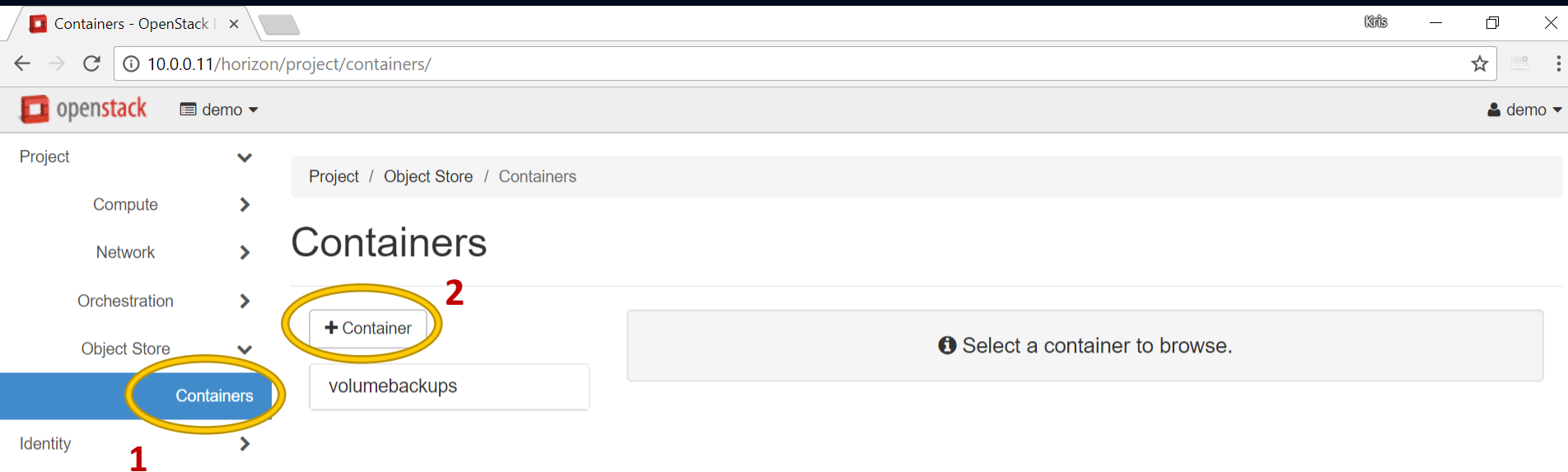


Preparing to **Certified OpenStack Administrator** Exam

Section 9 – Swift Object Storage

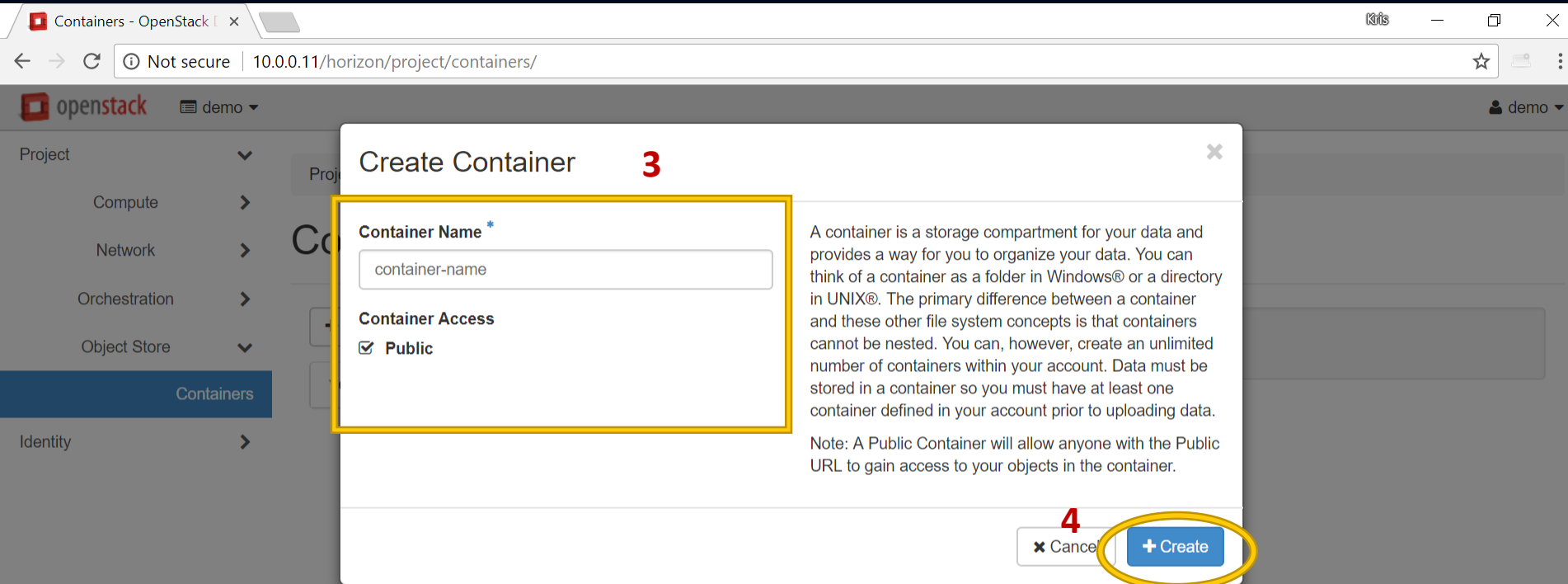
Lecture 39. Swift Summary and Review

Create a Swift Container



The screenshot shows the OpenStack Horizon web interface. The browser address bar displays `10.0.0.11/horizon/project/containers/`. The OpenStack logo and a 'demo' dropdown menu are visible in the top navigation bar. On the left sidebar, the 'Containers' menu item is highlighted with a blue background and circled in blue, with a red '1' next to it. The main content area shows the 'Containers' page title and a breadcrumb trail: 'Project / Object Store / Containers'. Below the title, there is a '+ Container' button circled in yellow with a red '2' next to it, and a list of existing containers including 'volumebackups'. A message box on the right says 'Select a container to browse.'

