SEIKO

運用ガイド
Operation Guide

SmartCS modules for Ansible

第6版
2023年10月13日
U00143054806

セイコーソリューションズ株式会社

U00143054801　2019 年　3 月
U00143054802　2019 年　10 月
U00143054803　2021 年　1 月
U00143054804　2021 年　3 月
U00143054805　2021 年　9 月
U00143054806　2023 年　10 月

目次

1　章　はじめに

1.1　SmartCS modules for Ansible について

SmartCS modules for Ansible は、レッドハット株式会社 から提供されている自動化ツールの Red Hat Ansible Automation Platform（以下, Ansible）を利用して、コンソールサーバ SmartCS を制御する為に必要なモジュールやプラグインを含んだパッケージの総称となります。

SmartCS を Ansible 経由で制御する事によって、SmartCS に接続されている機器の IP 設定などの初期構築作業を含むコンソール経由のオペレーションを Ansible 経由で実現できるようになり、運用の各フェーズにおいて IT インフラの運用負荷をさらに軽減する事が可能となります。

SmartCS modules for Ansible は、v1.3.0 より、Ansible Collections に対応しました。
本ドキュメントでは、Ansible Collections の提供形式に沿った内容を主に取り扱います。

※v1.0～v1.2 までは、弊社独自のパッケージ形式で提供していました。
　詳細については「12 章 付録 B. v1.0～v1.2 のオペレーション」、を参照ください。

本運用ガイドは、SmartCS modules for Ansible の使用方法や運用方法について記載したドキュメントとなります。

1.2 機能概要

SmartCS modules for Ansible は Ansible を使って SmartCS の設定や情報取得をするだけでなく、SmartCS のシリアルポートに接続されている機器のオペレーションについても行うことが可能です。

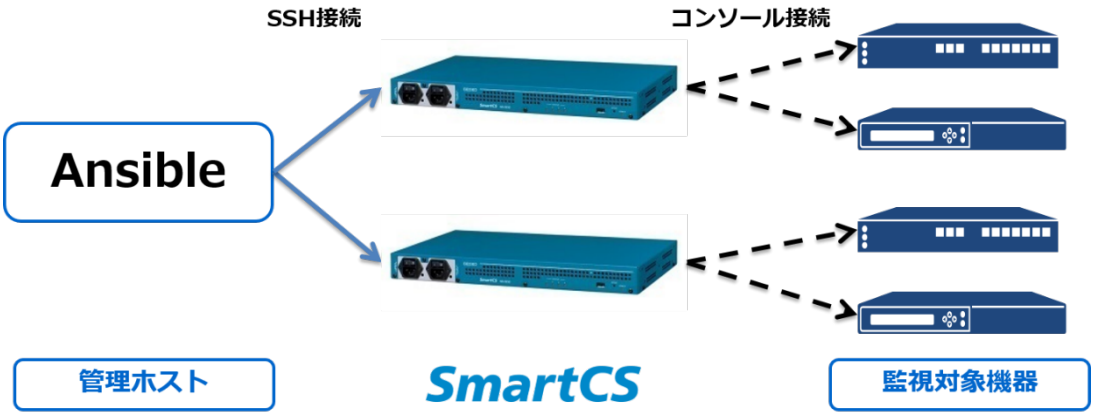
本ドキュメントでは、SmartCS に接続されている機器のオペレーションを実現できる機能を「コンソールアクセス機能」、SmartCS の設定や設定情報を取得する機能を「CLI コマンド機能」としています。
また、コンソールアクセス機能はネットワーク機器ベンダーから提供されている Ansible モジュールを SmartCS 経由で動作させる事も可能となっています。

SmartCS modules for Ansble が提供する機能と、モジュール一覧は以下の表の通りとなります。

機能名	モジュール名
コンソールアクセス機能	seiko.smartcs.smartcs_tty_command
CLI コマンド機能	seiko.smartcs.smartcs_command
	seiko.smartcs.smartcs_config
	seiko.smartcs.smartcs_facts

1.2.1 コンソールアクセス機能の概要

SmartCS に SSH ログイン後、CLI コマンドを実行する事で SmartCS のシリアルポートに接続されている機器のコンソールに対して文字列の送受信を行います。



従来は、Telnet または SSH プロトコルを使い、SmartCS のシリアルポートに接続して直接ネットワーク機器のコンソールオペレーションを実行していましたが、Ansible を使う事で、手動で行っていたオペレーション内容を Playbook に記述して実行する事が可能となります。

本機能に対応した Ansible モジュールは以下となります。

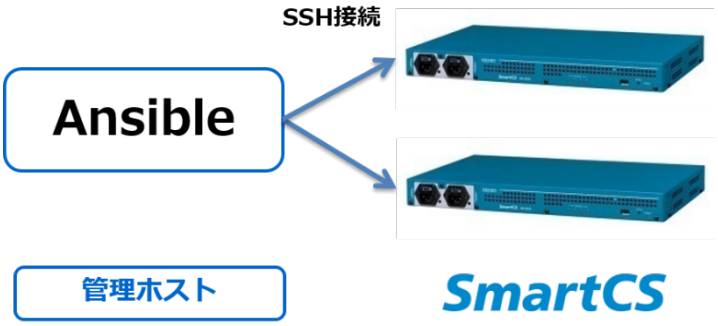
機能名	モジュール名
コンソールアクセス機能	seiko.smartcs.smartcs_tty_command
	ネットワーク機器ベンダーのモジュールと seiko.smartcs.smartcs_tty_command を連携

smartcs_tty_commnad モジュールを使う事で、Ansible モジュールのないネットワーク機器、IP 到達性のないネットワーク機器についても Ansible 経由でのオペレーションが可能となります。具体的な使い方については、「5 章 コンソールアクセス機能」を参照してください。

また、ネットワーク機器ベンダーの提供しているモジュールと連携して SmartCS に接続されている機器の操作を行う事も可能です。具体的な使い方については、「6 章 ネットワーク機器ベンダーのモジュールと連携する」を参照してください。

1.2.2 CLI コマンド機能の概要

SmartCS の CLI コマンドを Ansible 経由で実行する機能となります。



本機能に対応する Ansible モジュールは以下となります。

機能名	モジュール名
CLI コマンド機能	seiko.smartcs.smartcs_command seiko.smartcs.smartcs_config seiko.smartcs.smartcs_facts

CLI コマンド機能の各モジュールが提供する機能概要は以下となります。

(1) smartcs_command
SmartCS 上で任意の状態表示コマンド、メンテナンスコマンドを実行し、その結果を取得します。

(2) smartcs_config
SmartCS 上で任意の設定コマンドを実行します。

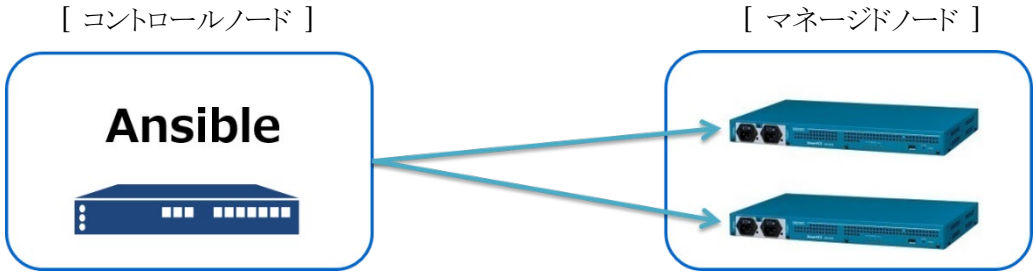
(3) smartcs_facts
SmartCS から装置モデルやソフトウェアのバージョン、コンフィグレーション情報などの装置情報を取得します。

具体的な使い方については、「7 章 CLI コマンド機能」を参照してください。

1.3　動作要件

1.3.1 コントロールノード・マネージドノード

本ドキュメントで使う文言についての説明となります。
コントロールノードは Ansible をインストールして動作させる環境となります。
マネージドノードは、SmartCS となり、Ansible から SmartCS にアクセスする際に、SSHv2 によるログインに対応している必要があります。



※コントロールノードとなる Ansible から見た場合、あくまでマネージドノードは SmartCS となります。
（SmartCS に接続している機器ではありません）

1.3.2 動作環境

コントロールノードである Ansible とマネージドノードとなる SmartCS の各動作要件、組み合わせについては以下の表の通りとなります。

＜v1.0 ～ v1.2＞:弊社独自のパッケージで提供しています。

SmartCS modules for Ansible	コントロールノード環境		マネージドノード環境 SmartCS ソフトウェア Ver	
	ansible	Python	NS-2250 series	NS-2240 series
v1.0	2.7.7	2.7 以降 3.6 以降	v2.0 以降	未サポート
v1.1 v1.1.1	2.8.4		v2.1 以降	
v1.2	2.9.15	3.6.8	v2.1 以降	

＜v1.3.0～＞:Ansible Collections 用のパッケージで提供しています。

SmartCS modules for Ansible	コントロールノード環境	マネージドノード環境 SmartCS ソフトウェア Ver	
	ansible	NS-2250 series	NS-2240 series
v1.3.0	2.10.x (>=2.10, < 2.11)	v2.1 以降	未サポート
v1.4.0	ansible 2.9.22～ ansible-base 2.10.x ansible-core 2.11.x (>=2.9.22, < 2.12)		
v1.4.1	ansible 2.9.10～ ansible-base 2.10.x ansible-core 2.11.x (>=2.9.10, < 2.12)		
v1.5.0	ansible-core 2.13.x ansible-core 2.14.x ansible-core 2.15.x (>=2.13, < 2.16)		

SmartCS modules for Ansible は、各バージョンに合ったコントロールノード環境、マネージドノード環境の組み合わせで動作させて下さい。

＜補足＞

(1) ansible-core2.11 環境で v1.3.0 を動作させた場合など、実行環境の ansible バージョンと SmartCS modules for Ansible のバージョンの組み合わせによっては、Playbook 実行時に以下のような Warning が出力されます。

```
[DEPRECATION WARNING]: Ansible will require Python 3.8 or newer on
the controller starting with Ansible 2.12.
```

ansible-core 2.12 以降で Python3.8 環境が必須となる為に出力される警告ですが、動作に影響はありません。

ansible.cfg に以下の設定を投入する事で出力しなくなります。

```
deprecation_warnings = False
```

1.3.3 Ansible 環境

Ansible はコントロールノード環境の Python にインストールする事で動作可能となります。Python の仮想技術である venv を使うと、コントロールノード上で動作する Python に影響を与えることなく Ansible 環境が構築可能となりますので venv による Ansible 環境の構築を推奨しています。

venv を利用した Ansible 環境の構築は「11 章 付録 A. Ansible 環境の構築」を参照してください。

1.4 ライセンス

SmartCS modules for Ansible のライセンスは、GNU General Public License Version3（以下, GPLv3）となります。

GPLv3 の詳細については「15.1 第三者ソフトウェアライセンス」、または、SmartCS modules for Ansible のパッケージに含まれる、LICENSE ファイルを参照して下さい。

また、GitHub に公開されているソースから Ansible Collections 用のパッケージを作成する手順については「15.2 Ansible Collections パッケージの作成方法」を参照して下さい。

SmartCS modules for Ansible は、Ansible を改変したプログラムを含んでいます。

1.5 SmartCS modules for Ansible の入手

SmartCS modules for Ansible は、3つの入手方法があります。

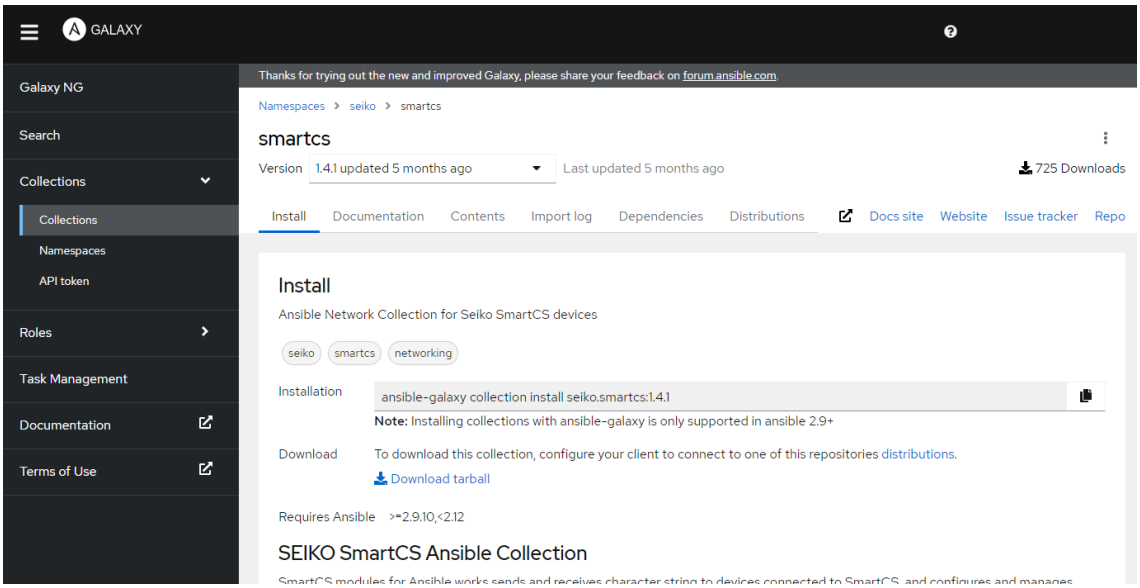
(1) Ansible Galaxy サイトから入手

SmartCS 用の Ansible Collections パッケージは、Ansible Galaxy サイトより入手が可能です。

https://galaxy.ansible.com/

「seiko」や「smartcs」といったキーワードを入力する事で、SmartCS 用 Ansible Collections パッケージのページが見つかります。

https://galaxy.ansible.com/ui/repo/published/seiko/smartcs/



[Install] タブの Installation に記載されているコマンドをコントロールノードから実行する事で、ダウンロード、インストールが可能となります。

詳細な手順については、「2 章 インストール」を参照してください。

(2) Ansible Automation Hub サイトから入手

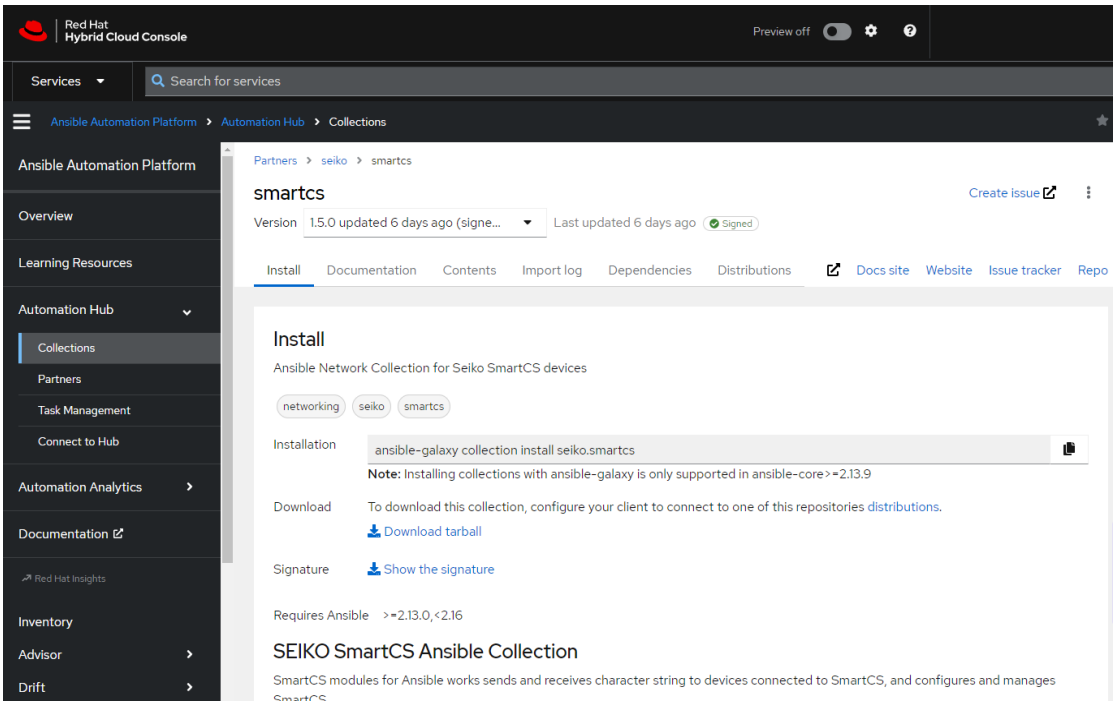
SmartCS 用の Ansible Collections パッケージは、Red Hat 社 及び 認定パートナーのコンテンツを配布するサイトである、Ansible Automation Hub からも入手が可能です。

Ansible Automation Hub は、Red Hat Ansible Automation Platform サブスクリプションの一部となっており、サポート契約を有するお客様のみがアクセス可能なサイトとなっています。

https://www.ansible.com/products/automation-hub

「seiko」や「smartcs」といったキーワードを入力する事で、SmartCS 用 Ansible Collections パッケージのページが見つかります。

https://console.redhat.com/ansible/automation-hub/repo/published/seiko/smartcs



[Install]タブの Installation に記載されているコマンドをコントロールノードから実行する事で、ダウンロード、インストールが可能となります。

詳細な手順については、「2 章 インストール」を参照してください。

2　章　インストール

2.1　インストール前の確認

Ansible 環境を構築するホストマシンに、Ansible がインストールされている事を確認して下さい。もしインストールされていなければ、CentOS などの場合、yum コマンドや pip コマンド等でインストールする事が可能です。
Ansible 環境の構築方法や、ansible.cfg ファイルの用意については、「11 章 付録 A. Ansible 環境の構築」を参照してください。

本章では、SmartCS modules for Ansible の v1.3.0 からサポートしている Ansible Collections の形式でインストールする手順について説明します。

v1.0～v1.2 の弊社独自のパッケージで提供していたバージョンのインストール手順については、「12 章 付録 B. v1.0～v1.2 のオペレーション」を参照してください。

＜提供形式とインストール手順＞

SmartCS modules for Ansible	提供形式	インストール手順
v1.0	弊社独自のパッケージ	「12 章 付録 B. v1.0～v1.2 のオペレーション」
v1.1 v1.1.1		
v1.2		
v1.3.0～	Ansible Collections 形式	「2 章 インストール」 （本章）

2.2　インストール

Ansible Collections 形式に対応した SmartCS modules for Ansible は、以下のファイル名で提供します。

ファイル名形式	備考
seiko-smartcs-x.y.z.tar.gz	例 v1.3.0 の場合 seiko-smartcs-1.3.0.tar.gz

※バージョンについては、Ansible バージョンとの関連性は特にありません。
弊社のリリースルールに沿ったバージョンを付与して提供しています。

また、ネームスペース、コレクション名は、以下となります。

Namespace （ネームスペース名）	seiko
Collection（コレクション名）	smartcs

本項では、Ansible Collections の形式で SmartCS modules for Ansible をインストールする手順について、弊社で動作確認ができている内容について説明します。
venv 環境に Ansible を構築した場合は、構築した venv 環境に遷移してからそれぞれの手順を実行して、インストールを行ってください。

インストール手順は、以下 3 つの方法となります。
インストール後の動作についてはどのインストール手順であっても変わりません。

(1) SmartCS modules for Ansible をダウンロード＆インストール
一番簡単な構築手順となります。コントロールノードから、Ansible Galaxy または Ansible Automation Hub サイトへのアクセスが可能な場合は本手順でのインストールを推奨します。

(2) ファイルを指定してインストール
Ansible Collections ファイルを直接指定してインストールを行う手順となります。弊社より SmartCS 用 Ansible モジュールを直接入手した場合は、こちらの手順でインストールを行います。

