Phase 6: Observability & Logging - Script to Upload Jenkins Build Logs to Azure Blob

Overview

This solution provides automated backup of Jenkins build logs to Azure Blob Storage for long-term storage, analysis, and compliance. The script supports both manual execution and automated scheduling.

Features

- ✅ Jenkins API integration with authentication
- ✅ Azure Blob Storage upload with error handling
- ✅ Configurable log retention policies
- ✅ Batch processing for multiple builds
- ✅ Detailed logging and monitoring
- ✅ Support for different log formats (console, test results, artifacts)

Prerequisites

System Requirements

- Python 3.8 or higher
- Jenkins server (from Phase 4 setup)
- Azure subscription and storage account
- Network connectivity to both Jenkins and Azure

Jenkins Requirements

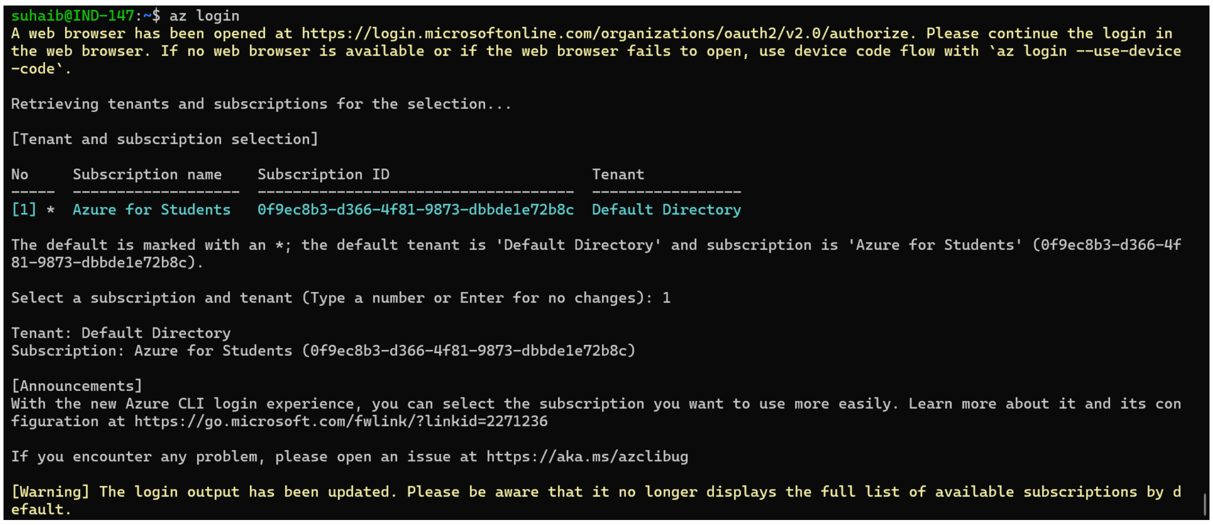
- Jenkins API enabled
- User with appropriate permissions

- Build jobs with logs to backup

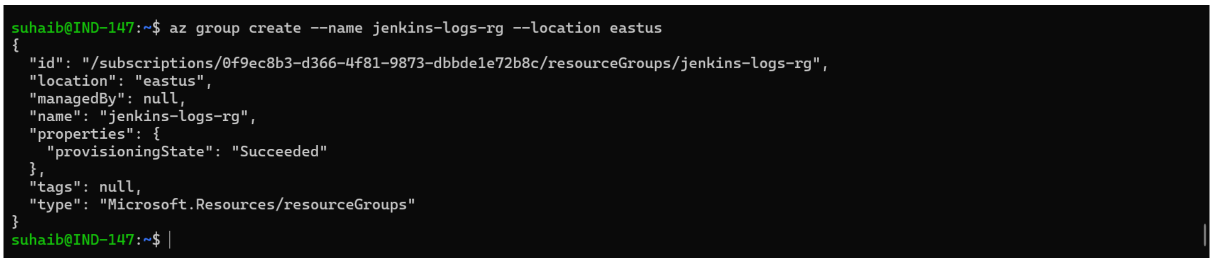
Azure Setup

1. Create Azure Storage Account

```
# Login to Azure
az login

# Create resource group (if not exists)
az group create --name jenkins-logs-rg --location eastus

# Create storage account
az storage account create \
   --name jenkinslogsstorage \
   --resource-group jenkins-logs-rg \
   --location eastus \
   --sku Standard_LRS \
   --kind StorageV2

# Create container for logs
az storage container create \
   --name jenkins-build-logs \
   --account-name jenkinslogsstorage \
   --public-access off
```

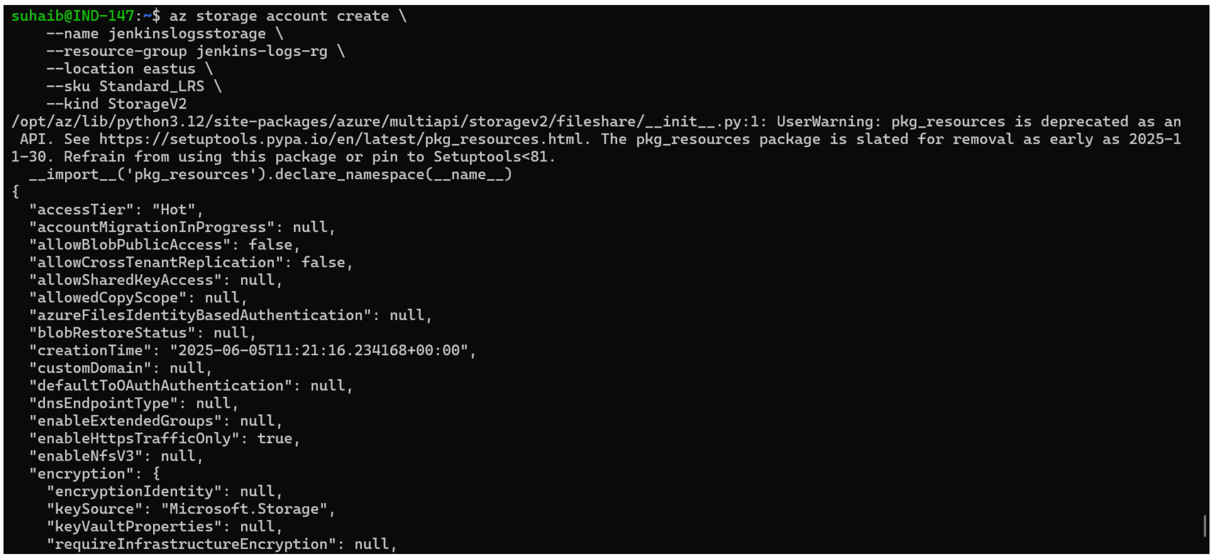


```
suhaib@IND-147:~$ az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue the login in
the web browser. If no web browser is available or if the web browser fails to open, use device code flow with `az login --use-device
-code`.

Retrieving tenants and subscriptions for the selection...

[Tenant and subscription selection]

No     Subscription name    Subscription ID                       Tenant
-----  -------------------  ------------------------------------  -----------------
[1] *  Azure for Students   0f9ec8b3-d366-4f81-9873-dbbde1e72b8c  Default Directory

The default is marked with an *; the default tenant is 'Default Directory' and subscription is 'Azure for Students' (0f9ec8b3-d366-4f
81-9873-dbbde1e72b8c).

Select a subscription and tenant (Type a number or Enter for no changes): 1

Tenant: Default Directory
Subscription: Azure for Students (0f9ec8b3-d366-4f81-9873-dbbde1e72b8c)

[Announcements]
With the new Azure CLI login experience, you can select the subscription you want to use more easily. Learn more about it and its con
figuration at https://go.microsoft.com/fwlink/?linkid=2271236

If you encounter any problem, please open an issue at https://aka.ms/azclibug

[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by d
efault.
```

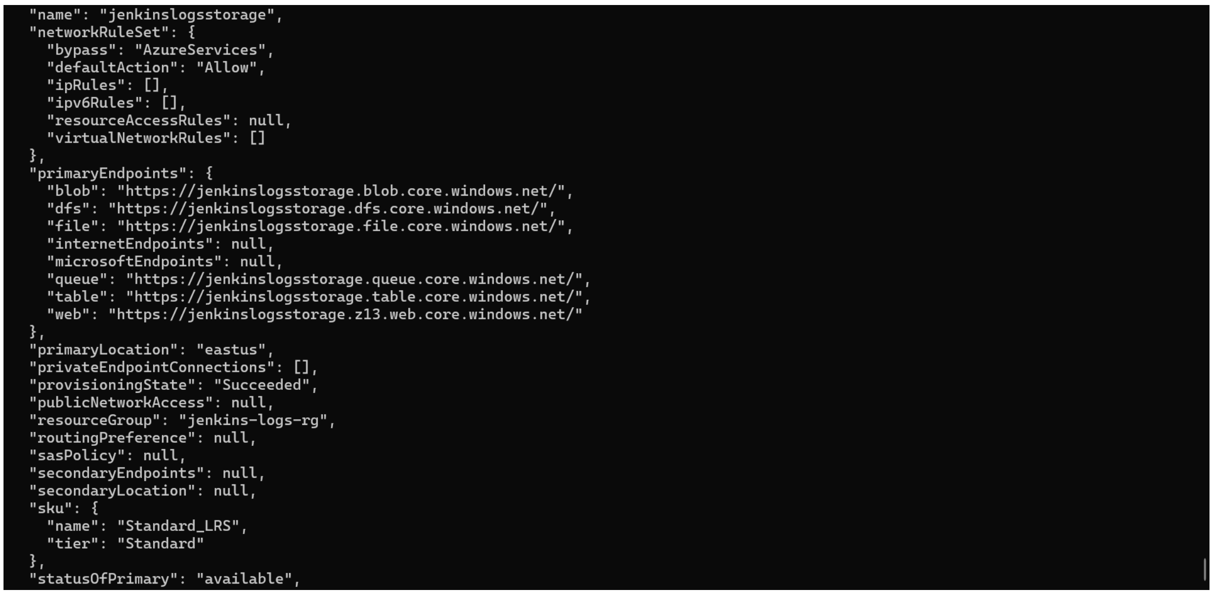


```
suhaib@IND-147:~$ az group create --name jenkins-logs-rg --location eastus
{
  "id": "/subscriptions/0f9ec8b3-d366-4f81-9873-dbbde1e72b8c/resourceGroups/jenkins-logs-rg",
  "location": "eastus",
  "managedBy": null,
  "name": "jenkins-logs-rg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
suhaib@IND-147:~$
```

