Cisco Intersight 実践編：

Intersight の初期設定から、Ansible / ServiceNow 連携まで、まるっと具体的な使い方をお見せします！

シスコシステムズ合同会社
データセンター / 仮想化事業部
相園 沙貴
2020年12月16日

アジェンダ

- Intersight 概要
- Intersight 初期設定
- Intersight によるサーバ運用管理
- Ansible 自動化
- Connected TAC
- ServiceNow 連携
- Intersight Workload Optimizer (IWO)

Intersight 概要

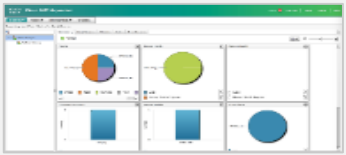
Cisco Intersight

Cisco データセンター製品である、UCS、HyperFlex、コンバージドインフラ、
3rd Party 製品のインテリジェントなライフサイクル管理

管理を Intersight 一つに統合し、シンプルに。

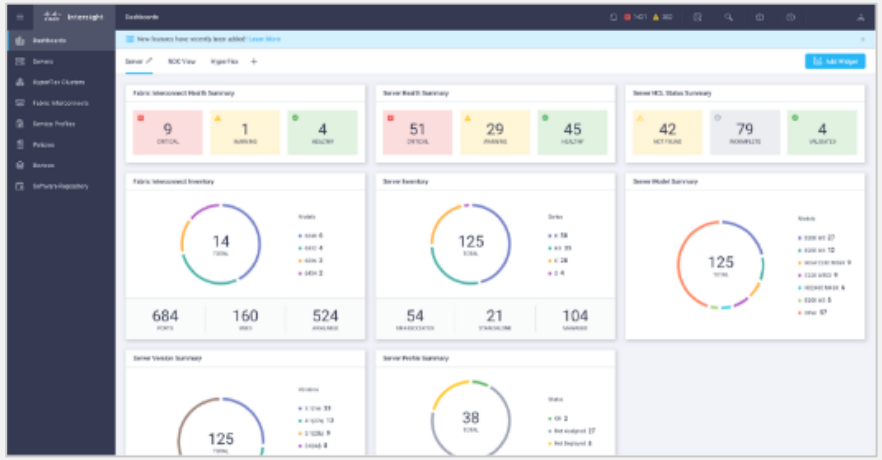
Operations

Remote/ Branch/ Edge



Performance

Single Interface / Single API