(3) requirements.yml を使ったインストール
requirements.yml というファイルを利用したインストール手順となります。ファイル内に各モジュールのバージョンや取得先を指定する事で、構築に使う各パッケージの環境を固定する事ができます。

(1) SmartCS modules for Ansible をダウンロード＆インストール

SmartCS modules for Ansible は、Ansible Galaxy のサイト
https://galaxy.ansible.com/

または Ansible Automation Hub サイトで公開しています。
https://www.ansible.com/products/automation-hub

Ansible Automation Hub サイトからダウンロード・インストールする場合は、事前に Automation Hub にアクセスして認証トークンを取得し、その内容を ansible.cfg に設定する必要があります。
詳細につきましては、以下のドキュメントを参照ください。
https://docs.ansible.com/ansible/latest/galaxy/user_guide.html#downloading-a-collection-from-automation-hub

以降の各オペレーションについて、Ansible Galaxy、Ansible Automation Hub のどちらから SmartCS modules for Ansible をダウンロード・インストールした場合でも内容については同様となります。

ansible 2.10 以降の環境の場合、ansible-galaxy コマンドを使って SmartCS modules for Ansible をインストールします。

```
$ ansible-galaxy collection install seiko.smartcs
$
```

インストールされた SmartCS modules for Ansible のバージョンを確認します。

```
$ ansible-galaxy collection list

# /home/test/xxx/xxx/ansible_collections
Collection    Version
------------- -------
seiko.smartcs 1.3.0
$
```

(2) ファイル名を指定してインストール

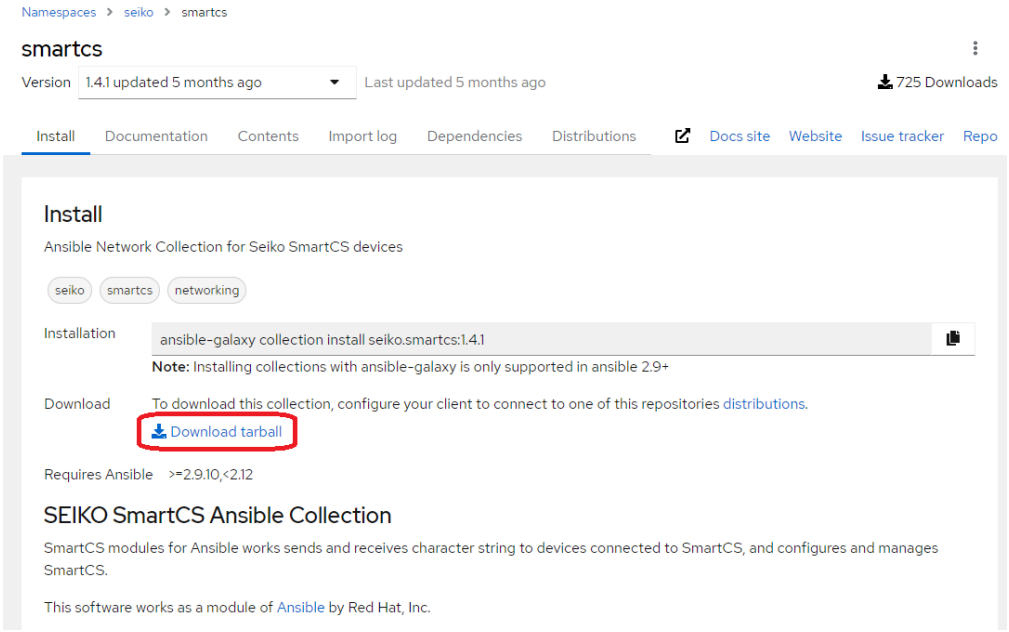
SmartCS 用の Ansible Collections パッケージは、以下の方法でファイルのダウンロード、入手が可能です。

① Ansible Galaxy、Ansible Automation Hub サイトから、ansible-galaxy コマンド使ってダウンロードする。

```
$ ansible-galaxy collection download seiko.smartcs
$ ls
$ seiko-smartcs-1.3.0.tar.gz
```

※ダウンロード先は、ansible.cfg の collections_paths で指定したディレクトリとなります。

② Ansible Galaxy、Ansible Automation Hub サイトから、Web ブラウザ経由でファイルをダウンロードする。
（以下の例は Ansible Galaxy の場合となります）



ダウンロードした SmartCS 用 Ansible Collections パッケージを、コントロールノードにコピー後、ansible-galaxy コマンドを使ってインストールします。

```
$ ansible-galaxy collection install seiko-smartcs-1.3.0.tar.gz
$
```

インストールされた SmartCS modules for Ansible のバージョンを確認します。

```
$ ansible-galaxy collection list

# /home/test/xxx/xxx/ansible_collections
Collection    Version
------------- -------
seiko.smartcs 1.3.0
$
```

(3) requirements.yml を使ったインストール

requirements.yml というファイルを用意してインストールを行うことができます。

```
---
collections:
  -   name: seiko.smartcs
      version: 1.3.0
```

(requirements.yml の例)

requirements.yml には、複数のモジュールを記載する事ができるので、Ansible Collections を使った環境を管理する場合に便利な仕組みとなっています。

requirement.yml を指定し、ansible-galaxy コマンドを使ってインストールをします。

```
$ ansible-galaxy collection install -r requirement.yml
$
```

インストールされた SmartCS modules for Ansible のバージョンを確認します。

```
$ ansible-galaxy collection list

# /home/test/xxx/xxx/ansible_collections
Collection    Version
------------- -------
seiko.smartcs 1.3.0
$
```

2.3　バージョンアップ

ansible-galaxy コマンドを使って、インストールした Ansible Collections のパッケージをバージョンアップする場合「--upgrade」オプション または、「--force」オプションのどちらかを使う事で、インストール済みのパッケージをバージョンアップする事が可能です。

(1) --force オプションを使う場合

```
$ ansible-galaxy collection install seiko.smartcs --force
$
```

既に AnsibleCollections パッケージがインストール済みの環境で、同じモジュールをインストール、ダウンロードすると通常はエラーとなりますが、各操作の実行時に--force オプションを付与する事で、上書きインストールが可能となります。

本オプションは、「-r requirements.yml」を使ったインストールの場合にも有効なオプションとなります。

```
$ ansible-galaxy collection install -r requirement.yml --force
$
```

(2) --upgrade オプションを使う場合（ansible-core2.11 以降）

```
$ ansible-galaxy collection install seiko.smartcs --upgrade
$
```

ansible-core2.11 からは、--upgrade オプションが追加となりました。本オプションを使ってモジュールのバージョンアップを行う事も可能です。

2.4　インストールしたコレクションファイルの削除

ansible-galaxy コマンドでは、インストールした AnsibleCollection パッケージを削除する為の専用コマンドは用意されていませんが、実際にインストールした Ansible Collections ファイルが格納されているフォルダを削除する事で、削除が可能です。

Ansible Collections パッケージがインストールされるディレクトリは、ansible.cfg で collections_paths というパラメータで指定が可能となっています。

https://docs.ansible.com/ansible/latest/reference_appendices/config.html#collections-paths

2.5　依存パッケージについて

SmartCS modules for Ansible は、コネクションプラグインとして、network_cli を使用します。その為、依存パッケージとして、ansible.netcommon を設定しています。

インストール時（ansible-galaxy collection install コマンド実行時）に network_cli コネクションプラグインを提供している、ansible.netcommon コレクションがコントロールノードにインストールされていない場合は、ansible.netcommon コレクションも合わせてインストール処理が行われます。

2.6 その他

その他、ansible-galaxy コマンドを使った Ansible Collections パッケージの管理方法について説明します。

(1) 特定のバージョンを指定してインストール(ダウンロード)する。

ansible-galaxy コマンドを使い、seiko.smartcs などと、ネームスペースとコレクション名を指定してインストールやダウンロードを行う場合、通常 Ansible Galaxy に登録されているバージョンの最新版が自動的にインストールされます。

特定のバージョンを指定したい場合、コレクション名の後に:(コロン)とバージョン名を指定する事で特定のバージョンのインストールやダウンロードが可能です。

```
$ ansible-galaxy collection install seiko.smartcs:1.4.0
$
```

また、バージョン名以外の識別子が付与されたパッケージについても同様の手順でインストール、ダウンロードが可能です。

```
$ ansible-galaxy collection install seiko.smartcs:1.4.0-dev1
$
```

requirements.yml で指定する場合、version は以下のように指定します。

```
---
collections:
  - name: seiko.smartcs
    version: 1.3.0
```

3　章　準備

3.1　SmartCS の準備

コンソールアクセス機能及び CLI コマンド機能のどちらを使う時にも必要な SmartCS の設定について下記に記載します。

(1) 端末出力制御の設定

SmartCS modules for Ansible が提供する各モジュールを正しく動作させる為、端末出力制御設定については下記の様に設定して下さい。

```
(0)NS-2250# set terminal default prompt device on
(0)NS-2250# set terminal default prompt hostname on
```

※上記設定は端末出力設定のデフォルト値となります。

3.2　コントロールノードの準備

コンソールアクセス機能及びCLIコマンド機能のどちらを使う時にも必要なAnsibleを実行するコントロールノード側での準備について下記に記載します。

(1) SSHホスト公開鍵の登録

Playbookを実行する前に、あらかじめコントロールノードからSmartCSにSSHログインを行い、SmartCSのホスト公開鍵をコントロールノードに登録して下さい。

```
[testuser@ansible-host ~]$ ssh smartcs
The authenticity of host 'smartcs (172.31.8.16)' can't be established.
ECDSA key fingerprint is SHA256:/DieiZVP5ggJlupmTPqj/djKRfVRhmhzBPLHZ20jNZ8.
ECDSA key fingerprint is MD5:98:ea:d9:8b:aa:bd:af:13:56:7c:62:ee:7c:6c:d7:61.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'smartcs,172.31.8.16' (ECDSA) to the list of known hosts.
Console Server Authentication.
```

または、ansible.cfg の設定で、ホスト鍵の登録のないマネージドノードにアクセスする場合のチェックを無効にして下さい。

デフォルトでは、下記の"host_key_checking"はコメントアウトされており、ホスト鍵認証が有効となっています。
コメントアウトした場合、SSH接続時のホスト鍵認証を行わなくなります。

```
# uncomment this to disable SSH key host checking
host_key_checking = False
```

4　章　Ansible Collections に対応した Playbook の作成

4.1　モジュールの指定

Ansible Collections の仕組みが標準となった Ansible2.10 以降、Playbook 作成におけるモジュール名の指定方法が変わりました。
Ansible Collections に対応した Playbook の作成方法について説明します。

(1) 従来(v1.0～v1.2)の指定

tasks 部分のモジュール指定について、SmartCS modules for Ansible v1.0～1.2 までは以下のような指定となっていました。

```
tasks:
- smartcs_command:
    commands:
    - show version
    - show config running
```

(2) FQCN 形式での指定

Ansible Collections の仕組みに対応した SmartCS modules for Ansible v1.3.0 以降では、モジュール名について Full Qualified Collection Name 形式(完全修飾コレクション名，以下 FQCN)での指定が必要となります。

SmartCS modules for Ansible は、Namespace が seiko、Collection が smartcs となりますので、seiko.smartcs.モジュール名、という指定方法となります。
smartcs_command モジュールを指定する場合の Playbook 例(task 部分のみ)は以下のような内容となります。

```
tasks:
- seiko.smartcs.smartcs_command:
    commands:
    - show version
    - show config running
```

SmartCS modules for Ansible が提供しているモジュールについて、それぞれ FQCN 形式での指定方法は以下の内容となります。

v1.0～v1.2 までの指定	v1.3.0 以降の指定（FQCN 形式） ※Ansible Collections 形式の指定
smartcs_tty_command	seiko.smartcs.smartcs_tty_command
smartcs_command	seiko.smartcs.smartcs_command
smartcs_config	seiko.smartcs.smartcs_config
smartcs_facts	seiko.smartcs.smartcs_facts

本ドキュメントでは、基本的に FQCN 形式で各モジュール名を記載します。
※一部画像等については v1.0～v1.2 までの指定方法となっている箇所が
ありますがご了承下さい。

(3) collections ディレクティブ形式での指定

Playbook 内で、collections ディレクティブを利用する事で、従来(v1.0～v1.2)のモジュール名の指定や、エイリアス(別名)での指定が可能となります。
SmartCS modules for Ansible で提供するモジュールを、collections ディレクティブで指定する場合、以下のような指定となります。

```
- name: "Collect the default facts "
  hosts: smartcs

  collections:
  - seiko.smartcs

  tasks:
  - name: "run smartcs_facts with default"
    facts:
      gather_subset:
      - default
```

collections ディレクティブを利用した場合に指定可能となるモジュール名については、以下の表の通りとなります。

v1.0～v1.2 までの指定	collections ディレクティブを指定した場合のモジュール名指定
smartc_tty_command	seiko.smartcs.smartcs_tty_command smartcs_tty_command tty_command
smartcs_command	seiko.smartcs.smartcs_command smartcs_command command
smartcs_config	seiko.smartcs.smartcs_config smartcs_config config
smartcs_facts	seiko.smartcs.smartcs_facts smartcs_facts facts

4.2　コネクションプラグイン（network_cli）の指定

SmartCS modules for Ansible はコネクションプラグインとして、network_cli を利用しています。Ansible Collections の仕組みが標準となった Ansible2.10 以降、network_cli コネクションプラグインは、ansible.netcommon コレクションが提供する機能となりますので、Playbook 作成におけるコネクションプラグインとそのオプション値についても FQCN 形式での指定が必要となります。
参考：https://galaxy.ansible.com/ui/repo/published/ansible/netcommon/

それぞれ FQCN 形式での指定方法は以下の内容となります。

v1.0～v1.2 までの指定 （～ansible2.9 系までの指定）	v1.3.0 以降の指定（FQCN 形式） ※Ansible Collections（ansible2.10～）形式の指定
ansible_connection: network_cli	ansible_connection: ansible.netcommon.network_cli
ansible_network_os: smartcs	ansible_network_os: seiko.smartcs.smartcs
ansible_become_method: enable	ansible_become_method: ansible.netcommon.enable

Playbook 例は以下の内容となります。

```
- name: "run show version on SmartCS "
  hosts: smartcs

  tasks:
  - name: "run smartcs_command"
    seiko.smartcs.smartcs_command:
      commands: show version

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: testuser01
  - ansible_password: testpassword01
  - ansible_become_method: ansible.netcommon.enable
  - ansible_become: yes
  - ansible_become_password: "¥n"
```

4.3　コネクションプラグイン（network_cli）のオプション指定について

network_cli コネクションプラグインは、内部で paramiko という Python のライブラリを利用して SmartCS などのネットワーク機器に接続をしています。
ansible-core2.13 を内包した Ansible6.0.0 以降、paramiko に加えて ansible-pylibssh というライブラリが追加となり、Playbook 実行時にどちらかを選択して動くように機能が拡張されました。（セキュリティの観点から pylibssh が推奨とされています）
参考：
https://docs.ansible.com/ansible/latest/collections/ansible/netcommon/network_cli_connection.html#parameter-ssh_type
デフォルトの動作は auto 指定（ansible 実行環境に ansible-pylibssh がインストールされている場合は、pylibssh を指定、なければ paramiko を指定）となります。（2023 年 9 月 ansible-core 2.15.3 時点）

Playbook 例は以下の内容となります。

```
- name: "run show version on SmartCS "
  hosts: smartcs

  tasks:
  - name: "run smartcs_command"
    seiko.smartcs.smartcs_command:
      commands: show version

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_cli_ssh_type: libssh
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: testuser01
  - ansible_password: testpassword01
```

また、Playbook 実行時に以下のようなワーニングが出力されている場合

```
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
```

ansible 実行環境に、ansible-pylibssh がインストールされておらず、paramiko を指定して動作したという内容のワーニングとなります。以下のコマンド等を実行して、ansible 実行環境に ansible-pylibssh をインストールする事でワーニングは解消されます。

```
$pip3 install ansible-pylibssh
```

※11 章 11.1 項 venv による Ansible 環境の構築 もご参照下さい。

5　章　コンソールアクセス機能

5.1　SmartCS の準備

本項では、smartcs_tty_command モジュールを使ったコンソールアクセス機能について説明します。
ネットワーク機器ベンダーの Ansible モジュールと連携して SmartCS に接続されている機器の操作を行う場合の説明については、「6 章 ネットワーク機器ベンダーのモジュールと連携する」を参照してください。

(1) バージョンの確認

SmartCS のシステムソフトウェアが v2.1 以上である事を確認して下さい。

```
(0)NS-2250# show version
System                : System Software Ver 2.1 (Build 2019-MM-DD)
 :

(0)NS-2250#
```

(2) SSH 接続の有効化

SSH サーバが有効となっているかを確認して下さい。
また、デフォルトで公開鍵認証となっているのでパスワード認証に設定して下さい。

```
(0)NS-2250# enable sshd
(0)NS-2250# set sshd auth basic
(0)NS-2250#
```

フィルター機能が有効化されている場合や、特定のホストからの接続許可設定をしている場合、コントロールノード PC からの SSH 接続を許可するように設定して下さい。

(3) 拡張ユーザグループのユーザ作成とコンソールアクセス権限の付与

コンソールアクセス機能を利用する為には、拡張ユーザグループに所属するユーザを作成する必要があります。その後、作成したユーザに対して tty マネージ機能とポートアクセスの権限を付与する設定が必要です。

```
(0)NS-2250# create user smartcs-ansible group extusr password
Changing password for user smartcs-ansible.
New password:
Retype new password:
Password for smartcs-ansible changed
(0)NS-2250# set user smartcs-ansible permission ttymanage on
(0)NS-2250# set user smartcs-ansible port 1-48
(0)NS-2250#
```

(4) コンソールアクセス機能の有効化

コンソールアクセス機能を有効化する為の設定を行います。

```
(0)NS-2250# enable ttymanage
(0)NS-2250#
```

smartcs_tty_command モジュールを使う為の SmartCS の設定は以上となります。

5.2 Playbook 作成の準備

smartcs_tty_command モジュールを使ったコンソールアクセス機能を使う為の Playbook の作成時に必要な設定値について説明します。

(1) モジュールの設定

使用するモジュールとして、”seiko.smartcs.smartcs_tty_command”を指定します。

(2) コネクションプラグインの設定

コネクションプラグインとして、”ansible.netcommon.network_cli” を設定します。

```
ansible_connection: ansible.netcommon.network_cli
```

(3) ネットワーク OS の設定

ネットワーク OS として、”seiko.smartcs.smartcs”を設定します。

```
ansible_network_os: seiko.smartcs.smartcs
```

(4) SmartCS に接続するユーザ名、パスワードの設定

SmartCS に接続するユーザ名、パスワードを指定します。

5-1 項で作成した拡張ユーザのグループに所属するユーザを設定します。

```
ansible_user: smartcs-ansible
ansible_password: password
```

5.3 Playbook 例

```
---
- hosts: smartcs
  gather_facts: no

  tasks:
  - name: "smartcs_tty_command"
    seiko.smartcs.smartcs_tty_command:
      tty: 1
      sendchar :
      - 'show version'

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: smartcs-ansible
  - ansible_password: password
```

smartcs_tty_command のモジュールの各オプションの説明については、8-1 項を参照下さい。

6　章　ネットワーク機器ベンダーのモジュールと連携する

6.1　概要

SmartCS の SSH トランスペアレント接続機能(sshxpt)を有効にすることで、ネットワーク機器ベンダーの提供している Ansible モジュールを SmartCS 経由で動作させ、SmartCS に接続している機器の設定や情報取得を行うことができます。

