

Deployit Cloud Pack Manual

Version 3.9.2

Table of Contents

Table of Contents	2
Introduction	3
Overview	3
Concepts	3
Configuration	3
Creating a host template	3
Validating the host template descriptor	4
Creating an environment template	5
Validating the environment template descriptor	5
Security	5
Usage	6
Instantiating and destroying environment templates using the wizard	6
Instantiating and destroying host and environment templates via control tasks	7
Reporting cloud activity	7
Troubleshooting	8
CI Reference	8
Configuration Item Overview	8
Containers	8
Other Configuration Items	8
Configuration Item Details	8
cloud.BaseHostTemplate	8
cloud.CifsHost	9
cloud.CloudEnvironmentParameters	11
cloud.Environment	11
cloud.EnvironmentTemplate	12
cloud.HostParameters	12
cloud.SshHost	12

Introduction

This manual describes how to use Deployit's Cloud Pack functionality.

Note: functionality described in this manual is only available if you have the Cloud Pack installed.

Overview

Deployit's Cloud Pack provides functionality to create and destroy environments on virtualized infrastructure that can be used as deployment targets. Cloud Pack supports several different virtualization vendors. The Cloud Pack functionality is accessible from within the Deployit GUI and ties in to Deployit's security system and reporting.

Concepts

The Cloud Pack introduces two new concepts:

- **Host template:** a host template is a template for creating new, virtualized hosts in Deployit. A host template describes the characteristics of the host that will be created (using virtualization vendor specific configuration settings) and describes which middleware will be available to Deployit once a new host based on the template is created. A host template is specific to a particular virtualization vendor such as Amazon EC2 or VMWare.
- **Environment template:** an environment template is a collection of one or more host templates that are instantiated or destroyed together, allowing creating or removal of an entire environment in one go. An environment template describes which middleware will be part of the environment in Deployit once a new environment based on the template is created. An environment template is not tied to a virtualization provider and can contain host templates from various virtualization vendors.

Configuration

To configure the Cloud Pack, follow these steps:

1. Create a *host template* instance for each host to be created.
2. Validate the host template descriptor using the `validateDescriptor` control task.
3. Create an *environment template* instance to group together a set of host templates.
4. Validate the environment template descriptor using the `validateDescriptor` control task.

Creating a host template

Host templates are created in the Repository under the **Configuration** root node. Because host templates are virtualization platform specific, there will be different configuration options depending on the exact vendor. The following configuration options are common to all vendors:

Property	Category	Type	Description
Instance descriptor	Common	String	Freemarker and XML descriptor of the middleware that is present once the host template is instantiated. For more details, see below.
OS Family	Connection	Enumeration	The operating system family used on the template.
Connection type	Connection	Enumeration	The connection method to use when connecting to an instantiated host template.
Username	Connection	String	The user name to use when connecting to an instantiated host template.
Password	Connection	String	The password to use when connecting to an instantiated host template.
Private Key File	Connection	String	The private key file to use when connecting to an instantiated host template.
Marker file path	Connection	String	The marker file to scan to determine when the instantiated host is ready for use. If no marker file path is specified, the host template is assumed to be ready for use when a network connection to the host is available.
Boot timeout	Connection	Integer	The amount of seconds to allow for the host to boot before reporting failure.
Retry delay	Connection	Integer	The amount of seconds to wait between attempts to check for existence of the marker file.

The marker file

When instantiating a host template, the resulting host may already have the desired

middleware installed on it. If this is the case, the host will be ready for use as soon as it has finished booting. It is also possible to provision a host using Puppet, Chef or a shell command after launching it. This is supported via the the notion of *marker file*. If the host template specifies a marker file, Deployit will poll the launched instance for its presence. When the file is found on the instance filesystem, Deployit will conclude the host is up and ready for deployment.