Automation / Orchestration

Multi-Site

Hyperconverged

Intersight 初期設定

- アカウント作成
- デバイス登録
- ユーザ、ロール設定

Intersight 初期設定

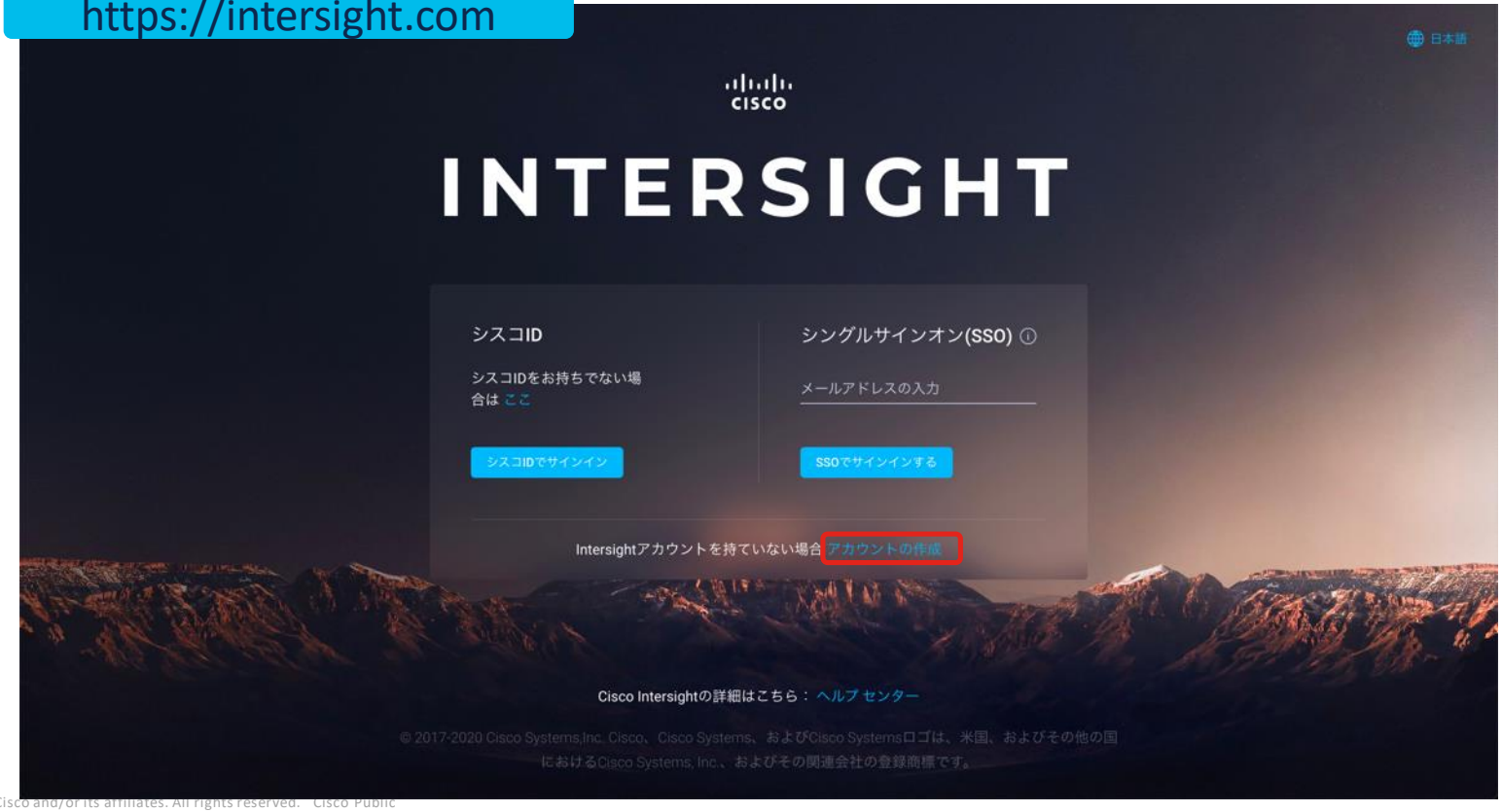
- アカウント作成
- デバイス登録
- ユーザ、ロール設定

Intersight アカウント作成に必要なもの

- **シスコ ID**
  - シスコ ID をお持ちでない場合は、こちらから作成します。
    https://www.cisco.com/c/m/ja_jp/partners/tools/how-to-make-ccoid.html
  - 現在は、シスコ ID があればまずはアカウントの作成が可能です。
- **Intersight に登録するシスコデバイス**
  - デバイスとは UCSM/IMC/HX Connect 等を指します。
  - デバイスは Intersight に通信出来る状態にしておく必要があります。
  - サーバ、ファブリックインターコネクトは管理 IP アドレス、DNS 設定を済ませておきます。
  - アカウントが作成されたら、デバイスを登録します。

