SINA End User Documentation

Last edited by **Ugur Gündüz** 1 hour ago

Changelog

Nothing to see here.

Table of Contents

Known Bugs

package.yaml

- Helm-values issue
  If you enter your own (optional) "values" in the helm-plugin there will be an error because the file can't be found.

```
installation:
  tasks:
    - template:
       tmpl: values.yaml
       target: sinaExampleValues.yaml
    - helm:
        install:
          values: sinaExampleValues.yaml
          tgz: sinaexample.helm
```

Error-Message:

```
[helm-plugin] -  Filepath of tgz is not valid
```

Work-around: Delete the line with the values-key:

```
installation:
  tasks:
    - template:
       tmpl: values.yaml
       target: sinaExampleValues.yaml
    - helm:
        install:
          tgz: sinaexample.helm
```

What is KubeOps?

KubeOps is a software suite and includes:

- **Lima:** Is a CLI application for creating a single control-plane Kubernetes cluster or a high-availability Kubernetes cluster.
- **SINA:** Is a tool that allows the user to install packages with all its dependencies via Helmcharts and list all packages from a dedicated YAML file.

The main goal behind KubeOps is to make Kubernetes **secure** and **easy to use** for everyone. KubeOps gives you the possibility to **automate** thousands of Kubernetes clusters. Our focus is to provide a secure, easy-to-manage and AirGap-compatible Kubernetes environment immediately after installation.

Prerequisites

- KubeOps-LIMA (aka. a working Kubernetes-Cluster)
- Helm

YAML file syntax

index.yaml

The index.yaml contains a list of all packages and versions available. This lays on the webserver for SINA to read it.

Shown below is the structure of the `index.yaml` file:

```yaml
apiversion: v1
entries:
- data:
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook/guestbookv0.1.0.sina
        version: 0.1.0
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook/guestbookv0.1.0.sina
        version: 0.1.1
  - wordpress:
      - description: Deploy a wordpress_0.1.0
        name: wordpress
        path: wordpress/wordpressv0.1.0.sina
        version: 0.1.0
      - description: Deploy a wordpress_0.1.2
        name: wordpress
        path: wordpress/wordpressv0.1.2.sina
        version: 0.1.2
  - kibana:
      - description: Deploy a kibana_0.1.0
        name: kibana
        path: kibana/kibanav0.1.0.sina
        version: 0.1.0
  - logstash:
      - description: Deploy a logstash_0.1.0
        name: logstash
        path: logstash/logstashv0.1.0.sina
        version: 0.1.0
  path: kubernative
- data:
  - nginx:
      - description: Deploy a nginx_0.1.0
        name: nginx
        path: nginx_0.1.0.sina
        version: 0.1.0
  - kibana:
      - description: Deploy a kibana_0.1.0
        name: kibana
        path: kibana/kibanav0.1.0.sina
        version: 0.1.0
  - logstash:
      - description: Deploy a logstash_0.1.0
        name: logstash
```

```
        path: logstash/logstashv0.1.0.sina
        version: 0.1.0
  path: mathias
- data:
  - guestbook:
      - description: Deploy an guestbook_0.1.0
        name: guestbook
        path: guestbook/guestbookv0.1.0.sina
        version: 0.1.0
  - elasticSearch:
      - description: Deploy an elasticSearch_0.1.0
        name: elasticSearch
        path: elasticsearch/elasticsearchv0.1.0.sina
        version: 0.1.0
  - kibana:
      - description: Deploy a kibana_0.1.0
        name: kibana
        path: kibana/kibanav0.1.0.sina
        version: 0.1.0
  - logstash:
      - description: Deploy a logstash_0.1.0
        name: logstash
        path: logstash/logstashv0.1.0.sina
        version: 0.1.0
  path: ugur
```

uservalues.yaml

> Note: This section is optional

The user can set one or multiple uservalues.yaml files in the install-command of SINA. The uservalues.yaml has the same structure and key-values as the values.yaml coming from the helm-chart. The purpose of the uservalues.yaml is that the user can set his own values for the deployment of the helm-chart which will override the values.yaml from the helm-chart.

Shown below is the Helm-Chart `values.yaml` file:

```
replicaCount: 2

image:
  repository: ibmcom/guestbook
  tag: v1
  pullPolicy: Always
service:
  type: LoadBalancer
  port: 3000
  nodePort: 31100
redis:
  master:
    image: redis:2.8.23
  slave:
    image: k8s.gcr.io/redis-slave:v2
  port: 6379
  slaveEnabled: true
```

Shown below is the `uservalues.yaml` file:

```
replicaCount: 7

image:
  repository: ibmcom/guestbook
  tag: v2
  pullPolicy: Never
service:
  type: NodePort
  port: 3000
  nodePort: 32333
redis:
  master:
    image: redis:2.8.23
  slave:
    image: k8s.gcr.io/redis-slave:v2
```

```
  port: 6379
  slaveEnabled: false
```

config.yaml

The config.yaml is the systemconfiguration file for the SINA.
The file defines the specification needed for SINA to work correctly.

