following command as root:

  ```
  snap remove microk8s
  ```

> This command does remove all microk8s pods and related microk8s storage volumes. **Once this command has been run, you need to reboot your server** - otherwise you may have networking issues. Add `--purge` flag to remove the data if disk usage is a concern.
> After the reboot, you can re-run the installer and have it re-deploy microk8s and Element Enterprise On-Premise for you.

Node-based pods failing name resoution

```
05:03:45:601 ERROR [Pipeline] Unable to verify identity configuration for bot-auditbot:
Unknown errcode Unknown error
05:03:45:601 ERROR [Pipeline] Unable to verify identity. Stopping
matrix-pipe encountered an error and has stopped Error: getaddrinfo EAI_AGAIN
synapse.prod.ourdomain
    at GetAddrInfoReqWrap.onlookup [as oncomplete] (node:dns:84:26) {
  errno: -3001,
  code: 'EAI_AGAIN',
  syscall: 'getaddrinfo',
  hostname: 'synapse.prod.ourdomain'
}
```

To see what Hosts are set, try:

```
kubectl exec -it -n element-onprem <pod name> getent hosts
```

So to do this on the adminbot-pipe-0 pod, it would look like:

```
kubectl exec -it -n element-onprem adminbot-pipe-0 getent hosts
```

and return output similar to:

```
127.0.0.1       localhost
127.0.0.1       localhost ip6-localhost ip6-loopback
10.1.241.27     adminbot-pipe-0
192.168.122.5   ems.onprem element.ems.onprem hs.ems.onprem adminbot.ems.onprem
auditbot.ems.onprem integrator.ems.onprem hookshot.ems.onprem admin.ems.onprem
eleweb.ems.onprem
```

Node-based pods failing SSL

```
2023-02-06 15:42:04 ERROR: IrcBridge Failed to fetch roomlist from joined rooms: Error: unable
to verify the first certificate. Retrying
MatrixHttpClient (REQ-13) Error: unable to verify the first certificate
```

```
at TLSSocket.onConnectSecure (_tls_wrap.js:1515:34)
at TLSSocket.emit (events.js:400:28)
at TLSSocket.emit (domain.js:475:12)
at TLSSocket. finishInit (_tls_wrap.js:937:8),
at TLSWrap.ssl.onhandshakedone (_tls_wrap.js:709:12) {
code: 'UNABLE TO VERIFY LEAF SIGNATURE
```

Drop into a shell on the pod

```
kubectl exec -it -n element-onprem adminbot-pipe-0 -- /bin/sh
```

Check it's abililty to send a request to the Synapse server

```
node


require=("http")
request(https://synapse.server/)
```

Default administrator

The installer creates a default administrator `onprem-admin-donotdelete` The Synapse admin user password is defined under the synapse section in the installer

