Fabric Services System

Release 24.5.1

Digital Sandbox Command Line Reference

3HE 20042 AAAA TQZZA
Edition: 1
May 2024

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Table of contents

1 About this document

This document describes CLI commands for the Nokia Fabric Services System Digital Sandbox.

This document is intended for network technicians, programmers, operators, and others who will create and deploy intents in a sandbox environment, including the use of traffic injection tools.

Note:
This manual covers the current release and may also contain some content that will be released in later maintenance loads. See the *Fabric Services System Release Notes* for information about features supported in each load.

1.1 What's new

There were no changes to this document for this release.

1.2 Precautionary and information messages

The following are information symbols used in the documentation.

DANGER: Danger warns that the described activity or situation may result in serious personal injury or death. An electric shock hazard could exist. Before you begin work on this equipment, be aware of hazards involving electrical circuitry, be familiar with networking environments, and implement accident prevention procedures.

WARNING: Warning indicates that the described activity or situation may, or will, cause equipment damage, serious performance problems, or loss of data.

Caution: Caution indicates that the described activity or situation may reduce your component or system performance.

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.3 Conventions

Commands use the following conventions

- **Bold** type indicates a command that the user must enter.
- Input and output examples are displayed in `Courier` text.

- An open right angle bracket indicates a progression of menu choices or simple command sequence (often selected from a user interface). Example: **start** > **connect to**
- Angle brackets (< >) indicate an item that is not used verbatim. For example, for the command **show ethernet <*name*>**, *name* should be replaced with the name of the interface.
- A vertical bar (|) indicates a mutually exclusive argument.
- Square brackets ([ ]) indicate optional elements.
- Braces ({ }) indicate a required choice. When braces are contained within square brackets, they indicate a required choice within an optional element.
- *Italic* type indicates a variable.

Examples use generic IP addresses. Replace these with the appropriate IP addresses used in your system.

2 CLI interface introduction

The Digital Sandbox (DS) CLI is an interface for configuring, monitoring, and maintaining the Fabric Services System Digital Sandbox. This chapter describes basic features of the CLI and how to use them.

For more information about the Digital Sandbox, see the *Fabric Services System User Guide*.

2.1 Accessing and using the CLI

2.1.1 Accessing the CLI

The Digital Sandbox software is installed on top of the Linux operating system. Once initialized, you can access the CLI. Standard Linux commands are accessible and formatting and filtering commands (such as grep, more, and so on) can be used with the Digital Sandbox CLI.

2.1.2 Executing CLI commands

After logging in, you are placed in the Digital Sandbox's Linux operating system. The **dsctl** command controls all aspects of the local instance and proceeds all Digital Sandbox CLI commands. For example, to use the **version** CLI command, you would enter:

```
dsctl version
```

This command produces the following output:

```
$ dsctl version
Digital Sandbox  v0.0.0-169
$
```

2.1.3 Formatting and filtering output

Command output can be formatted or filtered using Linux commands such as pipe ("|"), grep, awk, and so on.

2.1.4 Using the CLI help functions

The CLI help functions (**help**, **-h**, and **--help**) can assist in understanding command usage.

Enter **help**, **-h**, or **--help** after a command to display the command usage. For example, entering **-h** after the **dsctl** command shows its usage. For example:

```
[username1 ~]$ dsctl -h
Usage:
  dsctl [command]
```

```
Available Commands:
  abort       Aborts an ongoing update of a Digital Sandbox deployment
  connect     Connect (SSH) to a Network Element in a Digital Sandbox deployment
  create      Creation of a Digital Sandbox resource (deployment, phyconnect)
              or Digital Sandbox configuration
  delete      Delete a Digital Sandbox resource/configuration
  help        Help about any command
  intent      Contains all intent related Digital Sandbox commands
  list        List a Digital Sandbox resource/configuration
  load        Load commands.
  logs        Retrieve a Gatenet log
  status      Get the status of a Digital Sandbox resource
  traffic     Traffic related commands
  update      Update a resource
  version     Print the version number of Digital Sandbox

Flags:
  -c, --config string      configuration file
  -h, --help               help for dsctl
      --log.level string   log level (default "info")
Use "dsctl [command] --help" for more information about a command.
```

You can also enter **-h** after a specific CLI command to show its usage. For example:

```
$ dsctl traffic -h
Digital Sandboxes allows injecting/capturing traffic a deployment. The commands in this
 structure
       will allow you not only to start/stop traffic injection or capturing but also to trigger
 analysis of the
       captures and comparisons of two different traffic runs.


Usage:
  dsctl traffic [command]
Available Commands:
  analyze     Perform an analysis of traffic run
  compare     Compare two traffic runs
  create      Create Digital Sandbox traffic resources
  list        List Digital Sandbox traffic resources
  message     Message components of the Digital Sandbox.
Flags:
  -h, --help   help for traffic
Global Flags:
  -c, --config string      configuration file
      --log.level string   log level (default "info")
Use "dsctl traffic [command] --help" for more information about a command.
```

Within the help output, mandatory fields are denoted by (M).

2.1.5 Using the CLI auto-complete function

To reduce keystrokes or aid in remembering a command name, use the CLI auto-complete function.

Enter a tab at any level to auto-complete the next command level. If multiple options are available, a list will appear. The auto-complete function will also prompt for mandatory fields.

When a command is partially entered, the remainder of the command appears ahead of the prompt in lighter text. Press the Tab key to complete the command.

```
$ dsctl traffic
```

When the Tab key is pressed and there are multiple options, the options are shown:

```
$ dsctl traffic
analyze   compare   create   list   message
```

2.1.6 Using CLI shortcuts

Use shortcuts to move the cursor on the command line, complete commands, and recall commands previously entered. Shortcuts can also make syntax correction easier. Table 1: CLI keyboard shortcuts lists common shortcuts.

Table 1: CLI keyboard shortcuts

Task	Keystroke
Move cursor to the beginning of the line	Ctrl+A
Move cursor to the end of the line	Ctrl+E
Move cursor one character to the right	Ctrl+F or Right arrow key
Move cursor one character to the left	Ctrl+B or Left arrow key
Move cursor forward one word	Esc F
Move cursor back one word	Esc B
Transpose the character to the left of the cursor with the character the cursor is placed on	Ctrl+T
Complete a partial command	Enter the first few letters, then press the Tab key
Recall previous entry in the buffer	Page up key

2.2 Command structure

Each CLI command consists of a primary keyword (dsctl), command group, operations, and optional parameters.

```
dsctl <command_group> <operations> [-<parameter>]
```

- Primary key word - The primary keyword is always **dsctl**.
- Command group - The primary action (such as connect, list, logs). In this document, each command group has its own section.
- Operations - Secondary actions associated with the command group.
- Parameters - Any optional, positional, or required named parameters.

2.2.1 Global flags

Global flags are optional parameters that are associated with all command groups or sub-commands. The following global flags are available:

Table 2: Global flags

`-c, --config string`	Configuration file
`-h, --help`	Help option for current command or sub-command
`--log.level string`	Log level (default "info")

2.2.2 Field descriptions

Table 3: Field descriptions defines fields used to describe the CLI commands. Not all fields may be applicable for all commands.

Table 3: Field descriptions

Field	Description
Synopsis	Defines the syntax for the CLI command including any optional, positional, or required parameters.
Description	Describes the command.
Options	Defines any optional parameters associated with the command group or sub-command. Global flags are not described here. See Global flags for details. Mandatory fields are noted.
Examples	Provides select examples for reference.

3 Commands

The Fabric Services System Digital Sandbox (DS) CLI is organized by command groups in the sections that follow. Command groups include:

3.1 abort commands

Abort commands are used to abort an ongoing update of a Digital Sandbox deployment.

The following CLI abort commands are available:

dsctl

— abort

dsctl abort

Synopsis	`dsctl abort -i <ID_string>`
Description	The **dsctl abort** command is used to abort an ongoing update of a deployment. When used, the deployment will be labeled as Failed because there will be differences between what the user specified and what is available in the sandbox.A new update trigger must be used to remove the Failed label. The API is intended to be used in cases where a user knows they forgot to specify something before the update was triggered.
Options	The following are **dsctl abort** parameters: **Parameter:** `-i <ID_string>`

Used with operation: `abort`

Description: Specifies the Digital Sandbox deployment ID. Mandatory field.

Examples To abort a Digital Sandbox deployment with an ID of "star":

```
dsctl abort -i star
```

3.2 connect commands

Connect commands are used to connect via SSH to network elements in the Digital Sandbox deployment.

The following CLI connect commands are available:

dsctl

— connect

dsctl connect

Synopsis

```
dsctl connect {-n <name_string> | -e <expression_string>} -i
<ID_string> [-z <IPaddress_string>] [-o <ssh_opt_string>]
```

Description The **dsctl connect** command is used to connect to a network element in a Digital Sandbox deployment.

By default, the system uses kubectl exec to connect to the network element. If that fails, the system tries to connect using SSH.

Options The following are **dsctl connect** parameters:

Parameter: `-n <name_string>`

Used with operation: `connect`

Description: Exact name of the network element to connect to. Cannot be used with the -e flag.

Parameter: `-e <expression_string>`

Used with operation: connect

Description: Name (or part of) the network element to connect to. For example, if the network element is "dut-alpha", a valid expression could be "-e alpha". Cannot be used with the -n flag.

Parameter: `-i <ID_string>`

Used with operation: `connect`

Description:	Specifies the Digital Sandbox deployment ID. Mandatory field.
Parameter:	`-z <IPaddress_string>`
Used with operation:	`connect`
Description:	Specifies the IP address of the kubernetes node to connect to.
Parameter:	`-0 <ssh_opt_string>`
Used with operation:	`connect`
Description:	Specifies an option that can be passed to an SSH command. The option string must be enclosed in single or double quotes.

Examples

To connect to a Digital Sandbox network element with a deployment ID of "star" and a network element name of "networkelement1":

```
dsctl connect -i star -n networkelement1
```

To connect to a Digital Sandbox network element with a deployment ID of "star" and a unique shortened version of the network element name "dut2":

```
dsctl connect -i star -e dut2
```

To connect to a Digital Sandbox network element with a deployment ID of "star", a unique shortened version of the network element name "dut1", and a connection timeout value option of "10" (passed to an ssh command):

```
dsctl connect -i star -e dut1 -o "-o connectTimeout=10"
```

3.3 create commands

Create commands are used to create Digital Sandbox resources, such as configurations, images, and phyconnectors. The following CLI create commands are available:

dsctl

- create
  - checkpoint
  - config
    - defaults
    - fssconnect
    - nokia-srlinux-license
  - deployment
  - image

— phyconnector

— snapshot

dsctl create checkpoint

Synopsis

```
dsctl create checkpoint [-d <device_string>] -i <ID_string> -
n <name_string> [-t <timeout_integer>]
```

Description

The **dsctl create checkpoint** command is used to create a deployment checkpoint. A checkpoint is a saved version of the running file that you can revert to.

Options

The following are **dsctl create checkpoint** parameters:

Parameter: `-d <device_string>`

Used with operation: `create checkpoint`

Description: Specifies the device target name. Multiple names are separated by a semi-colon. Default is "*" (the asterisk symbol).

Parameter: `-i <ID_string>`

Used with operation: `create checkpoint`

Description: Specifies the deployment ID. Mandatory field.

Parameter: `-n <name_string>`

Used with operation: `create checkpoint`

Description: Specifies the name of the checkpoint being created. Mandatory field.

Parameter: `-t <timeout_integer>`

Used with operation: `create checkpoint`

Description: Specifies the timeout value as an integer. This is the time in minutes to wait for an ACK of the processed message. Default is 1.

Examples

To create a checkpoint with the name "testcheckpoint" associated with a deployment named "star" with a timeout value of 2 minutes:

```
dsctl create checkpoint -i star -n testcheckpoint -t 2
```

dsctl create config

Synopsis

```
dsctl create config defaults [-e <gluster_endpoint>] [-n
<gluster_endpoint_namespace>] [-v <gluster_volumename>] [-s
<imagePullSecrets_string>][-k <K8_nodeselector_string>]
```

```
dsctl create confg fssconnect -a <auth_URL_string> -d <DNS_
name_string> -s <host_string> -i <DNS_IP_resolution_string> -
p <password_string> -u <user_ID_string> -z <ZTP_URL_string>
```

```
dsctl create config nokia-srlinux-license [-f <filename_
string>] [-k <keyname_string>] [-v <version_string>]
```

Description

The **dsctl create config** command is used to create a Digital Sandbox configuration and has the following operations:

- The `defaults` operation creates the Digital Sandbox configuration defaults.
- The `fssconnect` operation creates the FSSconnect secret, or sensitive information such as passwords that are required to connect to the fabric manager (Fabric Services System).
- The `nokia-srlinux-license` operation creates a license key for a specified Nokia SR Linux version.

