Universal Task Documentation

Universal Automation Center to upload a local Linux or Windows directory to an Azure Container

ut-azure-blobstorage-directory-to-container-upload-linux

Associated Activities:

Date: 06 August 2018

Author: Nils Buer

Revision: 01

CONFIDENTIALITY INFORMATION

Distribution list: AXA

Revision	Date	Author	Changes
00	20180806	Nils Buer	Initial Document (WIP)

Abstract:

The here described Universal Tasks uploads a local Windows or Linux directory to an Azure Blob Storage Container. As a result, you can integrate uploads of an entire local directory into you existing or new scheduling workflows, providing a true hybrid cloud (on-premise and cloud computer) file transfer solution.

Contents

1 Disclaimer

No support and no warranty are provided by Stonebranch GmbH for this document and the related Universal Task. The use of this document and the related Universal Task is on your own risk.

Before using this task in a production system, please perform extensive testing.

Stonebranch GmbH assumes no liability for damage caused by the performance of the Universal Tasks

2 Scope

This document provides a documentation how to install and use the Universal Tasks to upload a local directory to an Azure Blob Storage Container.

3 Introduction

Storing data in the cloud becomes an integral part of most modern IT landscapes. With Universal Automation Center you can securely automate your AWS, Azure or any other Cloud File Transfer and integrate them into your existing scheduling flows.

As security is one of the blob concerns, when moving to the cloud, the provided solution supports multi-level of security:

- All Credential for Azure Blob Storage are stored in an encrypted form in the database

Some details about the universal tasks to upload a local directory to an Azure Blob Storage Container:

- The Universal Task is calling the Python blobxfr module
- The python *blobxfr* module is called by a Universal Agent running on a Linux Server or Windows Server – Note: This document focuses on the Linux Version
- The Server Running the Universal Agent needs to have Python 2.7.x or 3.6.x installed
- All Credential for Azure are stored in an encrypted form in the database
- You can select different log-levels e.g. Info and debug
- A proxy connection towards Azure is currently not implemented for this Universal Task (is it however possible with minor adjustments)

4 Installation

4.1 Software Requirements

Universal Task name: *ut_azure_directory_upload_<xxx>_linux*

Related UAC XML Files for template and task: *Github repository*

Software used:

For the set-up you need:

1. Python 2.7.x (or 3.6.x) for Linux installed on a server where a Universal Agent is installed.
2. For Python the following modules are required:

   - ***Re****,* to support regular expression matching operations
   - ***glob****, to* find Unix pathnames matching a specified pattern
   - ***os****,* to support operating system dependent commands
   - ***sys***, for output re-direct processing
   - ***datetime, date*** and time stamps for messages
   - ***logging***, to provide logging capabilities for debug, info etc.
   - ***argparse***, to allow testing of the Universal TPL. script on the command line
   - ***azure-storage-blob***, The Azure libraries for Python to use Azure services and manage Azure resources
   - ***azure-storage-logging***, provide enhanced logging for Azure storage services
   - ***blobxfr***, provides support for uploading a directory to an Azure container

   Note: Only the module ***azure-storage-blob*** *and* ***azure-storage-logging*** *need to be added to python 3.6.x. e.g. using pip.*

   - *pip install azure-storage-blob*
   - *pip install azure-storage-logging*
   - *pip install blobxfr*

3. Universal Controller 6.4.5.x or higher
4. Universal Agent 6.4.2.2 or higher installed on a Linux Server
5. An Azure account to try it out

4.2 Installation Steps

The following describes the installation steps:

1. **Install Python 2.7.x or 3.6.x for Linux on the Universal Controller server or any Linux Server running a Universal Agent.**

   Official Download link: https://www.python.org/downloads/

   Note:

   In most cases python is already available on Linux. Check availability with: *python -V*

2. **Add the *azure-storage-blob* and *azure-storage-logging* modules to your python installation**

   In a command shell run as root or sudo:

   - *pip install azure-storage-blob*
   - *pip install azure-storage-logging*
   - *pip install blobxfr*

Pre-requisites

(for details refer to: http://blobxfer.readthedocs.io/en/latest/01-installation/)

blobxfer is a pure Python package, however, some dependencies require a C compiler and supporting libraries if there is no binary wheel for that dependency and your platform. Please follow the pre-requisites section first prior to invoking installation via pip.
blobxfer has dependencies which require a C compiler if your platform does not have pre-made binary wheels for these dependencies. Please follow the instructions below for your platform. You will need to run the following commands via sudo or as root.

