



Get Familiar with Docker Enterprise Edition Client Bundles

By [Brian Kaufman](#) September 20 2017 [Twitter](#) [LinkedIn](#) [Reddit](#) [Google+](#) [Facebook](#) [YouTube](#) [Email](#)

◆ [Client Bundles](#), [Docker EE](#), [Docker Enterprise Edition](#), [ucp](#), [universal control plane](#)

Docker Enterprise Edition (EE) is the only Containers as a Service (CaaS) Platform for IT that manages and secures diverse applications across disparate infrastructure, both on-premises and in the cloud.

There's a little mentioned big feature in Docker Enterprise Edition (EE) that seems to always bring smiles to the room once it's displayed. Before I tell you about it, let me first describe the use case. You're a sysadmin managing a Docker cluster and you have the following requirements:

- Different individuals in your LDAP/AD need various levels of access to the containers/services in your cluster
- Some users need to be able to go inside the running containers.
- Some users just need to be able to see the logs
- You do NOT want to give SSH access to each host in your cluster.

Now, how do you achieve this? The answer, or feature rather, is a client bundle. When you do a *docker version* command you will see two entries. The client portion of the engine is able to connect to a local server AND a remote once a client bundle is invoked.

```
Documents $docker version
Client:
Version:      17.06.2-ce
API version:  1.30
Go version:   go1.8.3
Git commit:   cec0b72
Built:        Tue Sep  5 20:12:06 2017
OS/Arch:      darwin/amd64

Server:
Version:      17.06.2-ce
API version:  1.30 (minimum version 1.12)
Go version:   go1.8.3
Git commit:   cec0b72
Built:        Tue Sep  5 19:59:19 2017
OS/Arch:      linux/amd64
Experimental: true
```

What is a client bundle?

A client bundle is a group of certificates downloadable directly from the [Docker Universal Control Plane \(UCP\)](#) user interface within the admin section for "My Profile". This allows you to authorize a remote Docker engine to a specific user account managed in Docker EE, absorbing all associated RBAC controls in the process. You can now execute docker swarm commands from your remote machine that take effect on the remote cluster.

Example:

I have a user named 'bkauf' in my UCP. I download and extract a client bundle for this user.

Client Bundles

+ New Client Bundle

LABEL	PUBLIC KEY
<div>Generated on Tue, 12 Sep 2017 16:35:46 UTC by user admin</div> <div></div>	<div>-----BEGIN PUBLIC KEY-----</div> <div>MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE8PxM0</div> <div>7TPAUy6id/1ldt8SsHISxrYawTylNvgOaHmVYUEwS</div> <div>-----END PUBLIC KEY-----</div>

Name

▼ ucp-bundle-bkauf


ca.pem

cert.pem

cert.pub

env.cmd

env.ps1

 env.sh

key.pem

I open a terminal session with my docker for mac and issue a *docker version* command. You will see the server version matches the client. I can do a *docker ps* and verify nothing is running.

Client:

Version: 17.06.2-ce
API version: 1.30
Go version: go1.8.3
Git commit: cec0b72
Built: Tue Sep 5 20:12:06 2017
OS/Arch: darwin/amd64

Server:

Version: 17.06.2-ce
API version: 1.30 (minimum version 1.12)
Go version: go1.8.3
Git commit: cec0b72
Built: Tue Sep 5 19:59:19 2017
OS/Arch: linux/amd64
Experimental: true

Documents \$docker ps

CONTAINER ID	IMAGE	COMMAND
--------------	-------	---------

Documents \$

Now, I navigate to the extracted bundle directory and run the env.sh script (env.ps1 for windows)