Create a Swift Container



Containers - OpenStack

10.0.0.11/horizon/project/containers/

openstack demo

Project

- Compute
- Network
- Orchestration
- Object Store
- Containers**
- Identity

Create Container 3

Container Name *

container-name

Container Access

☒ Public

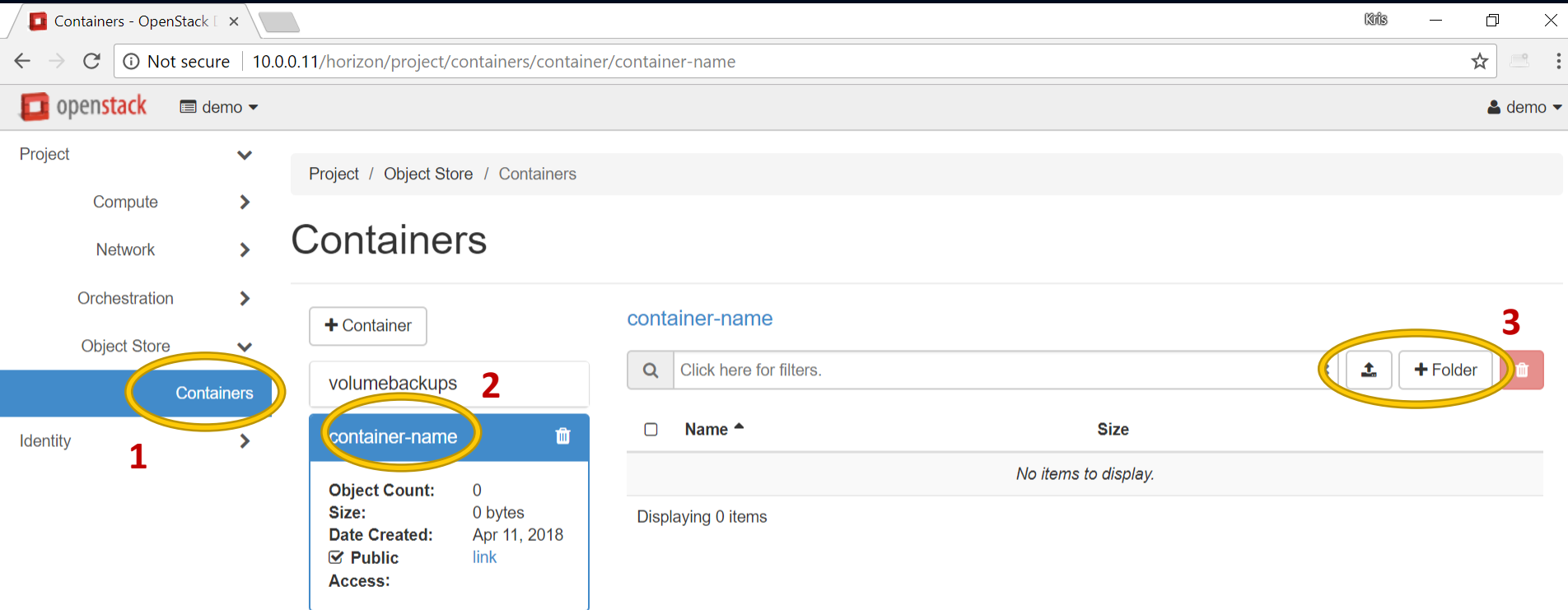
A container is a storage compartment for your data and provides a way for you to organize your data. You can think of a container as a folder in Windows® or a directory in UNIX®. The primary difference between a container and these other file system concepts is that containers cannot be nested. You can, however, create an unlimited number of containers within your account. Data must be stored in a container so you must have at least one container defined in your account prior to uploading data.

