

Deployit EC2 Plugin Manual

Version 3.9.0

Table of Contents

Table of Contents	2
Preface	3
Overview	3
Features	3
Requirements	3
Usage scenarios	3
Creating a single host	3
Creating an environment	3
Provisioning instantiated hosts	3
Configuration instructions	4
EC2 credentials	4
Host template	4
Environment template	5
Using the EC2 plugin	5
CI Reference	5
Configuration Item Overview	5
Containers	5
Other Configuration Items	5
Configuration Item Details	5
cloud.BaseHostTemplate	5
cloud.CloudEnvironmentParameters	6
cloud.Environment	6
cloud.EnvironmentTemplate	7
cloud.HostParameters	7
ec2.CifsHost	7
ec2.Credentials	9
ec2.HostTemplate	9
ec2.SshHost	10

Preface

This document describes the functionality provided by the EC2 plugin.

See the **Deployit Reference Manual** for background information on Deployit and deployment concepts.

Overview

The EC2 plugin is a Deployit plugin that supports launching, provisioning and terminating hosts and environments on the EC2 platform.

The EC2 plugin is part of the Deployit **Cloud Pack**. For more information about the Cloud Pack, see the **Deployit Cloud Pack Manual**.

Features

- Launching and terminating EC2 instances from Deployit;
- Combining EC2 instances into environments which can be used for application deployment;
- Registering middleware on EC2 instances as part of a created environment.

Requirements

- **Deployit requirements**
 - **Deployit:** version 3.9+
 - **Other Deployit Plugins:** cloud-plugin
- **Infrastructural requirements**
 - **EC2 credentials:** AWS key and secret for accessing the EC2 platform via its API.

Usage scenarios

This section describes the most common usage scenarios, further sections contain more detailed configuration instructions.

Creating a single host

In its simplest form, the EC2 plugin can launch a single instance on EC2 and register it in Deployit as a CI of type `ec2.SshHost` (when connecting to the host using SSH) or `ec2.CifsHost` (when connecting using CIFS). The resulting host CI can contain middleware CIs that are present on the host and can be used as normal containers for deployment. The host can also be destroyed, which causes Deployit to terminate the EC2 instance and remove the host CI and its children from the repository.

There is a special CI type `ec2.HostTemplate` which is used as a template to define all information about the future host.

Creating an environment

The EC2 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.

Provisioning instantiated hosts

When launching an instance on EC2, the AMI it is based on may already have the desired middleware installed. If this is the case, a launched 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 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 location of the marker file can be configured in the `ec2.HostTemplate`.

See the section on the marker file in the Cloud Pack Manual for additional details on the polling process.

Note:

- it is the responsibility of the AMI to invoke the provisioning process and to create the marker file when provisioning is completed, signaling that the host and middleware is ready for deployment.
- marker files are only supported on Unix-based AMIs.

Configuration instructions

EC2 credentials

The EC2 plugin requires access to the AWS key and secret to access the EC2 API. These credentials are specified under the *Configuration* root node in the repository using a `ec2.Credentials` CI. The CI has a control task `validateCredentials` that can test that the credentials can be used to communicate with EC2.

Host template

The next step is to define host template CIs (`ec2.HostTemplate`). A host template describes a single host that can be launched on EC2. In addition to the generic host template properties, the EC2 host template allows some EC2-specific properties:

Property	Type	Description
EC2 Credentials	<code>ec2.Credentials</code>	The EC2 credentials to use when communicating with the AWS API.
Groups	Set of String	The EC2 security groups to associate with the instance.
Image ID	String	The Amazon Machine Image ID (AMI) to use when creating an instance.
Region	String	The EC2 region to create the instance in. Default value is <code>eu-west-1</code> .
Instance Type	Enumeration	The EC2 instance type to create.
Key Pair	String	The EC2 key pair to use.

See the Cloud Pack Manual for the generic properties.

Here is an example of an EC2 host descriptor:

```
<#escape x as x?xml>
<list>
  <ec2.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>
  </ec2.SshHost>
  <www.ApacheHttpdServer id="{hostsPath}/{hostTemplate.name}_{hostAddress}/http">
    <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>
```

Every `ec2.HostTemplate` 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.

Please note that:

- EC2 hosts which you define here should be either `ec2.SshHost` or `ec2.CifsHost`.
- The `ec2.SshHost` or `ec2.CifsHost` in the template must contain the XML fragments for `address`, `cloudId` and `template`. These are needed for the proper functioning of the plugin.
- 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 [Freemarker documentation](#) for details.

Environment template

Host templates can be collected in an environment template (`cloud.EnvironmentTemplate`).

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

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 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.
- Here as well, you need to pay attention to XML quoting.

Using the EC2 plugin

Please see the [Cloud Pack manual](#) for instructions on how to use the environment and host templates provided in the EC2 plugin.