The polling is done based on the connection settings of the Host template, not based on the connection settings that may appear in the host template descriptor itself. As a result some of the connection types will not work out of the box when using the marker file. If you extend the `cloud.BaseHostTemplate` (or a more specific template type) with additional properties, these will be copied over to the overthere connection used for polling if the property name coincides.

It may be a good approach to start with a connection type that works well out of the box like SFTP or SCP, and then change to a connection type that requires customization.

The host template descriptor

The host template descriptor is a Freemarkers template which is transformed into an XML-based definition of all CIs that are registered in Deployit after the host is launched on the virtualization platform. For more details regarding the XML format please check the [Deployit REST API documentation](#).

The Freemarkers template can make use of the variables described in the following table:

Variable	Type	Meaning
hostsPath	String	Folder which will contain the created middleware. This parameter is entered during creation of a host template / environment template instance.
hostTemplate	<code>ec2.HostTemplate</code>	Reference to this template. Allows reuse of some of its properties.
cloudId	String	Unique id of the created instance within the virtualization platform.
hostAddress	String	Public IP address of the created host.

Here is an example of a host descriptor from the **EC2 plugin**:

```
<#escape x as x?xml>
<list>
  <cloud.SshHost id="{hostsPath}/{hostTemplate.name}_{hostAddress}">
    <template ref="{hostTemplate.id}"/>
    <cloudId>{cloudId}</cloudId>
    <address>{hostAddress}</address>
    <#if hostTemplate.privateKeyFile??><privateKeyFile>{hostTemplate.privateKeyFile}</privateKeyFile></#if>
    <username>{hostTemplate.username}</username>
    <#if hostTemplate.password??><password>{hostTemplate.password}</password></#if>
    <os>{hostTemplate.os}</os>
    <connectionType>{hostTemplate.connectionType}</connectionType>
  </cloud.SshHost>
  <www.ApacheHttpdServer id="{hostsPath}/{hostTemplate.name}_{hostAddress}/httpd">
    <host ref="{hostsPath}/{hostTemplate.name}_{hostAddress}"/>
    <startCommand>sudo apachectl stop</startCommand>
    <startWaitTime>3</startWaitTime>
    <stopCommand>sudo apachectl stop</stopCommand>
    <stopWaitTime>3</stopWaitTime>
    <restartCommand>sudo apachectl restart</restartCommand>
    <restartWaitTime>10</restartWaitTime>
    <defaultDocumentRoot>/var/www/defaultDocumentRoot>
    <configurationFragmentDirectory>/etc/apache2/conf.d</configurationFragmentDirectory>
  </www.ApacheHttpdServer>
</list>
</#escape>
```

Please note that:

- You should always use `<list>` as a parent XML element, even if you define only one CI.
- If you do not propagate the Host template connection properties to the created CI, as is done in the above example, you may get unexpected behavior.
- Since XML is being generated you have to make sure that values are properly encoded. You can achieve this by enclosing the template in `<#escape x as x?xml>...</#escape>`, or alternatively use `{exampleKey?xml}`. See the [Freemarkers documentation](#) for details.

Validating the host template descriptor

Every host template CI provides a `validateDescriptor` control task which processes

the Freemarker template, parses the resulting XML and reports errors if something is wrong. No actual changes are made to the repository during execution of this control task.

Creating an environment template

The cloud plugin can combine several hosts into a `cloud.Environment` CI. This type of CI behaves similar to `udm.Environment` and can immediately be used for application deployment. Analogous to the single-host scenario, a special CI type `cloud.EnvironmentTemplate` is used to define which host to create and allows the user to fine-tune the set of members which will end up in the new environment.

Environment templates are created in the Repository under the **Configuration** root node. The following configuration options are available for environment templates:

Property	Type	Description
Environment descriptor	String	XML descriptor of the middleware that is present once the host template is instantiated. For more details, see below.
Host templates	List of CI	The list of host templates that together make up this environment template. Host templates will be instantiated in order when the environment template is instantiated.
Environment descriptor	String	Freemarker and XML descriptor of the middleware that is present once the host template is instantiated. For more details, see below.