1. Intersight ポータルにアクセスし、”アカウントの作成” をクリック

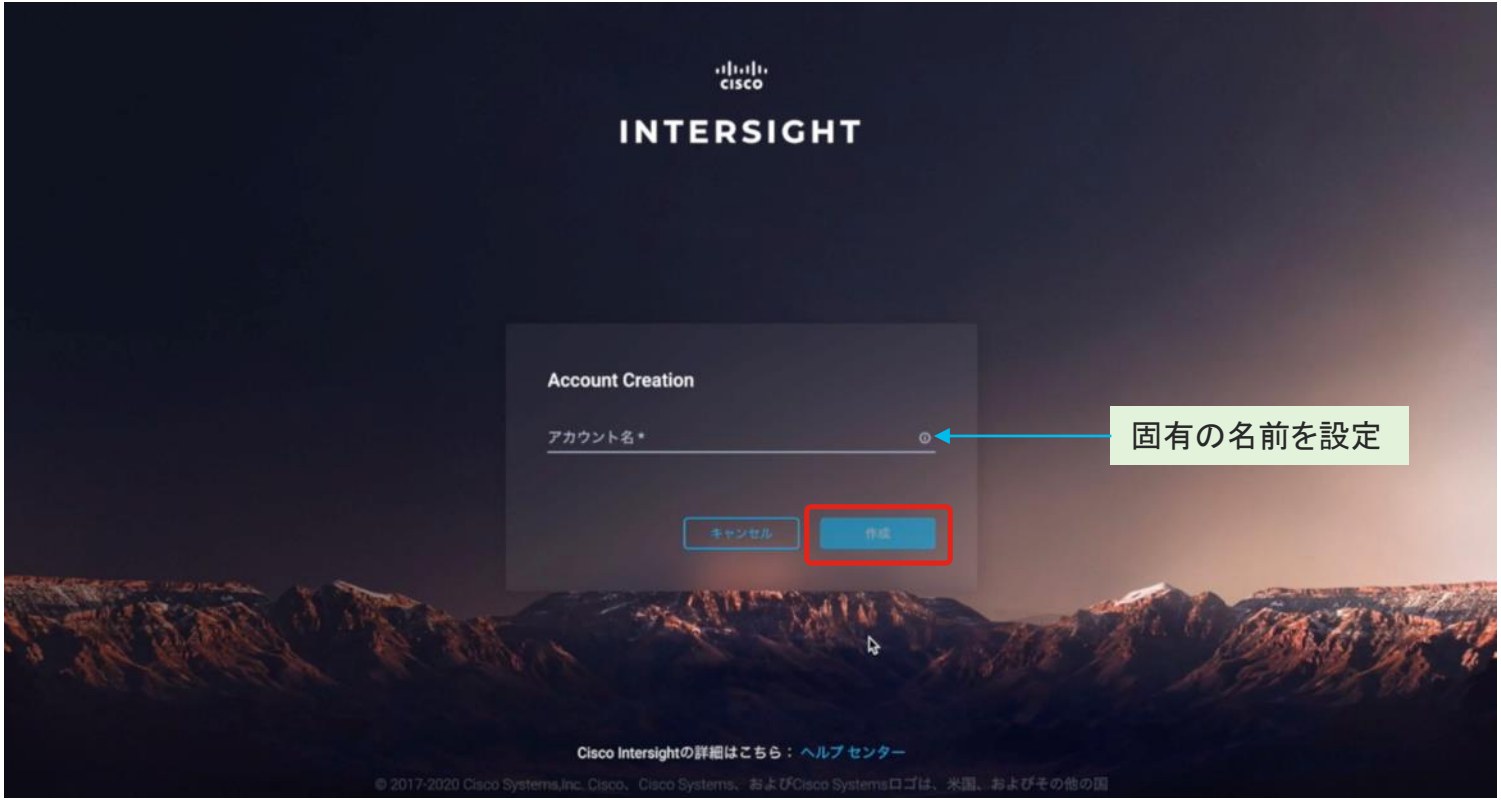
https://intersight.com



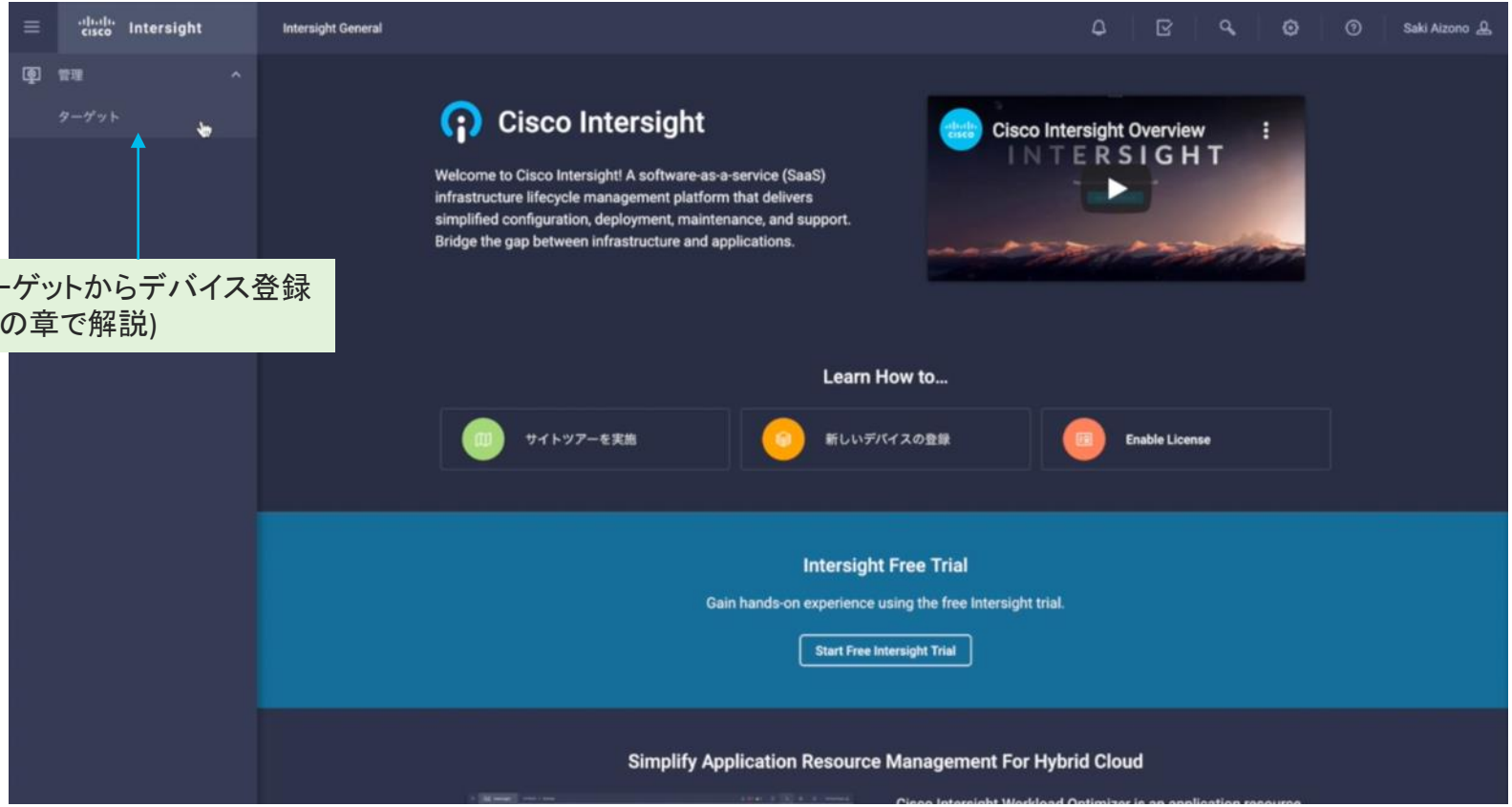
2. シスコ ID を入力してサインイン

3. サービス内容をよく読み、同意した後に “次へ” をクリック

4. アカウント名を入力してアカウントを作成



5. アカウント名・デバイスID・要求コードを入力してアカウントを作成



ターゲットからデバイス登録
(次の章で解説)