ネットワーク機器をコンソール経由で制御するため、工場出荷状態の場合や、IP アドレスが未設定の場合であっても、SmartCS を介して Ansible からの制御が可能となります。Playbook 内に記載する制御コマンドや戻り値のエラー判定などは、使用するベンダーの Ansible モジュールでの定義通りに動作します。

SmartCS と連携する際に使うモジュールの使い方については、各ベンダーから提供されているドキュメントや Web サイトを参照下さい。

6.2 SmartCS の準備

ネットワーク機器ベンダーのモジュールと連携してSmartCSに接続している機器の制御を行う場合、SmartCS に以下の設定を行う必要があります。
手順(2)(3)(4)は、smartcs_tty_command モジュールを使う為の設定、
手順(5)(6)(7)は、ネットワーク機器ベンダーのモジュールを連携して動作させる為の設定となります。

(1) バージョンの確認

SmartCS のシステムソフトウェアが v2.1 以上である事を確認してください。

```
(0)NS-2250# show version
System                : System Software Ver 2.1 (Build 2019-MM-DD)
 :

(0)NS-2250#
```

(2) SSH 接続の有効化

SSH サーバが有効となっているかを確認して下さい。
また、デフォルトで公開鍵認証となっているのでパスワード認証に設定して下さい。

```
(0)NS-2250# enable sshd
(0)NS-2250# set sshd auth basic
(0)NS-2250#
```

フィルター機能が有効化されている場合や、特定のホストからの接続許可設定をしている場合、コントロールノードからの SSH 接続を許可するように設定して下さい。

(3) 拡張ユーザグループのユーザ作成とコンソールアクセス権限の付与

コンソールアクセス機能を利用する為には、拡張ユーザグループに所属するユーザを作成する必要があります。その後、作成したユーザに対して tty マネージ機能とポートアクセスの権限を付与する設定が必要です。

```
(0)NS-2250# create user smartcs-ansible group extusr password
Changing password for user smartcs-ansible.
New password:
Retype new password:
Password for smartcs-ansible changed
(0)NS-2250# set user smartcs-ansible permission ttymanage on
(0)NS-2250# set user smartcs-ansible port 1-48
(0)NS-2250#
```

(4) コンソールアクセス機能の有効化

コンソールアクセス機能を有効化する為の設定を行います。

```
(0)NS-2250# enable ttymanage
(0)NS-2250#
```

(5) ポートユーザグループのユーザ作成とコンソールアクセス機能の権限の付与

SSH トランスペアレント接続機能(sshxpt)を利用するために、ポートユーザグループのユーザを作成し、ポートアクセスの権限を付与する設定を行います。

```
(0)NS-2250# create user smartcs-port group portusr password
Changing password for user smartcs-port.
New password:
Retype new password:
Password for smarcs-port changed
(0)NS-2250# set user smartcs-port port 1-48
(0)NS-2250#
```

(6) 接続ポートの開放

SSH トランスペアレント接続機能(sshxpt)を有効にするシリアルポートに対して sshxpt オプションを設定し、TCP ポートを開放します。ポートの開始番号は 9301 が初期値となっており、開始番号から連続してシリアルポートの数だけ使用されます。
このポート番号は、Ansible からアクセスする際の「ansible_port」で指定するポート番号に該当します。

```
(0)NS-2250# set portd tty 1 session both both sshxpt
(0)NS-2250#
```

ポートの開始番号は、設定で変更することが可能です。
設定可能範囲は 1025～65000 です。

```
(0)NS-2250# set portd sshxpt 9301
(0)NS-2250#
```

フィルター機能が有効化されている場合や、特定のホストからの接続許可設定をしている場合、コントロールノードからポートサーバへの SSH 接続を許可するように設定して下さい。

(7) 改行コードの設定

SSH トランスペアレント接続機能(sshxpt)を使用してネットワーク機器へ接続した際に、改行コードを送信するよう設定します。
改行コードを送信することによってネットワーク機器のプロンプトを受信し、その後 Playbook に記載した各種コマンドが実行されます。

送信する改行コードは CR、LF、CRLF、none(送信なし)から選択して設定します。
※デフォルトでは改行コードを送信しない設定になっています。

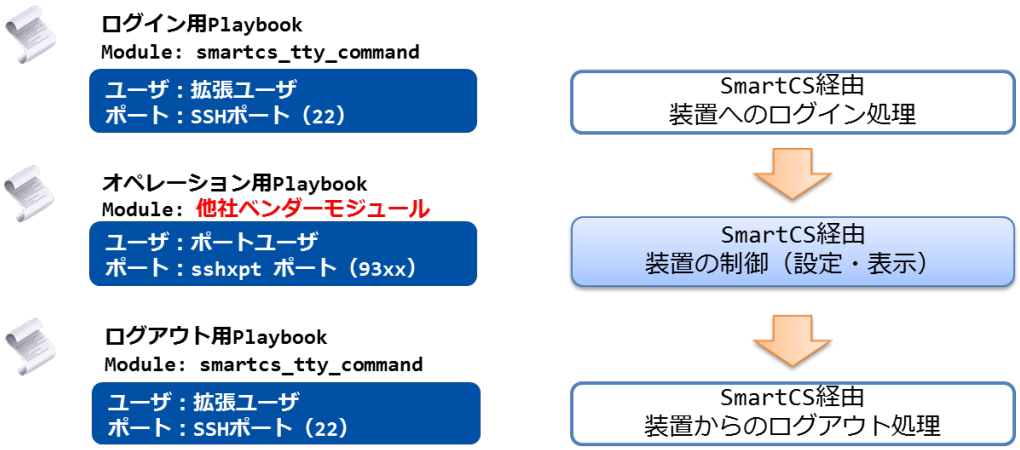
```
(0)NS-2250# set portd tty 1 connted send_nl cr
(0)NS-2250#
```

ネットワーク機器ベンダーのモジュールと連携して制御を行う為の SmartCS の設定は以上となります。

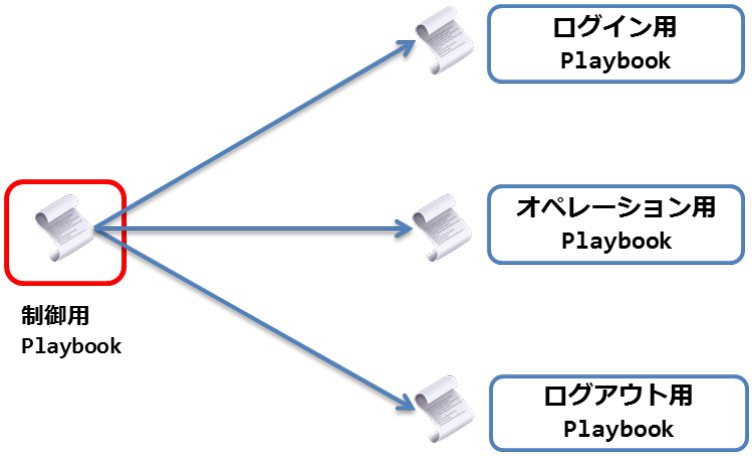
6.3 Playbook 作成の準備

ネットワーク機器ベンダーの提供する network_cli コネクションプラグインを使う Ansible モジュールは、通常 SSH 経由でアクセスする事を想定してつくられています。

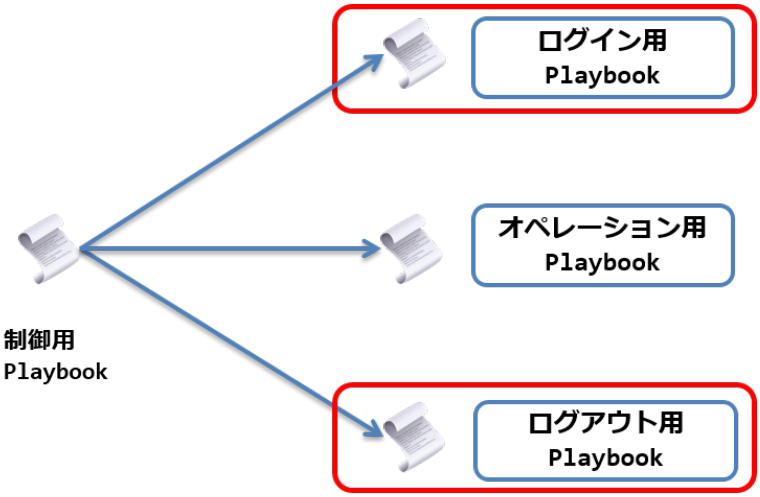
その為、SmartCS との連携を実現する為には、ネットワーク機器のコンソールへのログイン、ログアウト処理については別途 Playbook を作成する必要があります。

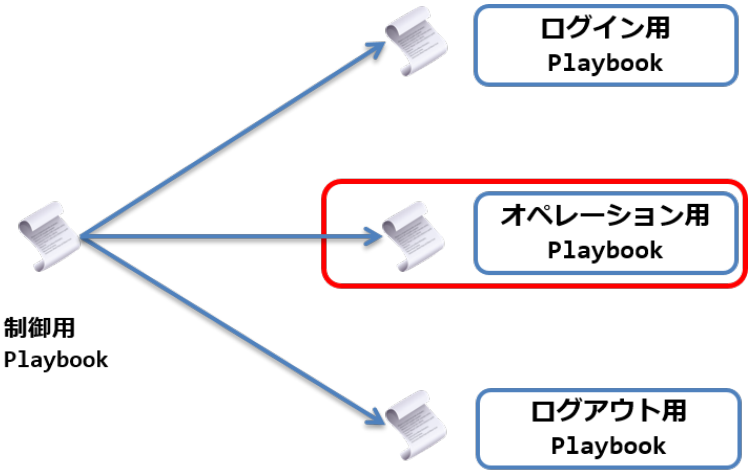


また、下記の様に制御用 Playbook を作成することで、ログイン用 Playbook、ログアウト用 Playbook、オペレーション用 Playbook を一連の操作として実行することが可能になります。



ログイン用 Playbook およびログアウト用 Playbook では、smartcs_tty_command モジュールを使用します。各 Playbook の recvchar にはログインおよびログアウト処理において受信するプロンプト、sendchar にはユーザ名、パスワード、ログアウトコマンドなどを指定し、ネットワーク機器へログインおよびログアウトができるような Playbook を作成します。





オペレーション用 Playbook では、連携するネットワーク機器ベンダーのモジュールを指定し、実行したいコマンドを記述します。ネットワーク機器への接続には、SSH トランスペアレント接続機能(sshxpt)を有効化する際に開放した TCP ポートを使用します。
（デフォルト設定の場合、tty1 への操作は 9301 ポートを使用します。）

オペレーション用 Playbook の作成に必要な設定値について説明します。

(1) モジュール

連携するネットワーク機器ベンダーのモジュールを指定します。

（例）

”xxx_command”

“xxx_config”

“xxx_facts”

(2) コネクションプラグイン

コネクションプラグインとして、”network_cli” を設定します。

```
ansible_connection: ansible.netcommon.network_cli
```

(3) ネットワーク OS

連携するネットワーク機器ベンダーのモジュールで指定するネットワーク OS を設定します。

```
ansible_network_os: xxx
```

(4) SmartCS に接続するユーザ名、パスワードの設定

「6.2 (2) ポートユーザグループのユーザ作成とコンソールアクセス権限の付与」で作成した SmartCS のポートユーザグループのユーザ名、パスワードを指定します。

(5) ポート番号

「6.2 の(3) 接続ポートの開放」で設定した、接続ポートを指定します。

```
ansible_port: 9301
```

6.4 Playbook 例

例)制御用 Playbook

```
---
- name: "Login with smartcs_tty_command"
  import_playbook: login.yml

- name: "Exec Task"
  import_playbook: operation.yml

- name: "Logout with smartcs_tty_command"
  import_playbook: logout.yml
```

例)ログイン用 Playbook(login.yml)

```
---
- hosts: smartcs
  tasks:
  - name: "Login by Console"
    seiko.smartcs.smartcs_tty_command:
      tty: 1
      recvchar:
      - 'username: '
      - 'password: '
      - 'switch>'
      sendchar:
      - '__NL__'
      - 'user'
      - 'secret'

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: smartcs-ansible
  - ansible_password: password
```

例)ログアウト用 Playbook(logout.yml)

```
---
- hosts: smartcs
  tasks:
  - name: "Logout by Console"
    seiko.smartcs.smartcs_tty_command:
      tty: 1
      recvchar:
      - 'username: '
      - 'password: '
      - 'switch>'
      sendchar:
      - 'exit'

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: smartcs-ansible
  - ansible_password: password
```

smartcs_tty_command のモジュールの各オプションの説明については、8.1.2 項を参照してください。

例)オペレーション用 Playbook(operation.yml)

```
---
- hosts: smartcs
  gather_facts: no
  tasks:
  - name: "Task"
    xxx.xxx.xxx_command:
      commands:
      - show version
      - show interfaces
      - show arp
      - show ip route

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_user: smartcs-port
  - ansible_password: password
  - ansible_port: 9301
  - ansible_network_os: xxx.xxx.xxx
  - ansible_become: yes
  - ansible_become_method: ansible.netcommon.enable
  - ansible_become_password: secret
  - ansible_command_timeout: 60
```

6.5 使用上の注意

ネットワーク機器ベンダーの Ansible モジュールと連携して制御を行う際の、制限事項および注意事項を下記に記載します。

(1) network_cli コネクションプラグインのサポート
本装置と連携可能な他ベンダーのモジュールは、コネクションプラグインとして network_cli をサポートしているものに限ります。

(2) ネットワーク機器のプロンプト仕様
SSH でのアクセス時とコンソールからのアクセス時において、ネットワーク機器のプロンプト定義が同一である必要があります。

(3) 処理速度のケア(タイムアウト時間の調整)
SmartCS に接続されている機器のコンソール経由で制御することになりますので、ネットワーク機器に直接 SSH 接続する、通常の Ansible のオペレーションと比べて処理速度が遅くなります。
そのため、「ansible_command_timeout」で指定するタイムアウト時間を長めに設定する必要があります。

7　章　CLI コマンド機能

7.1　SmartCS の準備

CLI コマンド機能を使う為の SmartCS の準備について下記に記載します。

(1) SSH 接続の有効化

SSH サーバが有効となっているかを確認して下さい。

また、デフォルトで公開鍵認証となっているのでパスワード認証に設定して下さい。

```
(0)NS-2250# enable sshd
(0)NS-2250# set sshd auth basic
(0)NS-2250#
```

フィルター機能が有効化されている場合や、特定のホストからの接続許可設定をしている場合、コントロールノードからの SSH 接続を許可するように設定して下さい。

7.2 Playbook 作成の準備

CLI コマンド機能を実行する際の Playbook 作成に必要な設定値について説明します。

(1) モジュールの設定
以下の中から使用するモジュールを指定します。
”smartcs_command”
“smartcs_config”
“smartcs_facts”

(2) コネクションプラグインの設定
コネクションプラグインとして、”network_cli” を設定します。

```
ansible_connection: ansible.netcommon.network_cli
```

(3) ネットワーク OS の設定
ネットワーク OS として、”smartcs”を設定します。

```
ansible_network_os: seiko.smartcs.smartcs
```

(4) SmartCS に接続するユーザ名、パスワードの設定
SmartCS に接続するユーザ名、パスワードを指定します。
ユーザは以下のどちらのグループで作成したユーザでも動作します。
・一般ユーザのグループ
・拡張ユーザのグループ

(5) 管理者ユーザの設定
使用するモジュールやオプションによっては装置管理ユーザに遷移しないとモジュールが正しく動作しません。
その場合、Playbook に以下のように設定して下さい。

```
ansible_become: yes
ansible_become_method: ansible.netcommon.enable
ansible_become_password: password（装置管理ユーザのパスワード）
```

7.3　モジュールと装置管理ユーザ

装置管理ユーザに遷移しないと正しく動作しないモジュール及びオプションについて下記の表に記載します。

○:　装置管理ユーザに遷移する指定が必要。

―:　装置管理ユーザに遷移してもしなくてもどちらでも正常に動作する。

モジュール名	オプション名	装置管理ユーザ に遷移
smartcs_command	commands	commands オプションで 指定するコマンドによります。
smartcs_config	lines	○
	src	○
smartcs_facts	all	○
	default	―
	tty	―
	config	○

smartcs_command の commands オプションで指定するコマンドについて、装置管理ユーザに遷移する必要があるかどうかについては、コマンドリファレンスを参照して下さい。

smartcs_facts のオプション名は、gather_subset オプションで指定する値となります。また、例えば"!tty"のような指定をした場合、tty オプション以外を指定する事になる為、装置管理ユーザ遷移する事が必要となります。

7.4 Playbook 例

```
---
- name: smartcs_command
  hosts: smartcs
  gather_facts: no
  tasks:
  - seiko.smartcs.smartcs_command:
      commands:
      - show version
      - show config running

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: somebody
  - ansible_password: password
  - ansible_become: yes
  - ansible_become_method: ansible.netcommon.enable
  - ansible_become_password: "\r"
```

smartcs_command のモジュールの各オプションの説明については、8.2.2 項を参照下さい。

管理者ユーザにパスワードを付与していない場合、ansible_become_password には上記例のように"\r"と設定して下さい。

8　章　モジュール

8.1　seiko.smartcs.smartcs_tty_command

コンソールアクセス機能用のモジュールについて説明します。

8.1.1 概要

SmartCS のシリアルポートに接続されているネットワーク機器のコンソールに対して、指定された文字列を送信し、コンソールの入出力結果を取得します。

8.1.2 オプション

このモジュールのオプションについて下記に記載します。

オプション名	必須	省略時	設定範囲	内容
cmd_timeout		10	1～7200	sendchar/src で指定した文字列を送信後、recvchar で指定した文字列を受信するまでのタイムアウト時間（秒）を設定します。
error_detect_on_sendchar		cancel	cancel exec	sendchar/src で指定した文字列を送信後、エラーが発生した場合、以降の文字列を送信するか、しないかを設定します。 cancel と設定した場合、文字列の送信がエラーとなると、エラー発生以降の文字列については送信しません。 exec と設定した場合、文字列の送信がエラーとなっても次の文字列を送信します。

オプション名	必須	省略時	設定範囲	内容
error_detect_on_module		ok	ok failed	sendchar/src で指定した文字列を送信後にエラーが発生した場合、ansible コマンド(ansible-playbook コマンド)の実行結果を ok とするか、failed とするかの設定を行います。 ok と設定した場合、文字列を送信後にエラーが発生しても ansible コマンドはエラーとならず、ok となります。 failed と設定した場合、文字列を送信後にエラーが発生すると ansible コマンドはエラーとなり、failed となります。
error_recvchar_regex				sendchar/src で指定した文字列を送信後、受信した文字列内に特定の文字列が含まれていた場合にエラーとして検出する為の文字列のリストを正規表現で設定します。 本設定は、list 形式で最大 8 個設定が可能です。 エラーを検出した際に、ansible コマンドを ok とするか、failed とするかのスイッチは、error_detect_on_module オプションの設定となります。
nl		cr	crlf cr lf	sendchar/src で送信する文字列の改行コードを設定します。

オプション名	必須	省略時	設定範囲	内容
recvchar				sendchar/src で指定した文字列を送信後、待ち受ける受信文字列のリストを設定します。 本設定は list 形式で、最大 16 個設定が可能です。
recvchar_regex				recvchar で設定する文字列のリストを正規表現で設定します。 本設定は list 形式で、最大 8 個設定が可能です。