Ubuntu

- for Python3 (recommended)
  apt-get update
  apt-get install -y build-essential libssl-dev libffi-dev python3-dev python3-pip

- for Python2
  apt-get update
  apt-get install -y build-essential libssl-dev libffi-dev python-dev python-pip

CentOS/RHEL

- for Python3 (recommended)
  yum install -y epel-release
  yum install -y python34 python34-devel gcc openssl-devel libffi-devel
  curl -fSsL https://bootstrap.pypa.io/get-pip.py | python3

- for Python2
  yum install -y gcc openssl-devel libffi-devel python-devel
  curl -fSsL https://bootstrap.pypa.io/get-pip.py | python

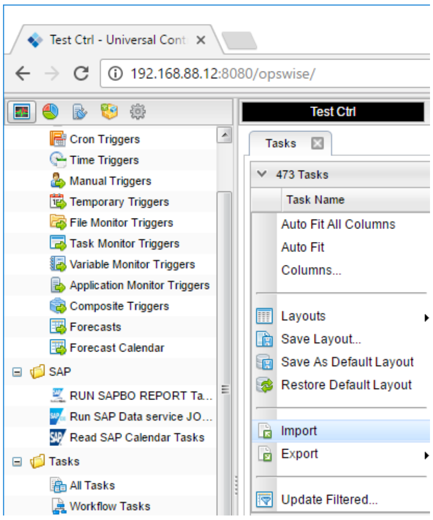
SLES/OpenSUSE

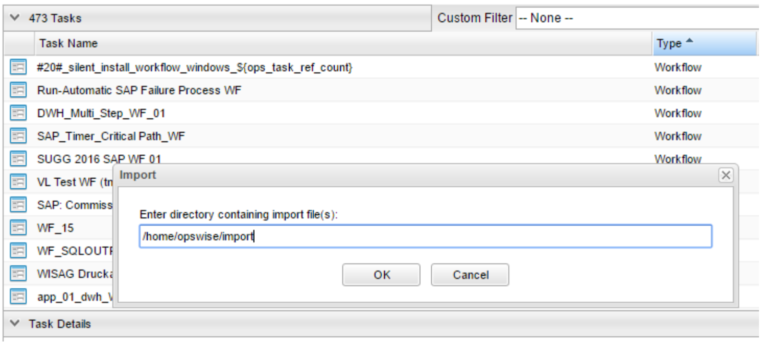
- for Python3 (recommended)
  ```
  zypper ref
  zypper -n in gcc libopenssl-devel libffi48-devel python3-devel
  curl -fSsL https://bootstrap.pypa.io/get-pip.py | python3
  ```

- for Python2
  ```
  zypper ref
  zypper -n in gcc libopenssl-devel libffi48-devel python-devel
  curl -fSsL https://bootstrap.pypa.io/get-pip.py | python
  ```

3. Import the Azure directory upload Universal Template into your Controller

Go to “All Tasks” and load via the Import functionality the Universal Task configuration into the Controller.



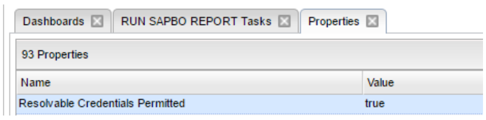


Task Name	Type
#20#_silent_install_workflow_windows_${ops_task_ref_count}	Workflow
Run-Automatic SAP Failure Process WF	Workflow
DWH_Multi_Step_WF_01	Workflow
SAP_Timer_Critical Path_WF	Workflow
SUGG 2016 SAP WF 01	Workflow
VL Test WF (tn	
SAP: Commiss	
WF_15	
WF_SQLOUTF	
WISAG Druck	
app_01_dwh_\	