The environment template descriptor

Analogous to the host template, the environment template also specifies a special property `xmlDescriptor` which describes the contents of the new environment.

The environment descriptor is a Freemarker template which is transformed into an XML-based definition of the environment that is registered in Deployit after the instances are launched on the virtualization platform. For more details regarding the XML format please check the [Deployit REST API documentation](#).

The Freemarker template can make use of the variables described in the following table:

Variable	Type	Meaning
<code>environmentId</code>	String	Id of the future environment. This parameter is entered during control task invocation.
<code>environmentTemplate</code>	<code>cloud.EnvironmentTemplate</code>	Reference to this template. Allows reuse of some of its properties.
<code>hosts</code>	<code>Set<overthere.Host></code>	Set of hosts that were created as a result of instantiating the environment template.

Here is an example of an environment descriptor:

```
<#escape x as x?xml>
<list>
  <cloud.Environment id="${environmentId}">
    <members>
      <#list hosts as h>
        <ci ref="{h.id}" />
        <#if h.name?starts_with("Apache")>
          <ci ref="{h.id}/httpd" />
        </#if>
      </#list>
    </members>
  </cloud.Environment>
</list>
</#escape>
```

Please note that:

- You should always use `<list>` as a parent XML element even if you define only one CI;
- `validateDescriptor` task processes the template with an empty set as a value of the `hosts` variable.
- It is possible to include other CIs in addition to the environment, such as dictionaries.
- You need to make sure to properly quote substituted values for XML output.

Validating the environment template descriptor

Every `cloud.EnvironmentTemplate` CI defines the `validateDescriptor` control task which processes the Freemarker template, parses the resulting XML and reports errors if something is wrong. No actual changes are made to the repository during this control task execution.

Security

In order to perform cloud operations, the following security permissions are important.

In order to create a cloud environment:

- You need read permission on the Environment Template CI
- you need `controltask#execute` permissions on the Environment Template CI
- you do **not** need to have read or edit permissions on the target Environment CI or Host Path for the cloud operation to be able to create it.

The cloud plugin operates as an administrator user concerning to repository acces and therefor the user that initiates the cloud operation does not need permissions on the target Environment or Host Path. The security for cloud is solely managed by the permissions to execute a `controltask`.

The cloud plugin will not automatically grant permissions on newly created environments and hosts. Instead, make sure that the target location for the new environment and hosts has the intended permission setup.

In order to destroy a cloud environment:

- You need read permission on the Environment CI
- you need `controltask#execute` on the Environment CI
- You do **not** need to have read or edit permissions on the Environment Template CI or the Host Path

Usage

After creating the needed configuration CIs (described above) you can instantiate either an *environment template* or a *host template*. There are two ways to do this in the Deployit GUI, either using the wizards on the Cloud tab or via control tasks in the Repository tab.

Instantiating and destroying environment templates using the wizard

The Cloud tab consist of two wizards, one for instantiating cloud environments, and one for destroying cloud environments. The wizard helps you to gather the required information to perform the create or destroy task.

Creating an environment

This wizard can help you to create a cloud environment. This wizard consists of four steps:

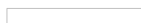
- **Select template** - First you need to select an Environment Template. The selection component shows all available instances of `cloud.EnvironmentTemplate` and `udm.Directory` available under the `Configuration` root node of the repository. When you select a template, it will display the description on the right side of the screen.



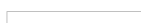
- **Fill in parameters** - The second step allows you to fill in additional parameters. The available parameters depends on the selected template. Common parameters that could be found here are environment id, which lets you specify the name of the environment that will be created (prefixed by Environments), and a second hosts parameters where you can specify where you want to store the created host CIs (prefixed by Infrastructure).