Shown below is the structure of the file `config.yaml` :

```
apiversion: v1
spec:
  packages:
  - default: true
    host: file:///srv/sina-data/
  - default: false
    host: http://hub.kubernative.net/
  plugins: /srv/sina-plugins/
  registry: registry.kubernative.net/
  workspace: /tmp/sina/process/
```

package.yaml

The package.yaml defines a package in a specific version as well as the tasks needed to install it.

Shown below is the structure of the `package.yaml` file:

```
apiVersion: v1
description: A Helm chart to deploy Guestbook three tier web application.
name: guestbook
version: 0.1.0
includes:
  files:
    - values: "values.yaml"
    - guestbook: "guestbookv0.1.0.helm"
  containers:
    - guestbook:
        image: registry.kubernative.net/guestbook/guestbook
        tag: v1
    - redis:
        image: gcr.io/redis
        tag: v1
installation:
  tasks:
    - print:
        message: "START GUESTBOOK"
    - template:
        tmpl: values.yaml
        target: gbValues.yaml
    - helm:
        install:
          values: gbValues.yaml
          tgz: guestbook.tgz
    - print:
        message: "END GUESTBOOK"
```

Install packages

SINA provides the `install` command installing the given package, its given version with its dependencies and subpackages. If there is no file given, SINA will not parse the template and the package will be installed with default values coming from the default values.yaml.

Install the packages from `values.yaml` with the following command:

```
sina install --file <filepath values.yaml> <username>/<packagename:version>
```

> Note: The `--file` flag is optional and must be a YAML file.

SINA

1. Checks if the YAML file from the flag exists:

If the file does not exist, following error occurs:

```
[root@stefan1 ~]# sina install --file notExists.yaml <username>/wordpress:0.6.0
Error: <filepath values.yaml> file does not exist
Usage:
  sina install [flags]

Flags:
  -f, --file string   One file or multiple files in .yaml format
  -h, --help          help for install
```

1. Checks if the `--file` flag from the flag exists:

```
[root@stefan1 ~]# sina install notExists.yaml <username>/wordpress:0.6.0
[SINA] Invalid format. Hint: > <packagename>:<version> < but you entered 'notExists.yaml'
```

1. Checks if the parameter has the following format: `<packagename:version>` .
   If the format is not valid an error will occur.

If the format is left completely empty:

```
[SINA] Usage: sina install --file <filepath> <username>/<packagename>:<version>
```

If the left side of the colon is empty:

```
[SINA] Invalid format. For example: > wordpress < but you entered ':<version>'
```

If the right side of the colon is empty:

```
[SINA] Invalid format. For example: > 0.0.1 < - > 0.1.0 < - > 1.0.0 but you entered '<packagename>:
```

The parameter is split by the colon and the split values are used to find the right entry in the index.yaml.

If an entry is found:

```
[SINA] Trying to access: registry.kubernative.net/index.yaml. It can take up to 10-15 seconds if unr
[SINA] Looking up the following entry in /tmp/sina/process/index.yaml

Packagename:    <packagename>
Version:        <version>

[SINA] Trying to access: registry.kubernative.net/<packagename>_<version>.yaml. It can take up to 10
[SINA] Extracting Template
```

If an entry is not found:

```
[SINA] Trying to access: registry.kubernative.net/index.yaml. It can take up to 10-15 seconds if unr
[SINA] Looking up the following entry in /tmp/sina/process/index.yaml

Packagename:    <packagename>
Version:        <version>

[SINA] No Entries found
```

If an entry is found, SINA will extract the URL for the package and download it.

List all available packages

SINA provides the `search` command printing out the currently available packages listed in the index.yaml.

By entering following command:

```
sina search
```

SINA fetches the current index.yaml from the package repository and an output is generated as a result.