Options

The following are **dsctl create config defaults** parameters:

Parameter:	`-e <gluster_endpoint_string>`
Used with operation:	`create config defaults`
Description:	Creates the default configuration with the specified gluster service endpoint.
Parameter:	`-n <gluster_endpoint_namespace>`
Used with operation:	`create config defaults`
Description:	Creates the default configuration with the specified namespace of the gluster endpoint.
Parameter:	`-v <gluster_volumename>`
Used with operation:	`create config defaults`
Description:	Creates the default configuration with the specified gluster volume name.
Parameter:	`-s <imagePullSecrets_string>`
Used with operation:	`create config defaults`
Description:	Defines the kubernetes image pull secrets (stored in the digital-sandbox-system namespace) that can be used during Digital Sandbox deployment creation. Default = regcred.
Parameter:	`-k <K8_nodeselector_string>`

Used with operation: `create config defaults`

Description: Defines the kubernetes node selector to use to specify which kubernetes nodes can be used by the Digital Sandbox.

The following are **dsctl create config fssconnect** parameters:

Parameter: `-a <auth_URL_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager authorization URL. Mandatory field.

Parameter: `-d <DNS_name_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager service DNS name. Mandatory field.

Parameter: `-s <host_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager host. Mandatory field.

Parameter: `-i <DNS_IP_resolution_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager service DNS IP resolution. Mandatory field.

Parameter: `-p <password_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager password. Mandatory field.

Parameter: `-u <user ID_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager user ID. Mandatory field.

Parameter: `-z <ZTP_URL_string>`

Used with operation: `create config fssconnect`

Description: Defines the fabric manager ZTP URL. Mandatory field.

The following are **dsctl create config nokia-srlinux-license** parameters:

Parameter: `-f <filename_string>`

Used with operation: `create config nokia-srlinux-license`

Description: Defines the name and path of the SR Linux license file.

Parameter: `-k <keyname_string>`

Used with operation: `create config nokia-srlinux-license`

Description: Defines the license key that corresponds to the Nokia SR Linux version.

Parameter: `-v <version_string>`

Used with operation: `create config nokia-srlinux-license`

Description: Defines the version of the Nokia SR Linux key.

Examples

To create a default configuration with the gluster endpoint "digital-sandbox-gluster-endpoints", a gluster endpoint namespace of "default", and a gluster volume name of "digitalsandbox":

```
dsctl create config defaults -e digital-sandbox-gluster-endpoints -n default -v digitalsandbox
```

dsctl create deployment

Synopsis

```
dsctl create deployment -f <filename_string> -i <ID_string> [-t <tar_filename_string>]
```

Description

The **dsctl create deployment** command is used to create a Digital Sandbox simulation platform or deployment.

Options

The following are **dsctl create deployment** parameters:

Parameter: `-f <filename_string>`

Used with operation: `create deployment`

Description: Specifies the name of the file containing the deployment path and filename. Default is "*/tmp/devices.yaml*". Mandatory field.

Parameter: `-i <ID_string>`

Used with operation: `create deployment`

Description: Specifies the deployment ID. Mandatory field.

Parameter: `-t <tar_filename_string>`

Used with operation: `create deployment`

Description: Specifies the name of the path and tar file that contains the configuration and state details required to build a topology.

Examples

To create a deployment with a deployment ID named "star", with the filename of "/tmp2/7live.yaml", and with a configuration tar file named "/tmp2/7live.tar":

```
dsctl create deployment -i star -f /tmp2/7live.yaml -t /tmp2/7live.tar
```

dsctl create image

Synopsis

```
dsctl create image {-p <RPM_path_string> | -r <SRL_release_string>}
```

Description

The **dsctl create image** command is used to create a custom SR Linux image with the designated RPMs.

Options

The following are **dsctl create image** parameters:

Parameter: `-p <RPM_path_string>`

Used with operation: `create image`

Description: Specifies the filename and path that contains the list of RPMs to install. Either this parameter (-p) must be provided or the -r parameter.

Parameter: `-r <SRL_release_string>`

Used with operation: `create image`

Description: Specifies the SR Linux release. When used, all the RPMs associated with this release are included and it will ignore any other RPMs.

Either this parameter (-r) must be provided or the -p parameter.

Examples To create an image using the RPMs associated with the SR Linux release "srl_rel_21_06_":

```
dsctl create image -r srl_rel_21_06_
```

dsctl create phyconnector

Synopsis

```
dsctl create phyconnector -s <interface_list_string> -k <k8node_string> -n <resource_name_string>
```

Description The **dsctl create phyconnector** command is used to create a phyconnector in the kubernetes cluster. A phyconnector is used to connect a physical interface, device, or network to the containerized network.

Options The following are **dsctl create phyconnector** parameters:

Parameter:	`-s <interface_list_string>`
Used with operation:	`create phyconnector`
Description:	Specifies a list of interfaces, using a semi-colon (;) to separate each, available on the node. Mandatory field.
Parameter:	`-k <k8node_string>`
Used with operation:	`create phyconnector`
Description:	Specifies the name of the k83 node containing the interfaces. Mandatory field.
Parameter:	`-n <resource_name_string>`
Used with operation:	`create phyconnector`
Description:	Specifies the name of the resource. Names must be DNS compatible. Mandatory field.

Examples To create a physical connection in the kubernetest cluster for interfaces eth3, eth4, and eth5 in the kubernetes node named "k8sworker1" for resource named "phyconnectOnNode1":

```
dsctl create phyconnector -s eth3;eth4;eth5 -n phyconnectOnNode1 -k k8sworker1
```

dsctl create snapshot

Synopsis

```
dsctl create snapshot -i <ID_string> [-l <location_string>] [-t <timeout_integer>]
```

Description The **dsctl create snapshot** command is used to create a snapshot of the deployment. A snapshot is a copy of the deployment with all the intents and configurations at the snapshot creation time. This snapshot can be used to create a duplicate of the deployment using the "dsctl load snapshot" command.

Options The following are **dsctl create snapshot** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`create snapshot`
Description:	Specifies the deployment ID. Mandatory field.
Parameter:	`-l <location_string>`
Used with operation:	`create snapshot`
Description:	Specifies filename and path of destination tar-file. Default is “/tmp/snapshot.tar”.
Parameter:	`-t <timeout_integer>`
Used with operation:	`create snapshot`
Description:	Specifies the timeout value as an integer. This is the time in minutes to wait for an ACK of the processed message. Default is 1.

3.4 delete commands

Delete commands are used to delete a Digital Sandbox resource configuration.

The following delete CLI commands are available:

dsctl
- delete
  - checkpoint
  - config
    - defaults
    - fssconnect
    - nokia-srlinux-license
  - deployment
  - phyconnector

dsctl delete checkpoint

Synopsis

```
dsctl delete checkpoint i <ID_string> [n <name_string>]
```

Description

The **dsctl delete checkpoint** command is used to delete a specific checkpoint.

Options

The following are **dsctl delete checkpoint** flags:

Parameter: `-h, --help`

Description: Obtains help for checkpoint

Parameter: `-i, <ID_string>`

Description: Specifies the deployment ID. Mandatory parameter.

Parameter: `-n, <name_string>`

Description: Specifies the checkpoint name.

The following are **dsctl delete checkpoint** global flags:

Parameter: `-c, --config string`

Description: Specifies the configuration file

Parameter: `-c, <log.level_string>`

Description: Specifies the logging level (the default is "info")

dsctl delete config

Synopsis

```
dsctl delete config defaults
dsctl delete config fssconnect
dsctl delete config nokia-srlinux-license -v <version>
```

Description

The **dsctl delete config** command is used to delete Digital Sandbox configuration details and has the following operations:

- The `defaults` operation deletes the Digital Sandbox default configuration.
- The `fssconnect` operation deletes the FSSconnect secret, or sensitive information such as passwords that are required to connect to the fabric manager (Fabric Services System).
- The `nokia-srlinux-license` operation deletes a license key for a specified Nokia SR Linux version. You can use the **dsctl list config nokia-srlinux-license** command to list existing licenses.

Options

The following are **dsctl delete config nokia-srlinux-license** parameters:

Parameter:	`-v <version>`
Used with operation:	`delete config nokia-srlinux-license`
Description:	Specifies the version of the Nokia SR Linux key to delete. Mandatory field.

Examples

To delete the Digital Sandbox default configuration:

```
dsctl delete config defaults
```

To delete the FSSconnect secret:

```
dsctl delete config fssconnect
```

To delete the license key for version "srl_rel_0.0.0" of the Nokia SR Linux key:

```
dsctl delete config nokia-srlinux-license -v srl_rel_0.0.0
```

dsctl delete deployment

Synopsis

```
dsctl delete deployment -i <ID_string> [-f]
```

Description

The **dsctl delete deployment** command is used to delete a Digital Sandbox deployment.

Options

The following are **dsctl delete deployment** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`delete deployment`
Description:	Specifies the deployment ID to delete. Mandatory field.

Parameter:	`-f`
Used with operation:	`delete deployment`
Description:	Specifies that the deletion should be forced even if another operation is in progess.

Examples

To delete a deployment with an ID of "star" and force the operation:

```
dsctl delete deployment -i star -f
```

dsctl delete phyconnector

Synopsis

```
dsctl delete phyconnector -n <name_string>
```

Description

The **dsctl delete phyconnector** command is used to delete a phyconnector in the kubernetes cluster. A phyconnector is used to connect a physical interface, device, or network to the containerized network.

Options The following are **dsctl delete phyconnector** parameters:

Parameter:	`-n <name_string>`
Used with operation:	`delete phyconnector`
Description:	Specifies the name of the phyconnector to delete. Mandatory field.

Examples To delete a phyconnector named "phyconnectOnNode1" in the kubernetes node:

```
dsctl delete phyconnector -n phyconnectOnNode1
```

3.5 help commands

Help commands are used to obtain help for any Digital Sandbox command.

The following help CLI commands are available:

dsctl

— help

dsctl help

Synopsis `dsctl help <command_name>`

Description The **dsctl help** command provides online assistance for Digital Sandbox CLI commands. Used alone without a command name, it provides a summary of the commands and how they are used. Alternatively, you can supply a command name for assistance in understanding a specific command.

Options The following are **dsctl help** parameters:

Parameter:	`<command_name>`
Used with operation:	`help`
Description:	Specifies the name of the command to get additional information for.

Examples To obtain a summary of available CLI commands:

```
dsctl help
```

To obtain help information for the **create** Sandbox CLI command:

```
dsctl help create
```

3.6 intent commands

Intent commands are used to create, delete, and obtain status for underlays (fabrics), workloads, and fully-qualified workloads.

The following intent CLI commands are available:

dsctl

— intent

— create

— underlay

— workload

— fully-qualified-workload

— delete

— underlay

— workload

— fully-qualified-workload

— list

— active-workloads

— underlay

— workload

— fully-qualified-workload

— status

— update

dsctl intent create

Synopsis

```
dsctl intent create underlay [-f <filename_string>] {-i <DNS_
ID_string> | -d <ID_name_string>} {-n <name_string> | -x
<DNS_name_string>}
```

```
dsctl intent create workload [-f <filename_string>] {-i <DNS_
ID_string> | -d <ID_name_string>} {-n <name_string> | -x
<DNS_name_string>}
```

```
dsctl intent create fully-qualified-workload [-f <filename_
string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n
<name_string> | -x <DNS_name_string>}
```

Description

The **dsctl intent create** command is used to create a Digital Sandbox intent context and has the following operations:

- The `underlay` operation specifies the creation of an underlay (or fabric) intent.
- The `workload` operation specifies the creation of a workload intent.

- The `fully-qualified-workload` operation specifies the creation of a fully qualified workload intent. A fully qualified workload is a workload that includes Customer Equipment (CE) endpoints and server configuration information.

Options

The following are **dsctl intent create {fully-qualified-workload | workload | underlay}** parameters:

Parameter: `-f <filename_string>`

Used with operation: `intent create {fully-qualified-workload | workload | underlay}`

Description: Specifies the name of the file containing the workload, fully qualified workload, or underlay definition.

Parameter: `-i <DNS_ID_string>`

Used with operation: `intent create {fully-qualified-workload | workload | underlay}`

Description: Specifies the DNS compatible intent ID. Either this parameter (`-i`) must be specified or `-d`.