- **Run create task** - This screen shows the create task. You can review the steps. The cloud create task works exactly the same as the deployment task screen. If you want to continue with creating the environment, click execute. If you do not want to continue, click cancel. You can click previous to change the parameters for this task. This only works if the task has not been started yet. When the task is finished, you can click next to continue.



- **View result** - The view result page shows the created infrastructure, if any. Typically it will search for all CIs that are part of the created environment. Nodes can be expanded to see contained CIs.



To finish the create wizard you have two options:

- **Close** - This will close the wizard. The resulting infrastructure CIs have already been added to the repository.
- **Deploy** - This will start an initial deployment with the created environment selected. From this point you can continue the initial deployment as usual.

Destroying an environment

This wizard can help you destroying a cloud environment. This wizard consist of two steps:

- **Select environment** - Select the environment you want to destroy. The selection component will only show instances of the `cloud.Environment`, and `udm.Directory`. You can expand directories. When you have selected the Environment you want to destroy, click next.

- **Run destroy task** - This screen shows the destroy task. You can review the steps. If you want to continue with destroying the environment, click execute. If you do not want to continue, click cancel. You can also go back by clicking previous to select a different environment. When the destroy task is complete, you can close the task, and the wizard will restart.

Instantiating and destroying host and environment templates via control tasks

Both host and environment templates have `instantiate` and `destroy` control tasks that can be invoked directly from the Repository tab. The control tasks create and destroy either hosts or environments. The instantiation of host templates is intended for testing, as such these actions will not appear in the cloud reports.

Creating hosts and environments

The `instantiate` control task provides the following functionality:

CI type	Control task	Purpose
<code>cloud.BaseHostTemplate</code>	<code>instantiate</code>	Launches an instance on a virtualization platform using the parameters provided in the template. Virtualization vendor specific. Creates a host CI and children representing the middleware present in the repository.
<code>cloud.EnvironmentTemplate</code>	<code>instantiate</code>	Creates instances for each of the host templates specified in the HostTemplates property. Creates an environment CI with all instantiated cloud hosts attached to it in the repository. Members of this environment are customizable via the XML descriptor.

Destroying hosts and environments

The `destroy` control task provides the following functionality:

CI type	Control task	Purpose
Subtypes of <code>overthere.Host</code> that were created via the Cloud Pack	<code>destroy</code>	Destroys the virtual host associated with this CI. Removes this CI and all its children from the repository.
<code>cloud.Environment</code>	<code>destroy</code>	Destroys all hosts that were created as part of the environment. Removes all host CIs, all associated middleware CIs and the cloud.Environment CI itself from the repository.

Please note that:

Cloud environments manage all launched hosts that were part of the environment template on which they are based. If other containers are added to a cloud environment, these must be removed from the environment prior to destroying it. Similarly, cloud environment members must not be members of other environments when the cloud environment is destroyed.

Reporting cloud activity

The Cloud Pack includes reporting functionality that shows information about the instantiation and destruction of cloud environments. Enter a date range and press the 'Generate Report' button to generate the standard report. It is also possible to filter on the

cloud environment template used to instantiate the environment.

No filtering. By default, the report shows all operations in the date range in tabular format.

The report shows the following columns:

- **Template** - The template used to create the environment.
- **Environment** - The name of the created or destroyed environment.
- **Operation** - The operation executed: instantiate or destroy.
- **User** - The user that performed the operation.
- **Status** - The status of the operation.
- **Start Date** - The date on which the operation was started.
- **Completion Date** - The date on which the operation was completed.

Double-click on a row to show the instantiation/destruction steps and logs for the steps.

Filter by. The report allows you to filter on the cloud template used to create the environment.