ucp-bundle-bkauf \$source env.sh

```
ucp-bundle-bkauf $docker version
Client:
  Version:      17.06.2-ce
  API version:  1.30
  Go version:   go1.8.3
  Git commit:   cec0b72
  Built:        Tue Sep  5 20:12:06 2017
  OS/Arch:      darwin/amd64

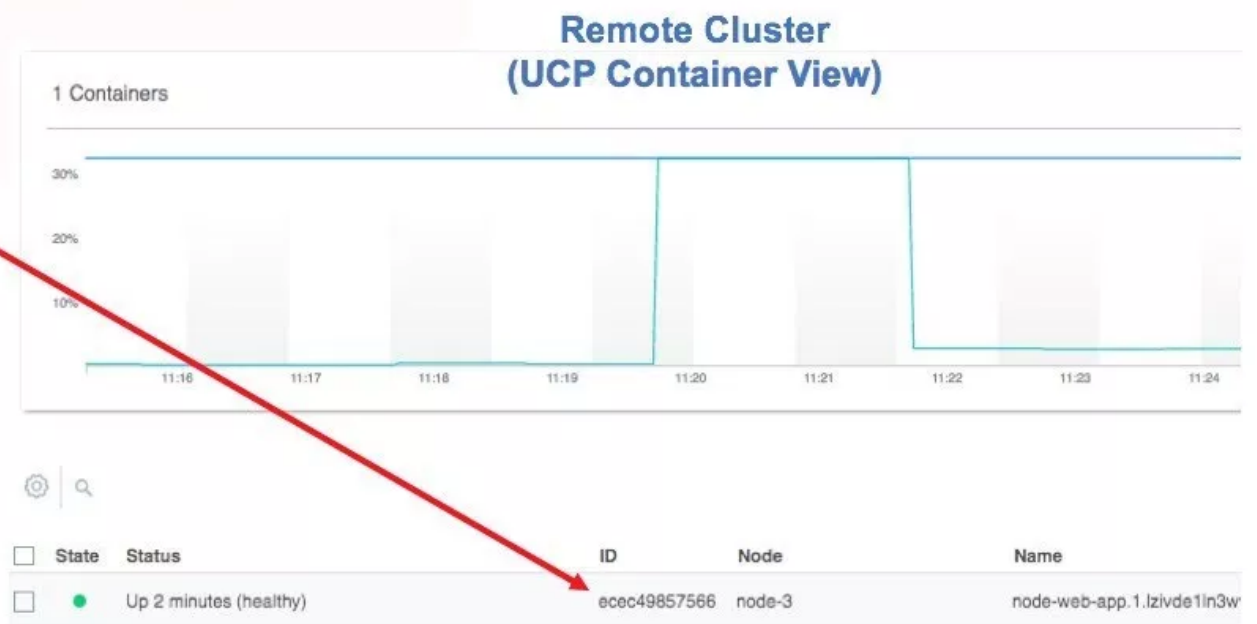
Server:
  Version:      ucp/2.2.2
  API version:  1.30 (minimum version 1.20)
  Go version:   go1.8.3
  Git commit:   94d1abdf3
  Built:        Wed Aug 30 23:30:05 UTC 2017
  OS/Arch:      linux/amd64
  Experimental: false
ucp-bundle-bkauf $
```

Notice the server now lists my version as ucp/2.2.2. This is the version of my UCP manager; I'm remotely connected from my laptop to my remote cluster assuming the bkauf user's access levels. I can now do various things such as create a service, view its tasks(containers) and even log into this REMOTE container from my laptop all through the API, no SSH access needed. I need not worry about what host the container is on! This is made possible by the role/permission set up for the use with the granular Role Based Access Control available with Docker EE.