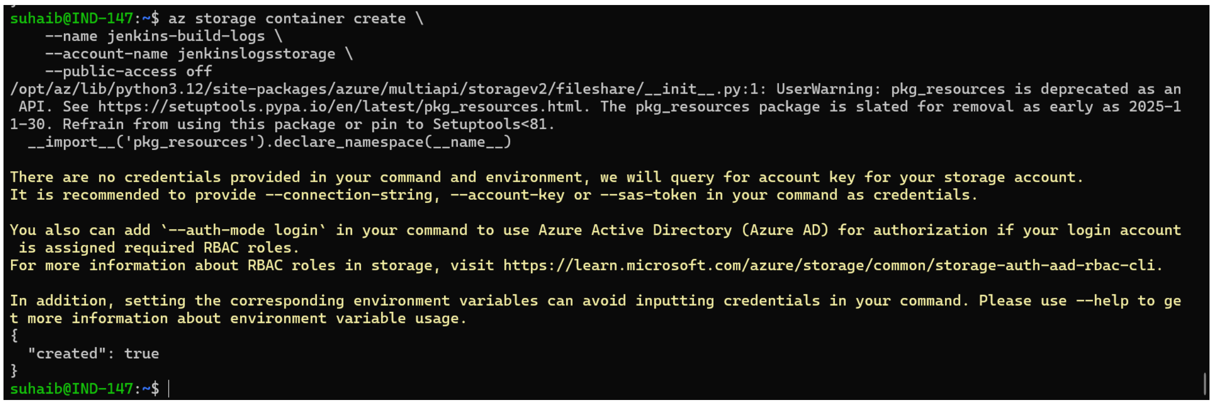
```
{
  "id": "/subscriptions/0f9ec8b3-d366-4f81-9873-dbbde1e72b8c/resourceGro
  "location": "eastus",
  "managedBy": null,
  "name": "jenkins-logs-rg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```



```
suhaib@IND-147:~$ az storage account create \
    --name jenkinslogsstorage \
    --resource-group jenkins-logs-rg \
    --location eastus \
    --sku Standard_LRS \
    --kind StorageV2
/opt/az/lib/python3.12/site-packages/azure/multiapi/storagev2/fileshare/__init__.py:1: UserWarning: pkg_resources is deprecated as an
 API. See https://setuptools.pypa.io/en/latest/pkg_resources.html. The pkg_resources package is slated for removal as early as 2025-1
1-30. Refrain from using this package or pin to Setuptools<81.
  __import__('pkg_resources').declare_namespace(__name__)
{
  "accessTier": "Hot",
  "accountMigrationInProgress": null,
  "allowBlobPublicAccess": false,
  "allowCrossTenantReplication": false,
  "allowSharedKeyAccess": null,
  "allowedCopyScope": null,
  "azureFilesIdentityBasedAuthentication": null,
  "blobRestoreStatus": null,
  "creationTime": "2025-06-05T11:21:16.234168+00:00",
  "customDomain": null,
  "defaultToOAuthAuthentication": null,
  "dnsEndpointType": null,
  "enableExtendedGroups": null,
  "enableHttpsTrafficOnly": true,
  "enableNfsV3": null,
  "encryption": {
    "encryptionIdentity": null,
    "keySource": "Microsoft.Storage",
    "keyVaultProperties": null,
    "requireInfrastructureEncryption": null,
```



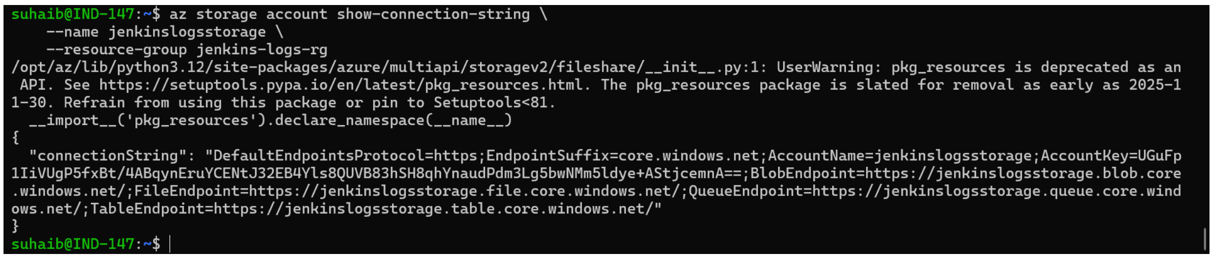
```
  "name": "jenkinslogsstorage",
  "networkRuleSet": {
    "bypass": "AzureServices",
    "defaultAction": "Allow",
    "ipRules": [],
    "ipv6Rules": [],
    "resourceAccessRules": null,
    "virtualNetworkRules": []
  },
  "primaryEndpoints": {
    "blob": "https://jenkinslogsstorage.blob.core.windows.net/",
    "dfs": "https://jenkinslogsstorage.dfs.core.windows.net/",
    "file": "https://jenkinslogsstorage.file.core.windows.net/",
    "internetEndpoints": null,
    "microsoftEndpoints": null,
    "queue": "https://jenkinslogsstorage.queue.core.windows.net/",
    "table": "https://jenkinslogsstorage.table.core.windows.net/",
    "web": "https://jenkinslogsstorage.z13.web.core.windows.net/"
  },
  "primaryLocation": "eastus",
  "privateEndpointConnections": [],
  "provisioningState": "Succeeded",
  "publicNetworkAccess": null,
  "resourceGroup": "jenkins-logs-rg",
  "routingPreference": null,
  "sasPolicy": null,
  "secondaryEndpoints": null,
  "secondaryLocation": null,
  "sku": {
    "name": "Standard_LRS",
    "tier": "Standard"
  },
  "statusOfPrimary": "available",
```



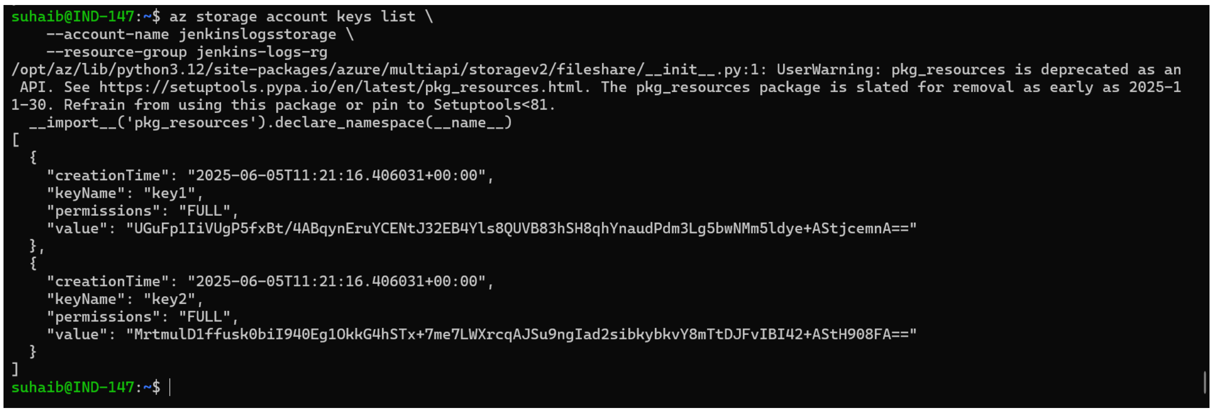
```
suhaib@IND-147:~$ az storage container create \
    --name jenkins-build-logs \
    --account-name jenkinslogsstorage \
    --public-access off
/opt/az/lib/python3.12/site-packages/azure/multiapi/storagev2/fileshare/__init__.py:1: UserWarning: pkg_resources is deprecated as an
 API. See https://setuptools.pypa.io/en/latest/pkg_resources.html. The pkg_resources package is slated for removal as early as 2025-1
1-30. Refrain from using this package or pin to Setuptools<81.
  __import__('pkg_resources').declare_namespace(__name__)

There are no credentials provided in your command and environment, we will query for account key for your storage account.
It is recommended to provide --connection-string, --account-key or --sas-token in your command as credentials.

You also can add `--auth-mode login` in your command to use Azure Active Directory (Azure AD) for authorization if your login account
 is assigned required RBAC roles.
For more information about RBAC roles in storage, visit https://learn.microsoft.com/azure/storage/common/storage-auth-aad-rbac-cli.

In addition, setting the corresponding environment variables can avoid inputting credentials in your command. Please use --help to ge
t more information about environment variable usage.
{
  "created": true
}
suhaib@IND-147:~$
```

2. Get Azure Storage Credentials

```
# Get storage account connection string
az storage account show-connection-string \
    --name jenkinslogsstorage \
    --resource-group jenkins-logs-rg

# Get storage account key
az storage account keys list \
    --account-name jenkinslogsstorage \
    --resource-group jenkins-logs-rg
```



```
suhaib@IND-147:~$ az storage account show-connection-string \
    --name jenkinslogsstorage \
    --resource-group jenkins-logs-rg
/opt/az/lib/python3.12/site-packages/azure/multiapi/storagev2/fileshare/__init__.py:1: UserWarning: pkg_resources is deprecated as an
 API. See https://setuptools.pypa.io/en/latest/pkg_resources.html. The pkg_resources package is slated for removal as early as 2025-1
1-30. Refrain from using this package or pin to Setuptools<81.
  __import__('pkg_resources').declare_namespace(__name__)
{
  "connectionString": "DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=jenkinslogsstorage;AccountKey=UGuFp
1IiVUgP5fxBt/4ABqynEruYCENtJ32EB4Yls8QUVB83hSH8qhYnaudPdm3Lg5bwNMm5ldye+AStjcemnA==;BlobEndpoint=https://jenkinslogsstorage.blob.core
.windows.net/;FileEndpoint=https://jenkinslogsstorage.file.core.windows.net/;QueueEndpoint=https://jenkinslogsstorage.queue.core.wind
ows.net/;TableEndpoint=https://jenkinslogsstorage.table.core.windows.net/"
}
suhaib@IND-147:~$ |
```

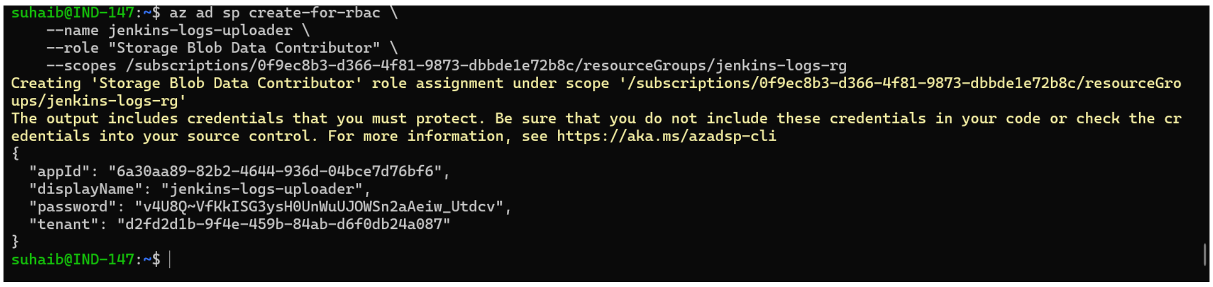