Parameter: `-d <ID_name_string>`

Used with operation: `intent create {fully-qualified-workload | workload | underlay}`

Description: Specifies the intent name. Either this parameter (`-d`) must be specified or `-i`.

Parameter: `-n <name_string>`

Used with operation: `intent create {fully-qualified-workload | workload | underlay}`

Description: Specifies the name of the workload, fully qualified workload, or underly intent. Either this parameter (`-n`) must be specified or `-x`.

Parameter: `-x <DNS_name_string>`

Used with operation: `intent create {fully-qualified-workload | workload | underlay}`

Description: Specifies the DNS name of the workload, fully qualified workload, or underlay intent. Either this parameter (`-x`) must be specified or `-n`.

Examples

To create a workload intent using the file/definition located at "/tmp/workload_help.yaml", 'associated with the deployment ID "star", and name the workload "w_help":

```
dsctl intent create workload -i star -n w_help -f /tmp/workload_help.yaml
```

dsctl intent delete

Synopsis

```
dsctl intent delete underlay {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
dsctl intent delete workload {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
dsctl intent delete fully-qualified-workload {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
```

Description

The **dsctl intent delete** command is used to delete a Digital Sandbox intent resource and has the following operations:

- The `underlay` operation specifies the deletion of a underlay (fabric) intent.
- The `workload` operation specifies the deletion of a workload intent.
- The `fully-qualified-workload` operation specifies the deletion of a fully qualified workload intent. A fully qualified workload is a workload that includes Customer Equipment (CE) endpoints and server configuration information.

Options

The following are **dsctl intent delete {fully-qualified-workload | workload | underlay}** parameters:

Parameter: `-i <DNS_ID_string>`

Used with operation: `intent delete {fully-qualified-workload | workload | underlay}`

Description: Specifies the DNS compatible intent ID. Either this parameter (`-i`) must be specified or `-d`.

Parameter: `-d <ID_name_string>`

Used with operation: `intent delete {fully-qualified-workload | workload | underlay}`

Description: Specifies the intent name. Either this parameter (`-d`) must be specified or `-i`.

Parameter: `-n <name_string>`

Used with operation: `intent delete {fully-qualified-workload | workload | underlay}`

Description:	Specifies the name of the workload, fully qualified workload, or underly intent. Either this parameter (`-n`) must be specified or `-x`.
Parameter:	`-x <DNS_name_string>`
Used with operation:	`intent delete {fully-qualified-workload | workload | underlay}`
Description:	Specifies the DNS name of the workload, fully qualified workload, or underlay intent. Either this parameter (`-x`) must be specified or `-n`.

Examples

To delete a workload intent named "workload-alpha" associated with ID "intentexample":

```
dsctl intent delete workload -i intentexample -n workload-alpha
```

dsctl intent list

Synopsis

```
dsctl intent list
```

```
dsctl intent list active-workloads {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>} [-o <output_format>]
```

```
dsctl intent list underlay [-s <datastore_string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>} [-o <output_format>]
```

```
dsctl intent list workload [-s <datastore_string>>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>} [-o <output_format>]
```

```
dsctl intent list fully-qualified-workload [-s <datastore_string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>} [-o <output_format>]
```

Description

The **dsctl intent list** command is used to retrieve a list of intent resources and has the following operations:

- The `active workload` operation retrieves a list of all currently active workload intents.
- The `underlay` operation retrieves a list of underlay (fabric) intents.
- The `workload` operation retrieves a list of workload intents.
- The `fully-qualified-workload` operation retrieves a list of fully qualified workload intents. A fully qualified workload is a workload that includes Customer Equipment (CE) endpoints and server configuration information.

Options

The following are **dsctl intent list {active-workload | fully-qualified-workload | workload | underlay}** parameters:

Parameter:	`-s <datastore_string>`
Used with operation:	`intent list {fully-qualified-workload | workload | underlay}`
Description:	Specifies the datastore. Valid options are: • candidate (default) • running Candidates show new/modified configuration that have not been committed. Running mode includes running or active configurations.
Parameter:	`-i <DNS_ID_string>`
Used with operation:	`intent list {active-workload | fully-qualified-workload | workload | underlay}`
Description:	Specifies the DNS compatible intent ID. Either this parameter (`-i`) must be specified or `-d`.
Parameter:	`-d <ID_name_string>`
Used with operation:	`intent list {active-workload | fully-qualified-workload | workload | underlay}`
Description:	Specifies the intent name. Either this parameter (`-d`) must be specified or `-i`.
Parameter:	`-n <name_string>`
Used with operation:	`intent list {active-workload | fully-qualified-workload | workload | underlay}`
Description:	Specifies the name of the workload, fully qualified workload, or underly intent. Either this parameter (`-n`) must be specified or `-x`.
Parameter:	`-x <DNS_name_string>`
Used with operation:	`intent list {active-workload | fully-qualified-workload | workload | underlay}`
Description:	Specifies the DNS name of the workload, fully qualified workload, or underlay intent. Either this parameter (`-x`) must be specified or `-n`.

Parameter:	`-o <output_format_string>`
Used with operation:	`intent list {active-workload | fully-qualified-workload | workload | underlay}`
Description:	Defines the output format for the list. Valid options are: • json (default) • yaml

Examples

To retrieve a list of all intent deployments and their corresponding defined workloads/underlays:

```
dsctl intent list
```

To retrieve a list of all workload types and underlays associated with the deployment named "star":

```
dsctl intent list -i star
```

To retrieve a list of all underlays associated with the deployment named "star":

```
dsctl intent list underlay -i star
```

To show the details of the underlay named "underlay1" associated with the deployment named "star":

```
dsctl intent list underlay -i star -n underlay1
```

To show all the workloads associated with the deployment named "star" that are in the running datastore:

```
dsctl intent list workload -i star -s running
```

dsctl intent status

Synopsis

```
dsctl intent status {-i <DNS_ID_string> | -d <ID_name_string>}
```

Description

The **dsctl intent status** command is used to obtain the status for intents.

Options

The following are **dsctl intent status** parameters:

Parameter:	`-i <DNS_ID_string>`
Used with operation:	`intent status`
Description:	Specifies the DNS compatible intent ID. Either this parameter (`-i`) must be specified or `-d`.
Parameter:	`-d <ID_name_string>`
Used with operation:	`intent status`

Description:	Specifies the intent name. Either this parameter (`-d`) must be specified or `-i`.

Examples

To show the status of all intent deployments and their corresponding defined workloads/underlays:

```
dsctl intent status
```

To show the status of all intent deployments and their corresponding defined workloads/underlays associated with the deployment named "star":

```
dsctl intent status -i star
```

dsctl intent update

Synopsis

```
dsctl intent update underlay [-f <filename_string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
```

```
dsctl intent update workload [-f <filename_string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
```

```
dsctl intent update fully-qualified-workload [-f <filename_string>] {-i <DNS_ID_string> | -d <ID_name_string>} {-n <name_string> | -x <DNS_name_string>}
```

Description

The **dsctl intent update** command is used to update an existing Digital Sandbox intent context. During the update, the candidate versions are promoted to running, and links or nodes are added, removed, and updated as needed. This command has the following operations:

- The `underlay` operation specifies the update of a underlay (fabric) intent.
- The `workload` operation specifies the update of a workload intent.
- The `fully-qualified-workload` operation specifies the update of a fully qualified workload intent. A fully qualified workload is a workload that includes Customer Equipment (CE) endpoints and server configuration information.

Options

The following are **dsctl intent update {fully-qualified-workload | workload | underlay}** parameters:

Parameter:	`-f <filename_string>`
Used with operation:	`intent update {fully-qualified-workload |workload | underlay}`
Description:	Specifies the name of the file containing the workload, fully qualified workload, or underlay definition.
Parameter:	`-i <DNS_ID_string>`

Used with operation:	`intent update {fully-qualified-workload | workload | underlay}`
Description:	Specifies the DNS compatible intent ID. Either this parameter (`-i`) must be specified or `-d`.
Parameter:	`-d <ID_name_string>`
Used with operation:	`intent update {fully-qualified-workload | workload | underlay}`
Description:	Specifies the intent name. Either this parameter (`-d`) must be specified or `-i`.
Parameter:	`-n <name_string>`
Used with operation:	`intent update {fully-qualified-workload | workload | underlay}`
Description:	Specifies the name of the workload, fully qualified workload, or underly intent. Either this parameter (`-n`) must be specified or `-x`.
Parameter:	`-x <DNS_name_string>`
Used with operation:	`intent update {fully-qualified-workload | workload | underlay}`
Description:	Specifies the DNS name of the workload, fully qualified workload, or underlay intent. Either this parameter (`-x`) must be specified or `-n`.

Examples

To update a workload intent named "w_help" using the file/definition located at "/tmp/workload_help.yaml", and that is associated with the deployment ID "star":

```
dsctl intent update workload -i star -n w_help -f /tmp/workload_help.yaml
```

3.7 list commands

List commands are used to list Digital Sandbox resources and configurations.

The following list CLI commands are available:

dsctl

- list
  - checkpoints
  - config

— constants
— defaults
— fssconnect
— nokia-srlinux-license
— deployments
— image
— links
— nodes
— phyconnector

dsctl list checkpoint

dsctl list config

Synopsis

```
dsctl list config constants
dsctl list config defaults
dsctl list config fssconnect
dsctl list config nokia-srlinux-license [-o <output_format_
string>] [-v <version_string>]
```

Description

The **dsctl list config** command is used to list Digital Sandbox configuration details and has the following operations:

- The `constants` operation lists the Digital Sandbox constants configurations. This configuration can be set, but not changed. The initial setting occurs at installation.
- The `defaults` operation lists the Digital Sandbox default configurations.
- The `fssconnect` operation lists the FSSconnect configurations.
- The `nokia-srlinux-license` operation lists all Nokia SR Linux licenses (version and key) or a specified version.

Options

The following are **dsctl list config nokia-srlinux-license** parameters:

Parameter:	`-o <output_format_string>`
Used with operation:	`list config nokia-srlinux-license`
Description:	Lists the SR Linux licenses in a specified format. Valid options are: • table (default) • text
Parameter:	`-v <version_string>`
Used with operation:	`list config nokia-srlinux-license`

Description: Lists the specific license for the version specified.

Examples

To list the Nokia SR Linux licenses in text format (instead of a table) for version "srl_rel_0.0.0":

```
dsctl list config nokia-srlinux-license -o text -v srl_rel_0.0.0
```

To list the Digital Sandbox default configuration:

```
dsctl list config defaults
```

Synopsis

```
dsctl list checkpoint -i <ID_string> [-n <name_string>]
```

Description

The **dsctl list checkpoint** command is used to list available checkpoints. A checkpoint is a saved version of the running file that you can revert back to. If you do not provide the name (-n), all the checkpoints available in the deployment are given. If the name is given, the details of a specific checkpoint are returned.

Options

The following are **dsctl list checkpoint** parameters:

Parameter: `-i <ID_string>`

Used with operation: `list checkpoint`

Description: Specifies the deployment ID. Mandatory field.

Parameter: `-n <name_string>`

Used with operation: `list checkpoint`

Description: Name of the checkpoint.

Examples

To list all the checkpoints associated with a deployment ID named "star":

```
dsctl list checkpoint -i star
```

To list the details of a checkpoint named "checkpt1" associated with deployment ID named "star":

```
dsctl list checkpoint -i star -n checkpt1
```

dsctl list deployments

Synopsis

```
dsctl list deployments [-i <ID_string>] [-o <output_format>]
```

Description

The **dsctl list deployments** command is used to list Digital Sandbox configuration details for all deployments or for a specific deployment ID.

Options

The following are **dsctl list deployments** parameters:

Parameter: `-i <ID_string>`

Used with operation: `list deployments`

Description:	Specifies the deployment ID.
Parameter:	`-o <output_format>`
Used with operation:	`list deployments`
Description:	Displays the output in a specified format. Valid options are: • json (default) • yaml

Examples To list deployment details for deployment ID "star" in yaml format:

```
dsctl list deployments -i star -o yaml
```

dsctl list links

Synopsis `dsctl list links -i <ID_string> [-e <endpoint_string>]`

Description The **dsctl list links** command is used to list all links in a Digital Sandbox deployment.

Options The following are **dsctl list links** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`list links`
Description:	Specifies the deployment ID. Mandatory field.
Parameter:	`-e <endpoint_string>`
Used with operation:	`list links`
Description:	Specifies a network element name that needs to be a part of the definition.

Examples To list links associated with deployment ID "star" with endpoint "endpoint1":

```
dsctl list links -i star -e endpoint1
```

dsctl list nodes

Synopsis `dsctl list links -i <ID_string>`

Description The **dsctl list nodes** command is used to list nodes or network elements in a Digital Sandbox deployment.