Enter directory containing import file(s): /home/opswise/import

5 Universal Task Configuration

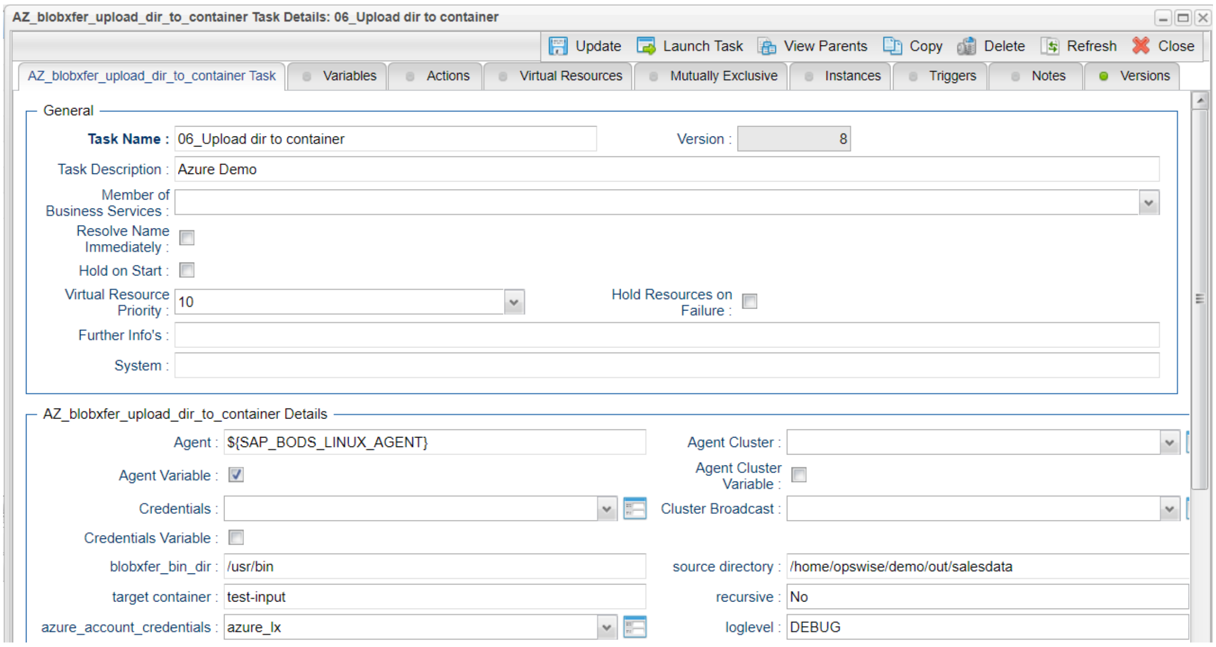
1. Activate: Resolvable Credentials in Universal Automation Center:



93 Properties

Name	Value
Resolvable Credentials Permitted	true

2. Fill Out the Universal Task:



AZ_blobxfer_upload_dir_to_container Task Details: 06_Upload dir to container

General

Task Name :	06_Upload dir to container	Version :	8
Task Description :	Azure Demo		
Member of Business Services :			
Resolve Name Immediately :	□		
Hold on Start :	□		
Virtual Resource Priority :	10	Hold Resources on Failure :	□
Further Info's :			
System :			

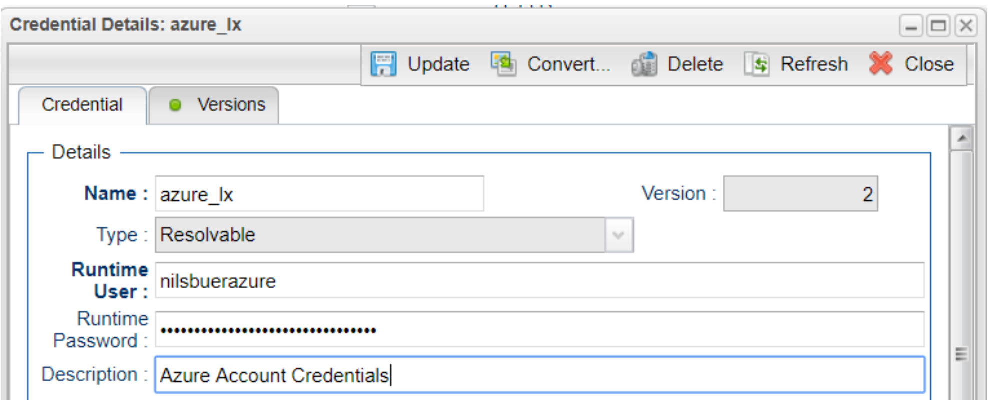
AZ_blobxfer_upload_dir_to_container Details