```
suhaib@IND-147:~$ az storage account keys list \
    --account-name jenkinslogsstorage \
    --resource-group jenkins-logs-rg
/opt/az/lib/python3.12/site-packages/azure/multiapi/storagev2/fileshare/__init__.py:1: UserWarning: pkg_resources is deprecated as an
 API. See https://setuptools.pypa.io/en/latest/pkg_resources.html. The pkg_resources package is slated for removal as early as 2025-1
1-30. Refrain from using this package or pin to Setuptools<81.
  __import__('pkg_resources').declare_namespace(__name__)
[
  {
    "creationTime": "2025-06-05T11:21:16.406031+00:00",
    "keyName": "key1",
    "permissions": "FULL",
    "value": "UGuFp1IiVUgP5fxBt/4ABqynEruYCENtJ32EB4Yls8QUVB83hSH8qhYnaudPdm3Lg5bwNMm5ldye+AStjcemnA=="
  },
  {
    "creationTime": "2025-06-05T11:21:16.406031+00:00",
    "keyName": "key2",
    "permissions": "FULL",
    "value": "MrtmulD1ffusk0biI940Eg1OkkG4hSTx+7me7LWXrcqAJSu9ngIad2sibkybkvY8mTtDJFvIBI42+AStH908FA=="
  }
]
suhaib@IND-147:~$ |
```

```
{
  "connectionString": "DefaultEndpointsProtocol=https;EndpointSuffix=core.w
}

[
  {
    "creationTime": "2025-06-05T11:21:16.406031+00:00",
    "keyName": "key1",
    "permissions": "FULL",
    "value": "UGuFp1IiVUgP5fxBt/4ABqynEruYCENtJ32EB4Yls8QUVB83hSH8q
  },
  {
    "creationTime": "2025-06-05T11:21:16.406031+00:00",
    "keyName": "key2",
    "permissions": "FULL",
    "value": "MrtmulD1ffusk0biI940Eg1OkkG4hSTx+7me7LWXrcqAJSu9ngIad2
  }
]
```

3. Create Service Principal (Recommended for Production)

```
# Create service principal
az ad sp create-for-rbac \
    --name jenkins-logs-uploader \
    --role "Storage Blob Data Contributor" \
    --scopes /subscriptions/{subscription-id}/resourceGroups/jenkins-logs-
rg

# Note down the output: appId, password, tenant
```



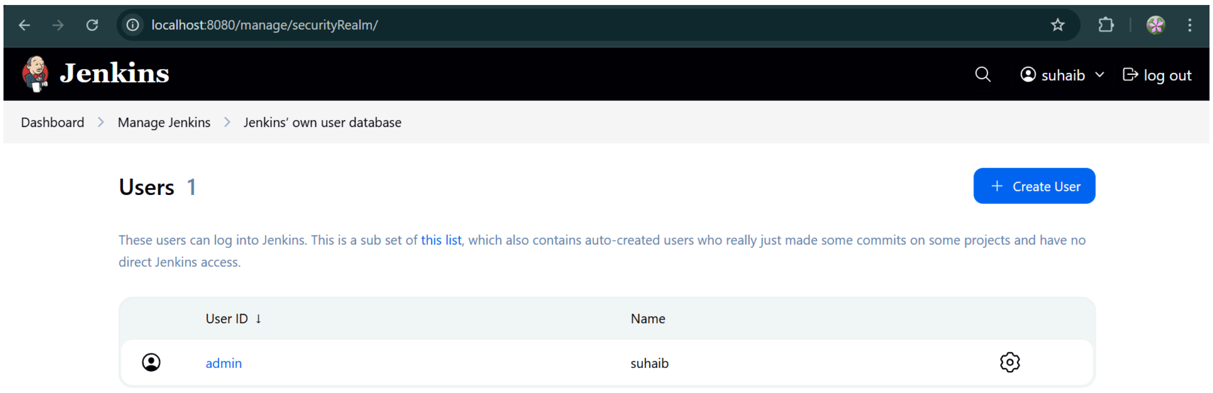
```
suhaib@IND-147:~$ az ad sp create-for-rbac \
    --name jenkins-logs-uploader \
    --role "Storage Blob Data Contributor" \
    --scopes /subscriptions/0f9ec8b3-d366-4f81-9873-dbbde1e72b8c/resourceGroups/jenkins-logs-rg
Creating 'Storage Blob Data Contributor' role assignment under scope '/subscriptions/0f9ec8b3-d366-4f81-9873-dbbde1e72b8c/resourceGro
ups/jenkins-logs-rg'
The output includes credentials that you must protect. Be sure that you do not include these credentials in your code or check the cr
edentials into your source control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "6a30aa89-82b2-4644-936d-04bce7d76bf6",
  "displayName": "jenkins-logs-uploader",
  "password": "v4U8Q~VfKkISG3ysH0UnWuUJOWSn2aAeiw_Utdcv",
  "tenant": "d2fd2d1b-9f4e-459b-84ab-d6f0db24a087"
}
suhaib@IND-147:~$
```

```
{
  "appId": "6a30aa89-82b2-4644-936d-04bce7d76bf6",
  "displayName": "jenkins-logs-uploader",
  "password": "v4U8Q~VfKkISG3ysH0UnWuUJOWSn2aAeiw_Utdcv",
  "tenant": "d2fd2d1b-9f4e-459b-84ab-d6f0db24a087"
}
```

Jenkins API Configuration

1. Create Jenkins API Token

1. Login to Jenkins web interface
2. Go to `Manage Jenkins` → `Manage Users` → Click your username


localhost:8080/manage/securityRealm/

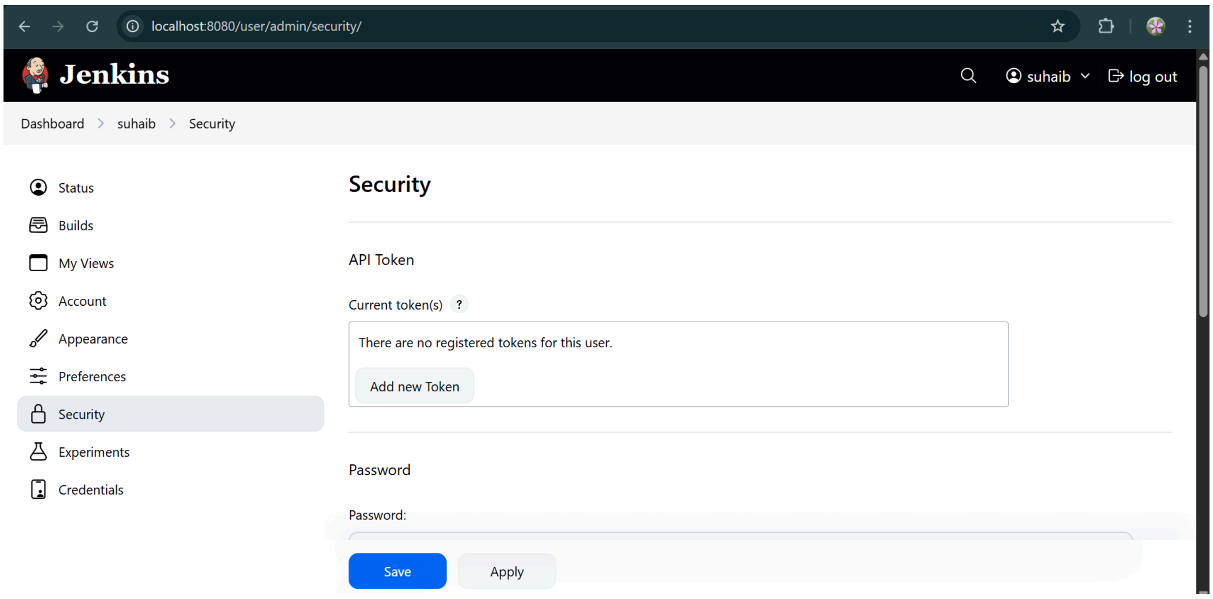
Dashboard > Manage Jenkins > Jenkins' own user database

Users 1

These users can log into Jenkins. This is a sub set of this list, which also contains auto-created users who really just made some commits on some projects and have no direct Jenkins access.

User ID ↓	Name
admin	suhaib

3. Click Security → API Token → Add new Token
4. Copy the generated token


localhost:8080/user/admin/security/

Dashboard > suhaib > Security

Security

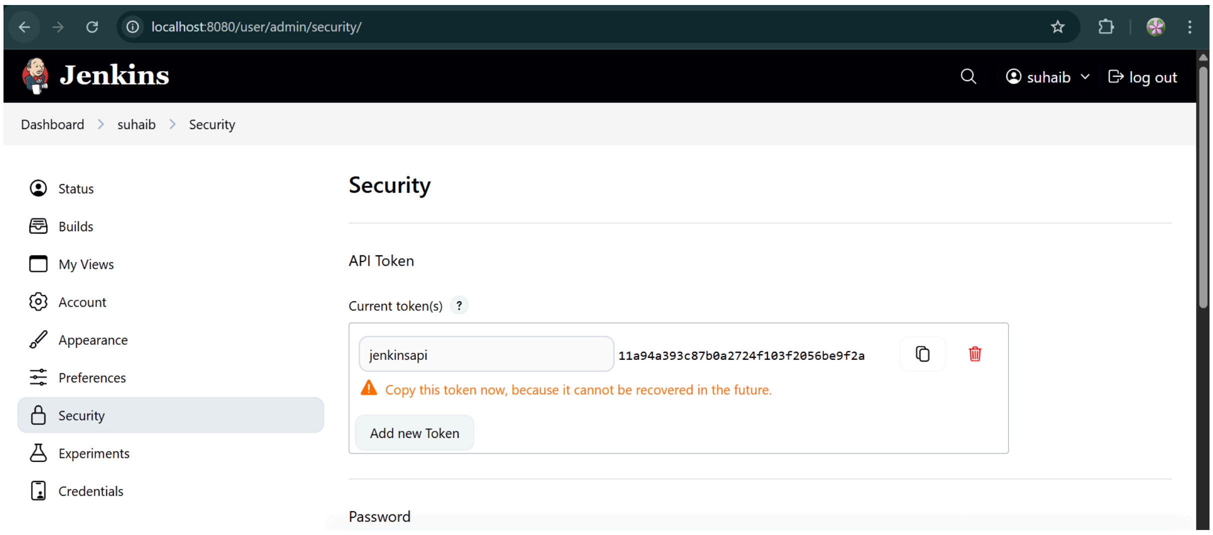
API Token

Current token(s)

There are no registered tokens for this user.

Password

Password:


localhost:8080/user/admin/security/

Dashboard > suhaib > Security

Security

API Token

Current token(s)

jenkinsapi 11a94a393c87b0a2724f103f2056be9f2a

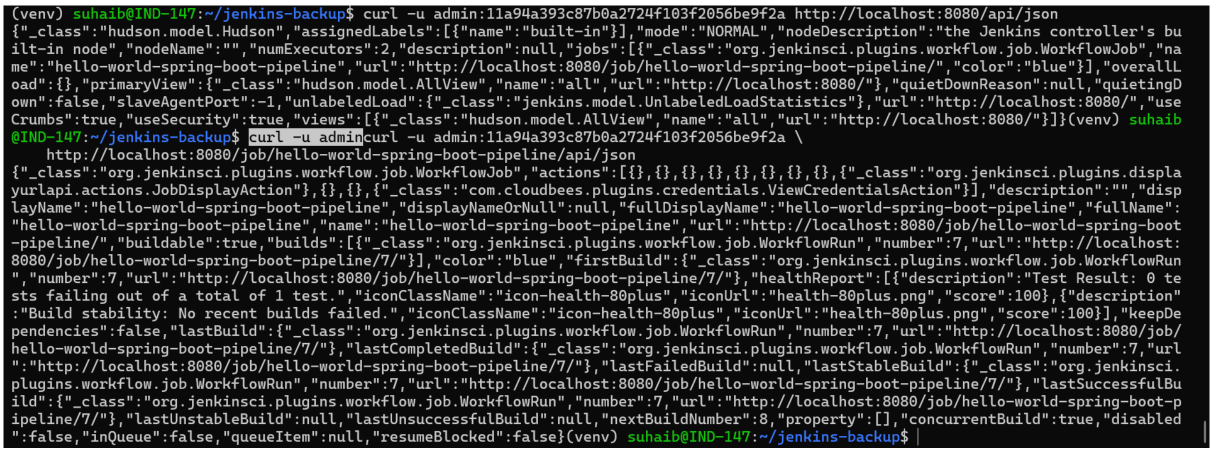
Copy this token now, because it cannot be recovered in the future.

Password

11a94a393c87b0a2724f103f2056be9f2a

2. Test Jenkins API Access