オプション名	必須	省略時	設定範囲	内容
sendchar	(○)			対象の tty に送信する文字列を設定します。送信文字列は sendchar で指定したリストについて上から順番に送信します。 sendchar オプションには、設定した文字列を送信する以外に以下のような送信方法が設定できます。 __NL__ 改行文字列だけを送信します。改行コードは、nl オプションで設定した値となります。 __CTL__:hex 制御文字を送信します。 指定する hex と制御文字の対応表は、「8.1.5-(4)-5 制御文字を送信する」を参照してください。 __WAIT__:sec 送信文字列毎に、recvchar で設定した文字列を受信するまでのタイムアウト時間を設定できます。未指定時は、cmd_timeout オプションで設定した値がタイムアウト時間として設定されます。 __NOWAIT__ 送信文字列毎に、recvchar で設定した文字列を待ちません。sendchar で指定した文字列を送信後、ただちに次の文字列を送信します。 __NOWAIT__:sec 送信文字列毎に、recvchar で設定した文字列を待ちません。sendchar で指定した文字列を送信後、設定された時間(秒)経過後、次の文字列を送信します。

				sendchar オプションで設定できる時間の範囲は 1～7200(sec)となります。 sendchar で送信する文字列のリストは、シングルクォーテーション、もしくは ダブルクォーテーションで囲むことを推奨します。 sendchar もしくは src のどちらかの指定が必須となります。 このオプションは、src オプションと排他で動作します。
src	(○)			対象の tty に送信する文字列を 1 行ずつ記載したファイルのパスを指定します。 このオプションは、指定するファイルの絶対パス、または Playbook の保存先ディレクトリからの相対パスを指定します。 src もしくは sendchar のどちらかの指定が必須となります。 このオプションは、sendchar オプションと排他で動作します。

オプション名	必須	省略時	設定範囲	内容
tty	○		1～48	文字列を送信する tty を設定します。 ttylist 形式（1-16、1,2-8,16）での設定が可能です。複数の tty 番号を指定した場合、tty 番号毎に sendchar/src で指定した文字列を送信します。
ttycmd_debug		off	off on detail	sendchar/src による文字列送信処理が終了した後に、以下の情報を表示します。 ・tty オプションの設定値 ・cmd_timeout オプションの設定値 ・nl オプションの設定値 ・error_detect_on_sendchar オプションの設定値 ・recvchar オプションの設定値 ・recvchar_regex オプションの設定値 ・error_recvchar_regex オプションの設定値 本オプションはデバッグ用途となります。

オプション名	必須	省略時	設定範囲	内容
custom_response		False	boolean 値	stdout, stdout_lines に加えて、送信文字列, 受信文字列を区別できるフォーマットの返り値を返します。 送信文字列毎に、execute_command と response を分けて出力します。
custom_response_delete_nl		False	boolean 値	custom_response の出力について、改行だけの行を削除します。
custom_response_delete_lastline		False	boolean 値	custom_response の出力について response の最後の 1 行を削除します。 ※CLI コマンド実行後のプロンプトを表示させないことを目的としています。

オプション名	必須	省略時	設定範囲	内容
initial_prompt				initial_prompt_check_cmd 送信後に受信を期待する文字列を指定します。本設定は正規表現での指定も可能です。 本設定がある場合、プレチェック処理が動作します。
initial_prompt_check_cmd		__NL__（改行）		sendchar/src で指定した文字列の送信前にコンソールのプロンプト状態を確認するためのコマンドを指定します。 未指定の場合、__NL__（改行）を送信します。
initial_prompt_check_cmd_timeout		5	1～30	initial_prompt_check_cmd を送信後、受信文字列をチェックするまでの時間を指定します。
escape_cmd				initial_prompt_check_cmd を送信後、initial_prompt を受信できなかった場合に送信する文字列を指定します。
escape_cmd_timeout		5	1～30	escape_cmd を送信後、initial_prompt を含んでいるかをチェックするまでの時間を指定します。
escape_cmd_retry		3	0～8	escape_cmd を送信後、initial_prompt を受信しなかった時に再度 initial_prompt_check_cmd を送信する際のリトライ回数を指定します。

8.1.3 Playbook 例

このモジュールの Playbook 例について下記に記載します。

```
---
- name: Configure_ipaddress
  hosts:  smartcs
  gather_facts: no

  tasks :
  - name : Configure  NS-2250  ipaddress  by Console
    seiko.smartcs.smartcs_tty_command :
      tty: 1
      nl : cr
      cmd_timeout : 5
      recvchar :
      - 'NS-2250 login: '
      - 'Password: '
      - '(c)NS-2250> '
      - '(c)NS-2250# '
      - '[y/n] ? '
      - 'logout: somebody/console'
      sendchar :
      - '__NL__'
      - 'somebody'
      - '__NL__'
      - 'su'
      - '__NL__'
      - 'set ipaddr eth1 192.168.0.1/24'
      - 'write'
      - 'y'
      - 'exit'
      - 'exit'

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: smartcs-ansible
  - ansible_password: password
  - ansible_command_timeout: 60
```

8.1.4 戻り値

このモジュールの戻り値ついて下記に記載します。

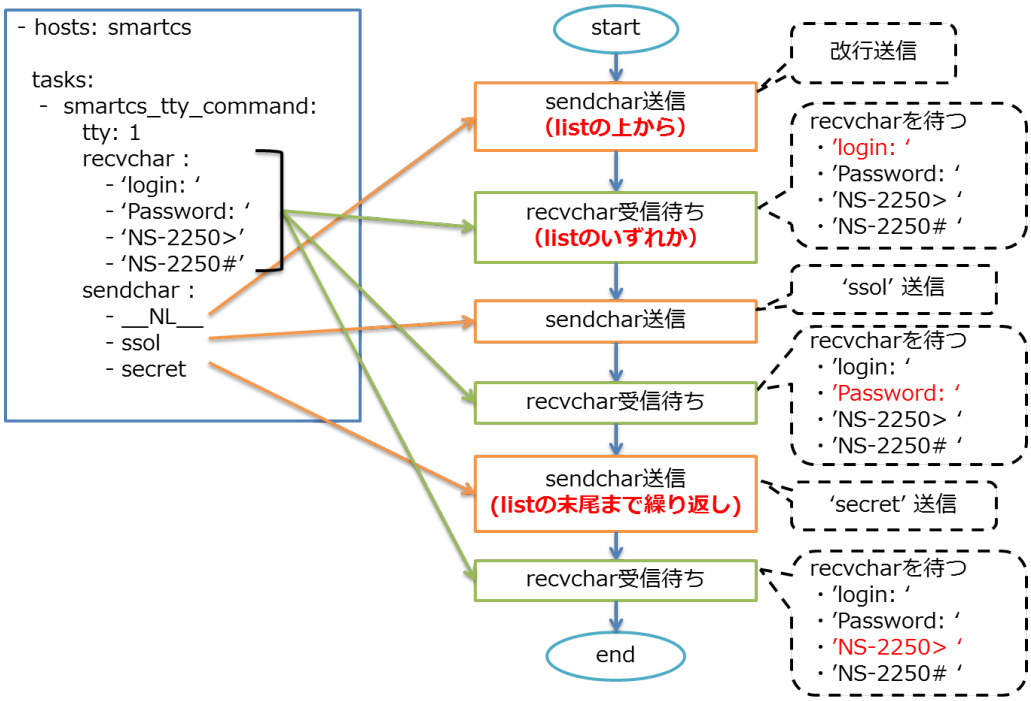
名前	説明	契機	タイプ
stdout_lines	コンソールの送受信文字列を改行文字列毎に分割したリストとなります。 sendchar/src で指定した送信文字列毎にリスト化します。	コマンドの実行に成功した場合	リスト
stdout	コンソールの送受信文字列となります。 sendchar/src で指定した送信文字列毎にリスト化します。	コマンドの実行に成功した場合	リスト
pre_stdout_lines	プレチェック機能実行時におけるコンソールの送受信文字列を、改行文字列毎に分割したリストとなります。	initial_prompt 設定があり、かつコマンドの実行に成功した場合	リスト
pre_stdout	プレチェック機能実行時におけるコンソールの送受信文字列となります。	initial_prompt 設定があり、かつコマンドの実行に成功した場合	リスト
stdout_lines_custom	コンソールの送受信文字列について、送信文字列(execute_command)、受信文字列(response)を区別した形式のリストとなります。	custom_response 設定が有効かつコマンドの実行に成功した場合	リスト

8.1.5 解説

各オプションの動作についてそれぞれ解説します。

(1) sendchar/src と recvchar の動作

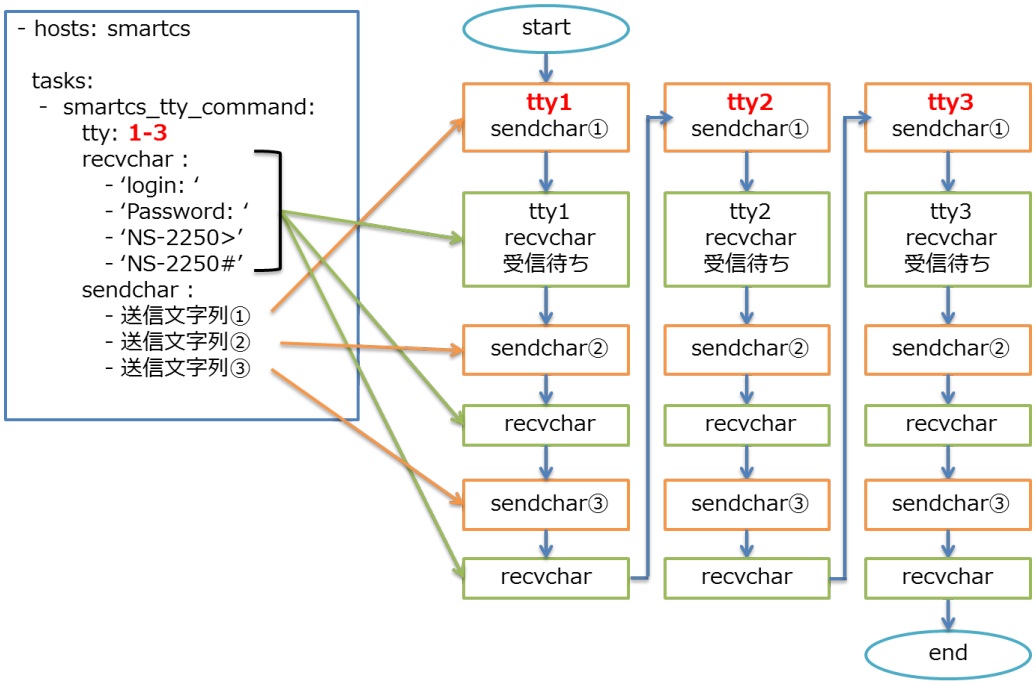
sendchar/src は、指定された文字列を上から順番に送信します。
recvchar は、文字列を送信後、一致する文字列が入出力内容に含まれるかどうかを待ちます。一致する文字列を受信した場合、次の文字列を送信します。
※下記の図は、sendchar オプションを指定した場合の例となります。



(2) 複数の tty を指定した場合の動作

tty オプションは ttylist 形式(ハイフンやカンマを含んだ複数指定)で設定する事が可能です。複数の tty が指定された場合、tty 毎に sendchar/src で指定した文字列を送信します。

※下記の図は、sendchar オプションを指定した場合の例となります。

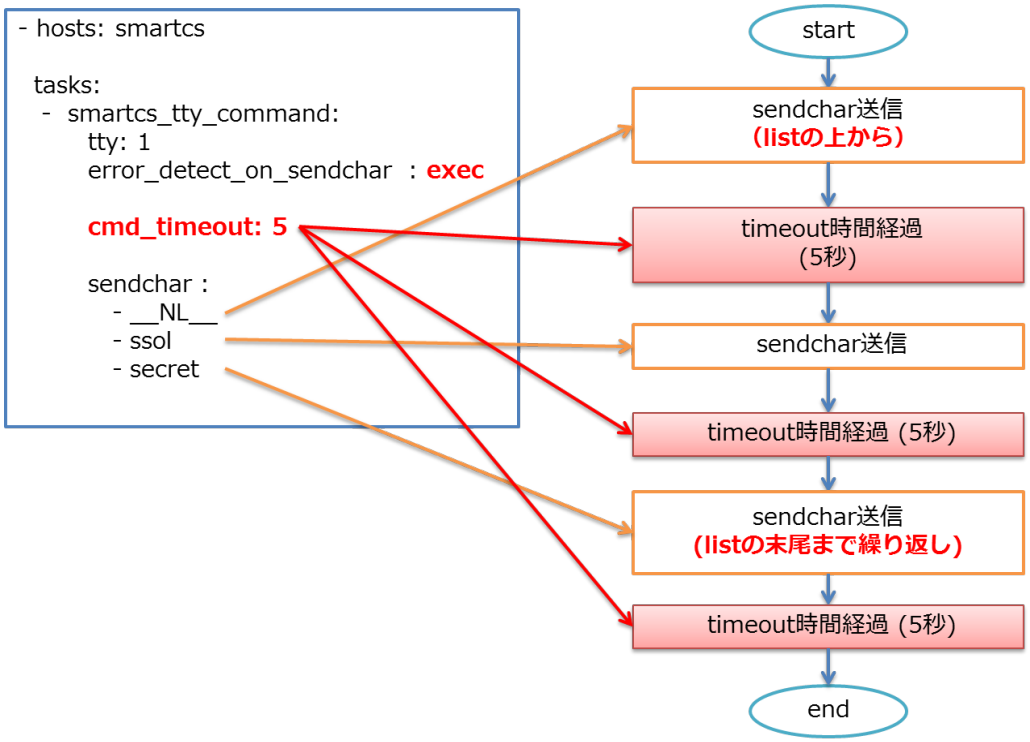


(3) recvchar を設定しない場合の動作

recvchar (recvchar_regex) を指定しなかった場合、sendchar/src は cmd_timeout 時間の経過を待って次の文字列を送信します。

※smartcs_tty_command モジュールの必須パラメータは、
tty と sendchar/src オプションのみとなります。

※下記の図は、sendchar オプションを指定した場合の例となります。



error_detect_on_sendchar の設定が cancel(デフォルト値)の場合、sendchar/src で指定した文字列を送信した際にエラーが発生すると(上記の場合タイムアウトエラー)、次の文字列を送信しません。その為、上記の例は error_detect_on_sendchar を exec と設定しています。

(4) sendchar/src の特殊な設定

sendchar/src は指定した文字列を送信する以外にもオプションを指定する事で特殊な送信方法を設定する事ができます。

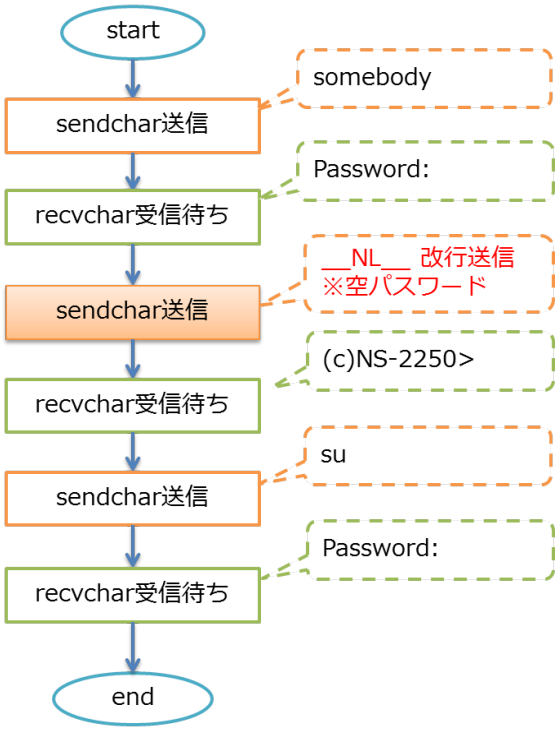
1. 改行文字列だけを送信する。

__NL__オプション

改行文字だけを送信する場合、送信文字列として”__NL__”を設定します。
送信される改行コードは、nl オプションで設定した値となります。
コンソール経由でログイン処理のオペレーションを行う時などのパスワード入力時に、空パスワードを設定する場合などに使う事ができます。
※下記の図は、sendchar オプションを指定した場合の例となります。

```
- hosts: smartcs

  tasks:
   -  smartcs_tty_command:
        tty: 1
        recvchar :
          - 'login: '
          - 'Password: '
          - 'NS-2250>'
          - 'NS-2250#'
        sendchar :
          - somebody
          - __NL__
          - su
          - __NL__
          - show version
```



2. 送信文字列毎に、タイムアウト時間を設定する。

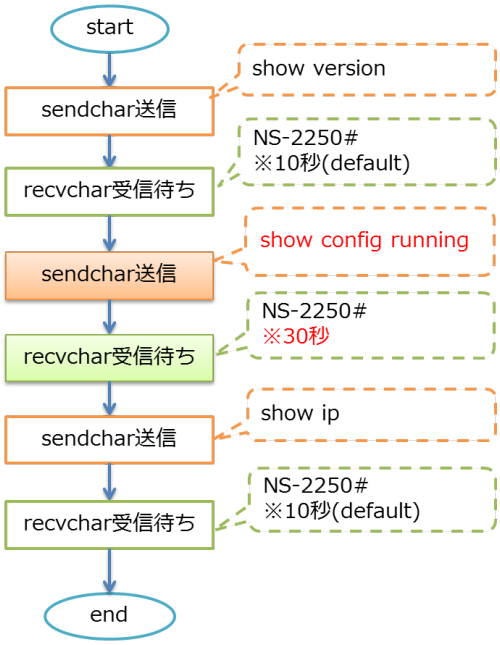
__WAIT__:sec オプション

sendchar/src で指定した文字列は、cmd_timeout オプションで設定された時間（デフォルト 10 秒）recvchar で設定された文字列を待ちます。
文字列の送信によって SmartCS に接続されている機器のコンソールで実行されるコマンドが、結果を出力するまでに時間が掛かる場合（ランニングコンフィグの取得コマンドやサポート情報の取得コマンドの実行）、特定の送信文字列のみタイムアウト時間を変更する事ができます。

以下の例の場合、recvchar のタイムアウト値はデフォルトの 10 秒ですが、「show config running」文字列の送信時のみ、タイムアウト時間を 30 秒に設定し動作します。
※下記の図は、sendchar オプションを指定した場合の例となります。

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       recvchar :
         - 'login: '
         - 'Password: '
         - 'NS-2250>'
         - 'NS-2250#'
       sendchar :
         - show version
         - show config running __WAIT__:30
         - show ip
```



3. 送信文字列毎に、recvchar を待たない設定をする。

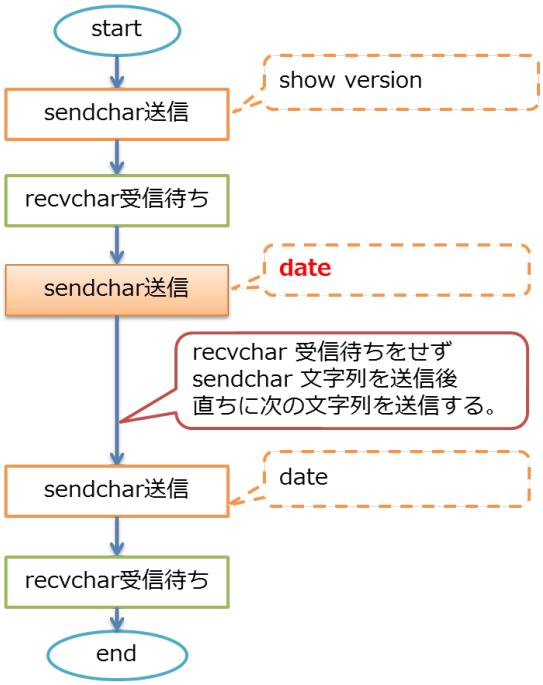
__NOWAIT__ オプション

cmd_timeout オプションで設定された時間を待たずにすぐ次の文字列を送信するオプションとなります。
※約1秒後に送信します。

コンソール接続先に対して、recvchar を待たずに連続して文字列を送信したい場合などに使う事ができます。
※下記の図は、sendchar オプションを指定した場合の例となります。