For example:

```
[root@dev ~] sina search
[SINA-Search]:
-------------------------------------------------------------------------------------------------
| User                    | Name                    | Version | Description
-------------------------------------------------------------------------------------------------
| kubernative             | guestbook               | 0.1.0   | Deploy a guestbook_0.1.0
| kubernative             | guestbook               | 0.1.1   | Deploy a guestbook_0.1.0
| kubernative             | wordpress               | 0.1.0   | Deploy a wordpress_0.1.0
| kubernative             | wordpress               | 0.1.2   | Deploy a wordpress_0.1.2
| kubernative             | kibana                  | 0.1.0   | Deploy a kibana_0.1.0
| kubernative             | logstash                | 0.1.0   | Deploy a logstash_0.1.0
| mathias                 | nginx                   | 0.1.0   | Deploy a nginx_0.1.0
| mathias                 | kibana                  | 0.1.0   | Deploy a kibana_0.1.0
| mathias                 | logstash                | 0.1.0   | Deploy a logstash_0.1.0
| ugur                    | guestbook               | 0.1.0   | Deploy an guestbook_0.1.0
| ugur                    | elasticsearch           | 0.1.0   | Deploy an elasticSearch_0.1.0
| ugur                    | kibana                  | 0.1.0   | Deploy a kibana_0.1.0
| ugur                    | logstash                | 0.1.0   | Deploy a logstash_0.1.0
-------------------------------------------------------------------------------------------------
```

reflecting the entries from the index.yaml.

Create packages

The `create` command creates example files of a package in the current directory which can be used to develop an own package. It is not usable like this but it gives a good example of how a package has to look like.
Usage:

```
sina create
```

Result:

```
[root@dev ~] sina create
[root@dev ~] ll
total 20
-rw-r--r--. 1 root root     1270 Aug  6 14:11 package.yaml
-rw-r--r--. 1 root root     3441 Aug  6 14:11 sinaexample.helm
-rw-r--r--. 1 root root      985 Aug  6 14:11 values.yaml
```

Check packages

The `lint` command checks the package.yaml in the current directory for all includes to be present or reachable. If there is an error, it will be displayed which file/container is not available.
Usage:

```
sina lint
```

Build packages

The `build` command builds a package from the current directory. There has to be a `.helm` file in it with a helm chart, as well as a `values.yaml` and a `package.yaml`. It than packs it all together to a compatible package for the hub then can be pushed with the `push` command. \

The content of the example-package:

```
sinaexamplev0.1.0.sina
├── package.yaml
├── sinaexamplev0.1.0.helm
│    ├── charts
│    ├── Chart.yaml
│    ├── templates
│    │   ├── deployment.yaml
│    │   ├── _helpers.tpl
│    │   ├── hpa.yaml
│    │   ├── ingress.yaml
│    │   ├── NOTES.txt
│    │   ├── serviceaccount.yaml
│    │   ├── service.yaml
│    │   └── tests
│    │   └── test-connection.yaml
```

```
|    └── values.yaml
|
└── values.yaml
```

Usage:

```
sina build
```

Log into the Hub

The `login` command lets you log into your Hub account to later push packages into (like docker push). You will be logged in for a limited time. If your username contains spaces it will be replaced by dots.

For example: If your username is test user:

test user -> test.user

Therefore -> sina login test.user -p

```
sina login -u <username> -p <password>
```

Push Packages into the Hub

The `push` command works like the docker push command. If you are logged in, you can push your package into the Hub.

```
sina push <package>
```

Attachments

Further explanation of the YAML file syntax

index.yaml API objects

Structure of the createCluster API Object with version `v1` .

An example to show the structure of the file `index.yaml` :

```yaml
apiversion: v1
entries:
- data:
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook_0.1.0.sina
        version: 0.1.0
  - wordpress:
      - description: Deploy a wordpress_0.1.0
        name: wordpress
        path: wordpress_0.1.0.sina
        version: 0.1.0
      - description: Deploy a wordpress_0.1.2
        name: wordpress
        path: wordpress_0.1.2.sina
        version: 0.1.2
  path: kubernative
- data:
  - nginx:
      - description: Deploy a nginx_0.1.0
        name: nginx
        path: nginx_0.1.0.sina
        version: 0.1.0
  path: mathias
- data:
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook_0.1.0.sina
        version: 0.1.0
  path: ugur
```

apiVersion:
Defines the API Version used for this index.yaml
Example:

```
apiVersion: v1
```

entries:
Defines all packages and versions available. It has a sub-item list with `name of package:` as the sub-item name.
`name of package:` has a value-list with `version:` as the first value, followed by `name`, `description` and `package`.
There can be one or more value-lists for different versions.
`version:` has to be a valid version number tag, such as '0.0.1'.
`name:` has to be the correct name of the package.
`description:` is a short description for the package.
`package:` has to be the url to the package.yaml. It has to be a valid address to a external repo or a relative path to a file in the filesystem.
`path:` the username