Troubleshooting

Problem	Solution
Error during descriptor validation: Can not process descriptor template. Expression x is undefined on line 1, column XXX in template.	You are using an expression or variable, which is undefined in the Freemarker scope. If your template is surrounded by ``#escape`` tags, remove these and try again to pinpoint the error.
Error during descriptor validation: Can not process descriptor template. Expression XXX is undefined on line YYY, column ZZZ in template.	You are using an expression or variable, which is undefined in the Freemarker scope. Please check the manual of the appropriate plugin for the list of available variables and their types.
Error during descriptor validation: Can not parse generated XML. Encountered unknown ConfigurationItem property [XXX] for ci [YYY]	One of CIs that is defined in your XML descriptor has an unknown field. Please check the CI Reference section of the plugin defining the CI type you are using.
Error during Create Environment: You do not have the required permission to perform this task. Permission CONTROLTASK_EXECUTE on [Configuration/XXX] is not granted to you.	You do not have the permission controltask#execute set on the Environment Template CI.
Error during Destroy Environment: You do not have the required permission to perform this task. Permission CONTROLTASK_EXECUTE on [Environments/XXX] is not granted to you.	You do not have the permission controltask#execute set on the Environment CI.

CI Reference

Configuration Item Overview

Containers

CI	Description
cloud.CifsHost	Cloud host with CIFS access
cloud.SshHost	Cloud host with SSH access

Other Configuration Items

CI	Description
cloud.BaseHostTemplate	Base class for instance templates, all instance templates must extend it
cloud.CifsHost	Cloud host with CIFS access
cloud.CloudEnvironmentParameters	Parameters for cloud environment instantiation
cloud.Environment	Cloud environment
cloud.EnvironmentTemplate	Cloud environment template
cloud.HostParameters	Parameters for host templates instantiation
cloud.SshHost	Cloud host with SSH access

Configuration Item Details

cloud.BaseHostTemplate

Virtual Type

Interfaces udm.ConfigurationItem

Base class for instance templates, all instance templates must extend it

Public Properties			
* bootTimeout	: INTEGER = 500		Maximal amount of time (in seconds) allowed to elapse before the instance is ready.
* xmlDescriptor	: STRING		Freemarker template of XML which describes instance and middleware
connectionType	: ENUM [SFTP, SFTP_CYGWIN, SFTP_WINSSHD, SCP, SUDO, INTERACTIVE_SUDO, TUNNEL, TELNET, WINRM, WINRM_HTTP, WINRM_HTTPS]		Connection type to be used for connecting to the host
markerPath	: STRING		Path to the file which should appear on the instance when provisioning completes.
os	: ENUM [WINDOWS, UNIX]		OS family
password	: STRING		Password
privateKeyFile	: STRING		Private key file to use for authentication
retryDelay	: INTEGER = 5		Delay (in seconds) after each connection attempt.
username	: STRING		Username

Control task	Parameter CI	Attributes	Description
instantiate	cloud.HostParameters		Create instance from template
validateDescriptor			Validate XML descriptor

cloud.CifsHost

Type Hierarchy	overthere.CifsHost >> overthere.RemoteHost >> overthere.Host >> udm.BaseContainer >> udm.BaseConfigurationItem
Interfaces	udm.Taggable, udm.ConfigurationItem, udm.Container, overthere.HostContainer

Cloud host with CIFS access

Public Properties	
* address : STRING	Address of the host
* cloudId : STRING	Unique ID within cloud platform
* connectionType : ENUM [TELNET, WINRM, WINRM_HTTP, WINRM_HTTPS] = WINRM	Connection Type
* os : ENUM [WINDOWS, UNIX] = WINDOWS	Operating system the host runs
* password : STRING	Password to use for authentication
* template : CI<cloud.BaseHostTemplate >	Template which was used to create this host
* username : STRING	Username to connect with
cifsPort : INTEGER = 445	Port on which the CIFS server runs
jumpstation : CI<overthere.Jumpstation>	Jumpstation that should be used to reach this host
pathShareMappings : MAP_STRING_STRING	Mapping from Windows paths to Windows share names, e.g. C:\IBM\WebSphere -> WebSphereShare
port : INTEGER	Port on which the Telnet or WinRM server runs
tags : SET_OF_STRING	If set, only deployables with the same tag will be automatically mapped to this container.
temporaryDirectoryPath : STRING	Directory into which temporary files are stored. Will be cleaned up when the connection is closed.
winrmEnableHttps : BOOLEAN = false	Enable SSL communication to the WinRM server