```
- hosts: smartcs

  tasks:
   -  smartcs_tty_command:
        tty: 1
        cmd_timeout: 5
        recvchar :
          - 'login: '
          - 'Password: '
          - '>'
          - '#'
          - '[y/n] ? '
        sendchar :
          - show version
          - date __NOWAIT__
          - date
```



4. 送信文字列毎に、recvchar を待たずに時間だけで待つように設定する。

__NOWAIT__:sec オプション

recvchar の設定がある場合、文字列送信後の入出力結果に recvchar が含まれているかの確認を行い、含まれている場合に次の文字列の送信を行います。ただし、SmartCS に接続されている機器のコンソールに対して行いたいオペレーションによっては、これらの基本動作によって意図通りに動作させる事ができない場合があります。

（例）

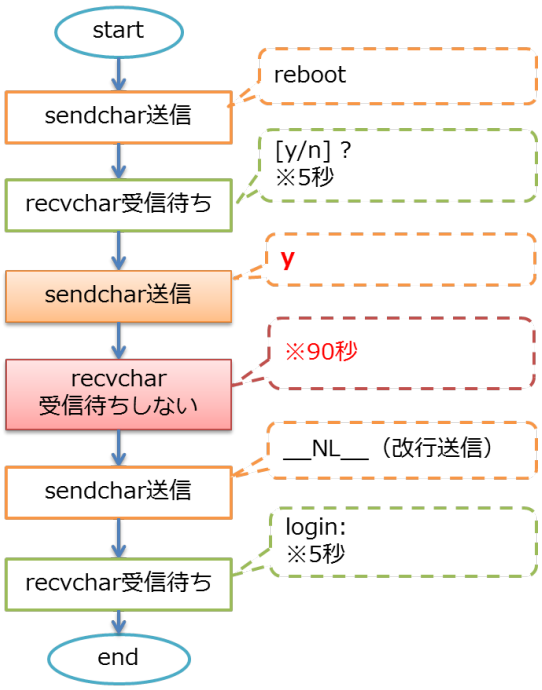
- recvchar に "#"、">" といった文字列を設定し 本来であれば SmartCS に接続されている機器のプロンプトを待ちたいが、実行したコマンドの出力に ">" が含まれており、次の文字列が送信されてしまう場合。
- リブート や バージョンアップコマンドを実行した際、コンソールに様々な文字や記号が出力されてしまう為、意図せず recvchar にマッチしてしまい、リブート中などに次の文字列が送信されてしまう場合。

上記のようなシチュエーションでも出来る限り意図通り Playbook を通してコンソールオペレーションが行えるように、送信文字列毎に recvchar を待たない設定をする事ができます。

※下記の図は、sendchar オプションを指定した場合の例となります。

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       cmd_timeout: 5
       recvchar :
          - 'login: '
          - 'Password: '
          - '>'
          - '#'
          - '[y/n] ? '
       sendchar :
          - show version
          - reboot
          - y __NOWAIT__:90
          - __NL__
          - show version
```



5. 制御文字を送信する

__CTL__ オプション

制御文字を送信する場合、sendchar/src に「__CTL__:hex」を指定します。
送信可能な制御文字は以下の範囲となります。

00 : [Ctrl-@]	08 : [Ctrl-H]	10 : [Ctrl-P]	18 : [Ctrl-X]
01 : [Ctrl-A]	09 : [Ctrl-I]	11 : [Ctrl-Q]	19 : [Ctrl-Y]
02 : [Ctrl-B]	0a : [Ctrl-J]	12 : [Ctrl-R]	1a : [Ctrl-Z]
03 : [Ctrl-C]	0b : [Ctrl-K]	13 : [Ctrl-S]	1b : [Ctrl-[ ] / ESC
04 : [Ctrl-D]	0c : [Ctrl-L]	14 : [Ctrl-T]	1c : [Ctrl-¥]
05 : [Ctrl-E]	0d : [Ctrl-M]	15 : [Ctrl-U]	1d : [Ctrl-]]
06 : [Ctrl-F]	0e : [Ctrl-N]	16 : [Ctrl-V]	1e : [Ctrl-^]
07: [Ctrl-G]	0f : [Ctrl-O]	17 : [Ctrl-W]	1f : [Ctrl-_]
			7f : [Delete] / Ctrl-?

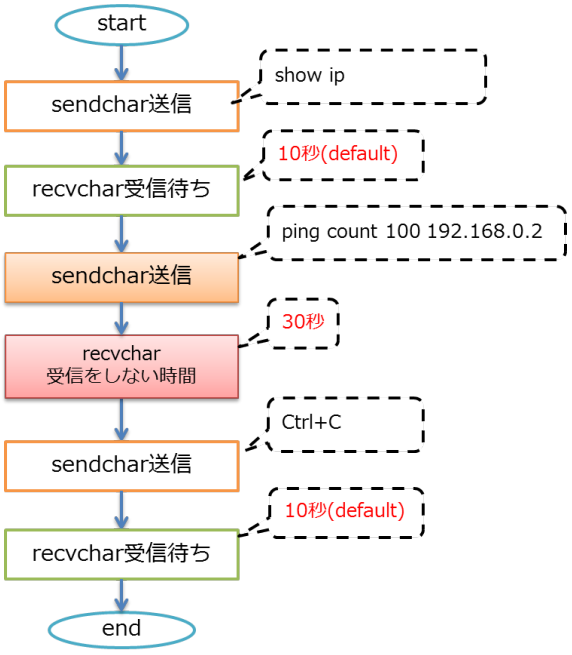
※一番左の値が__CTL__:hex の hex 部分で、playbook で指定する値です。

コンソール経由で制御文字を送信する場合に使うことができます。
例:ping の実行を停止する。特定の NW 機器のコマンド実行後に送信する。等

※下記の図は、sendchar オプションを指定した場合の例となります。

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       recvchar:
         - 'Username: '
         - 'Password: '
         - '(c)NS-2250>'
         - '(c)NS-2250#'
       sendchar:
         - show ip
         - ping count 1000 192.168.0.2 __NOWAIT__:30
         - __CTL__:03
```



6. 特殊な送信方法の組み合わせ

sendchar/src で送信できる方法の組み合わせは以下の通りとなります。

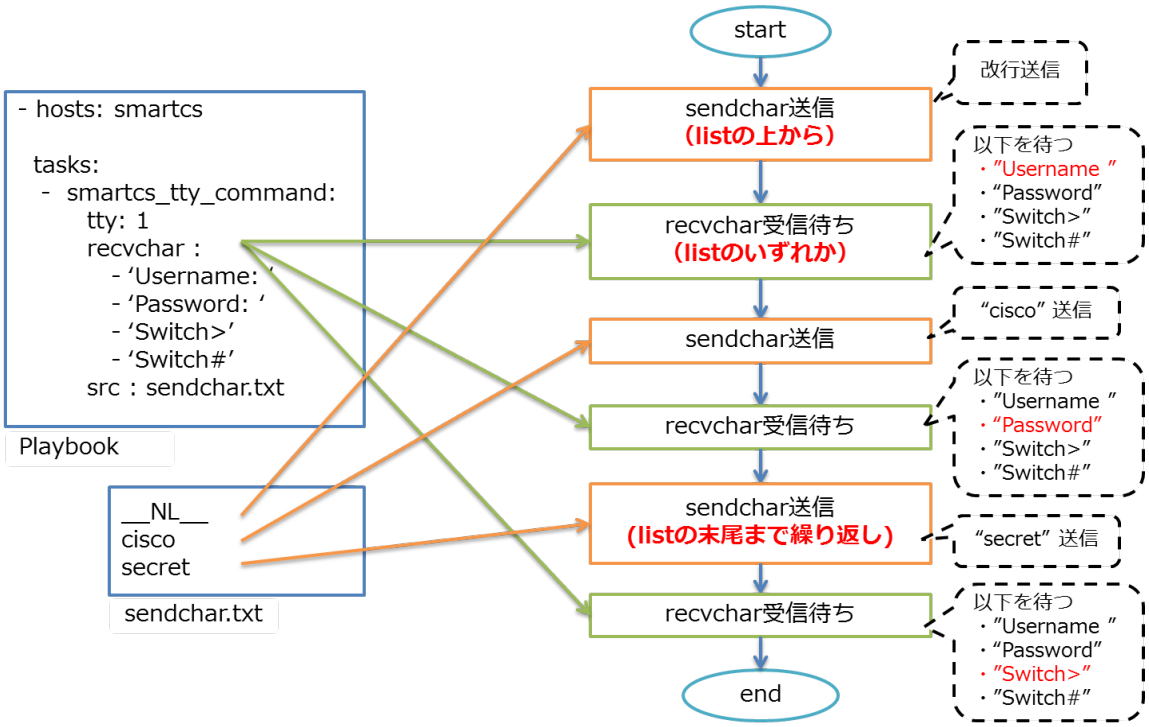
設定方法	備考
show version	文字列を送信
show version __WAIT__:sec	文字列を送信後、設定した時間 recvchar を待つ
show version __NOWAIT__	文字列を送信後、ただちに次の文字列を送信する。
show version __NOWAIT__:sec	文字列を送信後、設定した時間だけで待つ。
__NL__	改行を送信
__NL__ __WAIT__:sec	改行を送信後、設定した時間 recvchar を待つ
__NL__ __NOWAIT__	改行を送信後、ただちに次の文字列を送信する。
__NL__ __NOWAIT__:sec	改行を送信後、設定した時間だけで待つ。
__CTL__:03	制御文字を送信する。
__CTL__:03 __WAIT__:sec	制御文字を送信後、設定した時間 recvchar を待つ。
__CTL__:03 __NOWAIT__:sec	制御文字を送信後、設定した時間待って次の文字列を送信する。
__CTL__:03 __NOWAIT__	制御文字を送信後、ただちに次の文字列を送信する。

※"show version"部分は何らかの送信文字列を指定した場合の
例となります。

※"__CTL__"で指定している 03 は、ctrl+C を指定した場合の例となります。

(5) 送信文字列を外部ファイルで指定する

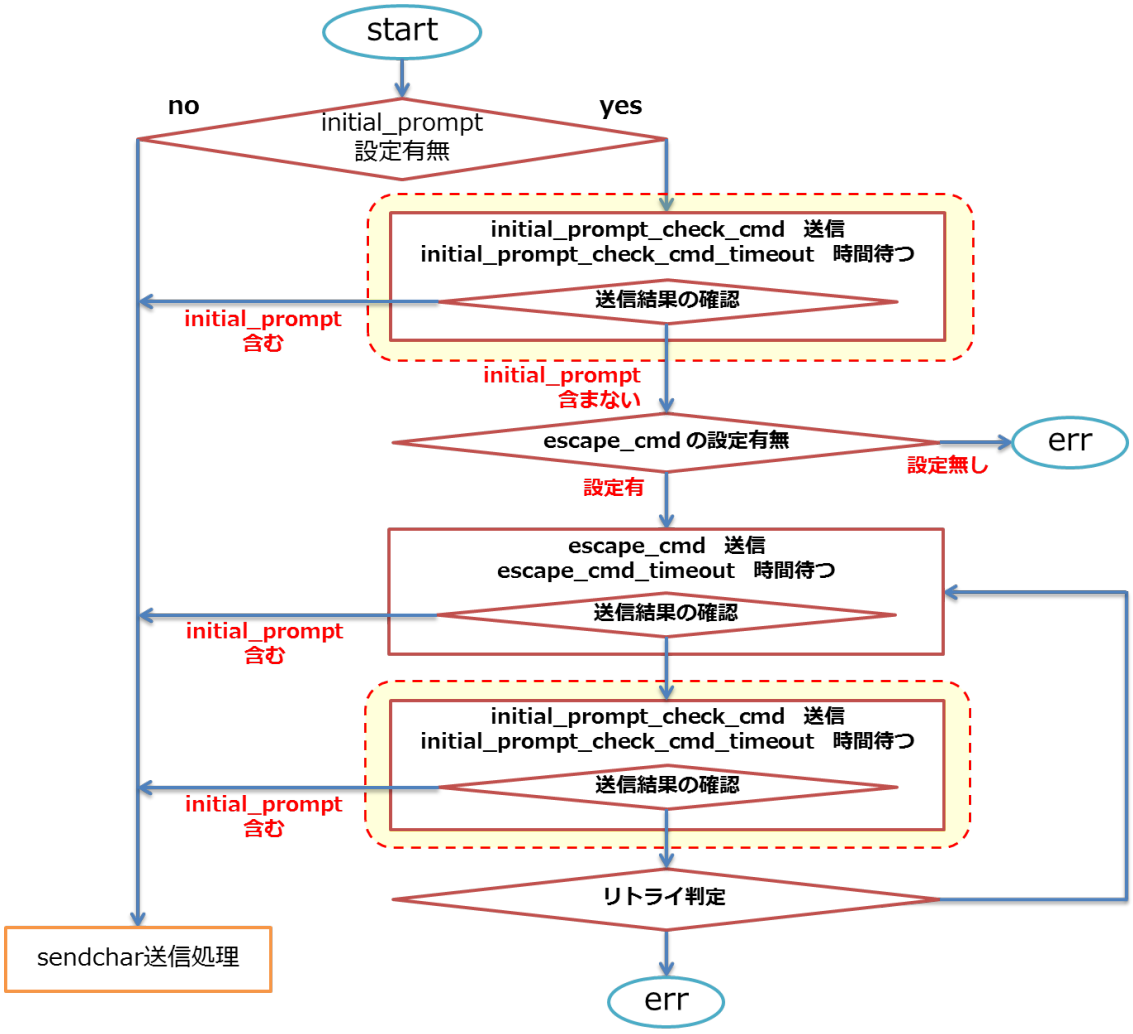
送信文字列は、Playbook 内で sendchar オプションにリスト形式で記載する以外に、src オプションで外部ファイルを読み込み実行することが可能です。
※下記の図は、src オプションを指定した場合の例となります。



(6) 文字列の送信前にコンソールの状態をチェックする機能(プレチェック機能)

sendchar/src オプションで指定した文字列を送信する前に、対象機器のコンソールが期待する状態(例:ログインプロンプト状態)であるかを確認することができます。
initial_prompt オプションを設定することで、sendchar/src オプションで指定した文字列を送信する前に、initial_prompt_check_cmd で指定した任意のコマンド(デフォルトは改行)を送信します。そのコマンドの結果に、initial_prompt で指定した文字列が含まれているかどうかを部分一致で確認します。一致しない場合、escape_cmd で指定した任意のコマンド(例:exit)を送信し、再度、結果を部分一致で確認します。
initial_prompt については正規表現での設定が可能です。

プレチェック機能の動作フローについては、下記図を参照してください。



＜補足＞

(ア) 本機能は initial_prompt オプションが設定された場合に動作します。
initial_prompt_check_cmd 未指定時は改行が送信されます。
※デフォルトでは__NL__が設定されており、改行コードは NL オプションに依存します。

(イ) 本機能の動作中、以下の場合に Playbook は sendchar/src で指定した文字列の送信前にエラーとなります。

・escape_cmd 設定がない場合
initial_prompt_check_cmd を送信し、期待する文字列を受信しなかった場合。

・escape_cmd 設定があり、リトライ回数が上限となった場合
escape_cmd をリトライ回数分送信後、期待する文字列を受信しなかった場合。

(ウ) 本機能が動作し、Playbook が正常に終了すると以下の戻り値が追加で出力されます。

initial_prompt_check_cmd 送信時、escape_cmd 送信時のコンソールの入出力結果についての値が出力されます。

・pre_stdout_lines
sendchar/src で指定した文字列送信前のチェック処理で送受信したコンソールの入出力内容を改行文字列毎に分割したリストで返します。

・pre_stdout
sendchar/src で指定した文字列送信前のチェック処理で送受信したコンソールの入出力内容を返します。

(エ) 本機能を使用したとしても、すべてのシチュエーションについて必ずコンソールを期待する状態に戻せる保証はありません。

(7) error_detect_on_sendchar の動作

sendchar/src で指定した文字列を送信する場合、以下の理由で文字列の送信がエラーとなる場合があります。

＜文字列の送信がエラーとなる要因＞

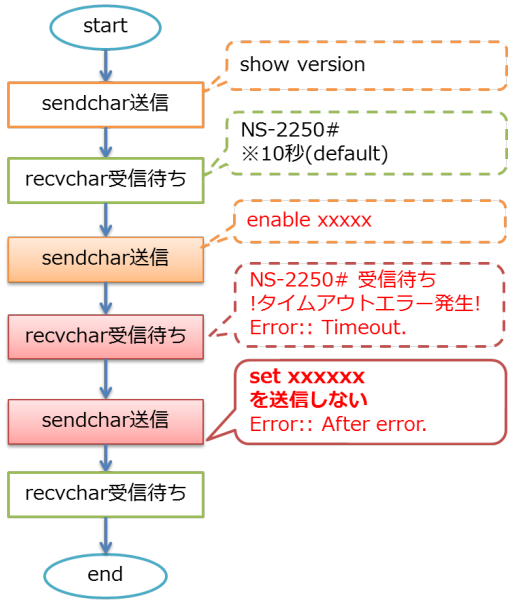
エラー要因		要因
recvchar をタイムアウト時間までに受信出来なかった		Error:: Timeout.
対象の tty に接続できない	接続許可設定がない為、接続できない。	Error:: Not allowed.
	排他制御により接続できない。	Error:: Session limit over.
	tty 管理用デーモンに接続ができない	Error:: Connection closed.
	error_recvchar_regex で設定した文字列を検出した。	Error:: Matched “xxx”.
error_detect_on_sendchar 設定が”cancel”の時に、次の送信文字列を送信しない		Error:: After error.

これらのエラーが発生した場合に次の文字列を送信してしまうと、本来想定していたオペレーションと異なる動作となってしまう恐れがあります。
その為
・エラー発生後も文字列をそのまま送信するか
・エラー発生後には文字列を送信しないか
についての動作を設定するオプションとして、error_detect_on_sendchar を用意しています。

1. error_detect_on_sendchar:cancel 設定時の動作

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       error_detect_on_sendchar: cancel
       recvchar :
         - 'login: '
         - 'Password: '
         - 'NS-2250>'
         - 'NS-2250#'
       sendchar :
         - show version
         - enable xxxxx
         - set xxxxxx
```