Options The following are **dsctl list nodes** parameters:

Parameter:	`-i <ID_string>`

Used with operation: `list nodes`

Description: Specifies the deployment ID. Mandatory field.

Examples To list a node or network elements associated with deployment ID "star":

```
dsctl list nodes -i star
```

dsctl list phyconnector

Synopsis `dsctl list phyconnector [-n <name_string>]`

Description The **dsctl list phyconnector** command is used to list all or one specified phyconnectors. A phyconnector is used to connect a physical interface, device, or network to the containerized network.

Options The following are **dsctl list phyconnector** parameters:

Parameter: `-n <name_string>`

Used with operation: `list phyconnector`

Description: Name of the phyconnector resource.

Examples To list all phyconnectors:

```
dsctl list phyconnector
```

To list a specific phyconnector named "phyconnector1":

```
dsctl list phyconnector -n phyconnector1
```

3.8 load commands

Load commands are used to load a saved version of a running file that you can convert back to.

The following load CLI commands are available:

dsctl

- load
  - checkpoint
  - snapshot

dsctl load checkpoint

Synopsis `dsctl load checkpoint -i <ID_string> -n <checkpoint_string>`

Description The **dsctl load checkpoint** command is used to load a specific checkpoint. A checkpoint is a saved version of the running file that you can revert back to.

Options The following are **dsctl load checkpoint** parameters:

Parameter: -i <ID_string>

Used with operation: `load checkpoint`

Description: Specifies the deployment ID. Mandatory field.

Parameter: -n <checkpoint_string>

Used with operation: `load checkpoint`

Description: Specifies the checkpoint name. Mandatory field.

Examples To load a checkpoint named "chkpoint1" that is associated with deployment ID "star":

```
dsctl load checkpoint -i star -n chkpoint1
```

dsctl load snapshot

Synopsis `dsctl load snapshot -i <ID_string> [-l <location_string> [-t <timeout_integer>]`

Description The **dsctl create checkpoint** command is used to load a specific snapshot.

Options The following are **dsctl load snapshot** parameters:

Parameter: `-h, --help`

Description: Help for snapshot

Parameter: `-i <ID_string>`

Description: Specifies the deployment ID. Mandatory field.

Parameter: `-l <location_string>`

Description: File name and path of destination tar file.

Parameter: `-t <timeout_integer>`

Used with operation: `create checkpoint`

Description: Specifies the timeout value as an integer. This is the time in minutes to wait for an ACK of the processed message. Default is 3.

The following are **dsctl load snapshot** global parameters:

Parameter: `-c, <config_string>`

Description: Specifies the configuration file.

Parameter:	`-c, <log.level_string>`
Description:	Specifies the logging level (default is "info").

3.9 log commands

Log commands are used to retrieve Digital Sandbox related logs.

The following log CLI commands are available:

dsctl

— logs

— all

— deployment

dsctl logs all

Synopsis `dsctl logs all [-t <tar_string>]`

Description The **dsctl logs all** command is used to retrieve all the Digital Sandbox logs. If a specific tar string is not specified (using the -t flag), the logs are sent to */tmp/logs.tar*.

Options The following are **dsctl logs all** parameters:

Parameter:	`-t <tar_string>`
Used with operation:	`logs all`
Description:	Optionally define the name of the local tar file that will contain the logs. The default is: "/tmp/logs.tar".

Examples To retrieve all the Digital Sandbox logs and send them to "/tmp/logs.tar":

```
dsctl logs all
```

To retrieve all the Digital Sandbox logs and send them to a tar file that you have named "newlogs":

```
dsctl logs all -t /tmp/newlogs.tar
```

dsctl logs deployment

Synopsis `dsctl logs deployment -i <ID_string> [-t <tar_string>]`

Description The **dsctl logs deployment** command is used to retrieve all the Digital Sandbox logs associated with a specific deployment ID.

Options The following are **dsctl logs deployment** parameters:

Parameter:	`-i <ID_string>`

Used with operation:	`logs deployment`
Description:	Defines the deployment ID. Mandatory field.
Parameter:	`-t <tar_string>`
Used with operation:	`logs deployment`
Description:	Optionally define the name of the local tar file that will contain the logs. The default is: "/tmp/logs.tar".

Examples

To retrieve all the Digital Sandbox logs associated with a deployment ID called "star" and send them to "/tmp/logs.tar":

```
dsctl logs deployment -i star
```

To retrieve all the Digital Sandbox logs associated with a deployment ID called "star" and send them to a tar file that you have named "newlogs":

```
dsctl logs deployment -i star -t /tmp/newlogs.tar
```

3.10 status commands

Status commands are used to obtain the status of Digital Sandbox resources.

The following status CLI commands are available:

dsctl

— status

— deployment

— phyconnector

dsctl status deployment

Synopsis

```
dsctl status deployment -i <ID_string>
```

Description

The **dsctl status deployment** command is used to retrieve the status of a specified Digital Sandbox deployment.

Options

The following are **dsctl status deployment** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`status deployment`
Description:	Defines the deployment ID. Mandatory field.

Examples

To retrieve the status of a Digital Sandbox deployment associated with deployment ID "star":

```
dsctl status deployment -i star
```

dsctl status phyconnector

Synopsis `dsctl status phyconnector -n <name_string>`

Description The **dsctl status phyconnector** command is used to retrieve the status of a specified phyconnector. A phyconnector is used to connect a physical interface, device, or network to the containerized network.

Options The following are **dsctl status phyconnector** parameters:

Parameter:	`-n <name_string>`
Used with operation:	`status phyconnector`
Description:	Defines the name of the phyconnector. Mandatory field.

Examples To retrieve the status of a phyconnector named "phyconnect1":

```
dsctl status phyconnector -n phyconnect1
```

3.11 traffic commands

Traffic commands are used to analyze and compare traffic, and obtain status.

The following traffic CLI commands are available:

dsctl
- traffic
  - analyze
    - traffic-run
  - compare
    - traffic-runs
  - create
    - tcap-profile
    - tgen-profile
  - list
    - stream
    - traffic-profiles
    - traffic-runs
  - message
    - tcap
    - tgen

dsctl traffic analyze

Synopsis

```
dsctl traffic analyze traffic-run -i <ID_string> -p <profile_
string> -r <run_string> [-s <stream_ID_string>
```

Description

The **dsctl traffic analyze** command is used to analyze a traffic run. It takes the raw capture data and performs an analysis on this data, and has the following operation:

- The `traffic-run` operation is used to analyze the traffic capture results from different traffic capturers.

Options

The following are **dsctl traffic analyze traffic-run** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`traffic analyze traffic-run`
Description:	Defines the deployment ID. Mandatory field.
Parameter:	`-p <profile_string>`
Used with operation:	`traffic analyze traffic-run`
Description:	Defines the traffic profile name. Mandatory field.
Parameter:	`-r <run_string>`
Used with operation:	`traffic analyze traffic-run`
Description:	Defines the name of the capture to use as the baseline run or the target. Mandatory field.
Parameter:	`-s <stream_ID_string>`
Used with operation:	`traffic analyze traffic-run`
Description:	Defines the ID of the stream to analyze. Default is "-1”.

Examples

To analyze a traffic run associated with deployment ID "star” with a traffic profile named "profile1”, and a capture named "baseline1” to use as the baseline run:

```
dsctl traffic analyze traffic-run -i star -p profile1 -r
baseline1
```

dsctl traffic compare

Synopsis

```
dsctl traffic compare traffic-runs -b <baseline_string> -i
<ID_string> -p <profile_string> -t <target_string>
```

Description

The **dsctl traffic compare** command is used to compare two traffic runs within the same profile (but with different run names), and has the following operation:

- The `traffic-runs` operation is used to generate a report that highlights differences between two traffic runs for a specified traffic profile.

Options

The following are **dsctl traffic compare traffic-runs** parameters:

Parameter:	`-b <baseline_string>`
Used with operation:	`traffic compare traffic-runs`
Description:	Defines the baseline run of the traffic capture. Mandatory field.
Parameter:	`-i <ID_string>`
Used with operation:	`traffic compare traffic-runs`
Description:	Defines the deployment ID. Mandatory field.
Parameter:	`-p <profile_string>`
Used with operation:	`traffic compare traffic-runs`
Description:	Defines the traffic profile of the run to compare. Mandatory field.
Parameter:	`-t <target_string>`
Used with operation:	`traffic compare traffic-runs`
Description:	Defines the target run name of the traffic run. Mandatory field.

Examples

For a traffic profile named "profile1" associated with deployment ID "star", compare the baseline traffic run named "baselinerun" with the target run named "targetrun":

```
dsctl traffic compare traffic-runs -i star -p profile1-b
baselinerun -t targetrun
```

dsctl traffic create

Synopsis

```
dsctl traffic create tcap-profile -i <ID_string> [-j] [-e] [-
p <profile_string>] [-s] [-t <timeout_integer>] [-m <capture_
type_string>]
```

```
dsctl traffic create tgen-profile -i <ID_string> [-d
<streampacket_interval_integer>]
```

Description

The **dsctl traffic create** command is used to create Digital Sandbox traffic resources and has the following operations:

- The `tcap-profile` operation creates a capture profile based on the current profile where the capture occurs on all interfaces or only the eternal gatenet interfaces (to and from the simulators at the edges of the Digital Sandbox).

- The `tgen-profile` operation creates a traffic profile that can be sent to the traffic generators based on the available routes in the topology. Output is directed to "stoudt".

Options

The following are **dsctl traffic create tcap-profile** parameters:

Parameter: `-i <ID_string>`

Used with operation: `traffic create tcap-profile`

Description: Defines the deployment ID. Mandatory fields.

Parameter: `-j`

Used with operation: `traffic create tcap-profile`

Description: If used, the capturer should use delay and jitter statistics gathering.

Parameter: `-e`

Used with operation: `traffic create tcap-profile`

Description: If used, creates a profile using only external interfaces.

Parameter: `-p <profile_string>`

Used with operation: `traffic create tcap-profile`

Description: Defines the profile name of the capture. Default is "default". Mandatory field.

Parameter: `-s`

Used with operation: `traffic create tcap-profile`

Description: If specified, traffic capturing should perform sequence number checks.

Parameter: `-t <timeout_integer>`

Used with operation: `traffic create tcap-profile`

Description: Defines the capture timeout value. Default is "30".

Parameter: `-m <capture_type_string>`

Used with operation: `traffic create tcap-profile`

Description: Defines the type of capture structure to generate. Options:

- init
- auto-timeout (default)

The following are **dsctl traffic create tgen-profile** parameters:

Parameter:	`-i <ID_string>`
Used with operation:	`traffic create tgen-profile`
Description:	Defines the deployment ID. Mandatory field.
Parameter:	`-d <streampacket_interval_integer>`
Used with operation:	`traffic create tgen-profile`
Description:	Defines the default time in milliseconds (ms) between two packets on the same simulator. Default is "10".

Examples

To create a capture profile based on the current profile with a deployment ID of "star", a profile name of "profile2", using delay/jitter checks, and with a capturing timeout of "25":

```
dsctl traffic create tcap-profile -i star -p profile2 -j -t 25
```

To create a traffic profile that can be sent to the traffic generators based on the available routes that are associated with deployment ID "star" with a time of 11ms between packets:

```
dsctl traffic create tgen-profile -i star -d 11
```

dsctl traffic list

Synopsis

```
dsctl traffic list stream [-s <stream_ID_string>] [-f <tgenfile_string>]
dsctl traffic list traffic-profiles -i <ID_string>
dsctl traffic list traffic-runs -i <ID_string> [-p <profile_string>]
```

Description

The **dsctl traffic list** command is used to list Digital Sandbox resources and has the following operations:

- The `stream` operation lists the stream parameters for a specific tgen profile.
- The `traffic-profiles` operation lists the available traffic profiles.
- The `traffic-run` operation lists the available traffic runs.

Options

The following are **dsctl traffic list stream** parameters:

Parameter:	`-s <stream_ID_string>`

Used with operation: `traffic list stream`

Description: Defines the stream ID of the stream to list. Default is "-1".

Parameter: `-f <tgenfile_string>`

Used with operation: `traffic list stream`

Description: Defines the name of the traffic generator init file.

The following are **dsctl traffic list traffic-profiles** parameters:

Parameter: `-i <ID_string>`

Used with operation: `traffic list traffic-profiles`

Description: Defines the deployment ID. Mandatory field.