```
# Test Jenkins API connection
curl -u admin:11a94a393c87b0a2724f103f2056be9f2a http://localhost:808
0/api/json

# Test specific job access
curl -u admin:11a94a393c87b0a2724f103f2056be9f2a \
   http://localhost:8080/job/hello-world-spring-boot-pipeline/api/json
```



```
(venv) suhaib@IND-147:~/jenkins-backup$ curl -u admin:11a94a393c87b0a2724f103f2056be9f2a http://localhost:8080/api/json
{"_class":"hudson.model.Hudson","assignedLabels":[{"name":"built-in"}],"mode":"NORMAL","nodeDescription":"the Jenkins controller's bu
ilt-in node","nodeName":"","numExecutors":2,"description":null,"jobs":[{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowJob","na
me":"hello-world-spring-boot-pipeline","url":"http://localhost:8080/job/hello-world-spring-boot-pipeline/","color":"blue"}],"overallL
oad":{},"primaryView":{"_class":"hudson.model.AllView","name":"all","url":"http://localhost:8080/"},"quietDownReason":null,"quietingD
own":false,"slaveAgentPort":-1,"unlabeledLoad":{"_class":"jenkins.model.UnlabeledLoadStatistics"},"url":"http://localhost:8080/","use
Crumbs":true,"useSecurity":true,"views":[{"_class":"hudson.model.AllView","name":"all","url":"http://localhost:8080/"}]}(venv) suhaib
@IND-147:~/jenkins-backup$ curl -u admincurl -u admin:11a94a393c87b0a2724f103f2056be9f2a \
    http://localhost:8080/job/hello-world-spring-boot-pipeline/api/json
{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowJob","actions":[{},{},{},{},{},{},{},{},{"_class":"org.jenkinsci.plugins.displa
yurlapi.actions.JobDisplayAction"},{},{},{"_class":"com.cloudbees.plugins.credentials.ViewCredentialsAction"}],"description":"","disp
layName":"hello-world-spring-boot-pipeline","displayNameOrNull":null,"fullDisplayName":"hello-world-spring-boot-pipeline","fullName":
"hello-world-spring-boot-pipeline","name":"hello-world-spring-boot-pipeline","url":"http://localhost:8080/job/hello-world-spring-boot
-pipeline/","buildable":true,"builds":[{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowRun","number":7,"url":"http://localhost:
8080/job/hello-world-spring-boot-pipeline/7/"}],"color":"blue","firstBuild":{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowRun
","number":7,"url":"http://localhost:8080/job/hello-world-spring-boot-pipeline/7/"},"healthReport":[{"description":"Test Result: 0 te
sts failing out of a total of 1 test.","iconClassName":"icon-health-80plus","iconUrl":"health-80plus.png","score":100},{"description"
:"Build stability: No recent builds failed.","iconClassName":"icon-health-80plus","iconUrl":"health-80plus.png","score":100}],"keepDe
pendencies":false,"lastBuild":{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowRun","number":7,"url":"http://localhost:8080/job/
hello-world-spring-boot-pipeline/7/"},"lastCompletedBuild":{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowRun","number":7,"url
":"http://localhost:8080/job/hello-world-spring-boot-pipeline/7/"},"lastFailedBuild":null,"lastStableBuild":{"_class":"org.jenkinsci.
plugins.workflow.job.WorkflowRun","number":7,"url":"http://localhost:8080/job/hello-world-spring-boot-pipeline/7/"},"lastSuccessfulBu
ild":{"_class":"org.jenkinsci.plugins.workflow.job.WorkflowRun","number":7,"url":"http://localhost:8080/job/hello-world-spring-boot-p
ipeline/7/"},"lastUnstableBuild":null,"lastUnsuccessfulBuild":null,"nextBuildNumber":8,"property":[],"concurrentBuild":true,"disabled
":false,"inQueue":false,"queueItem":null,"resumeBlocked":false}(venv) suhaib@IND-147:~/jenkins-backup$
```

Quick Setup

1. Clone/Download Files

Create a new directory and save these files:

- `jenkins_log_backup.py`

```
#!/usr/bin/env python3
"""
Jenkins Build Log Backup to Azure Blob Storage
Author: Auto-generated script for log backup automation
"""

import os
import sys
import json
import logging
from datetime import datetime, timedelta
```

```
import requests
from requests.auth import HTTPBasicAuth
from azure.storage.blob import BlobServiceClient
import argparse

# Configure logging
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    handlers=[
        logging.FileHandler('jenkins_backup.log'),
        logging.StreamHandler(sys.stdout)
    ]
)
logger = logging.getLogger(__name__)

class JenkinsLogBackup:
    def __init__(self, config_file='config.json'):
        """Initialize with configuration from file or environment variables"""
        self.config = self.load_config(config_file)
        self.jenkins_url = self.config['jenkins']['url']
        self.jenkins_user = self.config['jenkins']['username']
        self.jenkins_token = self.config['jenkins']['api_token']
        self.azure_connection_string = self.config['azure']['connection_string']
        self.container_name = self.config['azure']['container_name']

        # Initialize clients
        self.blob_service_client = BlobServiceClient.from_connection_string(
            self.azure_connection_string
        )
        self.ensure_container_exists()

    def load_config(self, config_file):
        """Load configuration from file or environment variables"""
        if os.path.exists(config_file):
            with open(config_file, 'r') as f:
                return json.load(f)
        else:
```

```
        # Fallback to environment variables
        return {
            "jenkins": {
                "url": os.getenv('JENKINS_URL', 'http://localhost:8080'),
                "username": os.getenv('JENKINS_USERNAME'),
                "api_token": os.getenv('JENKINS_API_TOKEN')
            },
            "azure": {
                "connection_string": os.getenv('AZURE_STORAGE_CONNECTION_
                "container_name": os.getenv('AZURE_CONTAINER_NAME', 'jenkin
            }
        }

    def ensure_container_exists(self):
        """Create Azure Blob container if it doesn't exist"""
        try:
            container_client = self.blob_service_client.get_container_client(
                self.container_name
            )
            container_client.create_container()
            logger.info(f"Created container: {self.container_name}")
        except Exception as e:
            if "ContainerAlreadyExists" in str(e):
                logger.info(f"Container {self.container_name} already exists")
            else:
                logger.error(f"Error creating container: {e}")
                raise

    def get_jenkins_jobs(self):
        """Get list of all Jenkins jobs"""
        try:
            url = f"{self.jenkins_url}/api/json?tree=jobs[name]"
            response = requests.get(
                url,
                auth=HTTPBasicAuth(self.jenkins_user, self.jenkins_token),
                timeout=30
            )
            response.raise_for_status()
            jobs_data = response.json()
            return [job['name'] for job in jobs_data['jobs']]
        except Exception as e:
            logger.error(f"Error getting Jenkins jobs: {e}")
            raise

    def get_job_builds(self, job_name, days_back=7):
        """Get recent builds for a specific job"""
        try:
            url = f"{self.jenkins_url}/job/{job_name}/api/json?tree=builds[number,ti
            response = requests.get(
                url,
                auth=HTTPBasicAuth(self.jenkins_user, self.jenkins_token),
                timeout=30
            )
            response.raise_for_status()
            builds_data = response.json()