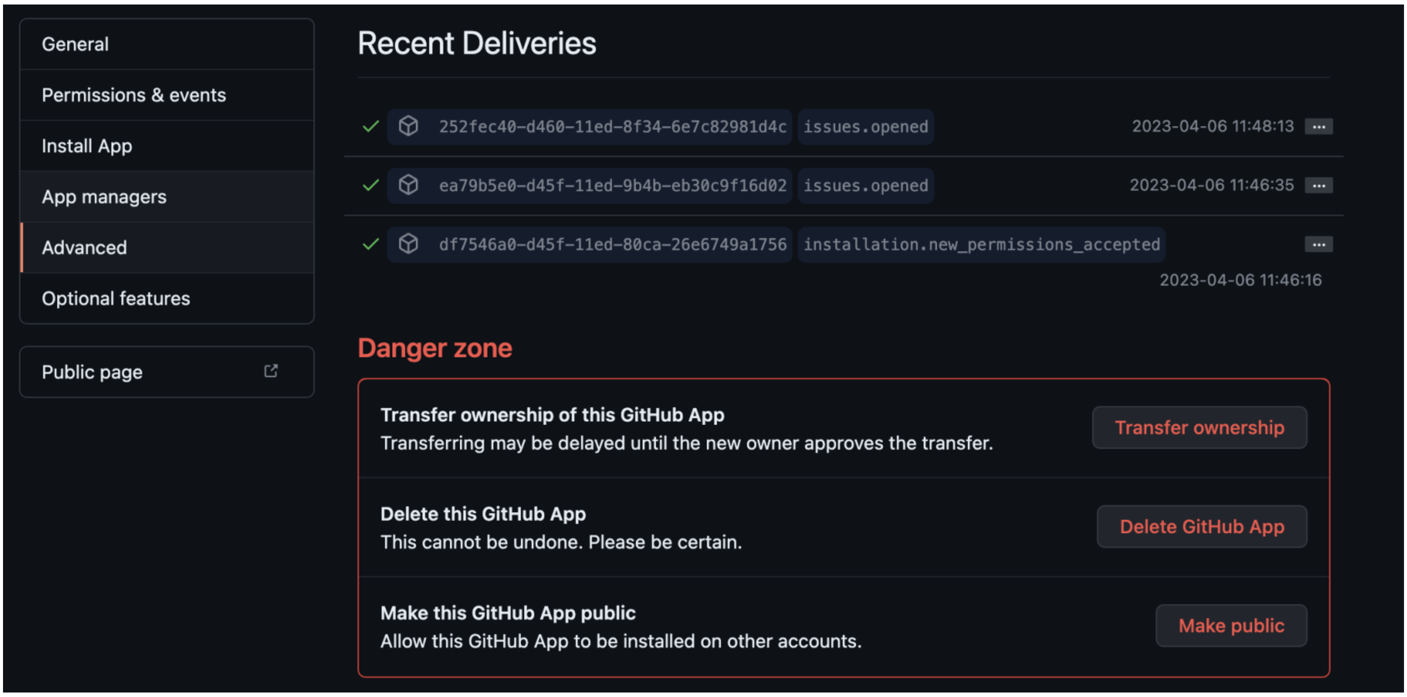
Integration issues

GitHub not sending events

You can trace webhook calls from your GitHub application under `Settings`/`developer settings`/`GitHub Apps`

Select your GitHub App

Click on `Advanced` and you should see queries issues by your app under `Recent Deliveries`



Recent Deliveries

```
252fec40-d460-11ed-8f34-6e7c82981d4c issues.opened 2023-04-06 11:48:13
ea79b5e0-d45f-11ed-9b4b-eb30c9f16d02 issues.opened 2023-04-06 11:46:35
df7546a0-d45f-11ed-80ca-26e6749a1756 installation.new_permissions_accepted 2023-04-06 11:46:16
```

Danger zone

Transfer ownership of this GitHub App
Transferring may be delayed until the new owner approves the transfer.

Delete this GitHub App
This cannot be undone. Please be certain.

Make this GitHub App public
Allow this GitHub App to be installed on other accounts.

Updater and Operator in `ImagePullBackOff` state

Check EMS Image Store Username and Token

Check to see if you can pull the Docker image:

```
kubectl get pods -l app.kubernetes.io/instance=element-operator-controller-manager -n operator-onprem  -o yaml | grep 'image:'
```

grab the entry like `image: gitlab-registry.matrix.org/ems-image-store/standard/kubernetes-operator@sha256:305c7ae51e3b3bfbeff8abf2454b47f86d676fa573ec13b45f8fa567dc02fcd1`

Should look like

```
microk8s.ctr image pull gitlab-registry.matrix.org/ems-image-store/standard/kubernetes-operator@sha256:305c7ae51e3b3bfbeff8abf2454b47f86d676fa573ec13b45f8fa567dc02fcd1  -u <EMS Image Store usenamer>:<EMS Image Store token>
```

Revision #31
Created 3 May 2022 17:41:38 by Karl Abbott
Updated 28 December 2023 11:51:06 by Lutz Lange