Agent :	${SAP_BODS_LINUX_AGENT}	Agent Cluster :	
Agent Variable :	☑	Agent Cluster Variable :	□
Credentials :		Cluster Broadcast :	
Credentials Variable :	□		
blobxfer_bin_dir :	/usr/bin	source directory :	/home/opswise/demo/out/salesdata
target container :	test-input	recursive :	No
azure_account_credentials :	azure_lx	loglevel :	DEBUG

In example above the local directory */home/opswise/demo/out/salesdata* is uploaded to the Azure container *test-input*.

Fill out or select the required Credentials for Azure and optionally a Proxy Server

In the example below the azure_account credentials are shown:



Credential Details: azure_lx

Details

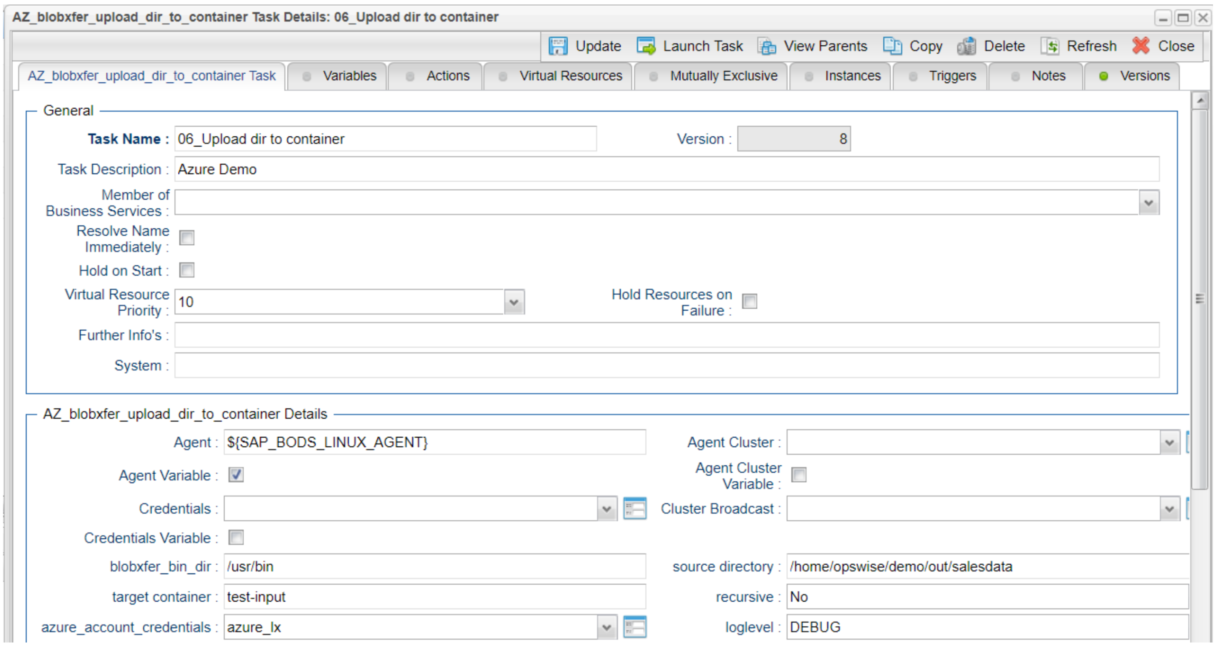
Name :	azure_lx	Version :	2
Type :	Resolvable		
Runtime User :	nilsbuerazure		
Runtime Password :	••••••••••••••••••••••••••••••••		
Description :	Azure Account Credentials		

6 Universal Tasks for Azure Blob Storage

The following chapter describes the provided Azure directory upload Universal Task.