            # Filter builds from last N days
            cutoff_time = datetime.now() - timedelta(days=days_back)
            cutoff_timestamp = int(cutoff_time.timestamp() * 1000)

            recent_builds = [
                build for build in builds_data['builds']
                if build['timestamp'] > cutoff_timestamp
            ]

            return recent_builds
        except Exception as e:
            logger.error(f"Error getting builds for job {job_name}: {e}")
            return []

    def get_build_log(self, job_name, build_number):
        """Download build log from Jenkins"""
        try:
            url = f"{self.jenkins_url}/job/{job_name}/{build_number}/consoleText"
            response = requests.get(
                url,
                auth=HTTPBasicAuth(self.jenkins_user, self.jenkins_token),
                timeout=60
            )
            response.raise_for_status()
            return response.text
        except Exception as e:
            logger.error(f"Error getting log for {job_name}#{build_number}: {e}")
            return None

    def upload_to_azure(self, job_name, build_number, log_content, build_info):
        """Upload log to Azure Blob Storage"""
        try:
            # Create blob name with timestamp
            timestamp = datetime.fromtimestamp(build_info['timestamp'] / 1000)
            blob_name = f"{job_name}/{timestamp.strftime('%Y/%m/%d')}/{job_n

            # Add metadata
            metadata = {
                'job_name': job_name,
                'build_number': str(build_number),
                'build_result': build_info.get('result', 'UNKNOWN'),
                'build_timestamp': str(build_info['timestamp']),
                'upload_timestamp': str(int(datetime.now().timestamp() * 1000))
            }

            # Upload to blob
            blob_client = self.blob_service_client.get_blob_client(
                container=self.container_name,
                blob=blob_name
            )

            blob_client.upload_blob(
                log_content,
                overwrite=True,
                metadata=metadata
            )

            logger.info(f"Uploaded: {blob_name}")
            return blob_name
        except Exception as e:
            logger.error(f"Error uploading {job_name}#{build_number}: {e}")
            return None

    def backup_job_logs(self, job_name, days_back=7):
        """Backup logs for a specific job"""
        logger.info(f"Starting backup for job: {job_name}")

        builds = self.get_job_builds(job_name, days_back)
        if not builds:
            logger.info(f"No recent builds found for {job_name}")
            return

        uploaded_count = 0
        for build in builds:
            build_number = build['number']

            # Check if already uploaded
            timestamp = datetime.fromtimestamp(build['timestamp'] / 1000)
            blob_name = f"{job_name}/{timestamp.strftime('%Y/%m/%d')}/{job_na

            try:
                blob_client = self.blob_service_client.get_blob_client(
                    container=self.container_name,
                    blob=blob_name
                )
                if blob_client.exists():
                    logger.info(f"Skipping {job_name}#{build_number} - already exists
                    continue
            except:
                pass  # Continue with upload if check fails

            # Get and upload log
            log_content = self.get_build_log(job_name, build_number)
            if log_content:
                uploaded_blob = self.upload_to_azure(job_name, build_number, log_
                if uploaded_blob:
```

```
                uploaded_count += 1

        logger.info(f"Completed backup for {job_name}: {uploaded_count} logs u

    def backup_all_jobs(self, days_back=7, job_filter=None):
        """Backup logs for all jobs or filtered jobs"""
        try:
            jobs = self.get_jenkins_jobs()

            if job_filter:
                jobs = [job for job in jobs if job_filter.lower() in job.lower()]

            logger.info(f"Found {len(jobs)} jobs to backup")

            total_uploaded = 0
            for job in jobs:
                try:
                    self.backup_job_logs(job, days_back)
                except Exception as e:
                    logger.error(f"Error backing up job {job}: {e}")
                    continue

            logger.info("Backup process completed")

        except Exception as e:
            logger.error(f"Error in backup process: {e}")
            raise

def main():
    parser = argparse.ArgumentParser(description='Backup Jenkins logs to Az
    parser.add_argument('--job', help='Specific job name to backup')
    parser.add_argument('--days', type=int, default=7, help='Number of days b
    parser.add_argument('--filter', help='Filter jobs by name pattern')
    parser.add_argument('--config', default='config.json', help='Configuration f

    args = parser.parse_args()

    try:
        backup = JenkinsLogBackup(config_file=args.config)

        if args.job:
            backup.backup_job_logs(args.job, args.days)
        else:
            backup.backup_all_jobs(args.days, args.filter)

    except Exception as e:
        logger.error(f"Backup failed: {e}")
        sys.exit(1)

if __name__ == "__main__":
    main()
```

- `requirements.txt`

```
requests==2.31.0
azure-storage-blob==12.19.0
python-dateutil==2.8.2
```

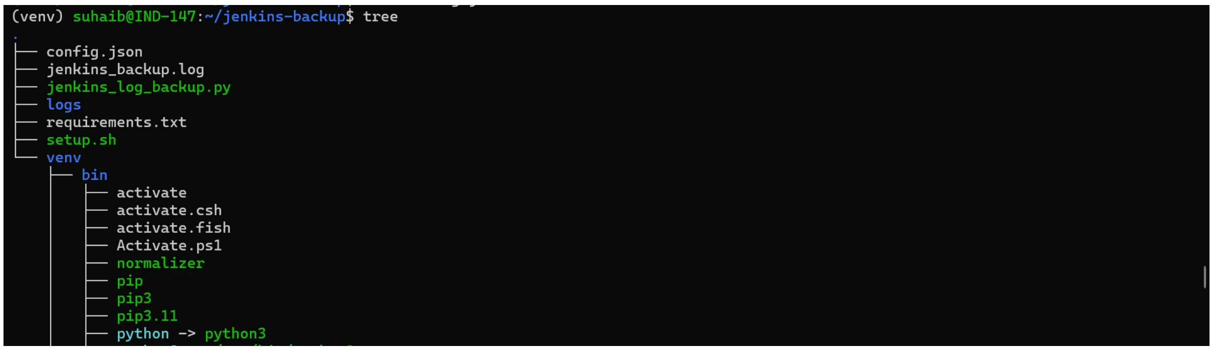
- `config.json`

```
{
  "jenkins": {
    "url": "http://localhost:8080",
    "username": "your_jenkins_username",
    "api_token": "your_jenkins_api_token"
  },
  "azure": {
    "connection_string": "your_azure_storage_connection_string",
    "container_name": "jenkins-logs"
  }
}

#eg:
{
  "jenkins": {
    "url": "http://localhost:8080",
    "username": "admin",
    "api_token": "11a94a393c87b0a2724f103f2056be9f2a"
  },
  "azure": {
    "connection_string": "DefaultEndpointsProtocol=https;EndpointSuffix=core.
    "container_name": "jenkins-logs"
  }
}
```

- `setup.sh`

```
#!/bin/bash

# Jenkins Log Backup Setup Script for Debian WSL
# This script sets up the environment for backing up Jenkins logs to Azure

set -e

echo "🚀 Setting up Jenkins Log Backup to Azure..."

# Update system packages
echo "📦 Updating system packages..."
sudo apt update && sudo apt upgrade -y

# Install Python3 and pip if not installed
echo "🐍 Installing Python3 and pip..."
sudo apt install -y python3 python3-pip python3-venv

# Create project directory
PROJECT_DIR="$HOME/jenkins-backup"
echo "📁 Creating project directory: $PROJECT_DIR"
mkdir -p "$PROJECT_DIR"
cd "$PROJECT_DIR"

# Create virtual environment
echo "🏗️ Creating Python virtual environment..."
python3 -m venv venv
source venv/bin/activate
```

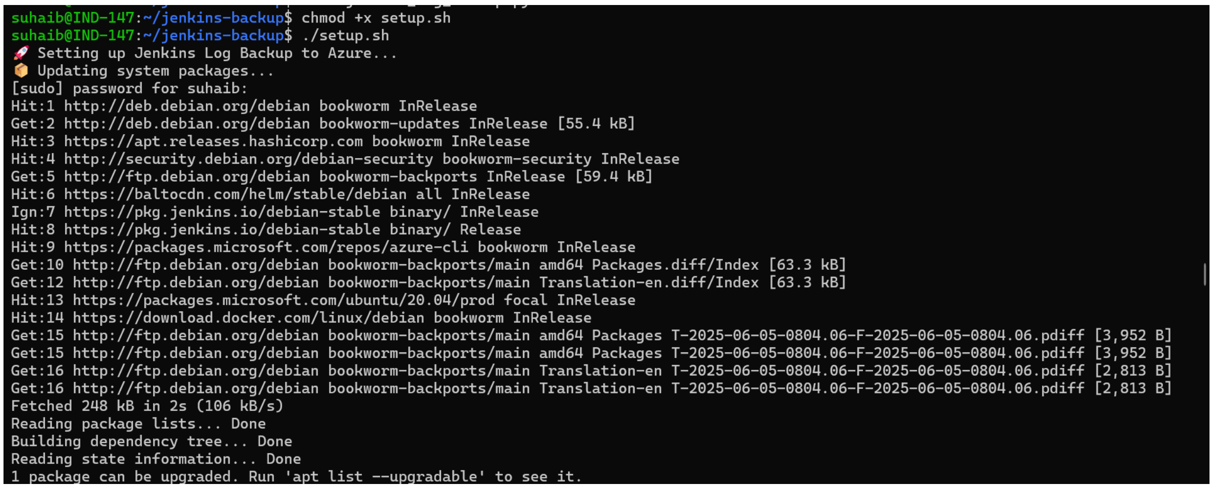
```
# Install Python dependencies
echo "📚 Installing Python dependencies..."
pip install --upgrade pip
pip install -r requirements.txt

# Make script executable
chmod +x jenkins_log_backup.py

# Create logs directory
mkdir -p logs

echo "✅ Setup completed successfully!"
echo ""
echo "📋 Next steps:"
echo "1. Edit config.json with your Jenkins and Azure credentials"
echo "2. Activate virtual environment: source $PROJECT_DIR/venv/bin/activat
echo "3. Test the script: python3 jenkins_log_backup.py --help"
echo ""
echo "🔧 Configuration needed:"
echo "- Jenkins URL, username, and API token"
echo "- Azure Storage connection string"
echo "- Container name (optional, defaults to 'jenkins-logs')"
```



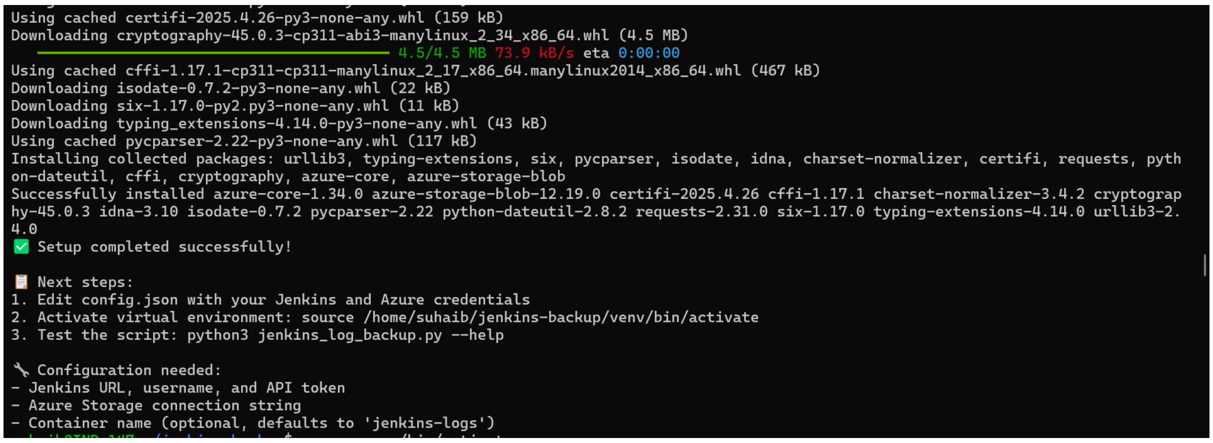
```
(venv) suhaib@IND-147:~/jenkins-backup$ tree
.
├── config.json
├── jenkins_backup.log
├── jenkins_log_backup.py
├── logs
├── requirements.txt
├── setup.sh
└── venv
    ├── bin
    │   ├── activate
    │   ├── activate.csh
    │   ├── activate.fish
    │   ├── Activate.ps1
    │   ├── normalizer
    │   ├── pip
    │   ├── pip3
    │   ├── pip3.11
    │   ├── python -> python3
```

2. Run Setup Script

```
chmod +x setup.sh
./setup.sh
```



```
suhaib@IND-147:~/jenkins-backup$ chmod +x setup.sh
suhaib@IND-147:~/jenkins-backup$ ./setup.sh
🚀 Setting up Jenkins Log Backup to Azure...
📦 Updating system packages...
[sudo] password for suhaib:
Hit:1 http://deb.debian.org/debian bookworm InRelease
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Hit:3 https://apt.releases.hashicorp.com bookworm InRelease
Hit:4 http://security.debian.org/debian-security bookworm-security InRelease
Get:5 http://ftp.debian.org/debian bookworm-backports InRelease [59.4 kB]
Hit:6 https://baltocdn.com/helm/stable/debian all InRelease
Ign:7 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:8 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:9 https://packages.microsoft.com/repos/azure-cli bookworm InRelease
Get:10 http://ftp.debian.org/debian bookworm-backports/main amd64 Packages.diff/Index [63.3 kB]
Get:12 http://ftp.debian.org/debian bookworm-backports/main Translation-en.diff/Index [63.3 kB]
Hit:13 https://packages.microsoft.com/ubuntu/20.04/prod focal InRelease
Hit:14 https://download.docker.com/linux/debian bookworm InRelease
Get:15 http://ftp.debian.org/debian bookworm-backports/main amd64 Packages T-2025-06-05-0804.06-F-2025-06-05-0804.06.pdiff [3,952 B]
Get:15 http://ftp.debian.org/debian bookworm-backports/main amd64 Packages T-2025-06-05-0804.06-F-2025-06-05-0804.06.pdiff [3,952 B]
Get:16 http://ftp.debian.org/debian bookworm-backports/main Translation-en T-2025-06-05-0804.06-F-2025-06-05-0804.06.pdiff [2,813 B]
Get:16 http://ftp.debian.org/debian bookworm-backports/main Translation-en T-2025-06-05-0804.06-F-2025-06-05-0804.06.pdiff [2,813 B]
Fetched 248 kB in 2s (106 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
```



```
Using cached certifi-2025.4.26-py3-none-any.whl (159 kB)
Downloading cryptography-45.0.3-cp311-abi3-manylinux_2_34_x86_64.whl (4.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 73.9 kB/s eta 0:00:00
Using cached cffi-1.17.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (467 kB)
Downloading isodate-0.7.2-py3-none-any.whl (22 kB)
Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Downloading typing_extensions-4.14.0-py3-none-any.whl (43 kB)
Using cached pycparser-2.22-py3-none-any.whl (117 kB)
Installing collected packages: urllib3, typing-extensions, six, pycparser, isodate, idna, charset-normalizer, certifi, requests, pyth
on-dateutil, cffi, cryptography, azure-core, azure-storage-blob
Successfully installed azure-core-1.34.0 azure-storage-blob-12.19.0 certifi-2025.4.26 cffi-1.17.1 charset-normalizer-3.4.2 cryptograp
hy-45.0.3 idna-3.10 isodate-0.7.2 pycparser-2.22 python-dateutil-2.8.2 requests-2.31.0 six-1.17.0 typing-extensions-4.14.0 urllib3-2.
4.0
✅ Setup completed successfully!

📋 Next steps:
1. Edit config.json with your Jenkins and Azure credentials
2. Activate virtual environment: source /home/suhaib/jenkins-backup/venv/bin/activate
3. Test the script: python3 jenkins_log_backup.py --help

🔧 Configuration needed:
- Jenkins URL, username, and API token
- Azure Storage connection string
- Container name (optional, defaults to 'jenkins-logs')
```

Script Implementation

Main Script Structure

The solution consists of several Python modules:

1. **jenkins_log_uploader.py** - Main script
2. **config.py** - Configuration management
3. **requirements.txt** - Python requirements
4. **setup.sh** - Environment setup

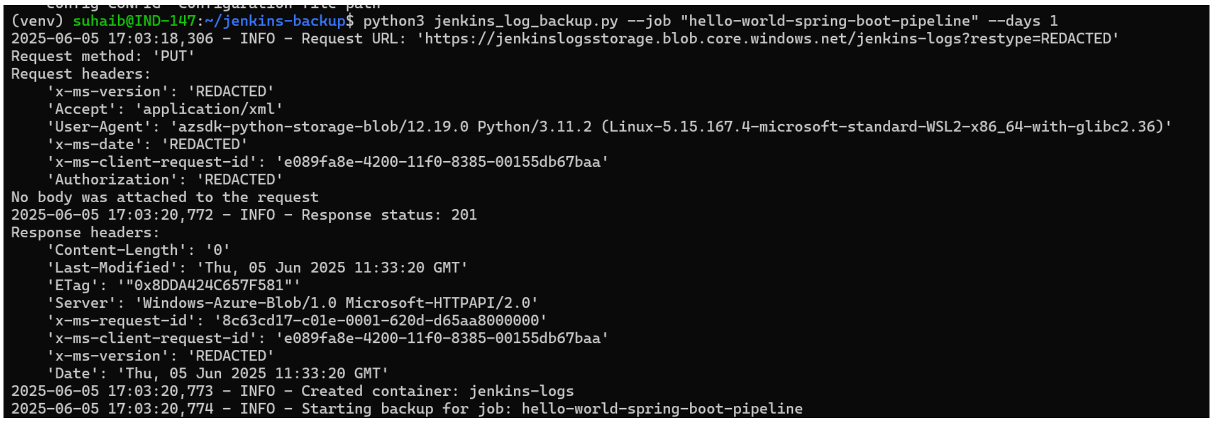
Usage

Activate Virtual Environment

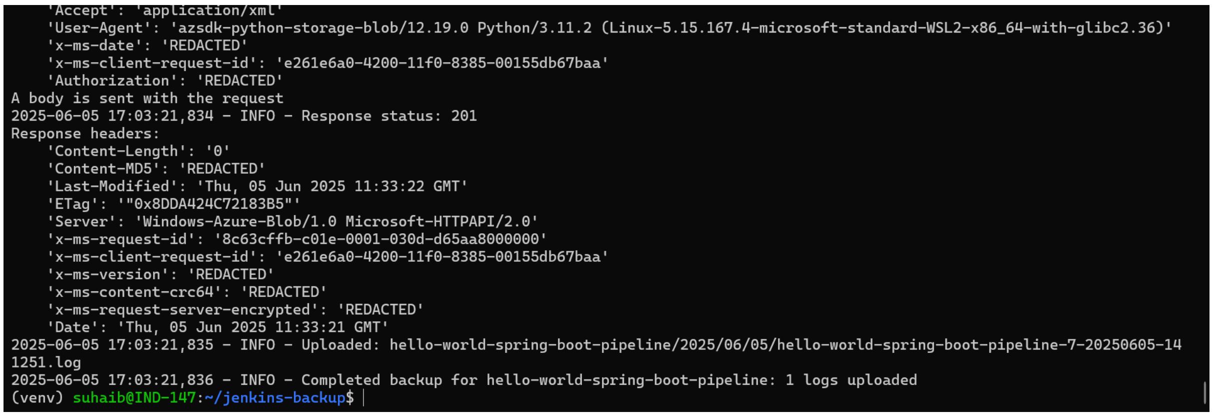
```
cd ~/jenkins-backup
source venv/bin/activate
```

Basic Usage

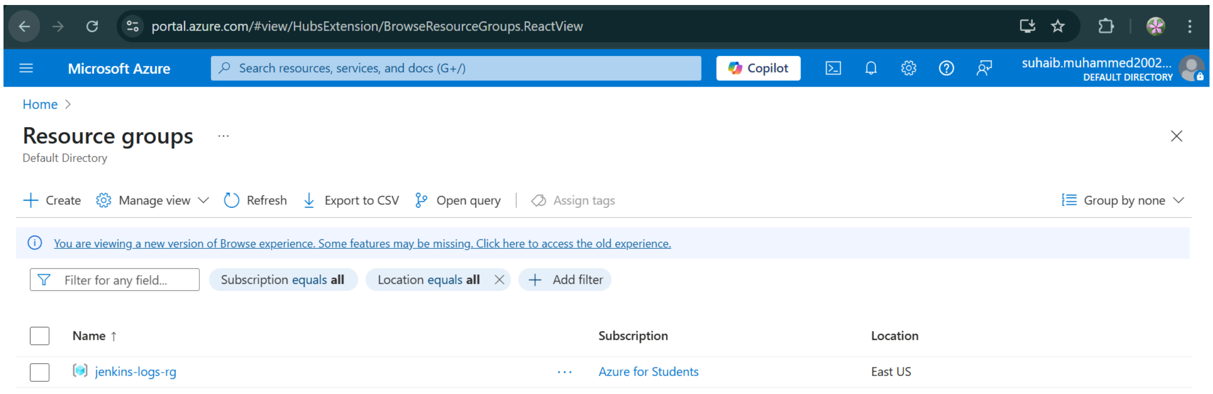
```
# Backup all jobs from last 7 days
python3 jenkins_log_backup.py