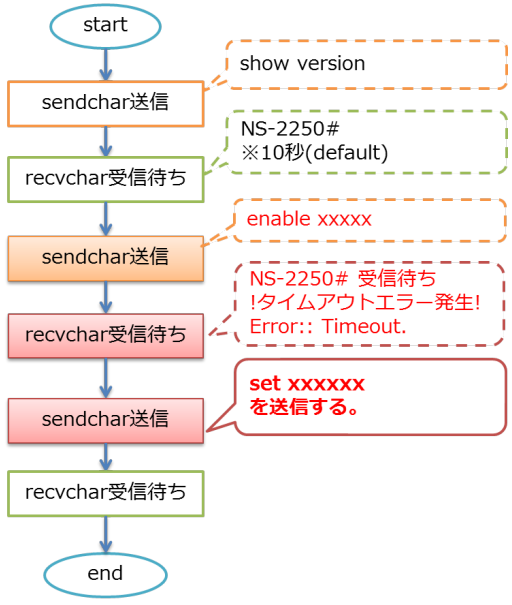


※デフォルト値は、error_detect_on_sendchar:cancel です。

2. error_detect_on_sendchar:exec 設定時の動作

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       error_detect_on_sendchar: exec
       recvchar :
         - 'login: '
         - 'Password: '
         - 'NS-2250>'
         - 'NS-2250#'
       sendchar :
         - show version
         - enable xxxxx
         - set xxxxxx
```



(8) error_detect_on_module の動作

smartcs_tty_command モジュールでは、コンソールの入出力をそのまま返す為、基本的には ansible コマンドの実行結果について、ok を返します。
ただし、error_detect_on_module オプションを使う事で、smartcs_tty_command モジュール内で発生したエラーを制御して、ansible コマンドを failed にする事が可能です。制御できるエラーは、以下の通り、「smartcs_tty_command モジュールで、sendchar オプションによって設定された文字列を送信した際に発生したエラー」に限ります。

制御できるエラー一覧	smartcs_tty_command モジュールで、sendchar オプションにより設定された文字列の送信時に発生するエラー ・Error:: Timeout. ・Error:: Not allowed. ・Error:: Session limit over. ・Error:: Connection closed. ・Error:: Matched “xxx” ・Error:: After error.
制御できないエラー例	・smartcs_tty_command モジュールのオプション指定を誤った場合。 ・smartcs_tty_command モジュールを実行した際にマネージドノードである SmartCS 上で実行した CLI が何らかの理由でエラーとなった場合。 ・Ansible 経由で通信を行っている SSH セッションに起因して発生するエラー （例）ansible_command_timeout で設定したコマンドタイムアウト時間を超過した場合など。

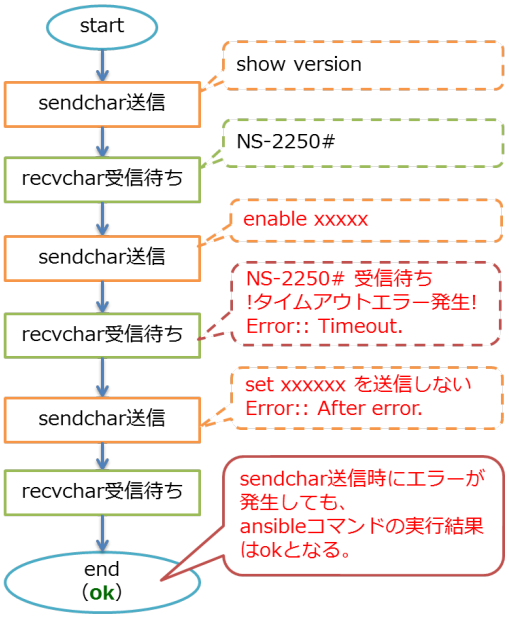
error_detect_on_module オプションの設定、sendchar/src で指定した文字列を送信後のエラー有無、ansible コマンドの実行結果の組み合わせは以下の通りです。

error_detect_on_module の設定値	文字列送信後のエラー発生有無	ansible コマンドの実行結果
ok	エラー発生	ok
	エラー未発生	ok
failed	エラー発生	failed
	エラー未発生	ok

1. error_detect_on_module:ok 設定時の動作

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       error_detect_on_sendchar: cancel
       error_detect_on_module: ok
       recvchar :
         - 'login: '
         - 'Password: '
         - 'NS-2250>'
         - 'NS-2250#'
       sendchar :
         - show version
         - enable xxxxx
         - set xxxxxx
```

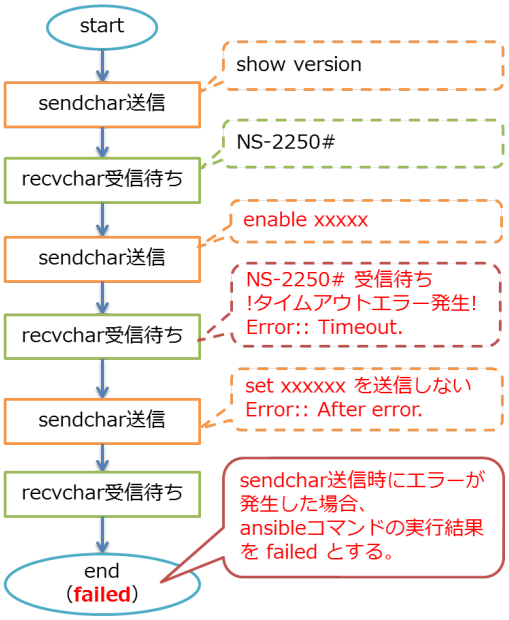


※デフォルト値は、error_detect_on_module:ok です。

2. error_detect_on_module:failed 設定時の動作

```
- hosts: smartcs

 tasks:
  -  smartcs_tty_command:
       tty: 1
       error_detect_on_sendchar: cancel
       error_detect_on_module: failed
       recvchar :
         - 'login: '
         - 'Password: '
         - 'NS-2250>'
         - 'NS-2250#'
       sendchar :
         - show version
         - enable xxxxx
         - set xxxxxx
```



(9) カスタマイズした戻り値を出力させる

smartcs_tty_command では、custom_response というオプションを設定することで、送信文字列毎に区別した stdout_lines_custom を出力することができます。

・stdout_lines_custom

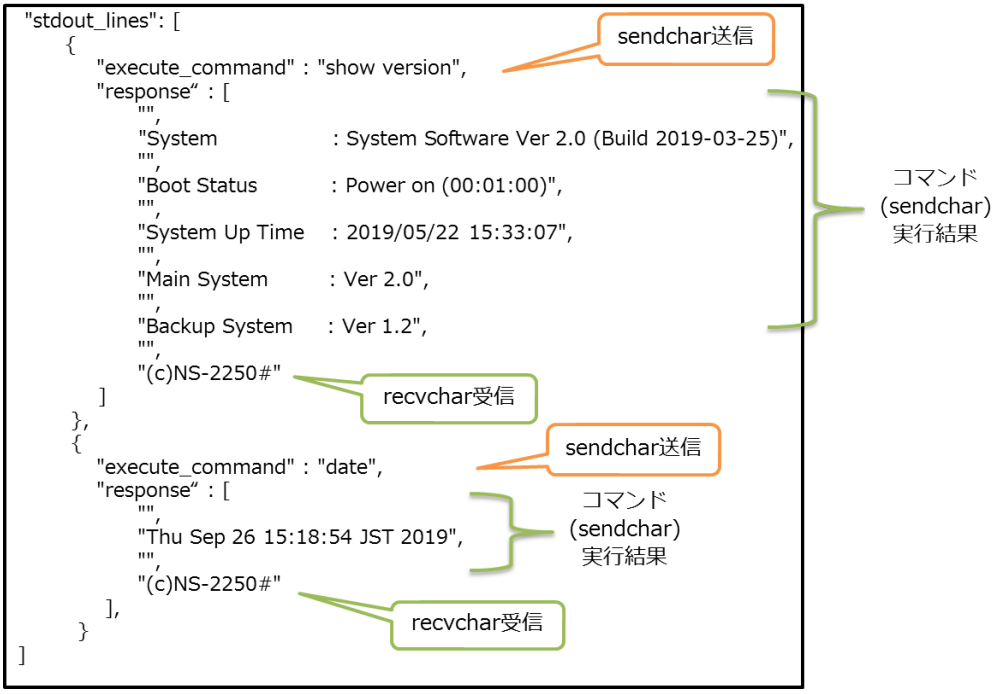
送信文字列毎に、execute_command, response というキーを用意し、それぞれに以下を格納します。

execute_command ：送信文字列

response ：sendchar 送信後、recvchar を受信するまでの出力内容

custom_response を有効にした Playbook と実行結果（stdout_lines_custom）

```
- hosts: smartcs
  tasks:
   - smartcs_tty_command:
       tty: 1
       custom_response: on
       recvchar_regex:
         - '(^|¥r|¥n|!)[a-zA-Z0-9_().-]*(>|#) '
       sendchar :
         - show version
         - date
```



custom_response オプションの設定値

・custom_response_delete_nl

stdout_lines_custom の response 内容から改行のみの行を削除します。

```
- hosts: smartcs
  tasks:
   -  smartcs_tty_command:
        tty: 1
        custom_response: on
        custom_response_delete_nl: on
        recvchar_regex:
          - '(^|¥r|¥n|!)[a-zA-Z0-9_().-]*(>|#) '
        sendchar :
          - show version
          - date
```

```
 "stdout_lines": [
      {
         "execute_command" : "show version",
         "response“ : [
              "System              : System Software Ver 2.0 (Build 2019-03-25)",
              "Boot Status         : Power on (00:01:00)",
              "System Up Time      : 2019/05/22 15:33:07",
              "Main System         : Ver 2.0",
              "Backup System       : Ver 1.2",
              "(c)NS-2250#"
         ]
       },
       {
         "execute_command" : "date",
         "response“ : [
              "Thu Sep 26 15:18:54 JST 2019",
              "(c)NS-2250#"
          ],
        }
]
```

・custom_response_delete_lastline

stdout_lines_custom の response 内容から最後の行を削除します。

```
- hosts: smartcs
  tasks:
   -  smartcs_tty_command:
        tty: 1
        custom_response: on
        custom_response_delete_nl: on
        custom_response_delete_lastline: on
        recvchar_regex:
          - '(^|¥r|¥n|!)[a-zA-Z0-9_().-]*(>|#) '
        sendchar :
          - show version
          - date
```

```
 "stdout_lines": [
      {
         "execute_command" : "show version",
         "response“ : [
              "System              : System Software Ver 2.0 (Build 2019-03-25)",
              "Boot Status         : Power on (00:01:00)",
              "System Up Time      : 2019/05/22 15:33:07",
              "Main System         : Ver 2.0",
              "Backup System       : Ver 1.2",
         ]
       },
       {
         "execute_command" : "date",
         "response“ : [
              "Thu Sep 26 15:18:54 JST 2019",
          ],
       }
]
```

＜補足＞

・本機能を有効にした場合でも、コンソールの送受信のタイミングによっては必ずしも期待したフォーマット通りに入出力が格納される保証はありません。

stdout_lines_custom を生成する元データは、stdout, stdout_lines となるため、Playbook で指定する送信文字列のタイムアウト時間を調整するなどして、意図通り Playbook が動作することを確認したうえで本オプションを指定してください。

8.1.6 使用上の注意

smartcs_tty_command モジュールを使う上での注意を記載します。

smartcs_tty_command モジュールは、SmartCS のシリアルポートに接続されている機器のコンソールに対し、指定された文字列の送受信を行います。以下の点に注意してご利用下さい。

(1) モジュールポリシー

1. コンソールの初期状態

SmartCS に接続されている機器のコンソール状態について、smartcs_tty_commnand モジュールでは管理、制御を行いません。最後に実行したコマンドによって、SmartCS に接続されている機器のコンソールは
・ログインプロンプト状態
・一般ユーザシェル状態
・管理者ユーザシェル状態
・設定投入用シェル状態
と様々な状態になり得ます。SmartCS に接続されている機器のコンソール状態を考慮して Playbook を作成して下さい。
例:必ず最後にログインプロンプト状態に戻す。プレチェック機能を使用する。等

2. コンソールの入出力結果について

SmartCS に接続されている機器のコンソールで実行された CLI コマンドの結果について、smartcs_tty_command はその実行結果のエラー有無等を自動で判別しません。
コンソール上で実行された CLI コマンドの結果に応じて ansible コマンドの実行結果(ok/failed)を制御したい場合は、
error_recvchar_regex オプション
error_detect_on_module オプション
を利用して制御を行って下さい。

(2) Playbook 作成時の注意点

1. コマンドタイムアウト時間(ansible_command_timeout)の延長

通常ネットワーク機器に対して Ansible 経由で xxx_command や xxx_config といったモジュールを実行する場合、内部では SSH による接続を行い、各コマンドを実行しています。
smartcs_tty_command モジュールは、SmartCS を介してシリアル通信で各コマンド実行の処理が行われる為、Ansible 経由でマネージドノードにコマンドを実行する際のタイムアウト時間(Playbook の task 時間)が長くなります。

その為 Playbook 内では、sendchar/src で指定する送信文字列やそのタイムアウト時間に応じて、ansible_command_timeout オプションの値を調整して下さい。

※ansible_command_timeout 値のデフォルト値は以下の通りです。
Ansible2.7 系:10 秒
Ansible2.8 系:30 秒

Playbook 例

```
---
- name: "get version and write"
  hosts: smartcs
  gather_facts: no

  tasks :
  - name : Configure  NS-2250  ipaddress  by Console
    seiko.smartcs.smartcs_tty_command :
      tty: 1-16
      recvchar :
      - '> '
      - '# '
      sendchar :
      - 'show version'
      - 'show ip'
      - 'write'
      - 'y'

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
  - ansible_user: smartcs-ansible
  - ansible_password: password
  - ansible_command_timeout: 600
```

8.2 seiko.smartcs.smartcs_command

8.2.1 概要

SmartCS の装置上で状態表示コマンド、メンテナンスコマンドを実行し、その実行結果を取得します。
このモジュールは、設定コマンドの実行をサポートしません。SmartCS の設定を行う場合は、smartcs_config モジュールを使用して下さい。

8.2.2 オプション

このモジュールのオプションについて下記に記載します。

オプション名	必須	省略時	設定範囲	内容
commands	○			SmartCS で実行するコマンドのリストを設定します。
interval		1		wait_for オプションで設定した条件が満たされなかった場合に、commands オプションで指定したコマンドの実リトライを行う際のインターバル時間（秒）を設定します。
match		all	any all	wait_for オプションで設定した条件の比較方式を設定します。 このオプションに all を設定した場合、wait_for オプションで設定した条件が全て満たされたときにタスクの実行を再開します。 any を設定した場合、条件のいずれかが満たされたときタスクの実行を再開します。
retries		10		wait_for オプションで設定した条件が満たされなかった場合に、commands オプションで指定したコマンド実行のリトライを行う回数を設定します。
wait_for				コマンドの実行結果が満たすべき条件のリストを設定します。 retries オプションで設定したリトライを行う回数を超えても条件が満たされなかった場合、タスクの実行は失敗します。

8.2.3 Playbook 例

このモジュールの Playbook 例について下記に記載します。

例1.　SmartCS で show version コマンドを実行

```
- name: run show version on SmartCS
  seiko.smartcs.smartcs_command :
      commands: show version
```

例2.　SmartCS で show version コマンドを実行し、
「Ver 2.0」という文字列が含まれるかを確認

```
- name: run show version and check to see if output contains 'Ver 2.0'
  seiko.smartcs.smartcs_command :
      commands: show version
      wait_for: result[0] contains 'Ver 2.0'
```

例3.　SmartCS で show version、show tty という複数のコマンドを実行

```
- name: run multiple command on SmartCS
  seiko.smartcs.smartcs_command :
      commands:
      - show version
      - show tty
```

例4.　SmartCS で show version、show tty という複数のコマンドを実行し、
それぞれ「Ver 2.0」、「1　　9600」という文字列が含まれているかを確認

```
- name: run multiple commands and avaluate the output
  seiko.smartcs.smartcs_command :
      commands:
      - show version
      - show tty
      wait_for:
      - result[0] contains 'Ver 2.0'
      - result[1] contains '1     9600'
```

例5.　SmartCS で、copy startup コマンドを実行する場合
（対話的なコマンドの対応）

```
  - name: run copy startup command on SmartCS
    seiko.smartcs.smartcs_command :
        commands:
        - command: 'copy startup 2 to startup 4'
          prompt: 'Do you really want to copy external startup2 to
external startup4 \[y/n\] ?'
          answer: 'y'
```

8.2.4 戻り値

このモジュールの戻り値について下記に記載します。

名前	説明	契機	タイプ
stdout_lines	コマンドの実行結果を改行文字列毎に分割したリストとなります。	コマンドの実行に成功した場合	リスト
stdout	コマンドの実行結果	コマンドの実行に成功した場合	リスト
failed_conditions	コマンド実行時に満たされなかった条件のリスト	条件が成立しなかった場合	リスト

8.3 seiko.smartcs.smartcs_config

8.3.1 概要

SmartCS に設定コマンドを実行します。

8.3.2 オプション

このモジュールのオプションについて下記に記載します。

オプション名	必須	省略時	設定範囲	内容
backup		False	boolean 値	現在のランニングコンフィグレーションのバックアップの取得を行うかを設定します。 True（yes, y, true 等）と設定した場合、バックアップファイルは Playbook の保存先ディレクトリの backup ディレクトリに保存されます。backup ディレクトリが存在しない場合、ディレクトリの作成を行います。 False（no, n, false 等）と設定した場合、バックアップの取得を行いません。
lines				SmartCS で実行する設定コマンドのリストを設定します。 対象となるコマンドは、show config ruuning で表示されるコマンドとなります。
match		line	line none	SmartCS に設定されているコンフィグレーションに対して、lines オプションで設定したコンフィグレーションを比較する際の比較方式を設定します。 このオプションに line を設定した場合、SmartCS に設定されているコンフィグレーションに対し、lines オプションで設定したコンフィグレーションをコマンド毎に比較し、装置に設定されていない場合に設定コマンドの実行を行います。 none を設定した場合、SmartCS に設定されているコンフィグレーションと lines オプションで設定されたコンフィグレーションコマンドの比較を行わずに設定コマンドを実行します。

オプション名	必須	省略時	設定範囲	内容
save_when		nerver	always never modified changed	コンフィグレーションの保存方法を設定します。 このオプションに always を設定した場合、常に write コマンドを実行し、コンフィグレーションを保存します。 modified を設定した場合、show config running の実行結果と、show config startup の実行結果を比較して、差分がある場合に write コマンドを実行し、コンフィグレーションを保存します。 changed を設定した場合、show config running の実行結果に lines で指定したコンフィグ行が設定されておらず、正常に設定が行えた場合に write コマンドを実行し、コンフィグレーションを保存します。 never を設定した場合、コンフィグレーションの保存を行いません。
src				設定対象のコンフィグレーションを記載したファイルのパスを設定します。 このオプションは ファイルの絶対パス または Playbook の保存先ディレクトリからの相対パスを設定します。 このオプションは lines オプションと排他で動作します。

8.3.3 Playbook 例

このモジュールの Playbook 例について下記に記載します。

例1.　SmartCS に接続されている TTY1のラベル名とボーレートを設定

```
- name: configuration tty 1 settings
  seiko.smartcs.smartcs_config :
    lines:
    - set pord tty 1 label SWITCH_1
    - set tty 1 baud 38400
```

例2.　SmartCS に接続されている TTY20 のラベル名とボーレートを設定後、
起動時のコンフィグと差分があったら write コマンドを実行する。

```
- name: configuration tty 20 settings and write
  seiko.smartcs.smartcs_config :
    lines:
    - set pord tty 20 label ROUTER
    - set tty 20 baud 19200
    save_when: modified
```

例3.　現在のランニングコンフィグレーションのバックアップを取得後、
SmartCS のホスト名を設定する。設定後、write コマンドを実行する。

```
- name: configuration host name and get backup file
  seiko.smartcs.smartcs_config :
    lines:
    - set hostname SmartCS_TEST1
    save_when: always
    backup: yes
```

例4.　ローカルにあるコンフィグレーションファイル(config_file.txt)に記載されている CLI コマンドを SmartCS に投入し、設定後 write コマンドを実行する。

※config_file.txt の内容

```
==================================
  1 #
  2 set hostname NS-2250-48_2
  3
==================================
```

※Playbook 記載例

```
  - name: configuration by local file and write
    seiko.smartcs.smartcs_config :
      src: config_file.txt
      save_when: changed
```

8.3.4 戻り値

このモジュールの戻り値について下記に記載します。

名前	説明	契機	タイプ
command	設定したコンフィグレーションのリスト(updates 戻り値と同じ値)	常時	リスト
updates	設定したコンフィグレーションのリスト(command 戻り値と同じ値)	常時	リスト
backup_path	バックアップファイルの絶対パス	backup オプションに yes を設定した場合	文字列