Note: A Public Container will allow anyone with the Public URL to gain access to your objects in the container.

4

Cancel + Create

Upload an Object to a Container



The screenshot shows the OpenStack Containers web interface. The left sidebar has a menu with 'Containers' highlighted (1). The main content area shows a list of containers, with 'volumebackups' (2) and 'container-name' (3) visible. The 'container-name' container is selected, showing details: Object Count: 0, Size: 0 bytes, Date Created: Apr 11, 2018, Public: checked, and Access: link. The right panel shows the 'container-name' container with a search bar and a '+ Folder' button (3).

Containers - OpenStack

10.0.0.11/horizon/project/containers/container/container-name

openstack demo

Project / Object Store / Containers

Containers

+ Container

volumebackups 2

container-name 3

Object Count: 0
Size: 0 bytes
Date Created: Apr 11, 2018
Public: ☒
Access: link

container-name

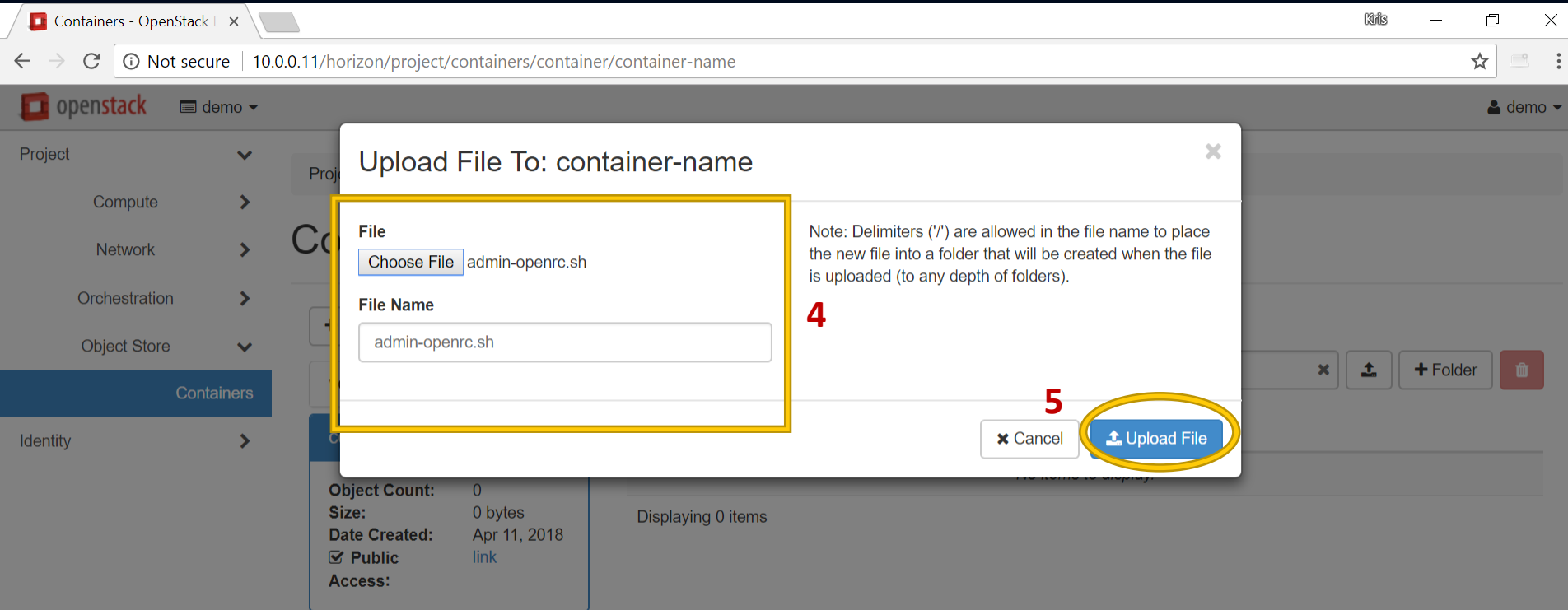
Click here for filters.