Command	UT Name	Description
Upload directory to container	ut_azure_directory_upload_linux	uploads a local Windows or Linux directory to an Azure Blob Storage Container

Task Screenshot:



AZ_blobxfer_upload_dir_to_container Task Details: 06_Upload dir to container

General

Task Name : 06_Upload dir to container

Version : 8

Task Description : Azure Demo

Member of Business Services :

Resolve Name Immediately : □

Hold on Start : □

Virtual Resource Priority : 10

Hold Resources on Failure : □

Further Info's :

System :

AZ_blobxfer_upload_dir_to_container Details

Agent : ${SAP_BODS_LINUX_AGENT}

Agent Cluster :

Agent Variable : ☑

Agent Cluster Variable : □

Credentials :

Cluster Broadcast :

Credentials Variable : □

blobxfer_bin_dir : /usr/bin

source directory : /home/opswise/demo/out/salesdata

target container : test-input

recursive : No

azure_account_credentials : azure_lx

loglevel : DEBUG

Field Description:

Field	Required	Description
Agent	Mandatory	The Linux Universal Agent, which runs the Python azure-storage-blob module to call the AZURE BLOB STORAGE commands
Credentials	Optional	The Credentials used on the Linux Server
Loglevel	Mandatory	logging settings DEBUG, INFO, WARNING, ERROR, CRITICAL
target Container	Mandatory	Target Container, where the local directory is uploaded to
Blobxfr_bin_dir	Mandatory	Directory of the blobxfr module CLI
Azure_account_credentials	Mandatory	Azure Credentials: Account Name and Password
Source directory	Mandatory	Local directory to be uploaded to an Azure Container

Field	Required	Description
Loglevel		logging settings DEBUG, INFO, WARNING, ERROR, CRITICAL
recursive		Recursive upload of the local directory

7 Test Cases

The following basic test cases have been performed:

Case#	Assumed behavior	Result
Upload a local directory to a container – directory exists, container is empty (recursive flag not set)	**Directory without subdirectories is uploaded.** INFO - blobxfer start time: 2018-08-06 10:23:03.427927+02:00 INFO - elapsed upload + verify time and throughput of 0.0000 GiB: 0.858 sec, 0.0054 Mbps (0.001 MiB/s) INFO - blobxfer end time: 2018-08-06 10:23:05.006844+02:00 (elapsed: 1.579 sec)	Correct
Upload a local directory to a container – directory exists, container is not empty	**Existing Blobs in the container will be overridden.** INFO - blobxfer start time: 2018-08-06 10:27:05.060135+02:00 INFO - elapsed upload + verify time and throughput of 0.0000 GiB: 0.864 sec, 0.0054 Mbps (0.001 MiB/s) INFO - blobxfer end time: 2018-08-06 10:27:06.664567+02:00 (elapsed: 1.604 sec)	Correct
Upload a local directory to a container – directory exists, container is empty (recursive flag set)	**Directory including subdirectories is uploaded.** INFO - blobxfer start time: INFO - elapsed upload + verify time and throughput of 0.0000 GiB: 1.040 sec, 0.0046 Mbps (0.001 MiB/s) INFO - blobxfer end time: 2018-08-06 10:34:34.055169+02:00 (elapsed: 1.845 sec)	Correct

Upload a local directory to a container – directory does not exist	ERROR - Directory not found: /home/opswise/demo/out/salesdatas	Correct
Wrong Azure credentials	Task goes to success: ValueError: specified storage account key is invalid for storage account: opswise	Error handling needs to be added
Upload a local directory to a container – container does not exist.	**New Container will be created** INFO - created blob container test-inputs on storage account liitextanalytics INFO - elapsed upload + verify time and throughput of 0.0000 GiB: 1.102 sec, 0.0043 Mbps (0.001 MiB/s) INFO - blobxfer end time: 2018-08-06 10:39:46.829207+02:00 (elapsed: 1.830 sec)	Correct

8 Document References

There are no document references.