The following are **dsctl traffic list traffic-runs** parameters:

Parameter: `-i <ID_string>`

Used with operation: `traffic list traffic-runs`

Description: Defines the deployment ID. Mandatory field.

Parameter: `-p <profile_string>`

Used with operation: `traffic list traffic-runs`

Description: Defines the name of the traffic profile. Default is "*".

Examples

To list the stream parameters associated with all streams:

```
dsctl traffic list stream -s -1
```

To list available traffic profiles associated with deployment ID "star":

```
dsctl traffic list traffic-profiles -i star
```

To list the available traffic runs associated with deployment ID "star" with a profile name of "profile1":

```
dsctl traffic list traffic-runs -i star -p profile1
```

dsctl traffic message

Synopsis

```
dsctl traffic message tcap -i <ID_string> -f <tcaprun_string>
[-p <profile_string>] [-r <profilerun_string>]
dsctl traffic message tgen -i <ID_string> -f <tgenrun_string>
```

Description

Traffic capturing and sending is controlled by sending the corresponding traffic message. The **dsctl traffic message** command is used to message Digital Sandbox components and has the following operations:

- The `tcap` operation sends a message to the traffic capturers running in the deployment.
- The `tgen` operation sends a message to the traffic generators running in the deployment.

Options

The following are **dsctl traffic message tcap** parameters:

Parameter: `-i <ID_string>`

Used with operation: `traffic message tcap`

Description: Defines the deployment ID. Mandatory field.

Parameter: `-f <tcaprun_string>`

Used with operation: `traffic message tcap`

Description: Defines the capture init file. Mandatory field.

Parameter: `-p <profile_string>`

Used with operation: `traffic message tcap`

Description: Specifies the profile name of the traffic run. When specified, overwrites the testname/profile of the associated capture.

Parameter: `-r <profilerun_string>`

Used with operation: `traffic message tcap`

Description: Specifies the profile run name of the traffic run. When specified, overwrites the Capture Name/Run of the capture control data.

The following are **dsctl traffic message tgen** parameters:

Parameter: `-i <ID_string>`

Used with operation: `traffic message tgen`

Description: Defines the deployment ID. Mandatory field.

Parameter: `-f <tgenrun_string>`

Used with operation: `traffic message tgen`

Description: Defines the traffic generator init file. Mandatory field.

Examples

To send a message to the traffic capturer associated with deployment ID "star" and with a path/filename to the capture profile of "tmp/tcapprofile". Overwrite the profile of the associated capture:

```
dsctl traffic message tcap -i star -f tmp/tcapprofile -p
```

To send a message to the traffic generators running in deployment ID "star" and with a path/filename to the capture generator of "tmp/tgenprofile":

```
dsctl traffic message tgen -i star -f tmp/tgenprofile
```

3.12 update commands

Update commands are used to update Digital Sandbox resources such as configurations and licenses.

The following update CLI commands are available:

dsctl

— update

— config

— defaults

— nokia-srlinux-license

— deployment

— topology

dsctl update config

Synopsis

```
dsctl update config defaults [-e <gluster_endpoint>] [-n
<gluster_endpoint_namespace>] [-v <gluster_volume-name>] [-s
<imagepullsecret_string>] [-k <K8_nodeselector_string>]
```

```
dsctl update config license
```

```
dsctl update config license-config
```

```
dsctl update config nokia-srlinux-license [-f <filename>] [-k
<license_key_name>] [-v <version>]
```

Description

The **dsctl update config** command is used to update Digital Sandbox configuration details and has the following operations:

- The `defaults` operation updates the Digital Sandbox default configuration.
- The `nokia-srlinux-license` operation updates a license key for a specified Nokia SR Linux version.

Options

The following are **dsctl update config default** parameters:

Parameter: `-e <gluster_endpoint_string>`

Used with operation:	`update config default`
Description:	Updates the default configuration with the specified gluster endpoint.
Parameter:	`-n <gluster_endpoint_namespace>`
Used with operation:	`update config default`
Description:	Updates the default configuration with the specified namespace of the gluster endpoint.
Parameter:	`-v <gluster_volume-name>`
Used with operation:	`update config default`
Description:	Updates the default configuration with the specified gluster volume-name.
Parameter:	`-s <imagepullsecret_string>`
Used with operation:	`update config default`
Description:	Updates the list of kubernetes image pull secrets, stored in the digital-sandbox-system namespace, that can be used during Digital Sandbox deployment creation. Default = regcred.
Parameter:	`-k <K8_nodeselector_string>`
Used with operation:	`update config default`
Description:	Updates the kubernetes node selector to use.

The following are **dsctl update config nokia-srlinux-license** parameters:

Parameter:	`-f <filename>`
Used with operation:	`update config nokia-srlinux-license`
Description:	Defines the name of the SR Linux license file.
Parameter:	`-k <license_key_name>`
Used with operation:	`update config nokia-srlinux-license`
Description:	Defines the license key that corresponds to the Nokia SR Linux version.

Parameter:	`-v <version>`
Used with operation:	`update config nokia-srlinux-license`
Description:	Defines the version of the Nokia SR Linux key.

Examples

To update the Digital Sandbox default configuration with a kubernetes endpoint value of "glusterEP2":

```
dsctl update config defaults -e glusterEP2
```

To update the license key for a specified Nokia SR Linux version:

```
dsctl update config nokia-srlinux-license -k 00000000-0000-0000-0000-000000000000 -v srl_rel_21_6
```

dsctl update deployment

Synopsis

```
dsctl update deployment -f <filename_string> -i <ID_string> [-t <tar_filename_string>]
```

Description

The **dsctl update deployment** command is used to update an existing Digital Sandbox deployment.

Options

The following are **dsctl update deployment** parameters:

Parameter:	`-f <filename_string>`
Used with operation:	`update deployment`
Description:	Specifies the pathname of the file containing the node data. Mandatory field.

Parameter:	`-i <ID_string>`
Used with operation:	`update deployment`
Description:	Specifies the deployment ID. Mandatory field.

Parameter:	`-t <tar_filename_string>`
Used with operation:	`update deployment`
Description:	Specifies the pathname of the tar file that contains the configuration and state details required to build a deployment.

Examples

To update a deployment with a deployment ID named "star", that has a filename of "/tmp2/7live.yaml", and with a new configuration tar file named "/tmp2/7live2.tar":

```
dsctl update deployment -i star -f /tmp2/7live.yaml -t /tmp2/
7live2.tar
```

dsctl update topology

Synopsis

```
dsctl update topology -f <filename_string> -i <ID_string> [-t
<tar_filename_string>]
```

Description

The **dsctl update topology** command is used to update an existing Digital Sandbox topology.

Options

The following are **dsctl update topology** parameters:

Parameter: `-f <filename_string>`

Used with operation: `update topology`

Description: Specifies the pathname of the file containing the topology data. Mandatory field.

Parameter: `-i <ID_string>`

Used with operation: `update topology`

Description: Specifies the deployment ID. Mandatory field.

Parameter: `-t <tar_filename_string>`

Used with operation: `update topology`

Description: Specifies the pathname of the tar file that contains the configuration and state details required to build a topology.

Examples

To update a topology associated with a deployment ID named "star", that has a filename of "/tmp2/7live3.yaml", and with a new configuration tar file named "/tmp2/toplog2":

```
dsctl update topology -i star -f /tmp2/7live3.yaml -t /tmp2/
toplog2.tar
```

3.13 version command

Version commands are used to display the current version of the Digital Sandbox.

The following version CLI commands are available:

dsctl

— version

— details

dsctl version

Synopsis	`dsctl version [details]`
Description	The **dsctl version** command displays the current version of the Digital Sandbox. When used with **details**, versions are provided for the main components of the Digital Sandbox (ds-apiserver, ds-cli, ds-imgsvc, g8netdock, sftpserver).
Options	No additional parameters. Global flags apply.

Examples	To display the current version of the Digital Sandbox:
	`dsctl version`
	To display the current version of all Digital Sandbox components:
	`dsctl version details`

4 Use cases

This chapter provides sample use cases for the Fabric Services System Digital Sandbox (DS) CLI commands. Examples of command sequence and sample output are provided. Two use cases are presented:

- Using the Digital Sandbox - standalone

  In this case, all configuration is performed using CLI commands in the Digital Sandbox.

- Using the Digital Sandbox - integrated with the Fabric Services System GUI

  In this case, the user can create some elements on the system GUI and deploy these to the Digital Sandbox. For the simulation session to fully work, some configuration (for example, endpoints) must be performed in the Digital Sandbox.

In addition, a section for Checking a deployments status is provided that outlines helpful commands to verify that your deployment was successfully created and running.

For more information about using the Fabric Services System and the Digital Sandbox, users should understand the concepts presented in the *Fabric Services System User Guide*.

4.1 Using the Digital Sandbox - standalone

In this use case, all configuration is performed using CLI commands in the Digital Sandbox.

When the Digital Sandbox is used standalone, a number of deployment sessions can be created verses only one deployment session if you are using the Digital Sandbox integrated with the Fabric Services System GUI. The total number is variable based on the DhcpRange specified. In addition, no configuration details are passed back to the Fabric Services System GUI.

Using the Digital Sandbox standalone, you must specify the different Network Elements (NEs) within the topology, and the L2 network segments that exist between the NEs. All elements at the border must be fully specified also.

4.1.1 Creating a deployment

To create a deployment (simulation session) for a standalone Digital Sandbox session, the following command is used:

```
dsctl create deployment -i <name> -f <file> [-t <tar-file>]
```

Where:

- <name> is the user-defined name of the deployment.
- <file> is the file (path+filename) that contains the necessary specification. This is a YAML encoded file used to define the deployment configuration.
- <tar-file> is a .tar file (path+filename) containing the configuration relevant to the NE defined in the 'file'. The .tar files contain a per-node directory that contain the config.json and state.json files of these nodes.

Using the **dsctl create deployment** command, there are two approaches to creating and updating a Digital Sandbox deployment:

- The topology is specified by the user.
- The topology is automatically derived using configuration/state files of networked SR Linux devices.

The sections that follow show an example of each and show how the .yaml file will differ for each method.

4.1.1.1 Creating a deployment (user specified topology)

This example shows the command and .yaml file content for an user specified topology.

In this example, the user specifies the different Network Elements (NEs) within the topology, and the L2 network segments that exist between those elements. All elements at the border are also fully specified.

Example:

```
dsctl create deployment -i star -f 7config.yaml -t 7config.tar
```

where the file `7config.yaml` contains the following:

```
deriveTopology: false
createSimulators: false
nodes:
  as--sim-1:
    deviceType: Simulator
    interfaces: [simitf-0, simitf-1, simitf-2]
    overrides:
      deriveConfig: false
  dut2:
    deviceType: NokSrLinux
    interfaces: [mgmt0, ethernet-1/1, ethernet-1/2, ethernet-1/3, ethernet-1/4, ethernet-1/5]
    mac: "00:01:02:00:00:00"
    uuid: helper
    chassisID: 00:01:02:ff:00:00
  dut3:
    deviceType: NokSrLinux
    interfaces: [ethernet-1/1, ethernet-1/2, ethernet-1/3, ethernet-1/4, mgmt0, ethernet-1/5]
    mac: "00:01:03:00:00:00"
    chassisID: 00:01:03:ff:00:00
  dut4:
    deviceType: NokSrLinux
    interfaces: [ethernet-1/1, ethernet-1/2, ethernet-1/3, ethernet-1/4, mgmt0, ethernet-1/5]
    mac: "00:01:04:00:00:00"
    chassisID: 00:01:04:ff:00:00
  dut5:
    deviceType: NokSrLinux
    interfaces: [ethernet-1/1, ethernet-1/2, ethernet-1/3, ethernet-1/4, mgmt0, ethernet-1/5]
    mac: "00:01:05:00:00:00"
    chassisID: 00:01:05:ff:00:00
  as--sim-2:
    deviceType: ThirdParty
    template: bgp-simulator
    interfaces: [simitf-1, simitf-2]
    overrides:
      deriveConfig: false
  as--sim-3:
    deviceType: Simulator
    interfaces: [simitf-1, simitf-2]
    overrides:
      deriveConfig: false
```

```
links:
- link:
  - node: as--sim-1
    itf: simitf-1
  - node: dut2
    itf: mgmt0
  - node: dut3
    itf: mgmt0
  - node: dut4
    itf: mgmt0
  - node: dut5
    itf: mgmt0
  isMgmtSegment: true
- link:
  - node: as--sim-1
    itf: simitf-0
  - node: dut2
    itf: ethernet-1/1
  isP2PSegment: true
- link:
  - node: as--sim-1
    itf: simitf-2
  - node: dut3
    itf: ethernet-1/1
- link:
  - node: dut2
    itf: ethernet-1/2
  - node: dut3
    itf: ethernet-1/2
  isP2PSegment: true
- link:
  - node: dut2
    itf: ethernet-1/3
  - node: dut4
    itf: ethernet-1/1
- link:
  - node: dut2
    itf: ethernet-1/4
  - node: dut5
    itf: ethernet-1/1
- link:
  - node: dut3
    itf: ethernet-1/3
  - node: dut4
    itf: ethernet-1/2
- link:
  - node: dut3
    itf: ethernet-1/4
  - node: dut5
    itf: ethernet-1/2
- link:
  - node: dut4
    itf: ethernet-1/3
  - node: dut5
    itf: ethernet-1/3
- link:
  - node: dut4
    itf: ethernet-1/4
  - node: as--sim-2
    itf: simitf-1
- link:
  - node: dut4
    itf: ethernet-1/5
  - node: as--sim-3
```

```
    itf: simitf-1
- link:
  - node: dut5
    itf: ethernet-1/4
  - node: as--sim-2
    itf: simitf-2
- link:
  - node: dut5
    itf: ethernet-1/5
  - node: as--sim-3
    itf: simitf-2
```

4.1.1.2 Creating a deployment (auto-derived topology)

This example shows the command and .yaml file content for an auto-derived topology.