8.4 seiko.smartcs.smartcs_facts

8.4.1 概要

SmartCS から装置情報を収集します。

8.4.2 オプション

このモジュールのオプションについて下記に記載します。

オプション名	必須	省略時	設定範囲	内容
gather_subset		!config	all default tty config	収集する装置情報の種別を設定します。このオプションの設定範囲は all, default, tty, config です。 all, config 設定時は、装置管理者モードに遷移する為、下記オプションが必要です。 vars: ansible_become: yes ansible_become_method: ansible.netcommon.enable ansible_become_password: xxxxx

8.4.3 Playbook 例

このモジュールの Playbook 例について下記に記載します。

例1.　smartcs_facts モジュールで取得できる全ての情報を取得する。

```
- name: Collect all facts from the SmartCS
  seiko.smartcs.smartcs_facts :
    gather_subset: all
```

例2.　SmartCS のランニングコンフィグレーションとデフォルト情報を取得する。

```
- name: Collect only the config and default facts
  seiko.smartcs.smartcs_facts:
    gather_subset: config
```

例3.　SmartCS の TTY 情報とデフォルト情報を取得する。

```
- name: Collect only the tty and default facts
  seiko.smartcs.smartcs_facts:
    gather_subset: tty
```

例4.　SmartCS の TTY 情報以外を取得する。

```
- name: Do not collect tty facts
  seiko.smartcs.smartcs_facts:
    gather_subset: "!tty"
```

例5.　smartcs_facts モジュールのデフォルト情報を取得する。(config 以外)

```
- name: Collect default facts
  seiko.smartcs.smartcs_facts:
```

8.4.4 戻り値

このモジュールの戻り値について下記に記載します。

名前	説明	契機	タイプ
ansible_net_gather_subset	SmartCSから収集した装置情報の種別のリスト	常時	文字列
ansible_net_model	SmartCS の装置モデル情報(show version コマンド表示における装置モデル)	常時	文字列
ansible_net_hostname	SmartCS のホスト名情報(show ip コマンド表示におけるホスト名)	常時	文字列
ansible_net_tty	SmartCS の tty 設定情報 (show tty コマンド表示における、baud、bitchar、flow、parity、stop、tty 情報、show portd コマンドにおける、label 情報)	gather_subsetオプションで tty が有効な場合	リスト
ansible_net_serialnum	SmartCS のシリアル番号情報(show version コマンド表示におけるシリアル番号)	常時	文字列
ansible_net_bootrom	SmartCS の BootROM 情報	常時	文字列
ansible_net_config	SmartCS のランニングコンフィグレーション	gather_subsetオプションで config が有効な場合	文字列
ansible_net_bootconfig	SmartCS の起動時のコンフィグ情報	常時	文字列

名前	説明	契機	タイプ
ansible_net_version	SmartCS のバージョン情報（show version コマンド表示におけるバージョン名）	常時	文字列
ansible_net_mainsystem	SmartCS のメインシステム面のバージョン情報	常時	文字列
ansible_net_backupsystem	SmartCS のバックアップシステム面のバージョン情報	常時	文字列
ansible_net_bond1	SmartCS のインターフェース設定情報（show ipinterface bond1 コマンド）	常時	ハッシュ
ansible_net_eth1	SmartCS のインターフェース設定情報（show ipinterface eth1 コマンド）	常時	ハッシュ
ansible_net_eth2	SmartCS のインターフェース設定情報（show ipinterface eth2 コマンド）	常時	ハッシュ

9　章　制限事項

9.1　smartcs_tty_command モジュールで SmartCS シリーズを制御する場合

smartcs_tty_commandモジュールを使ってSmartCSシリーズ（NS-2250、NS-2240）を制御する場合、Playbook に sendchar/src で設定した文字列を送信後、SmartCS で CLI エラー（no such command や、too many parameters）が発生すると ansible コマンドや ansible-playbook コマンドが終了します。
※通常コンソール経由のコマンド実行で CLI エラーが発生しても、そのエラー出力を Playbook の実行結果として取得できます。

smartcs_tty_command を使って SmartCS シリーズに対して設定コマンドや状態表示コマンドを実行する場合、sendchar/src に設定する文字列については、CLI エラーが発生しないようにして下さい。

9.2　gather_facts による装置情報の収集

ansible2.9 から、smartcs_facts モジュールを使わなくても、装置情報の収集が可能となりました。Playbook で gather_facts: yes を指定する事で、ansible_facts という変数にその内容が出力されます。(gather_subset: all 指定と同様の情報)

smartcs_facts モジュールを使わずに、SmartCS のバージョン情報を取得する事が可能となる為、従来と比較して Playbook をより簡易に記載する事が可能となっています。

※本機能は SmartCS 用モジュールの機能拡張ではなく、Ansible2.9 によって改善された Ansible が提供する機能となります。

Playbook 例

```
---
- name: "smartcs_command with fact "
  hosts: smartcs
  gather_facts: yes

  tasks :
  - name : "run show portd tty"
    seiko.smartcs.smartcs_command :
      commands:
      - show portd tty

  - name : "smartcs version"
    debug :
      var: ansible_facts.net_version

  vars:
  - ansible_connection: ansible.netcommon.network_cli
  - ansible_network_os: seiko.smartcs.smartcs
```

SmartCS modules for Ansible v1.3.0 以降で本機能を使う場合は、以下(1)(2)のどちらかで対応が可能です。

(1) smartcs_facts を実行して情報取得する。
Playbook 作成時に、smartcs_facts を実行して、SmartCS の装置情報を取得して対応してください。「gather_facts: yes」を指定した場合、smartcs_facts の gather_subset オプションに all を指定した場合の返り値を ansible_facts 変数に格納します。

(2) FACTS_MODULES オプションを指定する。
FACTS_MODULES オプションを指定する事で、「gather_facts: yes」指定時に、SmartCS の facts 情報を収集できるようになります。
https://docs.ansible.com/ansible/latest/reference_appendices/config.html#facts-modules

Playbook 内の vars セクションで変数を指定する場合は、以下のような指定となります。

```
vars:
  - ansible_facts_modules: seiko.smartcs.smartcs_facts
```

10 章　トラブルシューティング

本章では SmartCS modules for Ansible を使用して、Playbook 実行時にエラーとなった場合の対応について説明します。
各項毎にエラー出力と考えられる対処方法について記載しています。対処方法については必ずしもエラーを解消できると限りませんが、トラブルシュート時の情報として参考になりましたら幸いです。

10.1 “Unable to connect to port 22 on x.x.x.x”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "[Errno None] Unable to connect to port 22 on x.x.x.x"
}
```

対象の SmartCS に接続できません。コントロールノードに登録している SmartCS の IP アドレス、ホスト名や、コントロールノードと SmartCS の間のネットワークをご確認下さい。
また、SmartCS の SSH サーバが有効化されていない可能性があります。以下のコマンドを実行して、SmartCS の SSH サーバを有効化して下さい。

```
(0)NS-2250# enable sshd
```

10.2 “timed out”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "timed out"
}
```

対象の SmartCS に接続試行中にタイムアウトエラーが発生して接続できません。コントロールノードと SmartCS 間のネットワークをご確認下さい。
また、SmartCS のフィルター機能でパケットが廃棄されている可能性があります。以下のコマンドを実行して、コントロールノードからのパケットが正しく SmartCS に届く設定となっているかを確認し、必要に応じて設定を追加して下さい。

```
(0)NS-2250# show ipfilter input
(0)NS-2250# create ipfilter input accept …
```

10.3 “Error reading SSH protocol banner”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "Error reading SSH protocol banner[Errno 104] Connection reset by peer"
}
```

以下の原因が考えられます。それぞれ SmartCS の設定を確認して下さい。

(1) 対象の SmartCS に接続許可が無い為、エラーが発生して接続できません。
SmartCS の接続許可設定を確認し、必要に応じて設定を追加して下さい。

```
(0)NS-2250# show allowhost
(0)NS-2250# create allowhost …
```

(2) 対象の SmartCS のシリアルポートは既に RW セッションの最大接続数となっている可能性があります。portd 設定を確認し、必要に応じて設定を変更して下さい。

```
(0)NS-2250# show portd tty
(0)NS-2250# set portd tty x limit rw 2 ro 3
```

10.4 “The authenticity of host ‘x.x.x.x’ can’t be established.”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "paramiko: The authenticity of host 'x.x.x.x' can't be established.¥nThe ecdsa-sha2-nistp521 key fingerprint is b'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'."
}
```

SmartCS の SSH ホスト鍵がコントロールノードに登録されていません。SSH 接続を行い、ホスト鍵を登録するか、ansible.cfg の”host_key_checking = False”のコメントアウトを外し、SSH ホスト鍵のチェック行わない様にする等、確認をして下さい。
詳細は「3 章 準備」を参照して下さい。

10.5 ”Authentication failed.”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "Invalid/incorrect username/password. Authentication failed."
}
```

コントロールノードから SmartCS にログインする際の認証がエラーとなっています。
SmartCS にログインするユーザのユーザ名、パスワードを確認して下さい。

10.6 “Bad authentication type”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "Invalid/incorrect username/password. ('Bad authentication type', ['publickey']) (allowed_types=['publickey'])"
```

Ansible を実行するコントロールノードから SmartCS にログインする際のユーザ認証方式に誤りがある為、認証エラーとなっています。SmartCS の SSH サーバ ユーザ認証方式をコントロールノード側と合わせて下さい。

```
(0)NS-2250# set sshd auth basic
```

10.7 “Unable to automatically determine host network os.”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "Unable to automatically determine host network os. Please manually configure ansible_network_os value for this host"
}
```

コントロールノードから SmartCS に接続する為のネットワーク OS オプションに、”smartcs”が設定されていません。Playbook の”ansible_network_os” に “seiko.smartcs.smartcs” を設定して下さい。
詳細は「4 章 Ansible Collections に対応した Playbook の作成」を参照して下さい。

10.8 “unable to elevate privilege to enable mode”

```
fatal: [x.x.x.x]: FAILED! => {
    "msg": "unable to elevate privilege to enable mode, at prompt [b'¥¥n(2)NS-2250> '] with error: su¥r¥nPassword: ¥r¥nincorrect password¥r¥n(2)NS-2250> "
}
```

SmartCS にログイン後、装置管理ユーザへの遷移が失敗しました。Playbook の”ansible_become_password”等で指定したパスワードが正しいかを確認して下さい。
詳細は「4 章 Ansible Collections に対応した Playbook の作成」を参照して下さい。

10.9 “command timeout triggered, timeout value is X secs.”

```
    },
    "msg": "command timeout triggered, timeout value is 10 secs.¥nSee the timeout setting options in the Network Debug and Troubleshooting Guide."
}
```

SmartCS へのログインや、指定したコマンドを実行する際にタイムアウトが発生し、コマンドの実行がエラーとなりました。タイムアウト発生時には、様々な要因が考えられますので、以下のタイムアウトに関するドキュメントを参照ください。

コントロールノードの Ansible のタイムアウトに関する各設定（ansible.cfg）
https://docs.ansible.com/ansible/latest/reference_appendices/config.html

コネクションプラグインである、network_cli の各設定
https://docs.ansible.com/ansible/latest/collections/ansible/netcommon/network_cli_connection.html

Network Debug and Troubleshooting Guide の Timeout issues
https://docs.ansible.com/ansible/latest/network/user_guide/network_debug_troubleshooting.html#timeout-issues

10.10 “timeout value X seconds reached while trying to send～”

```
    },
    "msg": "timeout value 10 seconds reached while trying to send command: b'ttysendwaitset tty 1 timeout 15 nl cr string ¥"show version¥""'"
}
```

Playbook で指定したコマンドの実行において、タイムアウトが発生し、コマンドの実行がエラーとなりました。

「10.9 “command timeout triggered, timeout value is X secs.”」に記載している対処方法、「8 章 モジュール」の各モジュールのオプションを参照してください。

また、smartcs_tty_command モジュールを使用している場合は、

「8.1.2 オプション」の cmd_timeout のオプション値の設定

「8.1.5 解説」で解説している各オプションの動作

「8.1.6 使用上の注意」の(2)Playbook 作成時の注意点

についても確認して下さい。

10.11 “Ignoring timeout(10) for smartcs_facts”

```
TASK [Gathering Facts] ************************************
[WARNING]: Ignoring timeout(10) for smartcs_facts
ok: [xxx.xxx.xxx.xxx]
```

Ansible2.9 からネットワークモジュールの facts 収集は、gather_facts 経由で行われるようになり、SmartCS の場合は、smartcs_facts モジュールが内部的に動作します。
その為、ansible.cfg 等で設定されている facts 情報取集のタイムアウト値である、gather_timeout (DEFAULT_GATHER_TIMEOUT) については参照せず、コネクションプラグイン(network_cli)側のタイムアウト値で動作します。

このワーニングはその内容を警告しており、Ansible2.9 で SmartCS を操作する為の各モジュールを使った際に、gather_facts: yes と指定する事で出力されてしまいますが、動作や Playbook に問題ありません。

11 章　付録 A. Ansible 環境の構築

11.1 venv による Ansible 環境の構築

Ansible 環境を構築する際、Python の仮想技術である venv を使うとコントロールノード上で動作する Python に影響を与えることなく Ansible 環境が構築可能となります。

venv を使って Ansible 環境を構築し、venv 環境に SmartCS modules for Ansible を構築する手順を記載します。

＜構築環境例＞

・CentOS7(yum で python3 パッケージを追加している状態)

```
$ sudo yum install python3
```

(1) venv による仮想環境の構築

python3 環境で、venv による仮想環境を構築します。

```
$ python3 -m venv smartcs-ansible
$ source smartcs-ansible/bin/activate
(smartcs-ansible)$
$
```

(2) Ansible と必要なパッケージのインストール

venv 環境に Ansible と必要なパッケージをインストールします。

「1 章 はじめに 1.3 動作要件」の表を参照して、SmartCS modules for Ansible のバージョンに合った Ansible のバージョンをインストールしてください

v1.3.0 以降では、Ansible Collections 形式をサポートした Ansible v2.10 以降のバージョンををインストールする必要があります。Ansible v2.10 以降は、「ansible」「ansible-base」と異なるパッケージが提供されていますが、SmartCS modules for Ansible はどちらのパッケージでも動作します。

以下のオペレーション例は、「ansible-base 2.10.6」を構築する場合の例となります。

```
(smartcs-ansible)$ pip3 install ansible-base==2.10.6
(smartcs-ansible)$ pip3 install paramiko
(smartcs-ansible)$
```

※補足①

実行する環境によっては、以下のようなワーニングが出力され、pip コマンドのアップグレードが必要となる場合があります。

ワーニング文にある通り、pip コマンドのアップグレードを行い対処します。

```
You should consider upgrading via the 'pip install --upgrade pip'
command. $
$ pip install --upgrade pip
```

※補足②

Proxy 環境下で pip コマンドを実行する場合は、以下のように、pip コマンドに Proxy オプションを付与して対処します。

```
(smartcs-ansible)$ pip3 install ansible-base==2.10.6 --proxy
x.x.x.x:xxxx
(smartcs-ansible)$ pip3 install paramiko --proxy x.x.x.x:xxxx
(smartcs-ansible)$
```

(3) インストールした Ansible の確認

ansible --version コマンドを実行し、venv で構築した Ansible 環境を確認します。

```
(smartcs-ansible)$ ansible --version
```

インストールした Ansible のバージョンが表示されている事を確認します。

(4) SmartCS 用 Ansible モジュールのインストール

Ansible 環境が構築できたら SmartCS modules for Ansible をインストールして、SmartCS 用の Ansible モジュールを使えるようにします。

インストールするバージョンによってオペレーション内容が異なりますので、以下の表を参考にインストールを実行してください。

＜提供形式とインストール手順＞

SmartCS modules for Ansible	提供形式	インストール手順
v1.0	弊社独自のパッケージ	「12 章 付録 B. v1.0～v1.2 のオペレーション」
v1.1 v1.1.1		
v1.2		
v1.3.0～	Ansible Collections 形式	「2 章 インストール」

11.2 ansible.cfg の用意

venv を使って Ansible 環境を構築した場合、Ansible を動かす為に必要な ansible.cfg ファイルが自動的に生成されないので、ファイルを用意する必要があります。
ansible.cfg は、github の Ansible リポジトリより取得が可能です。構築した Ansible のバージョンに合わせたファイルをダウンロードして利用して下さい。

(例：ansible 2.10.6（ansible-base 2.10.6）のリポジトリ情報)
https://github.com/ansible/ansible/blob/v2.10.6/examples/ansible.cfg

以下は、github から ansible.cfg を取得するオペレーションの例となります。

```
$ wget https://raw.githubusercontent.com/ansible/ansible/v2.10.6/examples/ansible.cfg
$
```

ansible.cfg ファイルに記載した設定内容は、ファイルの配置場所によって優先順位がありますので、Ansible の実行環境に合わせて配置をしてください。

https://docs.ansible.com/ansible/latest/reference_appendices/general_precedence.html#id2

12 章　付録 B. v1.0～v1.2 のオペレーション

12.1 v1.0～v1.2 のオペレーション概要

SmartCS modules for Ansible の v1.0～v1.2 では、弊社独自のパッケージを提供して Ansible 環境に SmartCS 用モジュールをインストールしていました。

SmartCS modules for Ansible	コントロールノード環境		マネージドノード環境 SmartCS ソフトウェア Ver	
	ansible	Python	NS-2250 Series	NS-2240 series
v1.0	2.7.7	2.7 以降 3.6 以降	v2.0 以降	未サポート
v1.1 v1.1.1	2.8.4		v2.1 以降	
v1.2	2.9.15	3.6.8	v2.1 以降	

※弊社独自のパッケージ(モジュール、インストーラ)で提供しています。

本項では、v1.0～v1.2 の SmartCS modules for Ansible を利用したインストール、アンインストール、バージョンアップなどの使い方について説明します。

12.2 インストール前の確認

Ansible がインストールされている事を確認下さい。もしインストールされていなければ、CentOS などの場合、yum コマンドや pip コマンド等でインストールする事が可能です。
Ansible 環境の構築については、「11 章 付録 A. Ansible 環境の構築」を参照下さい。

12.3 インストール

SmartCS modules for Ansible のインストールは下記の手順で行います。
操作は必要に応じてコントロールノードの特権ユーザで行って下さい。
venv 環境に Ansible を構築した場合は、構築した venv 環境に遷移してから以下の手順を実行した、インストールを行ってください。

※SmartCS modules for Ansible は、バージョンによりファイル名が一部異なります。
＜バージョン 1.0＞
「smartcs_modules_for_ansible.tar.gz」…バージョン名を含みません。
＜バージョン 1.1 以降＞
「smartcs_modules_for_ansible_vXXX.tar.gz」…バージョン名を含みます。
（バージョン 1.1 の場合、v110 となります。）

(1) SmartCS modules for Ansible の解凍

ダウンロードしたファイル内に含まれている、
smartcs_modules_for_ansible_vXXX.tar.gz ファイルを任意のディレクトリに配置します。

```
$ ls
smartcs_modules_for_ansible_vXXX.tar.gz
$
$ tar zxvf smartcs_modules_for_ansible_vXXX.tar.gz
smartcs_modules_for_ansible_vXXX/
smartcs_modules_for_ansible_vXXX/readme
smartcs_modules_for_ansible_vXXX/install_smartcs_modules.sh
smartcs_modules_for_ansible_vXXX/COPYING
smartcs_modules_for_ansible_vXXX/smartcs_modules.tar.gz
$
```

(2) SmartCS modules for Ansible のインストール

解凍した際に作成された smartcs_modules_for_ansible_vXXX ディレクトリに移動して、install_smartcs_modules.sh を実行します。
SmartCS modules for Ansible は、コントロールノードの Ansible がインストールされている Python 配下にインストールされます。
正常にインストールされると、プロンプトだけが表示されます。