+ Folder

No items to display.

Displaying 0 items

Upload an Object to a Container



Containers - OpenStack

10.0.0.11/horizon/project/containers/container/container-name

openstack demo

Project

- Compute
- Network
- Orchestration
- Object Store
- Containers**
- Identity

Upload File To: container-name

File

Choose File admin-openrc.sh

File Name

admin-openrc.sh

Note: Delimiters ("/") are allowed in the file name to place the new file into a folder that will be created when the file is uploaded (to any depth of folders).

4

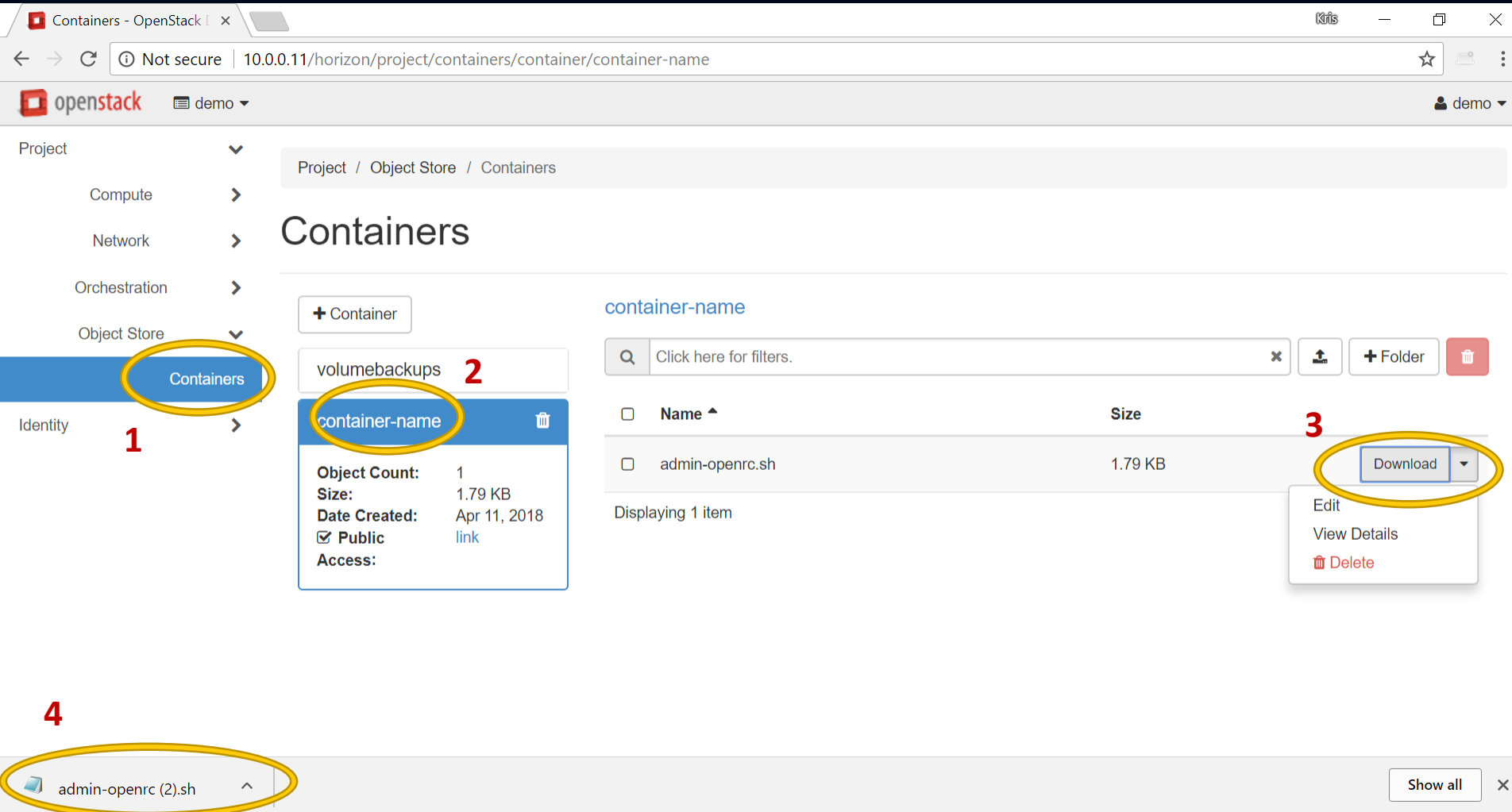
5

Cancel Upload File

Object Count: 0
Size: 0 bytes
Date Created: Apr 11, 2018
Public link
Access:

Displaying 0 items

Download an Object from a Container



The screenshot shows the OpenStack Horizon interface for managing containers. The browser address bar indicates the URL `10.0.0.11/horizon/project/containers/container/container-name`. The OpenStack logo and a 'demo' user are visible in the top header.

1 In the left sidebar, the 'Containers' link is highlighted.

2 In the 'Containers' section, the 'container-name' container is selected, showing its details:

- Object Count: 1
- Size: 1.79 KB
- Date Created: Apr 11, 2018
- Public: ☒ link
- Access:

3 In the 'container-name' container view, the 'Download' button is highlighted in the context menu for the file 'admin-openrc.sh'.

4 In the bottom bar, the downloaded file 'admin-openrc (2).sh' is highlighted.

CLI Review

```
$ swift list -l
```

```
$ swift list -lh <container-name>
```

```
$ swift post --help
```

```
$ swift post <private-container-name>
```

```
$ swift post <container> <object>
```

```
$ swift -r | --read-acl <acl>,<acl>,... <container>
```

```
$ swift -w | --write-acl <acl>,<acl>,... <container>
```

```
<acl> -> .r:*,.rlistings
```

```
<acl> -> <project>:<user>
```

```
<acl> -> <project>:*
```

```
<acl> -> *: <user>
```

CLI Review

```
$ swift post <container> <object> -H "X-Delete-After: <secs>"
```

```
$ swift post <container> <object> -H "X-Delete-At: <epoch>"
```

```
$ date +%s
```

```
$ date --date "May 21 2018 PST 10:00" +%s
```

```
$ swift post <container> <object> -H "X-Remove-Delete-At:"
```

```
$ swift post <container> -H "X-Storage-Policy: <policy>"