When the .yaml file below is given as input, the Digital Sandbox assumes four SR Linux devices are running on node 10.0.0.1 as docker containers. They make up the network that the user wants to replicate. If no .tar file is provided, the Digital Sandbox will login to the nodes, collect config and state files for the nodes, and auto-derive the topology and generate simulators at the edges of the network.

Example:

`dsctl create deployment -i star -f 7live.yaml -t 7live.tar`

where the file `7live.yaml` contains the following:

```
deriveTopology: true
createSimulators: true
nodes:
  srlinux_home_jvandena_ws_srlinux4_dut2:
    deviceType: NokSrLinux
    remoteConnect:
      ip: 10.0.0.1
      login: mainuser
      password: pwd01
      connectType: docker
      connectNames: [srlinux_home_jvandena_ws_srlinux4_dut2]
  srlinux_home_jvandena_ws_srlinux4_dut3:
    deviceType: NokSrLinux
    remoteConnect:
      ip: 10.0.0.1
      login: mainuser
      password: pwd01
      connectType: docker
      connectNames: [srlinux_home_jvandena_ws_srlinux4_dut3]
  srlinux_home_jvandena_ws_srlinux4_dut4:
    deviceType: NokSrLinux
    remoteConnect:
      ip: 10.0.0.1
      login: mainuser
      password: pwd01
      connectType: docker
      connectNames: [srlinux_home_jvandena_ws_srlinux4_dut4]
  srlinux_home_jvandena_ws_srlinux4_dut5:
    deviceType: NokSrLinux
    remoteConnect:
      ip: 10.0.0.1
      login: mainuser
      password: pwd01
      connectType: docker
```

```
      connectNames: [srlinux_home_jvandena_ws_srlinux4_dut5]
```

4.1.2 Updating a deployment

When updating a deployment, the following command is used:

`dsctl update deployment -i <name> -f <file> [-t <tar-file>]`

Example:

`dsctl update deployment -i star -f 7live.yaml -t 7live.tar`

Assuming the `7live.yaml` file content is different from what is currently configured, the Digital Sandbox would automatically determine which nodes are new, updated, or removed, and update the deployment as specified.

4.2 Using the Digital Sandbox - integrated with the Fabric Services System GUI

With this use case, a user can create some elements on the system GUI and deploy these to the Digital Sandbox. For the simulation session to fully work, some configuration (for example, endpoints) must be performed in the Digital Sandbox.

When using the Fabric Services System GUI, only one region/Digital Sandbox deployment can be created, and the information flow is unidirectional from the GUI to the Digital Sandbox. This means that if an intent is updated in the Digital Sandbox, the corresponding changes will not flow back to the Fabric Services System. Also, if a region or intent deployed in the Digital Sandbox is removed from the GUI, it will also be removed from the Digital Sandbox.

4.2.1 Working with intents

The Fabric Services System integrates with the Digital Sandbox using intents.

When a region, or fabric/workload intent is created in the Fabric Services System GUI, and the user indicates it should be deployed to the Digital Sandbox, a corresponding digital construct (underlay/workload) is automatically created in the Digital Sandbox.

Digital Sandbox intents that map to the Fabric Services System include:

- underlay intents: Corresponds to the Fabric Services System underlay of a fabric (Leaf-Spine/Back-Bone).
- workload intents: Corresponds to the Fabric Services System workloads that can span multiple fabrics. These define what kind of services that run on the servers in the rack and how they connect.

Table 4: GUI and digital sandbox equivalents defines the Fabric Services System generated contexts, their Digital Sandbox equivalents, and the equivalent Digital Sandbox CLI used to perform the task.

Table 4: GUI and digital sandbox equivalents

Task performed on the Fabric Services System	Digital Sandbox construct created	Equivalent Digital Sandbox CLI command
Create Region for deployment in the Digital Sandbox	Digital Sandbox Intent	`dsctl intent`
Create fabric for deployment in the Digital Sandbox	Digital Sandbox underlay	`dsctl intent createunderlay`
Create workload for deployment in the Digital Sandbox	Digital Sandbox workload	`dsctl intent createworkload`

The Digital Sandbox also has two unique workload types:

- active-workload: For every workload that is made active (via an update), the Digital Sandbox automatically generates an active-workload. This specifies the endpoint server and the interface(s) used to simulate the endpoint connected via the workload subnet subinterface.
- fully-qualified-workload intents: Allows the user to specify the IP address configuration of the endpoint interfaces. A workload created on the Fabric Services System does not contain this information, and it must be created in the Digital Sandbox.

For additional information on using the Fabric Services System GUI for intent management, see the *Fabric Services System User Guide*.

4.2.2 Creating an intent/region

A single region can be created in the Fabric Services System GUI and be deployed to the Digital Sandbox. If this region is created, updated, or removed in the GUI, the same action is performed in the Digital Sandbox. See the *Fabric Services System User Guide* for details on creating regions in the GUI.

The following CLI command can also be used to create an intent. An intent is the equivalent of a creating region in the Fabric Services System GUI. Creation of a Digital Sandbox intent triggers the creation of the Digital Sandbox deployment. Intents created in the Digital Sandbox will *not* appear in the Fabric Services System GUI.

```
dsctl intent create -i <name>
```

4.2.3 Creating an underlay (fabric) intent

Fabric intents can be created in the Fabric Services System GUI and be deployed to the Digital Sandbox. In the Digital Sandbox, a fabric intent is referred to as an underlay. If any fabric intent is created, updated, or removed in the GUI, the same action is performed in the Digital Sandbox. See the *Fabric Services System User Guide* for details on creating fabric intents in the GUI.

The following CLI command can also be used to create an underlay. An underlay is the equivalent of a creating a fabric intent in the Fabric Services System GUI. Note that if an underlay is created in the Digital Sandbox, it will *not* be created in the GUI.

```
dsctl intent create underlay -i <name> -n <underlay-name> -f <file>
```

Example:

The following will create an underlay intent in the Digital Sandbox. In this example, the underlay is associated with the DNS-compatible intent ID named 'star', and will be named 'u1'. Data from the file /tmp/u1 is used to configure the underlay:

`dsctl intent create underlay -i star -n u1 -f /tmp/u1`

where the file `/tmp/u1` contains the following:

```
meta:
  intent: fspi352854779224915968
  id: LagLs1_gatenet_fabric
  name: fspf352854958523023360
  version: 11
  uid: lxqiblwcmuajqp2m
nodes:
- name: ls1-leaf-1
  donotsimulate: false
  type: NokSrLinux
  hwdetails:
    chassis: "66"
    cpmtype: "177"
    cardtype: "177"
    mdatype: "194"
  interfaces:
  - mgmt0
  - ethernet-1/1
  ...
  - ethernet-1/34
  mac: ""
  uuid: "352854967851155456"
  overrides:
    tag: 0.0.0-27910
- name: ls1-spine-1
  donotsimulate: false
  type: NokSrLinux
  hwdetails:
    chassis: "42"
    cpmtype: "69"
    cardtype: "127"
    mdatype: "182"
  interfaces:
  - mgmt0
  - ethernet-1/1
  ...
  - ethernet-4/36
  mac: ""
  uuid: "352854969713426432"
  overrides:
    tag: 0.0.0-27910
- name: ls1-leaf-2
  donotsimulate: false
  type: NokSrLinux
  hwdetails:
    chassis: "66"
    cpmtype: "177"
    cardtype: "177"
```

```
    mdatype: "194"
  interfaces:
  - mgmt0
  - ethernet-1/1
  ...
  - ethernet-1/34
  mac: ""
  uuid: "352854967851220992"
  overrides:
    tag: 0.0.0-27910
links:
- endpoints:
  - node: ls1-leaf-1
    interface: ethernet-1/3
- endpoints:
  - node: ls1-leaf-1
    interface: ethernet-1/4
- endpoints:
  - node: ls1-leaf-1
    interface: ethernet-1/5
- endpoints:
  - node: ls1-leaf-1
    interface: ethernet-1/6
- endpoints:
  - node: ls1-leaf-1
    interface: ethernet-1/34
  - node: ls1-spine-1
    interface: ethernet-1/8
- endpoints:
  - node: ls1-leaf-2
    interface: ethernet-1/27
  - node: ls1-spine-1
    interface: ethernet-1/9
lags:
- endpoints:
  - endpoints:
    - node: ls1-leaf-1
      interface: ethernet-1/10
    - node: ls1-leaf-1
      interface: ethernet-1/11
  name: lag1e10e11
- endpoints:
  - endpoints:
    - node: ls1-leaf-1
      interface: ethernet-1/12
    - node: ls1-leaf-1
      interface: ethernet-1/13
  name: lag1e12e13
- endpoints:
  - endpoints:
    - node: ls1-leaf-2
      interface: ethernet-1/10
    - node: ls1-leaf-2
      interface: ethernet-1/11
  name: lag2e10e11
- endpoints:
  - endpoints:
    - node: ls1-leaf-2
      interface: ethernet-1/12
    - node: ls1-leaf-2
      interface: ethernet-1/13
  name: lag2e12e13
```

4.2.4 Creating a workload intent

Workload intents can be created in the Fabric Services System GUI and be deployed to the Digital Sandbox. In the Digital Sandbox, a workload intent is referred to by the same name; workload. If any workload intent is created, updated, or removed in the GUI, the same action is performed in the Digital Sandbox. See the *Fabric Services System User Guide* for details on creating workload intents in the GUI.

The following CLI command can also be used to create a workload. Note that if a workload is created in the Digital Sandbox, it will *not* be created in the GUI.

```
dsctl intent create workload -i <name> -n <workload-name> -f <file>
```

Example:

The following will create a workload intent. In this example, the workload is associated with the DNS-compatible intent ID named 'star', and will be named 'w1'. Data from the file /tmp/w1 is used to configure the underlay:

```
dsctl intent create workload -i star -n w1 -f /tmp/w1
```

where the file `/tmp/w1` contains the following:

```
meta:
  intent: fspi352854779224915968
  id: EvpnWl1
  name: fspw352859849752576000
  version: 1
  uid: 1sd7znoq5f818jul
config:
  type: 2
  subnets:
    Br1Gw:
      identifier:
        id: "352859919361245184"
        name: Br1Gw
      vrftype: 1
      interfaces:
        TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-1-7220 IXR-D3.lag1e12e13.2:
          config:
            name: TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-1-7220
              IXR-D3.lag1e12e13.2
            node: ls1-leaf-1
            interface: lag1e12e13
            index: 0
            encap:
              type: 1
              dot1q: 2
            ips:
            - 12.12.12.1/24
            - 12.12.22.1/24
        TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-2-7220 IXR-D3.lag2e12e13.2:
          config:
            name: TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-2-7220
              IXR-D3.lag2e12e13.2
            node: ls2-leaf-2
            interface: lag2e12e13
            index: 0
            encap:
              type: 1
              dot1q: 2
            ips:
            - 12.12.12.1/24
```

```
        - 12.12.22.1/24
  Rt1:
    identifier:
      id: "352859941238734848"
      name: Rt1
    vrftype: 2
    interfaces:
      TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220 IXR-D3.lag1e12e13.3:
        config:
          name: TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220
            IXR-D3.lag1e12e13.3
          node: ls2-leaf-1
          interface: lag1e12e13
          index: 0
          encap:
            type: 1
            dot1q: 3
          ips:
          - 14.14.14.1/24
  br1NoGw:
    identifier:
      id: "352859881830612992"
      name: br1NoGw
    vrftype: 1
    interfaces:
      TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-1-7220 IXR-D3.lag1e10e11.1:
        config:
          name: TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-1-7220
            IXR-D3.lag1e10e11.1
          node: ls1-leaf-1
          interface: lag1e10e11
          index: 0
          encap:
            type: 1
            dot1q: 1
          ips: []
      TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-2-7220 IXR-D3.lag2e10e11.1:
        config:
          name: TestWorkloadwithLag.LagLs1.LagLs1_gatenet_fabric.ls1-leaf-2-7220
            IXR-D3.lag2e10e11.1
          node: ls1-leaf-2
          interface: lag2e10e11
          index: 0
          encap:
            type: 1
            dot1q: 1
          ips: []
```

4.2.5 Configuring the network edge

Whether configured on the Fabric Services System GUI or Digital Sandbox, underlay and workload intents lack sufficient information about the network edge. This limits the number of routes within a simulation, impacts traffic injection, and traffic network traversal is not possible.