```
$ ls
smartcs_modules_for_ansible_vXXX
smartcs_modules_for_ansible_vXXX.tar.gz
$
$ cd smartcs_modules_for_ansible_vXXX/
$
$ ls
COPYING  install_smartcs_modules.sh  readme  smartcs_modules.tar.gz
$
$ sudo ./install_smartcs_modules.sh install
$
```

(3) インストールの確認

インストールされた SmartCS modules for Ansible のバージョンを確認します。

```
$ ./install_smartcs_modules.sh version
RUNNING version        : 1.x (rxxxx).
$
```

12.4 バージョンアップ

SmartCS modules for Ansible のバージョンアップは、以下に記載の手順で実行して下さい。
操作は必要に応じてコントロールノードの特権ユーザで行って下さい。

(1) インストール済みの SmartCS modules for Ansible をアンインストールする
アンインストールの手順は「12.5 アンインストール」を参照して下さい。

(2) Ansible をバージョンアップする
新しくインストールする SmartCS modules for Ansible が対応している Ansible のバージョンを確認し、バージョンアップを実行して下さい。対応しているバージョンは、「1.3 動作要件」の項を参照して下さい。

(3) 新しい SmartCS modules for Ansible をインストールする
インストールの手順は「12.3 インストール」を参照して下さい。
インストール後は、「12.3 (3)インストールの確認」手順と同様に、
「./install_smartcs_modules.sh version」コマンドを実行してバージョン名の確認を行って下さい。

12.5 アンインストール

SmartCS modules for Ansible のアンインストールは下記の手順で行います。
操作は必要に応じてコントロールノードの特権ユーザで行って下さい。

(1) バージョンの確認

インストールされている SmartCS modules for Ansible のバージョンを確認します。

```
$ ./install_smartcs_modules.sh version
RUNNING version        : 1.x (rxxxx).
$
```

(2) SmartCS modules for Ansible のアンインストール

確認したバージョン名を含む smartcs_modules_for_ansible_vXXX.tar.gz ファイルを解凍した際に作成された、smartcs_modules_for_ansible_vXXX ディレクトリに移動して、install_smartcs_modules.sh を実行します。

```
$ ls
smartcs_modules_for_ansible_vXXX
smartcs_modules_for_ansible_vXXX.tar.gz
$
$ cd smartcs_modules_for_ansible_vXXX/
$
$ ls
COPYING  install_smartcs_modules.sh  readme  smartcs_modules.tar.gz
$
$ sudo ./install_smartcs_modules.sh uninstall
$
```

Ansible のバージョンアップを行う場合などは、一度 SmartCS modules for Ansible をアンインストールして、Ansible のバージョンアップ後に再度インストールを行って下さい。

12.6 コマンドリファレンス（install_smartcs_modules）

SmartCS modules for Ansible の v1.0～v1.2 までに同梱していたインストーラコマンド(install_smartcs_module)について説明します。
操作は必要に応じてコントロールノードの特権ユーザで行って下さい。

機　　能　SmartCS Module for Ansible のインストール・アンインストールやバージョン情報の表示を行います。

フォーマット　**install_smartcs_modules.sh { install [*Ansible_Root*] | uninstall [*Ansible_Root*] | version [*Ansible_Root*] | package }**

パラメータ

install [*Ansible_Root*]
SmartCS modules for Ansible のインストールを行います。
Ansible_Root を指定しない場合、ansible --version コマンドを実行した際に表示される"ansible python module location"にインストールを行います。
*Ansible_Root*を指定した場合、指定したパス配下にファイルを展開します。

uninstall [*Ansible_Root*]
SmartCS modules for Ansible のアンインストールを行います。
Ansible_Root を指定しない場合、ansible --version コマンドを実行した際に表示される"ansible python module location"からアンインストールを行います。
*Ansible_Root*を指定した場合、指定したパス配下からファイルを削除します。

version [*Ansible_Root*]
インストールされているSmartCS modules for Ansibleのバージョン名を表示します。
Ansible_Root を指定しない場合、ansible --version コマンドを実行した際に表示される"ansible python module location"にインストールされているバージョン名を表示します。
*Ansible_Root*を指定した場合、指定したパス配下に展開されているバージョン名を表示します。

package
install_smartcs_modules.sh 実行するパス上にあるファイルのバージョン名を表示します

使 用 例　　SmartCS modules for Ansible のインストール

```
$ sudo ./install_smartcs_modules.sh install
$
```

インストールされている SmartCS modules for Ansible の
バージョン情報の表示

```
$ ./install_smartcs_modules.sh version
RUNNING version        : 1.x (rxxxx).
$
```

13 章　付録 C. Playbook での文字列の取り扱いについて

13.1 指定可能な文字種

SmartCS modules for Ansible に含まれる各モジュールについて、Playbook で設定可能な文字種を記載します。

表の記載内容について

・hex　　:16 進数を表します。

・character　　:指定できる文字について表します。

・text　　:Playbook でオプション値を設定する際に、
シングルクォーテーションやダブルクォーテーションで
囲まない設定方法を表します。
（例）

```
sendchar :
- show version
```

・single quotation:Playbook でオプション値を指定する際に、
シングルクォーテーションで囲む設定方法を表します。
（例）

```
sendchar :
- 'show version'
```

・double quotation:Playbook でオプション値を指定する際に、
ダブルクォーテーションで囲む定方法を表します。
（例）

```
sendchar :
- "show version"
```

smartcs_tty_command で文字列の送受信に使う各オプションにおける設定可能な文字種は以下の通りとなります。

○　　:送信可能

×　　:送信不可

それ以外:表内に記載の通り

hex	character	sendchar			recvchar recvchar_regex error_recvchar_regex		
		text	single quotation	double quotation	text	single quotation	double quotation
0x20	SPACE	×	○	○	×	○	○
0x21	!	×	○	○	×	○	○
0x22	“	×	○	\(0x5c) を付与	×	○	\(0x5c) を付与
0x23	#	×	○	○	×	○	○
0x24	$	○	○	○	○	○	○
0x25	%	×	○	○	×	○	○
0x26	&	×	○	○	×	○	○
0x27	‘	×	‘(0x27) を付与	○	×	‘(0x27) を付与	○
0x28	(	○	○	○	○	○	○
0x29	)	○	○	○	○	○	○
0x2a	*	×	○	○	×	○	○
0x2b	+	○	○	○	○	○	○
0x2c	,	×	○	○	×	○	○
0x2d	-	×	○	○	×	○	○
0x2e	.	○	○	○	○	○	○
0x2f	/	○	○	○	○	○	○
0x30	0	○	○	○	○	○	○
0x31	1	○	○	○	○	○	○
0x32	2	○	○	○	○	○	○
0x33	3	○	○	○	○	○	○
0x34	4	○	○	○	○	○	○
0x35	5	○	○	○	○	○	○
0x36	6	○	○	○	○	○	○

hex	character	sendchar	recvchar

					recvchar_regex error_recvchar_regex		
		text	single quotation	double quotation	text	single quotation	double quotation
0x37	7	○	○	○	○	○	○
0x38	8	○	○	○	○	○	○
0x39	9	○	○	○	○	○	○
0x3a	:	×	○	○	×	○	○
0x3b	;	○	○	○	○	○	○
0x3c	<	○	○	○	○	○	○
0x3d	=	×	○	○	×	○	○
0x3e	>	×	○	○	○	○	○
0x3f	?	×	○	○	×	○	○
0x40	@	×	○	○	×	○	○
0x41	A	○	○	○	○	○	○
0x42	B	○	○	○	○	○	○
0x43	C	○	○	○	○	○	○
0x44	D	○	○	○	○	○	○
0x45	E	○	○	○	○	○	○
0x46	F	○	○	○	○	○	○
0x47	G	○	○	○	○	○	○
0x48	G	○	○	○	○	○	○
0x49	I	○	○	○	○	○	○
0x4a	J	○	○	○	○	○	○
0x4b	K	○	○	○	○	○	○
0x4c	L	○	○	○	○	○	○
0x4d	M	○	○	○	○	○	○
0x4e	N	○	○	○	○	○	○
0x4f	O	○	○	○	○	○	○
0x50	P	○	○	○	○	○	○
0x51	Q	○	○	○	○	○	○
0x52	R	○	○	○	○	○	○
0x53	S	○	○	○	○	○	○
0x54	T	○	○	○	○	○	○

hex	character	sendchar			recvchar recvchar_regex error_recvchar_regex		
		text	single quotation	double quotation	text	single quotation	double quotation
0x55	U	○	○	○	○	○	○
0x56	V	○	○	○	○	○	○
0x57	W	○	○	○	○	○	○
0x58	X	○	○	○	○	○	○
0x59	Y	○	○	○	○	○	○
0x5a	Z	○	○	○	○	○	○
0x5b	[	×	○	○	×	○	○
0x5c	\	○	○	\(0x5c) を付与	○	○	\(0x5c) を付与
0x5d	]	×	○	○	×	○	○
0x5e	^	○	○	○	○	○	○
0x5f	_	○	○	○	○	○	○
0x60	`	×	○	○	×	○	○
0x61	a	○	○	○	○	○	○
0x62	b	○	○	○	○	○	○
0x63	c	○	○	○	○	○	○
0x64	d	○	○	○	○	○	○
0x65	e	○	○	○	○	○	○
0x66	f	○	○	○	○	○	○
0x67	g	○	○	○	○	○	○
0x68	h	○	○	○	○	○	○
0x69	i	○	○	○	○	○	○
0x6a	j	○	○	○	○	○	○
0x6b	k	○	○	○	○	○	○
0x6c	l	○	○	○	○	○	○
0x6d	m	○	○	○	○	○	○
0x6e	n	○	○	○	○	○	○
0x6f	o	○	○	○	○	○	○
0x70	p	○	○	○	○	○	○
0x71	q	○	○	○	○	○	○

hex	character	sendchar			recvchar recvchar_regex error_recvchar_regex		
		text	single quotation	double quotation	text	single quotation	double quotation
0x72	r	○	○	○	○	○	○
0x73	s	○	○	○	○	○	○
0x74	t	○	○	○	○	○	○
0x75	u	○	○	○	○	○	○
0x76	u	○	○	○	○	○	○
0x77	w	○	○	○	○	○	○
0x78	x	○	○	○	○	○	○
0x79	y	○	○	○	○	○	○
0x7a	z	○	○	○	○	○	○
0x7b	{	×	○	○	×	○	○
0x7c	|	×	○	○	×	○	○
0x7d	}	×	○	○	×	○	○
0x7e	~	×	○	○	×	○	○

ダブルクォーテーション(0x22)とバックスラッシュ(0x5c)を recvchar、recvchar_regex、error_recvchar_regex オプションで指定する場合、シングルクォーテーションで囲めば問題なく設定できますが、ダブルクォーテーションで囲む場合、Playbook 内では以下の様に設定する必要があります。

```
  seiko.smartcs.smartcs_tty_command:
    tty: 1
    cmd_timeout : 10
    recvchar_regex :
    - “(^|\\n|\\r)SmartCS> ”   <- 先頭文字を“SmartCS> ”と設定する
例
```

ダブルクォーテーションとバックスラッシュは正規表現を設定する場合に良く使う文字となっている為、正規表現を設定する場合には、シングルクォーテーションで囲んで設定する事を推奨します。

13.2 様々な文字種を送信する場合

SmartCS modules for Ansible に含まれる各モジュールで設定できる各オプションの中には、以下のように、特定の文字列やコマンド、正規表現を設定したい場合があります。

・任意のコマンド(送信したいコマンド文字列)を設定するもの
- smartcs_command の commands オプション
- smartcs_config の lines オプション
- smartcs_tty_command の sendchar、recvcar オプション

・正規表現を設定するもの
- smartcs_tty_command の、recvchar_regex、error_recvchar_regex オプション

これらのオプションを設定する場合の例、及び注意点を下記に記載します。

(1) オプションの設定値に、シングルクォーテーションやカンマを含むもの

例1:シングルクォーテーションを含む文字列を設定する場合

```
Version '1.0 (2019/xx/yy)'
```

上記の文字列を Playbook に記載する場合、文字列をダブルクォーテーションで囲んで設定して下さい。

```
seiko.smartcs.smartcs_tty_command:
  tty: 1
  recvchar :
  - "Version '1.0 (2019/xx/yy)'"
  sendchar :
  - 'show version'
```

例2:カンマを含む文字列を設定する場合

```
set tty 1,3,16 baud 115200
```

上記の文字列を Playbook に記載する場合、文字列をダブルクォーテーションで囲んで設定して下さい。

```
seiko.smartcs.smartcs_config:
  lines :
  - "set tty 1,3,16 baud 115200"
```

(2) オプションの設定値に、ダブルクォーテーションやバックスラッシュ(\)を含むもの

例1:ダブルクォーテーションを含む文字列を設定する場合

```
set portd tty 1 label "SWITCH A"
```

上記の設定コマンドを Playbook に記載する場合、コマンドをシングルクォーテーションで囲んで設定して下さい。

```
  seiko.smartcs.smartcs_config:
    lines:
    - 'set portd tty 1 label "SWITCH A"'
```

例2:バックスラッシュを含む文字列を設定する場合

```
(\r|\n|^)SmartCS
```

上記の設定コマンドを Playbook に記載する場合、正規表現をシングルクォーテーションで囲んで設定して下さい。

```
  seiko.smartcs.smartcs_tty_command:
    tty: 10
    recvchar_regex :
    - '(\r|\n|^)SmartCS'
    sendchar :
    -  'show version'
```

(3) オプションの設定値に、シングルクォーテーション、ダブルクォーテーションの両方を含むもの

例1:シングルクォーテーションとダブルクォーテーションを含む文字列を設定する場合

```
set logd tty 10 mail 1 subject "mail test 'A'"
```

上記の設定コマンドを Playbook に記載する場合、設定値全体をダブルクォーテーションで囲み、送信したいダブルクォーテーションをバックスラッシュでエスケープして設定して下さい。

```
    seiko.smartcs.smartcs_config:
      lines:
      - "set logd tty 10 mail 1 subject \"mail test 'A' \""
```

13.3 正規表現を設定する

smartcs_tty_commandモジュールを利用する際に、以下のオプション内において正規表現で設定を記載する事が可能です。

・recvchar_regex

・error_recvchar_regex

・initial_prompt

これらのオプションで設定できる正規表現について以下に記載します。

(1) 単一文字とマッチする表現

.	任意の1文字にマッチします。
[…]	(…は任意の文字)指定された任意の1文字にマッチします。
[^…]	(…は任意の文字)指定されていない任意の 1 文字にマッチします。
\k	(k が非英数字文字) 文字としてマッチします。
\d	0 から 9 の数字 1 文字にマッチします。
\D	\d 以外の 1 文字にマッチします。
\s	いずれかの空白文字にマッチします。
\S	\s 以外の 1 文字にマッチします。
\w	英数字と"_"(アンダーバー)の 1 文字にマッチします。
\W	\w 以外の 1 文字にマッチします。
\r	CR(0x0d)にマッチします。
\n	LF(0x0a)にマッチします。

(2) 付加することで反復したマッチを表す表現

*	0 回以上の反復マッチとなります。
+	1 回以上の反復マッチとなります。
?	0 回か 1 回のマッチとなります。
{*m*}	(*m* は 0 以上の整数)ちょうど *m* 回の反復マッチとなります。
{*m*,}	(*m* は 0 以上の整数) *m* 回以上の反復マッチとなります。
{*m*,*n*}	(*m*,*n* はそれぞれ 0 以上の整数) *m* 回から *n* 回までの反復マッチとなります。

(3) その他の表現

(re)	(re はあらゆる正規表現)re にマッチします。
|	この記号によって隔てられているいずれかの表現とマッチします。
[0-9]	0 から 9 の数字 1 文字にマッチします。
[a-z]	a から z の英字 1 文字にマッチします。
[A-Z]	A から Z の英字1文字にマッチします。

(4) 組合せの表現

(^|\n|\r)	行の先頭とマッチします。

(5) 正規表現の記述例

英字(大文字/小文字)、数字、記号(_(アンダーバー)、.(ドット)、-(ハイフン))から成る、複数種類のプロンプトを待ち受けたい場合。

＜マッチする文字列＞

例 : SmartCS_01> 、 SmartCS_01(config)# 、 SmartCS_01(config-if)# 、 SmartCS_01(config-line)#

```
recvchar_regex：
  - "(^|\\n|\\r)[a-zA-Z0-9_.-]*(\\(config)*(-if|-line)*\\)*(>|#)"
```

13.4 実行結果の出力文字について

Ansible の実行結果（stdout,stdout_lines）において、以下に記載する特定の文字は表の通り変換された状態で出力されます。

hex	character	stdout_lines	stdout
0x09	HT(\t)	\t	\t
0X0a	LF(\n)	2 回改行	\n\n
0x0c	FF(\f)	\f	\f
0x0d	CR(\r)	1 回改行	\n
0x22	“	\”	\”
0x5c	\	\\	\\

14 章　付録 D. SmartCS 用モジュールを便利に使う為の TIPS

14.1 src オプションを使用して送信文字列を指定する場合

sendchar オプションを使用して送信文字列を指定する際、ダブルクォーテーションやシングルクォーテーションで囲んで文字列を送信する事を推奨しています。src オプションを使い外部ファイルにて送信文字列を指定する場合は、クォーテーションで囲まずに記述する必要があります。

例 1：sendchar オプションを使用し送信文字列を指定する場合

```
sendchar：
  - 'somebody'
  - '__NL__'
  - 'su'
  - '__NL__'
  - 'set user testusr password'
  - ' !pass'
  - ' !pass'
  - 'exit'
  - 'exit'
```

例 2：src オプションを使用し、外部ファイルへの記述で送信文字列を指定する場合

```
somebody
__NL__
su
__NL__
set user testusr password
!pass
!pass
exit
exit
```

14.2 複数の SmartCS に接続されている機器に同時に文字列を送信する場合

/etc/ansible/ansible.cfg 内の forks の値を、同時に文字列を送信したい SmartCS に接続されている機器の数を考慮した値に変更して下さい。
例えば、本装置のシリアルポートに接続されている 48 台の SmartCS に接続されている機器に同時に文字列を送信する場合は、forks の値を 48 以上に設定する必要があります。なお、デフォルト値は 5 になっています。

15 章　ライセンス

15.1 第三者ソフトウェアライセンス

本項では、本ソフトウェアで利用している第三者ソフトウェアライセンスについて説明します。

SmartCS modules for Ansible のソースコードについては GitHub で公開しています。
https://github.com/ssol-smartcs/ansible-collections

弊社にて変更したソースコードをご要望のお客様は弊社までお問い合わせください。

GPLv3 のライセンス

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not

price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to

control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided

you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has

been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However,

nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or

hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify

or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have

permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively

state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year> <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program> Copyright (C) <year> <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

15.2 Ansible Collections パッケージの作成

SmartCS modules for Ansible の Ansible Collections 形式のパッケージ作成方法について説明します。
ソースについては GitHub で公開しています。
Ansible2.10 が実行可能な環境で、GitHub よりソースをダウンロード後、ansible-galaxy コマンドを実行する事で、Ansible Collections 形式のパッケージを作成する事ができます。

```
$ git clone https://github.com/ssol-smartcs/ansible-collections/seiko.smartcs
$
$ ansible-galaxy collection build seiko.smartcs
$
```

SEIKO

セイコーソリューションズ株式会社
〒261-8507 千葉県千葉市美浜区中瀬 1-8
support@seiko-sol.co.jp