Cohesity PowerShell

Software Design Document

May 27, 2018

Contents

[2 Introduction 2](#_Toc515212165)

[2.1 Purpose 2](#_Toc515212166)

[2.2 Scope 2](#_Toc515212167)

[3 System Overview 2](#_Toc515212168)

[4 System Architecture 2](#_Toc515212169)

[5 Data Design 2](#_Toc515212170)

[5.1 Data Description 2](#_Toc515212171)

[5.2 Data Dictionary 2](#_Toc515212172)

[6 Component Design 3](#_Toc515212173)

[6.1 CMDLET Component Design 3](#_Toc515212174)

[7 Human Interface Design 3](#_Toc515212175)

[7.1 CMDLET Interface Design 3](#_Toc515212176)

[7.2 Installation 4](#_Toc515212177)

# Introduction

## Purpose

This document is for internal use only and describes how the Cohesity PowerShell software is composed from a high-level. As such, the document is not meant to exhaustively specify the requirements or composition of the software but rather the basic composition and interaction of significant software components.

## Scope

Cohesity PowerShell provides a series of cmdlets which use the Cohesity model for managing Cohesity resources. Cohesity PowerShell is intended for use by Cohesity users and administrators.

The intended goal is to allow end-users to script common tasks using the Cohesity model with PowerShell.

# System Overview

Cohesity PowerShell has been built using Visual Studio 2015 and C#. The software is composed of two class libraries; one contains the Cohesity models and the other contains the PowerShell cmdlets.

# System Architecture

The cmdlets have been designed to allow for efficient scripting of common tasks. The is permitted with the use of shared connection and authentication information. By connecting to a Cohesity resource API once, the remainder of the script will utilize this known and authenticated connection.

The remainder of the cmdlets are designed to use recommended cmdlet design.

# Data Design

## Data Description

Cohesity PowerShell exposes the Cohesity model using plain-old-CLR-objects (POCOs) which are derived from the Cohesity resources Swagger documentation. These POCOs are used for queries and commands by cmdlets. Each cmdlet declares the POCO type when possible to allow for design-time scripting with tab completion and constraints.

## Data Dictionary

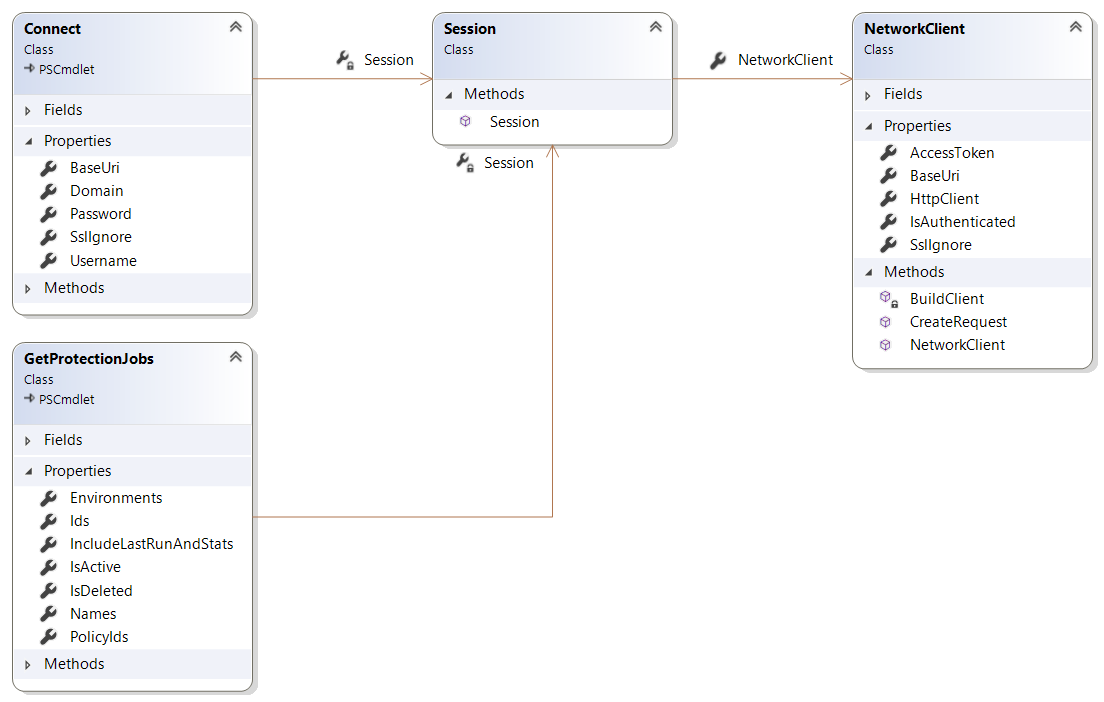
The Cohesity model is too large to document in its entirety however there is a one-to-one mapping of Cohesity resource model to PowerShell model.

# Component Design

## CMDLET Component Design

As described in the system architecture, each cmdlet shares a connection once connected to a Cohesity cluster. This session lives for the duration of a PowerShell session and is scoped to only that session. As seen in Figure 1, each cmdlet has access to an internal scoped session property. The Session class contains a NetworkClient property which is responsible for providing access to the HttpClient necessary to communicate with Cohesity resource APIs, properties to configure the HttpClient and a helper method for creating a HttpRequest meeting the requirements for communicating with the Cohesity resource API, specifically the authentication of requests.

Figure



# Human Interface Design

## CMDLET Interface Design

As is the common practice for cmdlet output to be stored in parameters to be iterated over and rationalized for further processing, so are the Cohesity PowerShell cmdlets. All outputs are in the form of objects which can be stored into variables for further processing.

## Installation

TODO: This is not yet implemented.

Installation of Cohesity PowerShell cmdlets will be in the form on an MSI built with the WIX Toolset.