This information is partially obtained through details in the active workload and through user input using the fully-qualified workload intent.

4.2.5.1 Generating an active workload

For every workload that is made active (via an update), the Digital Sandbox automatically generates an active-workload. This specifies the endpoint server and the interfaces or interfaces used to simulate the endpoint connected via the workload subnet subinterface. There is no equivalent of an active workload on the Fabric Services System GUI.

If you view an active workload, you can see that a peer section is added to the workload by the Digital Sandbox. This section contains:

- config: Specifies the endpoint node and interface, including potential encapsulation
- state: Specifies the virtual-sim and actual interfaces assigned to the interface specified in the config section
- protocols: Specifies the protocols running on the interface (not supported yet)

The following CLI command can be used to view an active workload:

`dsctl intent list active-workload -i <name> -n <active-workload-name>`

Example:

`dsctl intent list active-workload -i star -n aw1`

```
{
  "meta": {
    "intent": "star",
    "id": "w1",
    "name": "w-1c3qxz79ccypf82f",
    "version": 1,
    "uid": "65oun6n1r0w90yum"
  },
  "source": {
    "workload": {
      "uid": "3ixz5smw6lxnnl3e",
      "version": 1
    }
  },
  "config": {
    "subnets": {
      "Rt1": {
        "identifier": {
          "id": "352859941238734848",
          "name": "Rt1"
        },
        "vrf_type": 2,
        "interfaces": {
          "TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220 IXR-
D3.lag1e12e13.3": {
            "config": {
              "config": {
                "name": "TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220 IXR-
D3.lag1e12e13.3",
                "node": "ls2-leaf-1",
                "interface": "lag1e12e13",
                "encap": {
                  "type": 1,
                  "dot1q": 3
                },
                "ips": [
                  "14.14.14.1/24"
                ]
```

```
              },
              "peer": {
                "config": {
                  "node": "sim-0",
                  "interface": "lag-lag1e12e13",
                  "encap": {
                    "type": 1,
                    "dot1q": 3
                  }
                },
                "state": {
                  "vsim": "vsim-14",
                  "interfaces": [
                    "itf-14",
                    "itf-18"
                  ]
                },
                "protocols": {}
              }
            }
          }
        }
      }
    },
    "type": 2
  }
}
```

4.2.5.2 Creating a fully-qualified workload intent

To complete a deployment on the Digital Sandbox, the IP configuration and protocol definition of the peer is needed. This can only be created in the Digital Sandbox using a fully-qualified workload intent. There is no equivalent in the Fabric Services System GUI.

The following CLI command is used to create a fully-qualified workload intent. A fully-qualified-workload has the same sections and subsections as the active-workload, but also includes the IP configuration and protocol definition of the peer.

`dsctl intent create fully-qualified -i <name> -n <workload-name> -f <file>`

Example:

The following will create a fully-qualified workload intent. In this example, note that an 'ips' section is now included in the peer subsection.

`dsctl intent create fully-qualified-workload -i star -n fqw1 -f /tmp/fqw1`

where the file `/tmp/fqw1` contains the following:

```
{
  "meta": {
    "intent": "star",
    "id": "fqw1",
    "name": "w-1c3qxz79ccypf82f",
    "version": 1,
    "uid": "ujc2zn8i5e1kvh47"
  },
  "config": {
    "subnets": {
      "Rt1": {
        "identifier": {
```

```
          "id": "352859941238734848",
          "name": "Rt1"
        },
        "vrf_type": 2,
        "interfaces": {
          "TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220 IXR-
D3.lag1e12e13.3": {
            "config": {
              "config": {
                "name": "TestWorkloadwithLag.LagLs2.LagLs2_gatenet_fabric.ls2-leaf-1-7220 IXR-
D3.lag1e12e13.3",
                "node": "ls2-leaf-1",
                "interface": "lag1e12e13",
                "encap": {
                  "type": 1,
                  "dot1q": 3
                },
                "ips": [
                  "14.14.14.1/24"
                ]
              },
              "peer": {
                "config": {
                  "node": "sim-0",
                  "interface": "lag-lag1e12e13",
                  "encap": {
                    "type": 1,
                    "dot1q": 3
                  },
                  "ips": [
                    "14.14.14.76/24"
                  ]
                },
                "state": {
                  "vsim": "vsim-14",
                  "interfaces": [
                    "itf-14",
                    "itf-18"
                  ]
                },
                "protocols": {}
              }
            }
          }
        }
      }
    },
    "type": 2
  }
}
```


4.3 Verifying a deployment

This section provides examples of helpful commands to verify that your simulation session was successfully created.

4.3.1 Displaying details of a specific deployment

After creating a deployment, the following command can be used to display currently configured simulation sessions:

`dsctl list deployment`

Example:

The following will display all available deployments. In this case, we see that there are two successful deployments:

`dsctl list deployment`

```
+-------+------+------------+
|  ID   | NAME |   STATUS   |
+-------+------+------------+
| 00008 | star | Successful |
--------|------|------------|
| 00009 | fire | Successful |
+-------+------+------------+
*: indicates some pods /
containers have restarted
```

The ID references in this output are internal numbers generated by the Digital Sandbox. Numbers increase with each new deployment.

4.3.2 Displaying details of a specific deployment

After creating a deployment, the following command can be used to display the details of a specific deployment:

`dsctl list deployment -i <name>`

Example:

The following will display the details of a deployment that was named 'star':

`dsctl list deployment -i star`

```
{
 "nodes": {
  "h1-leaf-1": {
   "deviceType": 1,
   "interfaces": [
    "mgmt0",
    "ethernet-1/1",
    ...,
    "ethernet-1/34"
   ],
   "uuid": "348764238212235264",
   "overrides": {
    "tag": "0.0.0-27910"
   },
   "hwDetails": {
    "chassis": "66",
    "cpm": "177",
    "card": "177",
    "mda": "194"
```

```
  }
 },
 "h1-leaf-2": {
  "deviceType": 1,
  "interfaces": [
   "mgmt0",
   "ethernet-1/1",
   ...,
   "ethernet-1/34"
  ],
  "uuid": "348764238212300800",
  "overrides": {
   "tag": "0.0.0-27910"
  },
  "hwDetails": {
   "chassis": "66",
   "cpm": "177",
   "card": "177",
   "mda": "194"
  }
 },
 "h1-spine-1": {
  "deviceType": 1,
  "interfaces": [
   "mgmt0",
   "ethernet-1/1",
   ...,
   "ethernet-4/36"
  ],
  "uuid": "348764239571189760",
  "overrides": {
   "tag": "0.0.0-27910"
  },
  "hwDetails": {
   "chassis": "42",
   "cpm": "69",
   "card": "127",
   "mda": "182"
  }
 },
 "h1-spine-2": {
  "deviceType": 1,
  "interfaces": [
   "mgmt0",
   "ethernet-1/1",
   ...,
   "ethernet-4/36"
  ],
  "uuid": "348764239587966976",
  "overrides": {
   "tag": "0.0.0-27910"
  },
  "hwDetails": {
   "chassis": "42",
   "cpm": "69",
   "card": "127",
   "mda": "182"
  }
 },
 "m1-leaf-1": {
  "deviceType": 1,
  "interfaces": [
   "mgmt0",
   "ethernet-1/1",
```

```
    ...,
    "ethernet-1/34"
   ],
   "uuid": "348764942804975616",
   "overrides": {
    "tag": "0.0.0-27910"
   },
   "hwDetails": {
    "chassis": "66",
    "cpm": "177",
    "card": "177",
    "mda": "194"
   }
  },
  "m1-leaf-2": {
   "deviceType": 1,
   "interfaces": [
    "mgmt0",
    "ethernet-1/1",
    ...,
    "ethernet-1/34"
   ],
   "uuid": "348764942805041152",
   "overrides": {
    "tag": "0.0.0-27910"
   },
   "hwDetails": {
    "chassis": "66",
    "cpm": "177",
    "card": "177",
    "mda": "194"
   }
  },
  "m1-spine-1": {
   "deviceType": 1,
   "interfaces": [
    "mgmt0",
    "ethernet-1/1",
    ...,
    "ethernet-4/36"
   ],
   "uuid": "348764943526395904",
   "overrides": {
    "tag": "0.0.0-27910"
   },
   "hwDetails": {
    "chassis": "42",
    "cpm": "69",
    "card": "127",
    "mda": "182"
   }
  },
  "m1-spine-2": {
   "deviceType": 1,
   "interfaces": [
    "mgmt0",
    "ethernet-1/1",
    ...,
    "ethernet-4/36"
   ],
   "uuid": "348764943526461440",
   "overrides": {
    "tag": "0.0.0-27910"
   },
```

```
   "hwDetails": {
    "chassis": "42",
    "cpm": "69",
    "card": "127",
    "mda": "182"
   }
  }
 },
 "links": [
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/27"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/1"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/28"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/1"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/30"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/2"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/29"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/2"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/32"
    },
    {
```

```
     "node": "m1-spine-2",
     "itf": "ethernet-1/3"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/31"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/3"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/34"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/4"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-1",
     "itf": "ethernet-1/33"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/4"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/27"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/5"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/28"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/5"
    }
   ]
```

```
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/29"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/6"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/30"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/6"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/32"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/7"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/31"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/7"
    }
   ]
  },
  {
   "link": [
    {
     "node": "m1-leaf-2",
     "itf": "ethernet-1/33"
    },
    {
     "node": "m1-spine-1",
     "itf": "ethernet-1/8"
    }
   ]
  },
  {
   "link": [
    {
```

```
     "node": "m1-leaf-2",
     "itf": "ethernet-1/34"
    },
    {
     "node": "m1-spine-2",
     "itf": "ethernet-1/8"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-1",
     "itf": "ethernet-1/28"
    },
    {
     "node": "h1-spine-2",
     "itf": "ethernet-1/1"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-1",
     "itf": "ethernet-1/27"
    },
    {
     "node": "h1-spine-1",
     "itf": "ethernet-1/1"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-1",
     "itf": "ethernet-1/30"
    },
    {
     "node": "h1-spine-2",
     "itf": "ethernet-1/2"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-1",
     "itf": "ethernet-1/29"
    },
    {
     "node": "h1-spine-1",
     "itf": "ethernet-1/2"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-1",
     "itf": "ethernet-1/32"
    },
    {
```

```
    "node": "h1-spine-2",
    "itf": "ethernet-1/3"
   }
  ]
 },
 {
  "link": [
   {
    "node": "h1-leaf-1",
    "itf": "ethernet-1/31"
   },
   {
    "node": "h1-spine-1",
    "itf": "ethernet-1/3"
   }
  ]
 },
 {
  "link": [
   {
    "node": "h1-leaf-1",
    "itf": "ethernet-1/34"
   },
   {
    "node": "h1-spine-2",
    "itf": "ethernet-1/4"
   }
  ]
 },
 {
  "link": [
   {
    "node": "h1-leaf-1",
    "itf": "ethernet-1/33"
   },
   {
    "node": "h1-spine-1",
    "itf": "ethernet-1/4"
   }
  ]
 },
 {
  "link": [
   {
    "node": "h1-leaf-2",
    "itf": "ethernet-1/28"
   },
   {
    "node": "h1-spine-2",
    "itf": "ethernet-1/5"
   }
  ]
 },
 {
  "link": [
   {
    "node": "h1-leaf-2",
    "itf": "ethernet-1/27"
   },
   {
    "node": "h1-spine-1",
    "itf": "ethernet-1/5"
   }
  ]
```

```
  },
  {
   "link": [
    {
     "node": "h1-leaf-2",
     "itf": "ethernet-1/30"
    },
    {
     "node": "h1-spine-2",
     "itf": "ethernet-1/6"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-2",
     "itf": "ethernet-1/29"
    },
    {
     "node": "h1-spine-1",
     "itf": "ethernet-1/6"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-2",
     "itf": "ethernet-1/32"
    },
    {
     "node": "h1-spine-2",
     "itf": "ethernet-1/7"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-2",
     "itf": "ethernet-1/31"
    },
    {
     "node": "h1-spine-1",
     "itf": "ethernet-1/7"
    }
   ]
  },
  {
   "link": [
    {
     "node": "h1-leaf-2",
     "itf": "ethernet-1/34"
    },
    {
     "node": "h1-spine-2",
     "itf": "ethernet-1/8"
    }
   ]
  },
  {
   "link": [
    {
```

```
     "node": "h1-leaf-2",
     "itf": "ethernet-1/33"
    },
    {
     "node": "h1-spine-1",
     "itf": "ethernet-1/8"
    }
   ]
  },
  {
   "isMgmtSegment": true,
   "link": [
    {
     "node": "h1-spine-2",
     "itf": "mgmt0"
    },
    {
     "node": "h1-spine-1",
     "itf": "mgmt0"
    },
    {
     "node": "h1-leaf-2",
     "itf": "mgmt0"
    },
    {
     "node": "m1-spine-1",
     "itf": "mgmt0"
    },
    {
     "node": "m1-spine-2",
     "itf": "mgmt0"
    },
    {
     "node": "m1-leaf-2",
     "itf": "mgmt0"
    },
    {
     "node": "m1-leaf-1",
     "itf": "mgmt0"
    },
    {
     "node": "h1-leaf-1",
     "itf": "mgmt0"
    }
   ],
   "dhcpItf": "dhcp-0",
   "mgmtIP": "10.243.184.0/21"
  }
 ]
}
```