Example:

```
apiversion: v1
entries:
- data:
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook_0.1.0.sina
        version: 0.1.0
  - wordpress:
      - description: Deploy a wordpress_0.1.0
        name: wordpress
        path: wordpress_0.1.0.sina
        version: 0.1.0
      - description: Deploy a wordpress_0.1.2
        name: wordpress
        path: wordpress_0.1.2.sina
        version: 0.1.2
  path: kubernative
- data:
  - nginx:
      - description: Deploy a nginx_0.1.0
        name: nginx
        path: nginx_0.1.0.sina
        version: 0.1.0
  path: mathias
- data:
  - guestbook:
      - description: Deploy a guestbook_0.1.0
        name: guestbook
        path: guestbook_0.1.0.sina
        version: 0.1.0
  path: ugur
```

config.yaml API objects

Structure of the config.yaml API object with version `v1`.

An example to show the structure of the file `config.yaml`:

```
apiversion: v1
spec:
  hub: hub.kubernative.net:1337
  packages:
  - default: true
    host: http://hub.kubernative.net/
  plugins: /srv/sina-plugins/
  registry: registry.kubernative.net/
  workspace: /tmp/sina/process/
```

apiVersion:
Mandatory
Defines the API Version used for this config.yaml
Example:

```
apiversion: v1
```

spec
Mandatory
Defines the specification needed for SINA to work correctly.

packages:
Address to the package repo. Has to be either a valid `IP-adress` , a valid `DNS name` or a valid path with `http/https` or `file://` protocol. You can set a default package-host with the default flag

```
spec:
  packages:
  - default: true
    host: file:///srv/sina-data/
  - default: false
    host: http://hub.kubernative.net/
```

plugins:
Mandatory
The path where the plugin-binaries are stored.

```
plugins: /srv/sina-plugins/
```

hub:
Mandatory
The URL of the Hub.

```
plugins: hub.kubernative.net:1337
```

registry:
Mandatory
Address to the registry. Has to be either a valid `IP-adress` , a valid `DNS name` or a valid path without `http/https` .

```
registry: registry.kubernative.net/
```

workspace:
Mandatory
Its the path to the folder where the files are stored during the process.

```
workspace: /tmp/sina/process/
```

package.yaml API objects

Structure of the package.yaml API object with version `v1` .

An example to show the structure of the file `package.yaml` :

```
apiVersion: v1
description: A Helm chart to deploy Guestbook three tier web application.
name: guestbook
version: 0.1.0
includes:
  files:
    - values: "values.yaml"
    - guestbook: "guestbookv0.1.0.helm"
  containers:
    - guestbook:
        image: registry.kubernative.net/guestbook/guestbook
        tag: v1
    - redis:
        image: gcr.io/redis
        tag: v1
installation:
  tasks:
    - print:
        message: "START GUESTBOOK"
    - template:
        tpl: values.yaml
        saveTo: gbValues.yaml
    - helm:
        install: "guestbook.tgz"
```

```
        values: "gbValues.yaml"
    - print:
        message: "END GUESTBOOK"
```

apiVersion:
Mandatory
Defines the API Version used for this index.yaml Example:

```
apiVersion: v1
```

description:
Mandatory
Short description of the package.
Example:

```
description: "A Helm chart to deploy Guestbook three tier web application."
```

name:
Mandatory
Name of the package.
Example:

```
name: "guestbook"
```

version:
Mandatory
Version of the package in a valid form like '0.0.0'.
Example:

```
version: 0.1.0
```

includes:
Mandatory
Defines the helmchart and containers for the package.

files:
Mandatory
Has the path to the needed file as a list of key-value.
Example:

```
includes:
  files:
    - Datei1: "Datei1.rpm"
    - guestbook: "guestbookv0.1.0.tgz"
```

containers
Mandatory
Defines the needed containers.

The sub-items have `name of container` as the name of each item. The value of each item has to be the path to the container in the registry.
Example:

```
includes:
  containers:
    - guestbook:
        image: registry.kubernative.net/guestbook/guestbook
        tag: v1
    - redis:
        image: registry.kubernative.net/guestbook/redis
        tag: 2.8.23
    - redisSlave:
        image: registry.kubernative.net/guestbook/redis-slave
        tag: v2
```

installation
Mandatory
Installation routine for the package.

tasks
Mandatory
Includes all the tasks needed for the installation.
The sub-items are a list with `plugin-name` as the name of the item. Tasks names are modules/plugins.
Modules have sub-items/values and can have more in a tree. See the module documentation for the right parameters and values.
Example:

```
- plugin-name1:
    parameter1: value
    parameter2: value
- plugin-name2:
    parameter1: value
    parameter2: value
```