Hidden Properties			
* connectionTimeoutMillis : INTEGER = 1200000	Number of milliseconds Overthere waits for a connection to a remote host to be established		
* protocol : STRING = cifs	Protocol to use when connecting to this host		
* tmpFileCreationRetries : INTEGER = 1000	Number of times Overthere attempts to create a temporary file with a unique name		
* winrmContext : STRING = /wsman	Context used by the WinRM server		
* winrmEnvelopSize : INTEGER = 153600	Envelop size for WinRM messages		
* winrmHttpsCertificateTrustStrategy : ENUM [STRICT, SELF_SIGNED, ALLOW_ALL] = STRICT	HTTPS certificate trust strategy for WinRM over HTTPS		
* winrmHttpsHostnameVerificationStrategy : ENUM [STRICT, BROWSER_COMPATIBLE, ALLOW_ALL] = STRICT	HTTPS host name verification strategy for WinRM over HTTPS		
* winrmLocale : STRING = en-US	Locale to use for WinRM messages		
* winrmTimeout : STRING = PT60.000S	Timeout to use for WinRM messages in XML schema duration format		
tmpDeleteOnDisconnect : BOOLEAN = true	Whether to delete the temporary connection directory when the connection is closed		
winrmKerberosAddPortToSpn : BOOLEAN = false	Add the port number (e.g. 5985) to the service principal name (SPN) for which a Kerberos ticket is requested		
winrmKerberosDebug : BOOLEAN = false	Enable Kerberos debug messages		
winrmKerberosUseHttpSpn : BOOLEAN = false	Use the HTTP protocol in the service principal name (SPN) for which a Kerberos ticket is requested, instead of the default WSMAN protocol		
Control task	Parameter CI	Attributes	Description
checkConnection			Checks whether Deployit can transfer files to and execute commands on this host.
destroy		delegate = destroyHost	Shut down EC2 instance and remove all related CIs

cloud.CloudEnvironmentParameters

Type Hierarchy udm.Parameters >> udm.BaseConfigurationItem
Interfaces udm.ConfigurationItem

Parameters for cloud environment instantiation

Public Properties	
* environmentId : STRING = Environments	Id of the environment you want to create
* hostsPath : STRING = Infrastructure	Repository location where all created hosts will appear

cloud.Environment

Type Hierarchy udm.Environment >> udm.BaseConfigurationItem
Interfaces udm.ConfigurationItem

Cloud environment

Public Properties			
* linkedCis	: SET_OF_CI<udm.ConfigurationItem>		CIs that were described in the template of this cloud environment and created along with it
* template	: CI<cloud.EnvironmentTemplate >		Template which was used to create this environment
dictionaries	: LIST_OF_CI<udm.Dictionary>		The dictionaries providing placeholder values. If the same entry exists in multiple dictionaries, the first one in the list is taken.
members	: SET_OF_CI<udm.Container>		The infrastructure components of this Environment
smtpServer	: CI<mail.SmtServer>		The SMTP server used to send mails with when deploying to this Environment.
Control task	Parameter CI	Attributes	Description
destroy			Shut down all related cloud instances and remove all related CIs

cloud.EnvironmentTemplate

Interfaces udm.ConfigurationItem

Cloud environment template

Public Properties			
* hostTemplates	: LIST_OF_CI<cloud.BaseHostTemplate >		Host templates
* xmlDescriptor	: STRING		Freemarker template of XML which describes environment
description	: STRING		Description of the template
Control task	Parameter CI	Attributes	Description
instantiate	cloud.CloudEnvironmentParameters		Instantiate environment and all hosts, which templates are linked to this environment template
validateEnvironmentDescriptor			Validate XML descriptor of the environment template

cloud.HostParameters

Type Hierarchy udm.Parameters >> udm.BaseConfigurationItem

Interfaces udm.ConfigurationItem

Parameters for host templates instantiation

Public Properties			
* hostsLocation	: STRING = Infrastructure		Repository location where all created hosts will appear
instanceName	: STRING		Name of the instance after creation

cloud.SshHost

Type Hierarchy overthere.SshHost >> overthere.RemoteHost >> overthere.Host >> udm.BaseContainer >> udm.BaseConfigurationItem