4.3.3 Displaying nodes and links

After creating a deployment, the following commands can be used to display the nodes and links associated with a specific deployment:

`dsctl list nodes -i <name>`

Example - nodes:

The following will display the nodes and their status that are associated with a deployment named 'star':

dsctl list nodes -i star

```
+------------------+-------------+-------------+---------------+
|       NAME       | NODE STATUS |   MGMT-IP   | K8S-STATUS(*) |
+------------------+-------------+-------------+---------------+
| as--sim-1        | Running     |             | 0/3/3         |
| as--sim-2        | Running     |             | 0/3/3         |
| as--sim-3        | Running     |             | 0/3/3         |
| dut2             | Running     | 10.98.40.71 | 0/3/3         |
| dut3             | Running     | 10.98.40.72 | 0/3/3         |
| dut4             | Running     | 10.98.40.73 | 0/3/3         |
| dut5             | Running     | 10.98.40.67 | 0/3/3         |
| mgmt-dhcp-server | Running     |             | 0/1/1         |
+------------------+-------------+-------------+---------------+
```

Example - links:

The following will display the links and their status that are associated with a deployment named 'star':

dsclt list links -i star

```
+------+------------+---------------+-----------+------------+-------+
| UID  |    NODE    |   INTERFACE   |   STATE   | ERROR-CODE | ERROR |
+------+------------+---------------+-----------+------------+-------+
| 1009 | m1-leaf-1  | ethernet-1/27 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/1  |           |            |       |
|      |            |               |           |            |       |
| 1017 | m1-leaf-1  | ethernet-1/28 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/1  |           |            |       |
|      |            |               |           |            |       |
| 1018 | m1-leaf-1  | ethernet-1/30 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/2  |           |            |       |
|      |            |               |           |            |       |
| 1010 | m1-leaf-1  | ethernet-1/29 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/2  |           |            |       |
|      |            |               |           |            |       |
| 1019 | m1-leaf-1  | ethernet-1/32 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/3  |           |            |       |
|      |            |               |           |            |       |
| 1011 | m1-leaf-1  | ethernet-1/31 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/3  |           |            |       |
|      |            |               |           |            |       |
| 1020 | m1-leaf-1  | ethernet-1/34 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/4  |           |            |       |
|      |            |               |           |            |       |
| 1012 | m1-leaf-1  | ethernet-1/33 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/4  |           |            |       |
|      |            |               |           |            |       |
| 1013 | m1-leaf-2  | ethernet-1/27 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/5  |           |            |       |
|      |            |               |           |            |       |
| 1021 | m1-leaf-2  | ethernet-1/28 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/5  |           |            |       |
|      |            |               |           |            |       |
| 1014 | m1-leaf-2  | ethernet-1/29 | Installed |          0 |       |
|      | m1-spine-1 | ethernet-1/6  |           |            |       |
|      |            |               |           |            |       |
| 1022 | m1-leaf-2  | ethernet-1/30 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/6  |           |            |       |
|      |            |               |           |            |       |
| 1023 | m1-leaf-2  | ethernet-1/32 | Installed |          0 |       |
|      | m1-spine-2 | ethernet-1/7  |           |            |       |
|      |            |               |           |            |       |
```

```
| 1015 | m1-leaf-2  | ethernet-1/31 | Installed |          0 |        |
|      | m1-spine-1 | ethernet-1/7  |           |            |        |
|      |            |               |           |            |        |
| 1016 | m1-leaf-2  | ethernet-1/33 | Installed |          0 |        |
|      | m1-spine-1 | ethernet-1/8  |           |            |        |
|      |            |               |           |            |        |
| 1024 | m1-leaf-2  | ethernet-1/34 | Installed |          0 |        |
|      | m1-spine-2 | ethernet-1/8  |           |            |        |
|      |            |               |           |            |        |
| 1025 | h1-leaf-1  | ethernet-1/28 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/1  |           |            |        |
|      |            |               |           |            |        |
| 1001 | h1-leaf-1  | ethernet-1/27 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/1  |           |            |        |
|      |            |               |           |            |        |
| 1026 | h1-leaf-1  | ethernet-1/30 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/2  |           |            |        |
|      |            |               |           |            |        |
| 1002 | h1-leaf-1  | ethernet-1/29 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/2  |           |            |        |
|      |            |               |           |            |        |
| 1027 | h1-leaf-1  | ethernet-1/32 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/3  |           |            |        |
|      |            |               |           |            |        |
| 1003 | h1-leaf-1  | ethernet-1/31 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/3  |           |            |        |
|      |            |               |           |            |        |
| 1028 | h1-leaf-1  | ethernet-1/34 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/4  |           |            |        |
|      |            |               |           |            |        |
| 1004 | h1-leaf-1  | ethernet-1/33 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/4  |           |            |        |
|      |            |               |           |            |        |
| 1029 | h1-leaf-2  | ethernet-1/28 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/5  |           |            |        |
|      |            |               |           |            |        |
| 1005 | h1-leaf-2  | ethernet-1/27 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/5  |           |            |        |
|      |            |               |           |            |        |
| 1030 | h1-leaf-2  | ethernet-1/30 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/6  |           |            |        |
|      |            |               |           |            |        |
| 1006 | h1-leaf-2  | ethernet-1/29 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/6  |           |            |        |
|      |            |               |           |            |        |
| 1031 | h1-leaf-2  | ethernet-1/32 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/7  |           |            |        |
|      |            |               |           |            |        |
| 1007 | h1-leaf-2  | ethernet-1/31 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/7  |           |            |        |
|      |            |               |           |            |        |
| 1032 | h1-leaf-2  | ethernet-1/34 | Installed |          0 |        |
|      | h1-spine-2 | ethernet-1/8  |           |            |        |
|      |            |               |           |            |        |
| 1008 | h1-leaf-2  | ethernet-1/33 | Installed |          0 |        |
|      | h1-spine-1 | ethernet-1/8  |           |            |        |
|      |            |               |           |            |        |
| 1000 | h1-spine-2 | mgmt0         | Installed |          0 |        |
|      | h1-spine-1 | mgmt0         |           |            |        |
|      | h1-leaf-2  | mgmt0         |           |            |        |
|      | m1-spine-1 | mgmt0         |           |            |        |
|      | m1-spine-2 | mgmt0         |           |            |        |
|      | m1-leaf-2  | mgmt0         |           |            |        |
|      | m1-leaf-1  | mgmt0         |           |            |        |
```

```
|      | h1-leaf-1  | mgmt0         |           |            |       |
|      |            |               |           |            |       |
+------+------------+---------------+-----------+------------+-------+
```

4.3.4 Checking a deployments status

After creating a deployment, the following command can be used to display the current status of a deployment you are actively creating or updating:

`dsctl status deployment -i <name>`

Example:

The following will display the current status of a deployment named 'star':

`dsctl status deployment -i star`

```
+--------+----------+------------+------------+
| ISBUSY | ISFAILED | NODEERRORS | LINKERRORS |
+--------+----------+------------+------------+
| False  | False    |          0 |          0 |
+--------+----------+------------+------------+
Deployment is successful.
```

The following definitions apply to this output table:

- ISBUSY indicates if an update/create or delete is ongoing.
- ISFAILED indicates if the create/update/delete failed for some reason.
- NODEERRORS count indicates the number of node failures.
- LINKERRORS count indicates the number of known link failures.

4.3.5 Connecting to network elements

Once you have a working deployment, you may want to connect to SR Linux nodes to review status and run SR Linux commands. You can connect into network elements using the following command:

`dsctl connect {-n <name_string> | -e <expression_string>} -i <ID_string> [-z <IPaddress_string>] [-o <ssh_opt_string>]`

Example:

To connect into an SR Linux node for a simulation session named 'star', use the following:

`dsctl connect -i star -e dut2`

After entering this command, you will be prompted to start the SR Linux CLI:

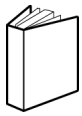
start cli: **`sr_cli`**

From here you can enter valid SR Linux commands.

```
Welcome to the srlinux CLI.
Type 'help' (and press <ENTER>) if you need any help using this.
--{ running }--[  ]--
dut2# show network-instance default protocols bgp neighbor
-------------------------------------------------------------------------------
BGP neighbor summary for network-instance "default"
```

```
Flags: S static, D dynamic, L discovered by LLDP, B BFD enabled, - disabled, * slow
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
+---------+-------+-------+-------+-------+---------+-------+-------+-------+
| Net-    | Peer  | Group | Flags | Peer- | State   | Uptim | AFI/S | [Rx/A |
| Inst    |       |       |       |  AS   |         |   e   |  AFI  | ctive |
|         |       |       |       |       |         |       |       | /Tx]  |
+=========+=======+=======+=======+=======+=========+=======+=======+=======+
| default | 192.1 | test  | S     | 65002 | esta-   | 2d:11 | ipv4- | [6/3/ |
|         | 68.11 |       |       |       | blished | h:54m | unica | 6]    |
|         | .2    |       |       |       |         | :43s  | st    | [6/3/ |
|         |       |       |       |       |         |       | ipv6- | 6]    |
|         |       |       |       |       |         |       | unica |       |
|         |       |       |       |       |         |       | st    |       |
| default | 192.1 | test  | S     | 65003 | esta-   | 2d:11 | ipv4- | [6/2/ |
|         | 68.12 |       |       |       | blished | h:55m | unica | 7]    |
|         | .2    |       |       |       |         | :13s  | st    | [6/2/ |
|         |       |       |       |       |         |       | ipv6- | 7]    |
|         |       |       |       |       |         |       | unica |       |
|         |       |       |       |       |         |       | st    |       |
| default | 2001: | test  | S     | 65002 | esta-   | 2d:11 | ipv4- | [0/0/ |
|         | 1::19 |       |       |       | blished | h:55m | unica | 0]    |
|         | 2:168 |       |       |       |         | :0s   | st    | [9/0/ |
|         | :11:2 |       |       |       |         |       | ipv6- | 9]    |
|         |       |       |       |       |         |       | unica |       |
|         |       |       |       |       |         |       | st    |       |
| default | 2001: | test  | S     | 65003 | esta-   | 2d:11 | ipv4- | [0/0/ |
|         | 1::19 |       |       |       | blished | h:55m | unica | 0]    |
|         | 2:168 |       |       |       |         | :20s  | st    | [9/0/ |
|         | :12:2 |       |       |       |         |       | ipv6- | 9]    |
|         |       |       |       |       |         |       | unica |       |
|         |       |       |       |       |         |       | st    |       |
+---------+-------+-------+-------+-------+---------+-------+-------+-------+
-------------------------------------------------------------------------------
Summary:
None configured neighbors, None configured sessions are established,None
disabled peers None dynamic peers
```

Customer document and product support



Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback