Technical Report

Building Storage as a Service with ServiceNow and NetApp Service Level Manager

YiZhao Zhuang, Rajesh V., Nayanthara S., NetApp

May 2019 | TR-4779

Abstract

To address the challenges that modern service providers are facing, the NetApp® IT department has developed and proven a new Storage-as-a-Service solution that uses ServiceNow and NetApp Service Level Manager.

NetApp®

TABLE OF CONTENTS

1 Customer Challenges 3

1.1 Overview 3

1.2 Solution 3

2 NetApp Service Level Manager 3

2.1 Overview 3

2.1.1 Performance Service Level Policy Definitions 4

2.2 How does NSLM work? 4

2.2.1 Standardize 4

2.2.2 Automate 5

2.2.3 Monitor and Regulate 5

2.2.4 Benefits using NSLM 5

3 An Example of Integration with ServiceNow and NSLM 5

3.1 Solution Components 5

3.2 Storage-as-a-Service Solution Configuration 6

3.3 Implementation Workflow 6

3.4 User Interface 7

3.5 Storage-as-a-Service Script Workflow 8

3.6 Storage-as-a-Service Sample Code 9

4 Where to Find Additional Information 12

LIST OF TABLES

Table 1) Predefined service levels. 4

Table 2) STaaS solution components. 5

LIST OF FIGURES

Figure 1) NSLM flow. 4

Figure 2) STaaS solution configuration. 6

Figure 3) STaaS workflow implementation. 6

Figure 4) ServiceNow Service Portal form. 7

Figure 5) STaaS Script workflow. 8

1 Customer Challenges

1.1 Overview

The IT world is changing at an increasingly rapid pace. IT generalists and service providers are looking for simple and reliable ways to deploy solutions that meet the dynamic storage environment demands of their end users. For example, DevOps professionals are rapidly prototyping storage solutions and iterating them continuously. These users require storage to create new applications on a regular basis.

Traditionally, end users requested storage by opening help tickets in the IT service portal. These requests were prioritized and put in the storage administrator's queue. The storage administrator would then find a storage location, provision the storage, send an e-mail to the requester, and perform other tasks before closing the ticket.

To improve agility and reduce costs, users would like to automate these processes, accelerate the service delivery time, and efficiently manage the performance and capacity of resource consumption.

1.2 Solution

To address these sorts of challenges, we present in this report a simple and reliable solution that uses ServiceNow and NetApp Service Level Manager (NSLM).

With this solution, service providers can implement an on-demand, self-service solution called storage as a service (STaaS) that enables end users to provision file-based storage when needed and without human interaction. With the STaaS solution, an end user can fill out a simple web form in ServiceNow. Within a few minutes of submitting the request, the requester is notified that storage is provisioned and ready to use. To improve the performance of storage resource management, NSLM supports efficient deployment, simplified data management, and ready control of NetApp ONTAP® systems.

This STaaS solution provides the following key features:

- Storage provisioning based on a user's requirements for capacity and performance.
- Three service level tiers—value, performance, and extreme—tailored, respectively, to high-capacity, database, and latency-sensitive applications.
- Support for NFS and SMB/CIFS shares for access across UNIX and Windows environments.
- Dynamic allocation of storage resources in the background based on an understanding of existing headroom, capacity, and underlying drive types of the ONTAP system.

The NetApp IT department has deployed and provisioned over 250 storage requests with this STaaS solution since Feb 2018.

NetApp has demonstrated that this solution minimizes the turnaround time from an average of 3 to 4 days down to 15 to 20 min for each service request. This solution removes the support-team from this critical path including the requirement for multiple interactions with end users.

2 NetApp Service Level Manager

2.1 Overview

NSLM is an automation software that simplifies workload management and provides STaaS provisioning for on-premises deployment. It uses the ONTAP Adaptive QoS engine and service level policy concept to regulate workload performance by using Performance Service Level (PSL) policy definitions.

2.1.1 Performance Service Level Policy Definitions

To standardize the service catalog, NSLM ships with three predefined PSLs that provide workloads with different level of performance and also allows you to customize PSLs. PSLs are tailored for different of I/O loads and uses and can be treated in a manner similar to adaptive QoS policies.

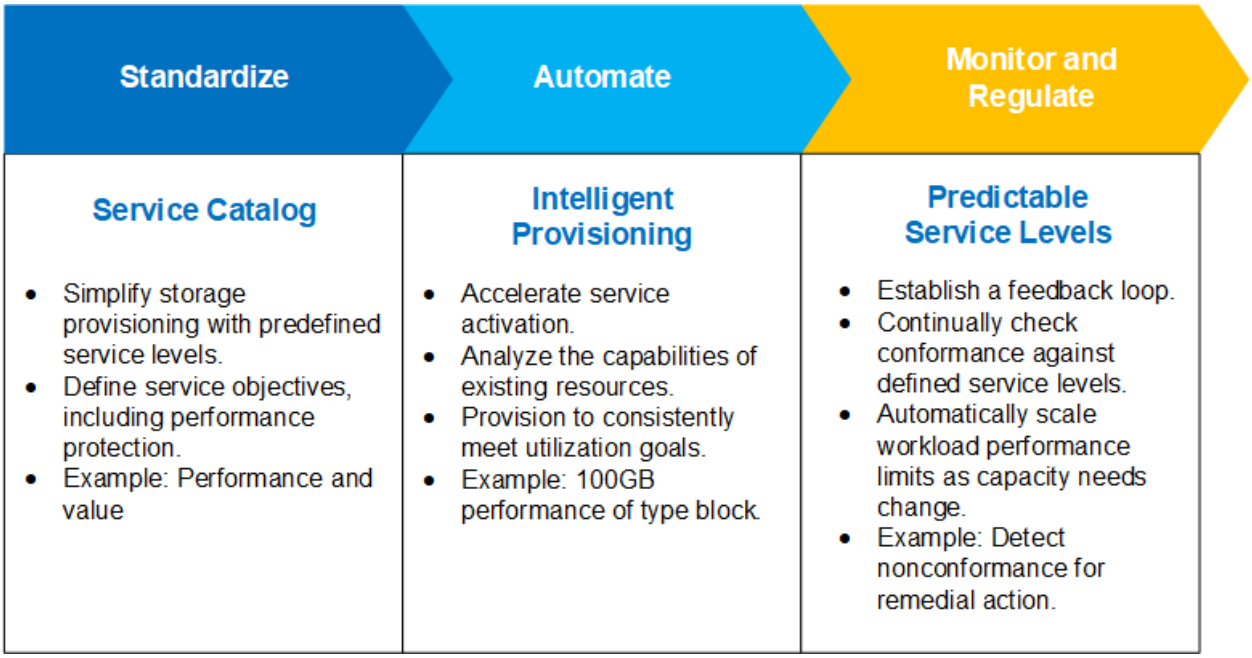
Table 1) Predefined service levels.

Service Level Policy	Value	Performance	Extreme
Workload type	Email, web, file shares, and backup	Databases and virtualized applications	Latency-sensitive applications
Minimum or expected SLA (IOPS/TB allocated)	128	2048	6144
Maximum or peak SLO (QoS limit in IOPS/TB stored)	512	4096	12288
Latency (ms)	17	2	1

2.2 How does NSLM work?

NSLM works in three phases, as is depicted in Figure 1.

Figure 1) NSLM flow.



2.2.1 Standardize

Users can define their service objectives based on performance, space, and efficiency requirements. Users might not be familiar with what level of PSL should be applied to their workload environment. To address this concern, NSLM 1.2 ships with artificial intelligence and machine-learning capabilities that can provide recommended PSL levels for each volume every 24 hours.

2.2.2 Automate

NSLM has built-in intelligence for provisioning and volume placement when given a specific storage virtual machine (SVM). It tracks the performance, capacity, and latency aspects of resources in the storage environment and automatically picks the best place for provisioning.

NSLM is built on a REST API platform. Therefore, users are given access to manage the storage environment and perform operations, at-scale, through REST API calls.

2.2.3 Monitor and Regulate

NSLM continuously checks the conformance of each volume with the latency threshold defined in the assigned PSLs. Upon detecting a nonconforming volume, that volume is flagged with an alert so that remediation actions can be taken by administrators. In addition, NSLM can also detect changes in IOPS usage on volumes and reflect that change in the PSL recommendation every 24 hours.

2.2.4 Benefits using NSLM

NSLM enables cloud economics for on-premises uses and it reduces human intervention by automating storage provisioning and management, ultimately reducing OPEX and CAPEX.

3 An Example of Integration with ServiceNow and NSLM

3.1 Solution Components

Table 2 provides a list of components used in this STaaS solution.

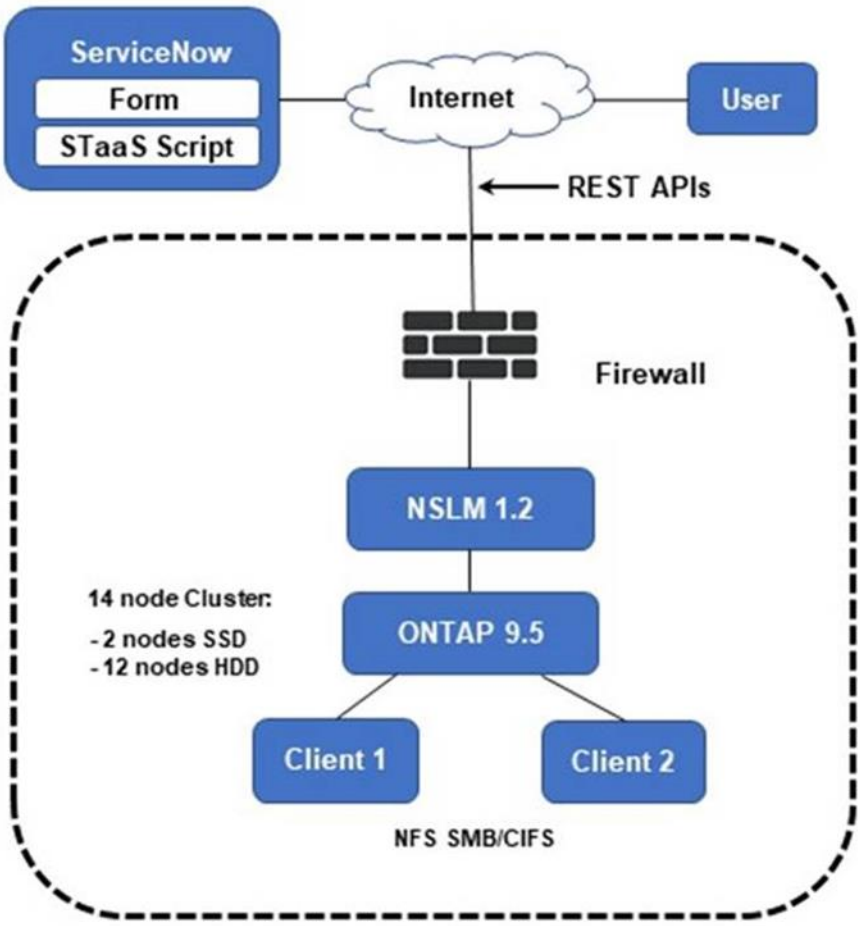
Table 2) STaaS solution components.

Component	Purpose	Netapp Solution	Your Solution
ONTAP 9.1 or later	Storage	ONTAP 9.5 (FAS, AFF)	ONTAP 9.1 or later
Service Portal	Hosts user interface	ServiceNow	Any service portal compatible with REST APIs
NetApp Service Level Manager (NSLM) 1.2	Service level management	VM: RHEL 7.3 • 12GB of RAM • 200GB of disk space	CentOS or Red Hat. Physical or virtual server: Minimum 8GB RAM, recommended 12GB Disk space:150GB. Capacity is allocated as follows: • 50GB allocated to the root partition • 100GB of free disk space allocated to the `/opt/netapp/data` directory, which is mounted on an LVM drive or on a separate local disk attached to the target system

3.2 Storage-as-a-Service Solution Configuration

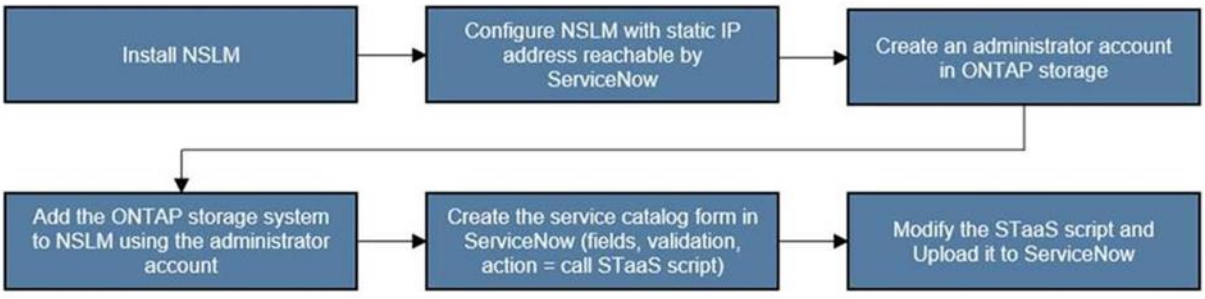
Figure 2 depicts the STaaS solution configuration.

Figure 2) STaaS solution configuration.



3.3 Implementation Workflow

Figure 3) STaaS workflow implementation.



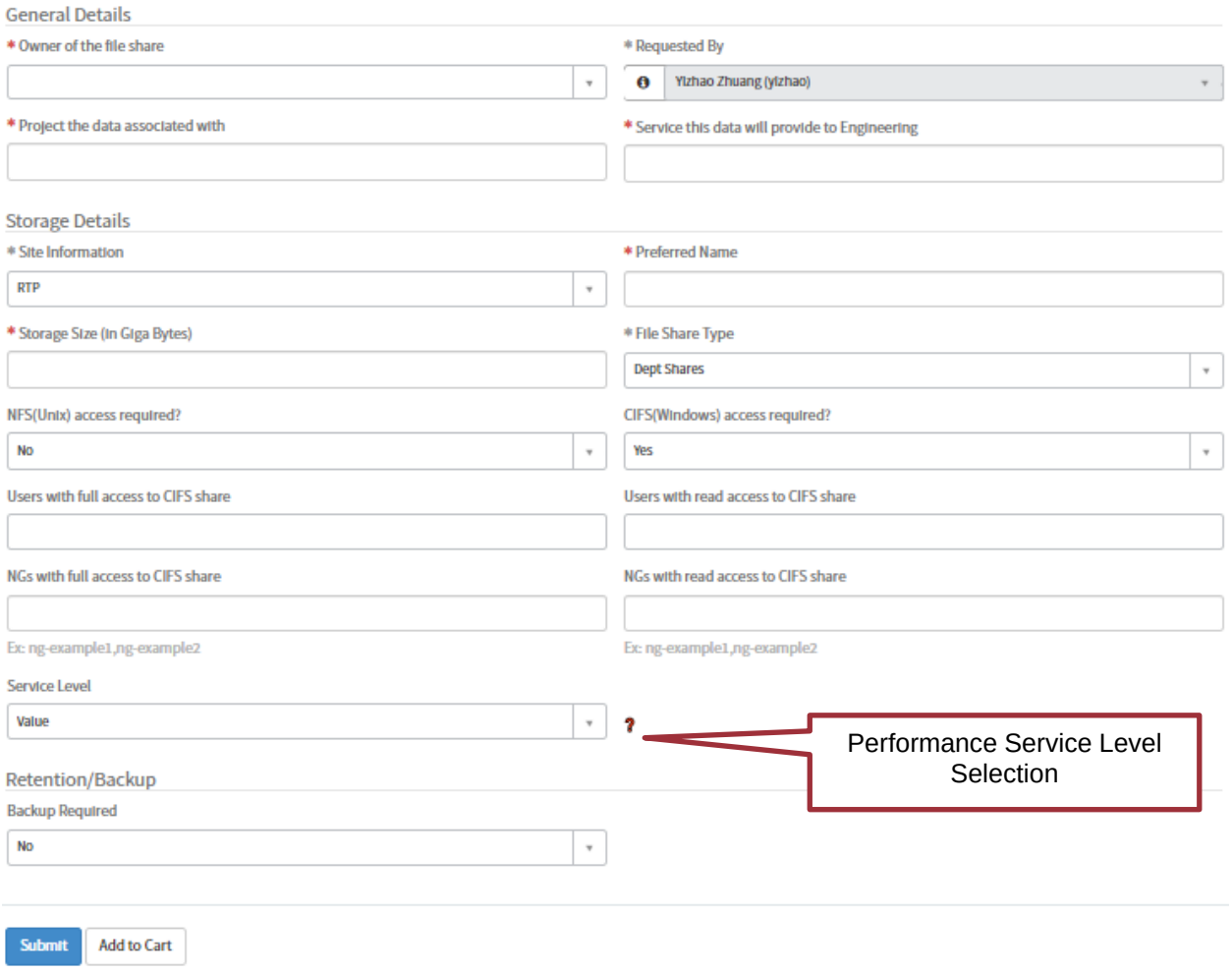
3.4 User Interface

For this solution, a ServiceNow interface builder was used to create a simple, web-based request form in the ServiceNow application. Although most of the fields are self-explanatory and can be customized to meet your environment needs, these two fields below require further explanation:

- **File Share Type.** Determines the type of file share:
  - TestBed, with support for NFS file shares only
  - Dept shares, with support for both NFS and CIFS file shares
- **Service Level.** Determines the service level: either value, performance, or extreme. Table 1 describes the differences between each level.

STaaS automatically scales performance based on the volume size, dynamically maintaining the ratio of IOPS to TBs as the size of the volume changes. Users are notified when a volume is noncompliant with the current service level.

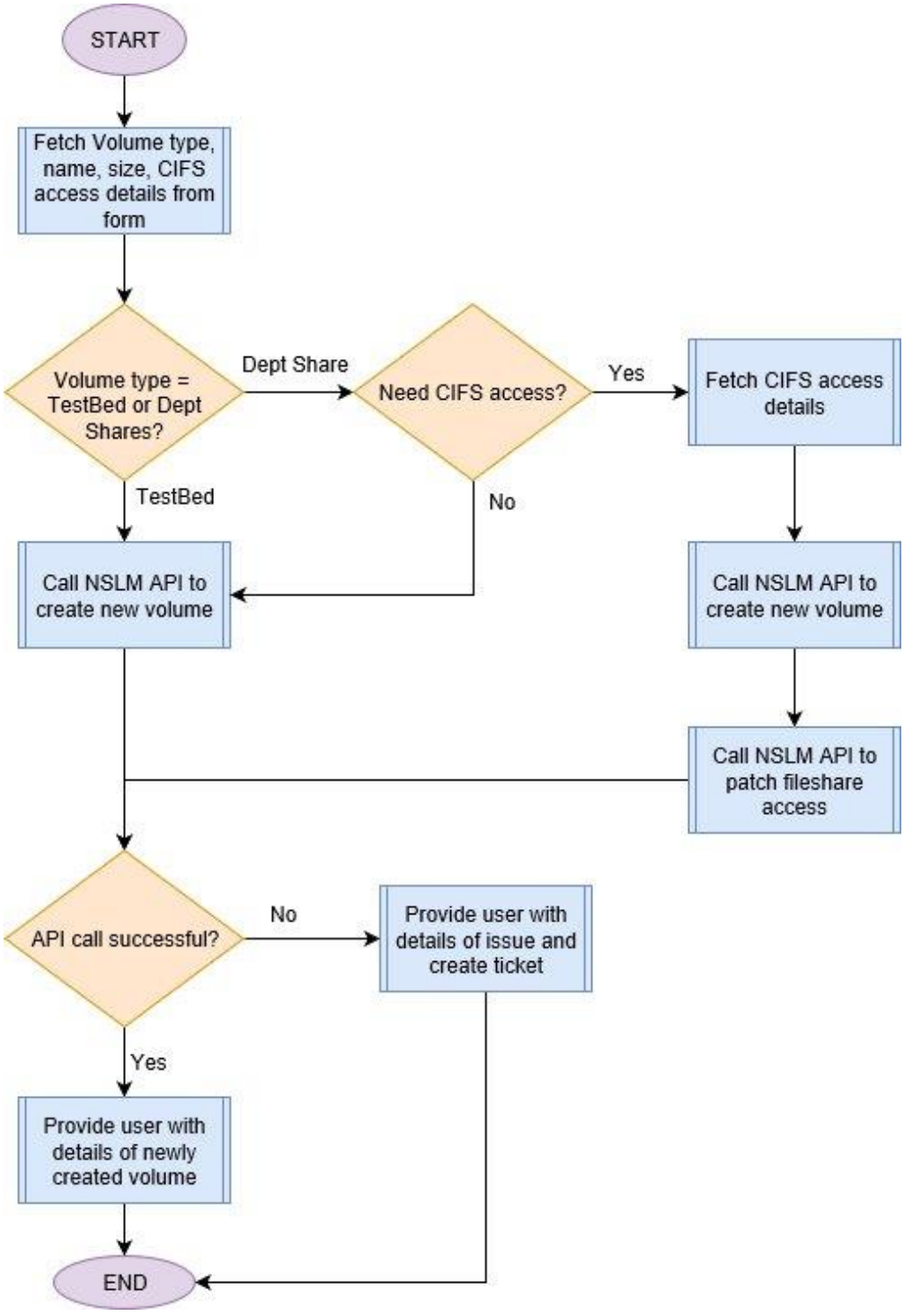
Figure 4) ServiceNow Service Portal form.



3.5 Storage-as-a-Service Script Workflow

Figure 5 illustrates the STaaS script workflow.

Figure 5) STaaS Script workflow.



3.6 Storage-as-a-Service Sample Code

The following code example demonstrates how to configure a STaaS system:

```
//function to create a file-shares
function create_fileshare(name,size,mount_point,svm_key){
	var output;
	try {
		var r = new sn_ws.RESTMessageV2('NSLM api', 'Create new fileshare');
		r.setStringParameterNoEscape('name', name);
		r.setStringParameterNoEscape('size',size);
		r.setStringParameterNoEscape('mount_point', mount_point);
		r.setStringParameterNoEscape('svm_key', svm_key);

		var response = r.execute();
		var responseBody = response.getBody();
		gs.log("NSLM create_fileshare response:"+response.getBody);
		var httpStatus = response.getStatusCode();
		var obj = JSON.parse(responseBody);
		if (httpStatus!=202){
			output="Failed to invoke file share creation:\nStatus code-
"+httpStatus+"\nError-"+obj.error.message+"\nTarget:"+obj.error.target;
			return output;
		}
		else{
			var job_key = obj.job_key;
			gs.log("NSLM job key:"+job_key);
			output = wait_for_job(job_key, 300);
			return output[0];
		}
	}
	catch(ex) {
		var message = ex.getMessage();
		gs.log("NSLM ERROR:"+message);
		return message;
	}
}

function patch_fileshare_access(access,fileshare_key){
	var output;
	try {
		var r = new sn_ws.RESTMessageV2('NSLM api', 'CIFS access');
		r.setEndpoint("https://nslmoldevnb.nb.openeng.netapp.com/api/storage-
provider/file-shares/"+fileshare_key);
		r.setStringParameterNoEscape('access', access);

		var response = r.execute();
		var responseBody = response.getBody();
		gs.log("NSLM patch_fileshare_access response:"+response.getBody);
		var httpStatus = response.getStatusCode();
		var obj = JSON.parse(responseBody);
		if (httpStatus!=202){
			output="Failed to patch fileshare access:\nStatus code-
"+httpStatus+"\nError-"+obj.error.message+"\nTarget:"+obj.error.target;
			return output;
		}
		else{
			var job_key = obj.job_key;
			gs.log("NSLM job key:"+job_key);
			output = wait_for_job(job_key, 300);
			return output[0];
		}
	}
	catch(ex) {
		var message = ex.getMessage();
		gs.log("NSLM ERROR:"+message);
		return message;
	}
}
```

```
function create_fileshare_cifs(name,size,mount_point,svm_key,access){
	var output;
	try {
		var r = new sn_ws.RESTMessageV2('NSLM api', 'Create new fileshare');
		r.setStringParameterNoEscape('name', name);
		r.setStringParameterNoEscape('size',size);
		r.setStringParameterNoEscape('mount_point', mount_point);
		r.setStringParameterNoEscape('svm_key', svm_key);

		var response = r.execute();
		var responseBody = response.getBody();
		gs.log("NSLM create_fileshare response:"+response.getBody);
		var httpStatus = response.getStatusCode();
		var obj = JSON.parse(responseBody);
		if (httpStatus!=202){
			output="Failed to invoke file share creation:\nStatus code-
"+httpStatus+"\nError-"+obj.error.message+"\nTarget:"+obj.error.target;
			return output;
		}
		else{
			var job_key = obj.job_key;
			gs.log("NSLM job key:"+job_key);
			output = wait_for_job(job_key, 300);
			if(output[0]==0){
				var fileshareKey = output[1].value;
				output=patch_fileshare_access(access,fileshareKey);
			}
			return output;
		}
	}
	catch(ex) {
		var message = ex.getMessage();
		gs.log("NSLM ERROR:"+message);
		return message;
	}

}


function wait_for_job(job_key, max_time_wait){
	var time_to_wait = max_time_wait;
	while (time_to_wait > 0){
		var r = new sn_ws.RESTMessageV2('NSLM api', 'Get job status');
		r.setEndpoint("https://nslmoldevnb.nb.openeng.netapp.com/api/management-
server/jobs/"+job_key);
		var response = r.execute();
		var responseBody = response.getBody();
		var httpStatus = response.getStatusCode();
		if (httpStatus == 200){
			var obj = JSON.parse(responseBody);
			status = obj.status;
			if (status == "NORMAL"){
				return [0,obj.job_results[0]];
			}
			else{
				var err_msg ="File share creation could not be completed.\n";
				var i=0;

	while((obj.task_reports[i].status!="ERROR")&&(obj.task_reports[i].name!="FINISH")){
					i++;
				}
				if(obj.task_reports[i].status=="ERROR"){
					err_msg+="NAME : "+obj.task_reports[i].name+"\n";
					err_msg+="STATUS : "+obj.task_reports[i].status+"\n";
					err_msg+="DESCRIPTION :
"+obj.task_reports[i].description+"\n";
					err_msg+="FAILURE REASON :
"+obj.task_reports[i].failure_reason+"\n";
				}
				return [1,err_msg];
			}
```

```
                }
                else if (time_to_wait > 0){
                        gs.sleep(5000);
                        time_to_wait -= 10;
                }
                else{
                        return [1,"FAILED to fetch Job results"];
                }
        }
}


function get_access(user_list,access_level){
        var access="";
        users=user_list.split(",");
        user_name="";
        for(i=0;i<users.length;i++){
                if(i>0 && users[i]==users[0])
                        continue;
                else if(users[i]){
                        if(users[i].indexOf("ng-") == -1 && users[i].length==32){
                                var user_rec = new GlideRecord('sys_user');
                                user_rec.addQuery('sys_id',users[i]);
                                user_rec.query();
                                if(user_rec.next()){
                                        user_name=user_rec.user_name;
                                }
                        }
                        else{
                                user_name=users[i].trim();
                        }
                }
                if(access){

        access+=",{\"permission\":\""+access_level+"\",\"user_or_group\":\"netapp\\\\\"+user_name+
"\"}";
                        }
                        else{

        access+="{\"permission\":\""+access_level+"\",\"user_or_group\":\"netapp\\\\\"+user_name+"
\"}";
                                }
                        }
                        return access;
                }

                //Size of fileshare in MB
                var size = parseInt(parseFloat(current.variables.size_data)*1024);
                // name of the share to create
                var name = current.variables.vname_data;
                // owner of the share
                var owner = current.variables.owner;
                // type of volume from the form
                var vol_type = current.variables.vol_type;
                var output_msg="";
                var path="";
                var SVM_KEY="";
                var SVM_DATA_LIF="";
                var mount_point="";
                if (vol_type=="testbed"){
                        SVM_DATA_LIF="10.195.21.236";
                        SVM_KEY="d4b539c9-3760-11e8-8661-00a0985fbe1a:type=vserver,uuid=5fd5f71a-
376d-11e8-8661-00a0985fbe1a";
                        mount_point="/testbedN/"+name;
                        path="Automount path: /x/eng/testbedN/"+name+"\n";
                        path+="NFS path: "+SVM_DATA_LIF+":/"+name+"\n";
                        output = create_fileshare(name,size,mount_point,SVM_KEY);
                }
                if (vol_type=="deptshares"){
                        SVM_DATA_LIF="10.195.21.236";
```

```
                        SVM_KEY="d4b539c9-3760-11e8-8661-00a0985fbe1a:type=vserver,uuid=5fd5f71a-
376d-11e8-8661-00a0985fbe1a";
                        mount_point="/"+name;
                        path="Automount path: /x/eng/deptshares/"+name+"\n";
                        path+="NFS path: "+SVM_DATA_LIF+":/"+name+"\n";
                        if(current.variables.cifs_data=="Yes"){
                                var full_access=owner.toString();
                                if(current.variables.full_access.toString()!=""){
                                        full_access+=","+current.variables.full_access.toString();

                                }
                                if(current.variables.full_access_ng.toString()!=""){

        full_access+=","+current.variables.full_access_ng.toString();

                                }
                                var access_list=get_access(full_access,"FULL_CONTROL");
                                if(current.variables.read_access.toString()!=""){

        access_list+=","+get_access(current.variables.read_access.toString(),"READ");
                                }
                                if(current.variables.read_access_ng.toString()!=""){

        access_list+=","+get_access(current.variables.read_access_ng.toString(),"READ");
                                }
                                gs.log("NSLM access: "+access_list);
                                output =
create_fileshare_cifs(name,size,mount_point,SVM_KEY,access_list);
                        }
                        else{
                                output = create_fileshare(name,size,mount_point,SVM_KEY);
                        }
                }
                if(output!=0){
                        output_msg="File Share creation failed: "+output+"\n\n";
                        //create SN ticket for the issue
                        var inc = new GlideRecord("incident");
                        inc.initialize();
                        inc.applyTemplate("NSLM error");
                        inc.short_description="ERROR: New File share creation-"+name;
                        inc.caller_id=current.variables.requester;
                        inc.opened_by=current.variables.requester;
                        inc.description="Request Item: "+current.number+"\n"+output_msg;
                        inc.insert();
                        output_msg=output_msg+"Ticket-"+inc.number+" has been raised with
Engineering support regarding this issue."+"\n";
                }
                else{
                        if((vol_type=="deptshares")&&(current.variables.cifs_data=="Yes")){
                                path+="CIFS path: \\\\"+SVM_DATA_LIF+"\\"+name+"\n";
                        }
                        current.stage="delivery";
                        output_msg="Successfully created file share.\n"+path;
                }
                current.comments=output_msg;
```

4 Where to Find Additional Information

The NSLM and ServiceNow integrated STaaS solution provide simple, reliable, and fast deployment options for the IT generalist and convenience to the end users.

To learn more about the information that is described in this document, review the following documents and/or websites:

- NSLM Download
  https://mysupport.netapp.com/products/p/nslm.html

- NSLM Product Documentation Fieldportal Collection
  https://fieldportal.netapp.com/collections/855594
- NSLM Blogs
  https://blog.netapp.com/?s=nslm
- NSLM Demo Youtube Playlist
  https://www.youtube.com/playlist?list=PLdXI3bZJEw7kNlnmcs_9Xu9khgsDjDB45
- NSLM Introduction Youtube Video
  https://youtu.be/a_RqTyP6asc
- NSLM 1.2 Lab On Demand
  https://labondemand.netapp.com/lab/sl10533

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Trademark Information