```


CLI Review

```
root@coa-lab: /home/coa
swift_hash_path_prefix = Open

# storage policies are defined here and determine various characteristics
# about how objects are stored and treated. Policies are specified by name on
# a per container basis. Names are case-insensitive. The policy index is
# specified in the section header and is used internally. The policy with
# index 0 is always used for legacy containers and can be given a name for use
# in metadata however the ring file name will always be 'object.ring.gz' for
# backwards compatibility. If no policies are defined a policy with index 0
# will be automatically created for backwards compatibility and given the name
# Policy-0. A default policy is used when creating new containers when no
# policy is specified in the request. If no other policies are defined the
# policy with index 0 will be declared the default. If multiple policies are
# defined you must define a policy with index 0 and you must specify a
# default. It is recommended you always define a section for
# storage-policy:0. Aliases are not required when defining a storage policy.
#
# A 'policy_type' argument is also supported but is not mandatory. Default
# policy type 'replication' is used when 'policy_type' is unspecified.
[storage-policy:0]
name = Policy-0
default = True
#policy_type = replication
aliases = yellow, orange

[storage-policy:1]
name = silver

# the following section would declare a policy called 'silver', the number of
# replicas will be determined by how the ring is built. In this example the
# 'silver' policy could have a lower or higher # of replicas than the
# 'Policy-0' policy above. The ring filename will be 'object-1.ring.gz'. You
# may only specify one storage policy section as the default. If you changed
# this section to specify 'silver' as the default, when a client created a new
# object it would be stored with 'silver' as the default policy.
/etc/swift/swift.conf" 187L, 7618C written
```

CLI Review

```
$ swift stat
```

```
$ swift-recon
```

```
$ swift-recon object -d
```

Preparing to **Certified OpenStack Administrator** Exam

Section 9 – Swift Object Storage

Lecture 39. Swift Summary and Review

Thank you!