CI Reference

Configuration Item Overview

Containers

CI	Description
ec2.CifsHost	EC2 instance with CIFS access
ec2.SshHost	EC2 instance with SSH access

Other Configuration Items

CI	Description
cloud.BaseHostTemplate	Base class for instance templates, all instance templates must extend it
cloud.CloudEnvironmentParameters	Parameters for cloud environment instantiation
cloud.Environment	Cloud environment
cloud.EnvironmentTemplate	Cloud environment template
cloud.HostParameters	Parameters for host templates instantiation
ec2.CifsHost	EC2 instance with CIFS access
ec2.Credentials	Amazon EC2 credentials
ec2.HostTemplate	Amazon EC2 instance template
ec2.SshHost	EC2 instance with SSH access

Configuration Item Details

cloud.BaseHostTemplate

Virtual Type
Type Hierarchy udm.BaseConfigurationItem
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

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**Type Hierarchy** udm.BaseConfigurationItem**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

ec2.CifsHost**Type Hierarchy** overthere.CifsHost >> overthere.RemoteHost >> overthere.Host >> udm.BaseContainer >> udm.BaseConfigurationItem**Interfaces** udm.Taggable, udm.ConfigurationItem, udm.Container, overthere.HostContainer

EC2 instance 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<ec2.HostTemplate >	Template which was used for this instance
* 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		
* 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 = destroyEC2Instance	Shut down EC2 instance and remove all related CIs

ec2.Credentials

Type Hierarchy udm.BaseConfigurationItem
Interfaces udm.ConfigurationItem

Amazon EC2 credentials

Public Properties			
* key : STRING	AWS key		
* secret : STRING	AWS secret		
Control task	Parameter CI	Attributes	Description
validateCredentials			Make a test call to EC2 with given credentials

ec2.HostTemplate

Type Hierarchy [cloud.BaseHostTemplate](#) >> udm.BaseConfigurationItem
Interfaces udm.ConfigurationItem

Amazon EC2 instance template

Public Properties			
* ami : STRING	Amazon Machine Image id		
* bootTimeout : INTEGER = 500	Maximal amount of time (in seconds) allowed to elapse before the instance is ready.		
* credentials : CI<ec2.Credentials >	EC2 credentials to be used for creating instances from this template		
* groups : SET_OF_STRING	Set of security groups to be applied to the instance		
* instanceType : ENUM [T1Micro, M1Small, M1Medium, M1Large, M1Xlarge, M2Xlarge, M22xlarge, M24xlarge, M3Xlarge, M32xlarge, C1Medium, C1Xlarge, Hi14xlarge, Cc14xlarge, Cc28xlarge, Cg14xlarge]	Instance type which defines how much resources are allocated		
* keyPair : STRING	Name of the key pair (should be already present on EC2)		
* region : STRING = eu-west-1	Amazon EC2 region to start instance in		
* 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

ec2.SshHost

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

EC2 instance 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 <ec2.HostTemplate >	Template which was used for this instance
* 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				
<div>* connectionTimeoutMillis : INTEGER = 1200000</div> <div>Number of milliseconds Overthere waits for a connection to a remote host to be established</div>				
<div>* interactiveKeyboardAuthRegex : STRING = .*Password:[]?</div> <div>Regular expression to look for in keyboard-interactive authentication before sending the password</div>				
<div>* protocol : STRING = ssh</div> <div>Protocol</div>				
<div>* sudoCommandPrefix : STRING = sudo -u {0}</div> <div>Sudo command to prefix to the original command. The placeholder {0} is replaced with the sudoUsername</div>				
<div>* sudoPasswordPromptRegex : STRING = .*[Pp]assword.*:</div> <div>Regular expression to look for in interactive sudo before sending the password</div>				
<div>* tmpFileCreationRetries : INTEGER = 1000</div> <div>Number of times Overthere attempts to create a temporary file with a unique name</div>				
<div>allocateDefaultPty : BOOLEAN = false</div> <div>If true, a default PTY (dummy:80:24:0:0) is allocated when executing a command</div>				
<div>allocatePty : STRING</div> <div>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</div>				
<div>sudoOverrideUmask : BOOLEAN = true</div> <div>If true, permissions are explicitly changed with chmod -R go+rX after uploading a file or directory</div>				
<div>sudoPreserveAttributesOnCopyFromTempFile : BOOLEAN = true</div> <div>If true, files are copied from the connection temporary directory using the -p flag to the cp command</div>				
<div>sudoPreserveAttributesOnCopyToTempFile : BOOLEAN = true</div> <div>If true, files are copied to the connection temporary directory using the -p flag to the cp command</div>				
<div>sudoQuoteCommand : BOOLEAN = false</div> <div>If true, the original command is quoted when it is prefixed with sudoCommandPrefix</div>				
<div>tmpDeleteOnDisconnect : BOOLEAN = true</div> <div>Whether to delete the temporary connection directory when the connection is closed</div>				
Control task	Parameter	CI	Attributes	Description
checkConnection				Checks whether Deployit can transfer files to and execute commands on this host.
destroy			delegate = destroyEC2Instance	Shut down EC2 instance and remove all related CIs