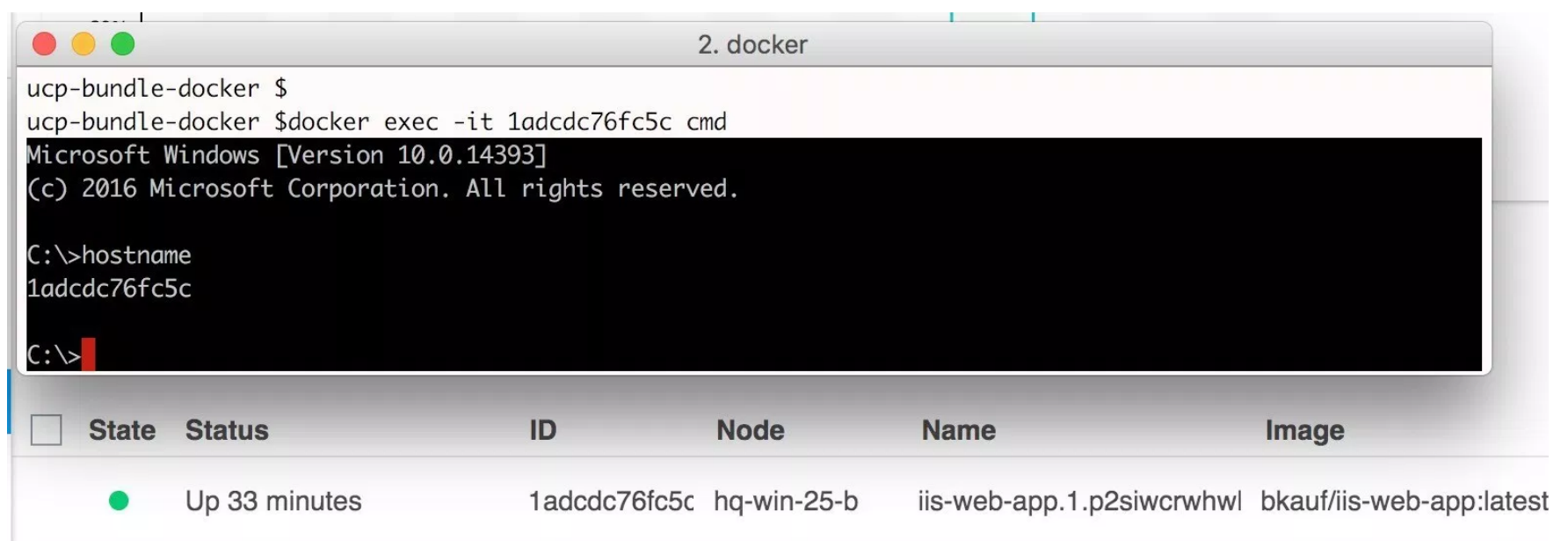
```
ucp-bundle-bkauf $docker service create --name node-web-app -p 8080 bkauf/node-web-app
h5rofdmjva9eto9kqb427cawo
Since --detach=false was not specified, tasks will be created in the background.
In a future release, --detach=false will become the default.
ucp-bundle-bkauf $docker service ls
ID                NAME                MODE                REPLICAS            IMAGE                PORTS
h5rofdmjva9e      node-web-app        replicated          1/1                 bkauf/node-web-app:latest *:0->8080/tcp
ucp-bundle-bkauf $docker service ps node-web-app
ID                NAME                IMAGE                NODE                DESIRED STATE        CURRENT STATE        ERROR
RTS
1zivde1ln3ww      node-web-app.1      bkauf/node-web-app:latest node-3              Running              Running 11 seconds ago
ucp-bundle-bkauf $
```



```
ucp-bundle-bkauf $docker ps
CONTAINER ID        IMAGE               COMMAND
ecec49857566       bkauf/node-web-app:latest  "npm start"
app.1.lzivde1ln3wwzcx6m4p41jqvv
ucp-bundle-bkauf $
ucp-bundle-bkauf $
ucp-bundle-bkauf $docker exec -it ecec49857566 sh
/usr/src/app # whoami
root
/usr/src/app # hostname
ecec49857566
/usr/src/app #
```



What about a Windows container on a Windows node in a UCP cluster you ask? Linux OR Windows nodes, remote access through your client bundle all works the same!



Docker Enterprise Edition (EE) is the only Containers as a Service (CaaS) Platform for IT that manages and secures diverse applications across disparate infrastructure, both on-premises and in the cloud. Docker EE embraces both traditional applications and microservices, built on Linux and Windows, and intended for x86 servers, mainframes, and public clouds. Docker EE unites all of these applications into single platform, complete with customizable and flexible access control, support for a broad range of applications and infrastructure, and a highly automated software supply chain.

Learn More

- Visit [IT Starts with Docker](#) and [learn more about MTA](#)
- Learn more about [Docker Enterprise Edition](#)
- Start a [hosted trial](#)
- Sign up for [upcoming webinars](#)

Get Familiar with #Docker Enterprise Edition Client Bundles

[CLICK TO TWEET](#)

[Continue reading...](#)