# Backup specific job
python3 jenkins_log_backup.py --job "my-project"

# Backup last 30 days
python3 jenkins_log_backup.py --days 30

# Filter jobs by name pattern
python3 jenkins_log_backup.py --filter "frontend"

# Use custom config file
python3 jenkins_log_backup.py --config /path/to/config.json

# Test script
python3 jenkins_log_backup.py --help

# Test with a specific job (replace 'your-job-name' with actual job)
python3 jenkins_log_backup.py --job "your-job-name" --days 1
```



```
    --config CONFIG  Configuration file path
(venv) suhaib@IND-147:~/jenkins-backup$ python3 jenkins_log_backup.py --job "hello-world-spring-boot-pipeline" --days 1
2025-06-05 17:03:18,306 - INFO - Request URL: 'https://jenkinslogsstorage.blob.core.windows.net/jenkins-logs?restype=REDACTED'
Request method: 'PUT'
Request headers:
    'x-ms-version': 'REDACTED'
    'Accept': 'application/xml'
    'User-Agent': 'azsdk-python-storage-blob/12.19.0 Python/3.11.2 (Linux-5.15.167.4-microsoft-standard-WSL2-x86_64-with-glibc2.36)'
    'x-ms-date': 'REDACTED'
    'x-ms-client-request-id': 'e089fa8e-4200-11f0-8385-00155db67baa'
    'Authorization': 'REDACTED'
No body was attached to the request
2025-06-05 17:03:20,772 - INFO - Response status: 201
Response headers:
    'Content-Length': '0'
    'Last-Modified': 'Thu, 05 Jun 2025 11:33:20 GMT'
    'ETag': '"0x8DDA424C657F581"'
    'Server': 'Windows-Azure-Blob/1.0 Microsoft-HTTPAPI/2.0'
    'x-ms-request-id': '8c63cd17-c01e-0001-620d-d65aa8000000'
    'x-ms-client-request-id': 'e089fa8e-4200-11f0-8385-00155db67baa'
    'x-ms-version': 'REDACTED'
    'Date': 'Thu, 05 Jun 2025 11:33:20 GMT'
2025-06-05 17:03:20,773 - INFO - Created container: jenkins-logs
2025-06-05 17:03:20,774 - INFO - Starting backup for job: hello-world-spring-boot-pipeline
```



```
    'Accept': 'application/xml'
    'User-Agent': 'azsdk-python-storage-blob/12.19.0 Python/3.11.2 (Linux-5.15.167.4-microsoft-standard-WSL2-x86_64-with-glibc2.36)'
    'x-ms-date': 'REDACTED'
    'x-ms-client-request-id': 'e261e6a0-4200-11f0-8385-00155db67baa'
    'Authorization': 'REDACTED'
A body is sent with the request
2025-06-05 17:03:21,834 - INFO - Response status: 201
Response headers:
    'Content-Length': '0'
    'Content-MD5': 'REDACTED'
    'Last-Modified': 'Thu, 05 Jun 2025 11:33:22 GMT'
    'ETag': '"0x8DDA424C72183B5"'
    'Server': 'Windows-Azure-Blob/1.0 Microsoft-HTTPAPI/2.0'
    'x-ms-request-id': '8c63cffb-c01e-0001-030d-d65aa8000000'
    'x-ms-client-request-id': 'e261e6a0-4200-11f0-8385-00155db67baa'
    'x-ms-version': 'REDACTED'
    'x-ms-content-crc64': 'REDACTED'
    'x-ms-request-server-encrypted': 'REDACTED'
    'Date': 'Thu, 05 Jun 2025 11:33:21 GMT'
2025-06-05 17:03:21,835 - INFO - Uploaded: hello-world-spring-boot-pipeline/2025/06/05/hello-world-spring-boot-pipeline-7-20250605-14
1251.log
2025-06-05 17:03:21,836 - INFO - Completed backup for hello-world-spring-boot-pipeline: 1 logs uploaded
(venv) suhaib@IND-147:~/jenkins-backup$
```



portal.azure.com/#view/HubsExtension/BrowseResourceGroups.ReactView

Microsoft Azure | suhaib.muhammed2002... DEFAULT DIRECTORY

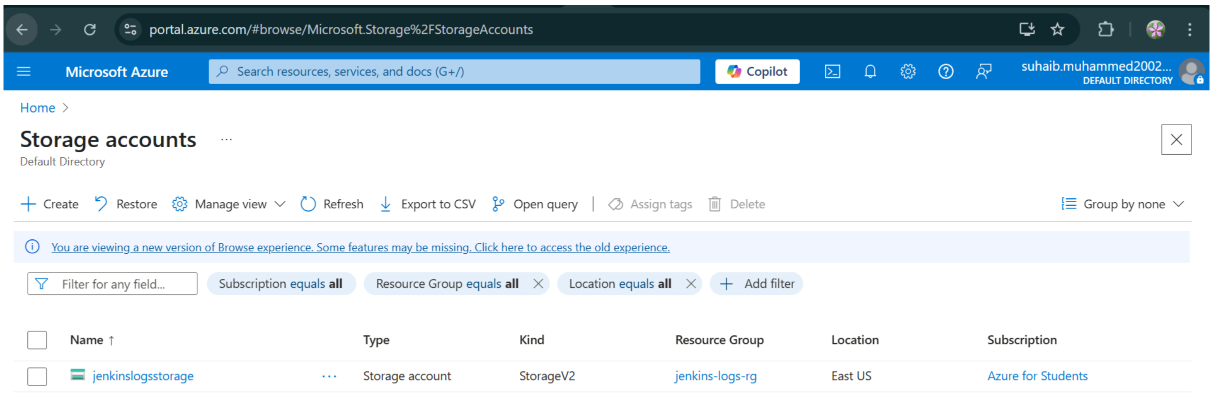
Home >

Resource groups
Default Directory

You are viewing a new version of Browse experience. Some features may be missing. Click here to access the old experience.

Filter for any field... | Subscription equals **all** | Location equals **all** | Add filter

Name ↑	Subscription	Location
jenkins-logs-rg	Azure for Students	East US



portal.azure.com/#browse/Microsoft.Storage%2FStorageAccounts

Microsoft Azure | suhaib.muhammed2002... DEFAULT DIRECTORY

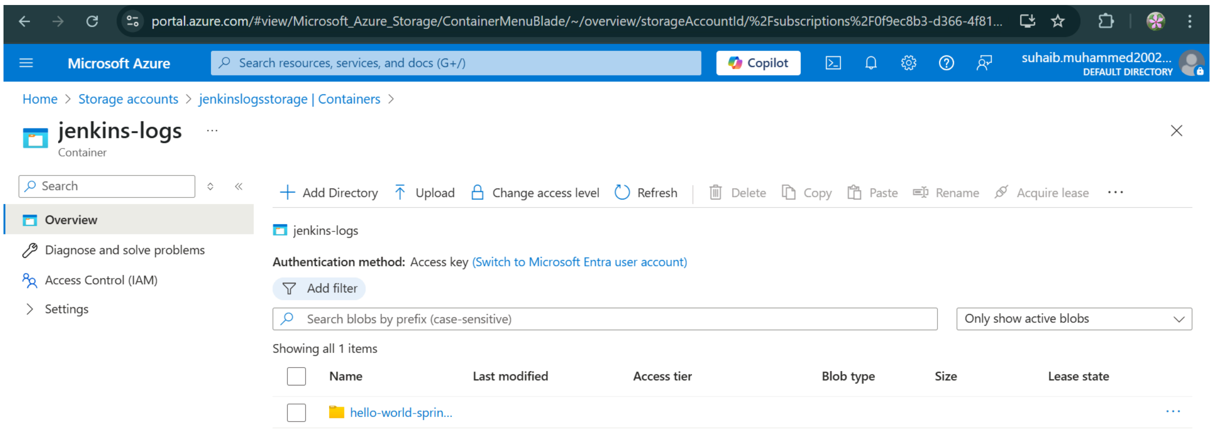
Home >

Storage accounts
Default Directory

You are viewing a new version of Browse experience. Some features may be missing. Click here to access the old experience.

Filter for any field... | Subscription equals **all** | Resource Group equals **all** | Location equals **all** | Add filter

Name ↑	Type	Kind	Resource Group	Location	Subscription
jenkinslogsstorage	Storage account	StorageV2	jenkins-logs-rg	East US	Azure for Students


portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2F0f9ec8b3-d366-4f81...

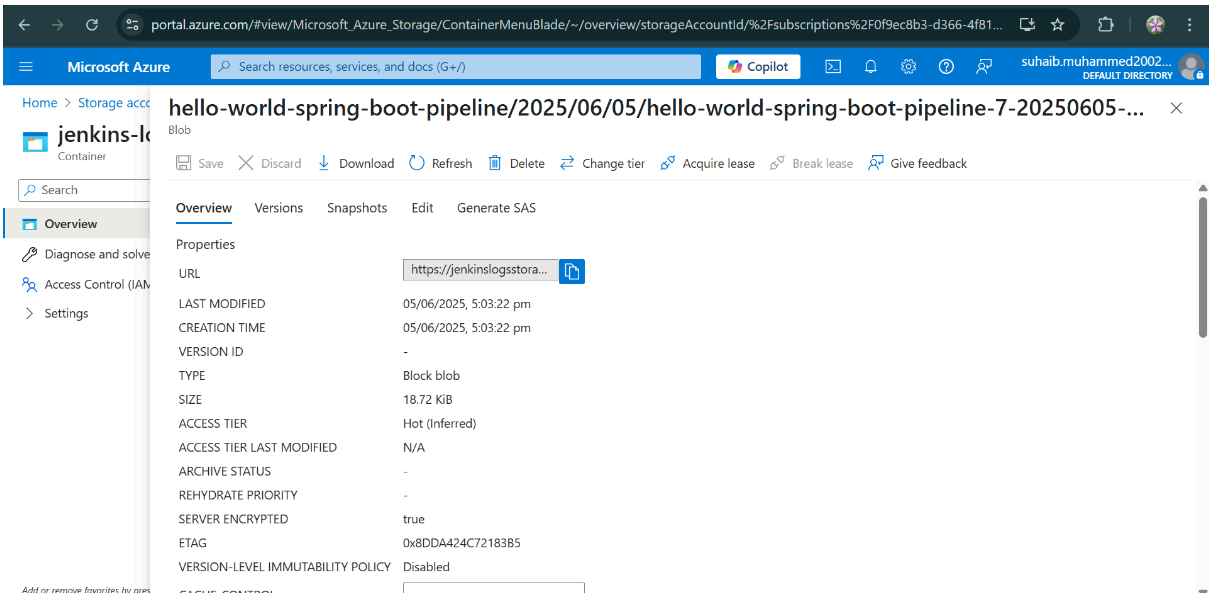
Home > Storage accounts > jenkinslogsstorage | Containers >

jenkins-logs
Container

Authentication method: Access key (Switch to Microsoft Entra user account)

Showing all 1 items

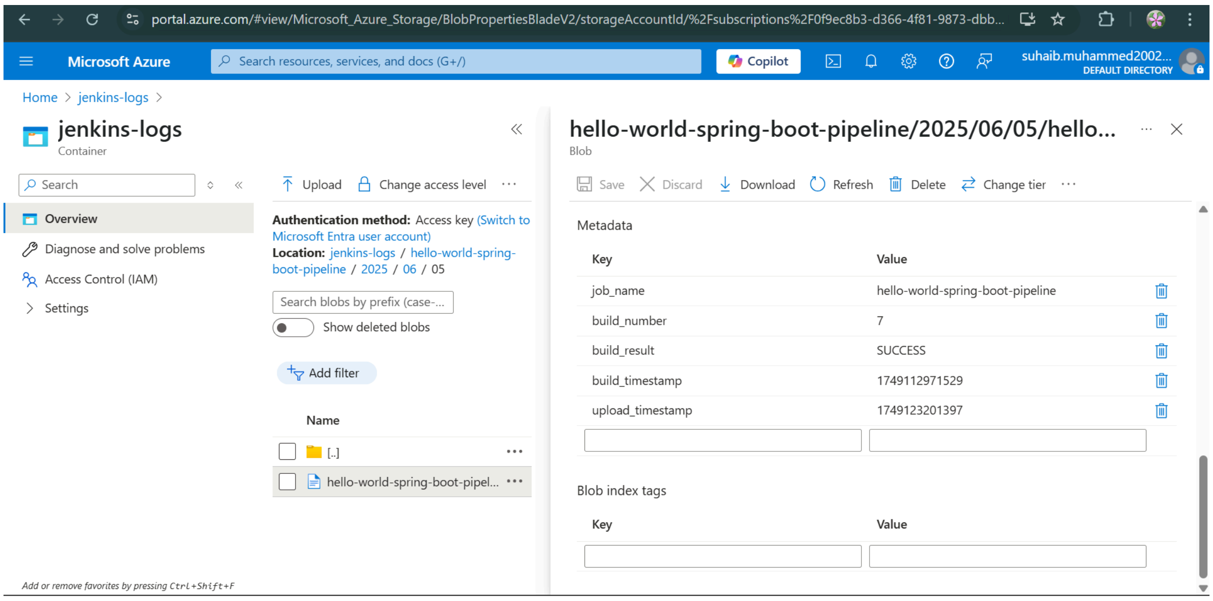
Name	Last modified	Access tier	Blob type	Size	Lease state
hello-world-sprin...					


portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2F0f9ec8b3-d366-4f81...

hello-world-spring-boot-pipeline/2025/06/05/hello-world-spring-boot-pipeline-7-20250605-...
Blob

Properties

URL	https://jenkinslogsstora...
LAST MODIFIED	05/06/2025, 5:03:22 pm
CREATION TIME	05/06/2025, 5:03:22 pm
VERSION ID	-
TYPE	Block blob
SIZE	18.72 KiB
ACCESS TIER	Hot (Inferred)
ACCESS TIER LAST MODIFIED	N/A
ARCHIVE STATUS	-
REHYDRATE PRIORITY	-
SERVER ENCRYPTED	true
ETAG	0x8DDA424C72183B5
VERSION-LEVEL IMMUTABILITY POLICY	Disabled


portal.azure.com/#view/Microsoft_Azure_Storage/BlobPropertiesBladeV2/storageAccountId/%2Fsubscriptions%2F0f9ec8b3-d366-4f81-9873-dbb...

Home > jenkins-logs >

jenkins-logs
Container

Authentication method: Access key (Switch to Microsoft Entra user account)
Location: jenkins-logs / hello-world-spring-boot-pipeline / 2025 / 06 / 05

hello-world-spring-boot-pipel...

hello-world-spring-boot-pipeline/2025/06/05/hello...
Blob

Metadata

Key	Value
job_name	hello-world-spring-boot-pipeline
build_number	7
build_result	SUCCESS
build_timestamp	1749112971529
upload_timestamp	1749123201397

Blob index tags

Advanced Options

```
# Upload with custom retention policy
python jenkins_log_backup.py \
    --job "hello-world-spring-boot-pipeline" \
    --build 10 \
    --retention-days 90

# Upload with compression
python jenkins_log_backup.py \
    --job "hello-world-spring-boot-pipeline" \
    --build 10 \
    --compress

# Dry run (simulate without uploading)
python jenkins_log_backup.py \
    --job "hello-world-spring-boot-pipeline" \
    --build 10 \
    --dry-run
```

Log Types

The script supports uploading different types of logs:

```
# Console logs only
python jenkins_log_backup.py \
    --job "hello-world-spring-boot-pipeline" \
    --build 10 \
    --log-types console

# Include test results and artifacts
python jenkins_log_backup.py \
    --job "hello-world-spring-boot-pipeline" \
    --build 10 \
    --log-types console,test-results,artifacts
```

Command Line Options

Option	Description	Default

`--job`	Specific job name to backup	All jobs
`--days`	Number of days back to backup	7
`--filter`	Filter jobs by name pattern	None
`--config`	Path to configuration file	config.json

Azure Blob Structure

Logs are organized in Azure Blob Storage as follows:

```
jenkins-logs/
├── job-name-1/
│   ├── 2024/
│   │   ├── 01/
│   │   │   ├── 15/
│   │   │   │   ├── job-name-1-123-20240115-143022.log
│   │   │   │   └── job-name-1-124-20240115-150145.log
│   │   │   └── 16/
│   │   └── 02/
│   └── 2025/
└── job-name-2/
    └── 2024/
```

Blob Metadata

Each uploaded log includes metadata:

- `job_name`: Jenkins job name
- `build_number`: Build number
- `build_result`: Build result (SUCCESS, FAILURE, etc.)
- `build_timestamp`: Original build timestamp
- `upload_timestamp`: Upload timestamp

Automation Setup

1. Cron Job Setup

```
# Edit crontab
crontab -e

# Add entry for daily backup at 2 AM
0 2 * * * cd /home/ubuntu/jenkins-log-uploader && ./venv/bin/python jenki
ns_log_uploader.py --all-jobs --recent 1 >> /var/log/jenkins-log-uploader.lo
g 2>&1

# Add entry for weekly cleanup (older than 30 days)
0 3 * * 0 cd /home/ubuntu/jenkins-log-uploader && ./venv/bin/python jenki
ns_log_uploader.py --cleanup --older-than 30
```

2. Systemd Service (Alternative)

```
# Create service file
sudo nano /etc/systemd/system/jenkins-log-uploader.service

# Enable and start service
sudo systemctl enable jenkins-log-uploader.service
sudo systemctl start jenkins-log-uploader.service
```

3. Jenkins Pipeline Integration

Add a post-build step to your existing Jenkins pipeline to automatically upload logs after each build.

Monitoring and Troubleshooting

1. Log Monitoring

```
# View recent logs
tail -f /var/log/jenkins-log-backup.log

# Check for errors
grep ERROR /var/log/jenkins-log-backup.log
```

```
# Monitor upload statistics
python jenkins_log_backup.py --stats
```

2. Health Checks

```
# Check service status
python jenkins_log_backup.py --health-check

# Verify Azure connectivity
python jenkins_log_backup.py --test-azure

# Check storage usage
python jenkins_log_backup.py --storage-info
```

3. Common Issues and Solutions

Issue: Jenkins API Authentication Failed

```
# Solution: Verify credentials
curl -u username:token http://localhost:8080/api/json
```

Issue: Azure Blob Upload Failed

```
# Solution: Check network connectivity
az storage blob list --container-name jenkins-build-logs --account-name je
nkinslogsstorage
```

Issue: Large Log Files

```
# Solution: Enable compression
python jenkins_log_uploader.py --job "job-name" --build 10 --compress --c
hunk-size 10MB
```

Best Practices

Security

- Use environment variables for sensitive data
- Implement proper access controls
- Regular credential rotation
- Enable Azure Storage encryption

Performance

- Use compression for large logs
- Implement chunked uploads
- Batch processing for multiple files
- Connection pooling

Monitoring

- Set up alerts for failed uploads
- Monitor storage costs
- Track upload statistics
- Regular health checks

Maintenance

- Implement log retention policies
- Regular cleanup of old logs
- Monitor storage usage
- Update dependencies regularly

Cost Optimization

Azure Storage Tiers

- Use Hot tier for recent logs (last 30 days)
- Use Cool tier for older logs (30-90 days)
- Use Archive tier for long-term retention (>90 days)

Compression

- Enable gzip compression to reduce storage costs
- Compress logs older than 7 days
- Use efficient compression algorithms

Compliance and Governance

Retention Policies

- Implement automated retention policies
- Support for legal hold requirements
- Audit trail for log access
- Data classification and tagging

Access Control

- Role-based access to logs
- Integration with Azure AD
- Audit logging for access
- Encryption in transit and at rest