Interfaces udm.Taggable, udm.ConfigurationItem, udm.Container, overthere.HostContainer

Cloud host with SSH access

Public Properties	
* address : STRING	Address of the host
* cloudId : STRING	Unique ID within cloud platform
* connectionType : ENUM [SFTP, SFTP_CYGWIN, SFTP_WINSSHD, SCP, SUDO, INTERACTIVE_SUDO, TUNNEL] = SFTP	Type of SSH connection to create
* os : ENUM [WINDOWS, UNIX]	Operating system the host runs
* port : INTEGER = 22	Port on which the SSH server runs
* template : CI <cloud.BaseHostTemplate >	Template which was used to create this host
* username : STRING	Username to connect with
jumpstation : CI <overthere.Jumpstation>	Jumpstation that should be used to reach this host
passphrase : STRING	Optional passphrase for the private key in the private key file
password : STRING	Password to use for authentication
privateKeyFile : STRING	Private key file to use for authentication
sudoUsername : STRING	Username to sudo to when accessing files or executing commands
tags : SET_OF_STRING	If set, only deployables with the same tag will be automatically mapped to this container.
temporaryDirectoryPath : STRING	Directory into which temporary files are stored. Will be cleaned up when the connection is closed.

Hidden Properties				
* connectionTimeoutMillis	: INTEGER	= 1200000	Number of milliseconds Overthere waits for a connection to a remote host to be established	
* interactiveKeyboardAuthRegex	: STRING	= .*Password:[]?	Regular expression to look for in keyboard-interactive authentication before sending the password	
* protocol	: STRING	= ssh	Protocol to use when connecting to this host	
* sudoCommandPrefix	: STRING	= sudo -u {0}	Sudo command to prefix to the original command. The placeholder {0} is replaced with the sudoUsername	
* sudoPasswordPromptRegex	: STRING	= .*[Pp]assword.*:	Regular expression to look for in interactive sudo before sending the password	
* tmpFileCreationRetries	: INTEGER	= 1000	Number of times Overthere attempts to create a temporary file with a unique name	
allocateDefaultPty	: BOOLEAN	= false	If true, a default PTY (dummy:80:24:0:0) is allocated when executing a command	
allocatePty	: STRING		Specification for the PTY to be allocated when executing a command. The format is TERM:COLS:ROWS:WIDTH:HEIGHT, e.g. xterm:80:24:0:0	
sudoOverrideUmask	: BOOLEAN	= true	If true, permissions are explicitly changed with chmod -R go+rX after uploading a file or directory	
sudoPreserveAttributesOnCopyFromTempFile	: BOOLEAN	= true	If true, files are copied from the connection temporary directory using the -p flag to the cp command	
sudoPreserveAttributesOnCopyToTempFile	: BOOLEAN	= true	If true, files are copied to the connection temporary directory using the -p flag to the cp command	
sudoQuoteCommand	: BOOLEAN	= false	If true, the original command is quoted when it is prefixed with sudoCommandPrefix	
tmpDeleteOnDisconnect	: BOOLEAN	= true	Whether to delete the temporary connection directory when the connection is closed	
Control task	Parameter	CI	Attributes	Description
checkConnection				Checks whether Deployit can transfer files to and execute commands on this host.
destroy			delegate = destroyHost	Shut down EC2 instance and remove all related CIs