次回以降は、シスコ ID でログイン可能

ログイン画面にて、シスコ ID を選択

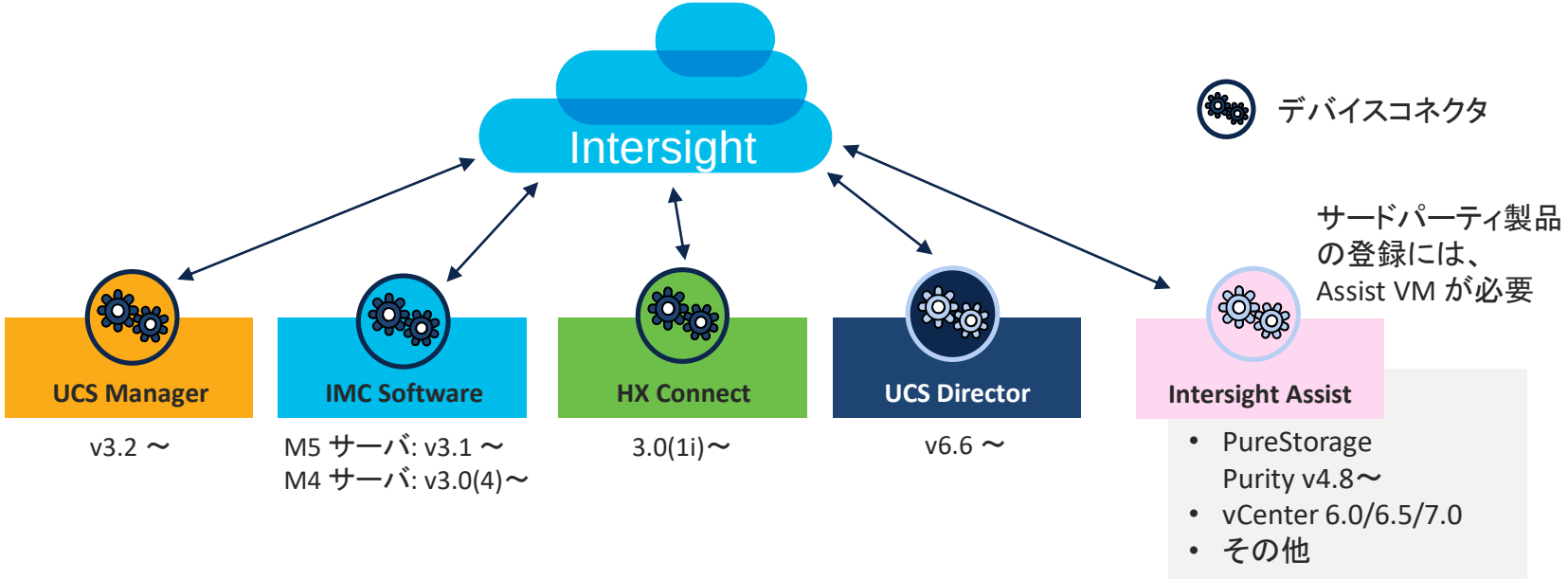
自身のアカウントに紐づく Intersight アカウントが表示される。

Intersight 初期設定

- アカウント作成
- デバイス登録
- ユーザ、ロール設定

Intersight へのデバイス登録 (デバイスクレーム)

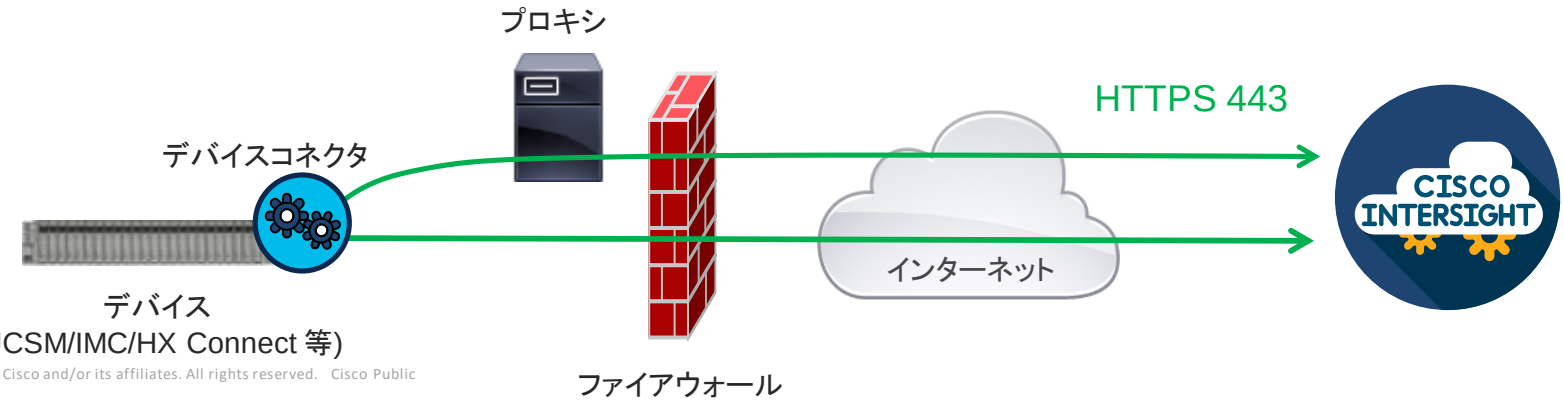
シスコデータセンター製品の管理ソフトウェアにプラグインされる “デバイスコネクタ” 機能により Intersight と通信



サポート詳細: https://www.intersight.com/help/supported_systems#supported_hardware_systems_and_software_versions

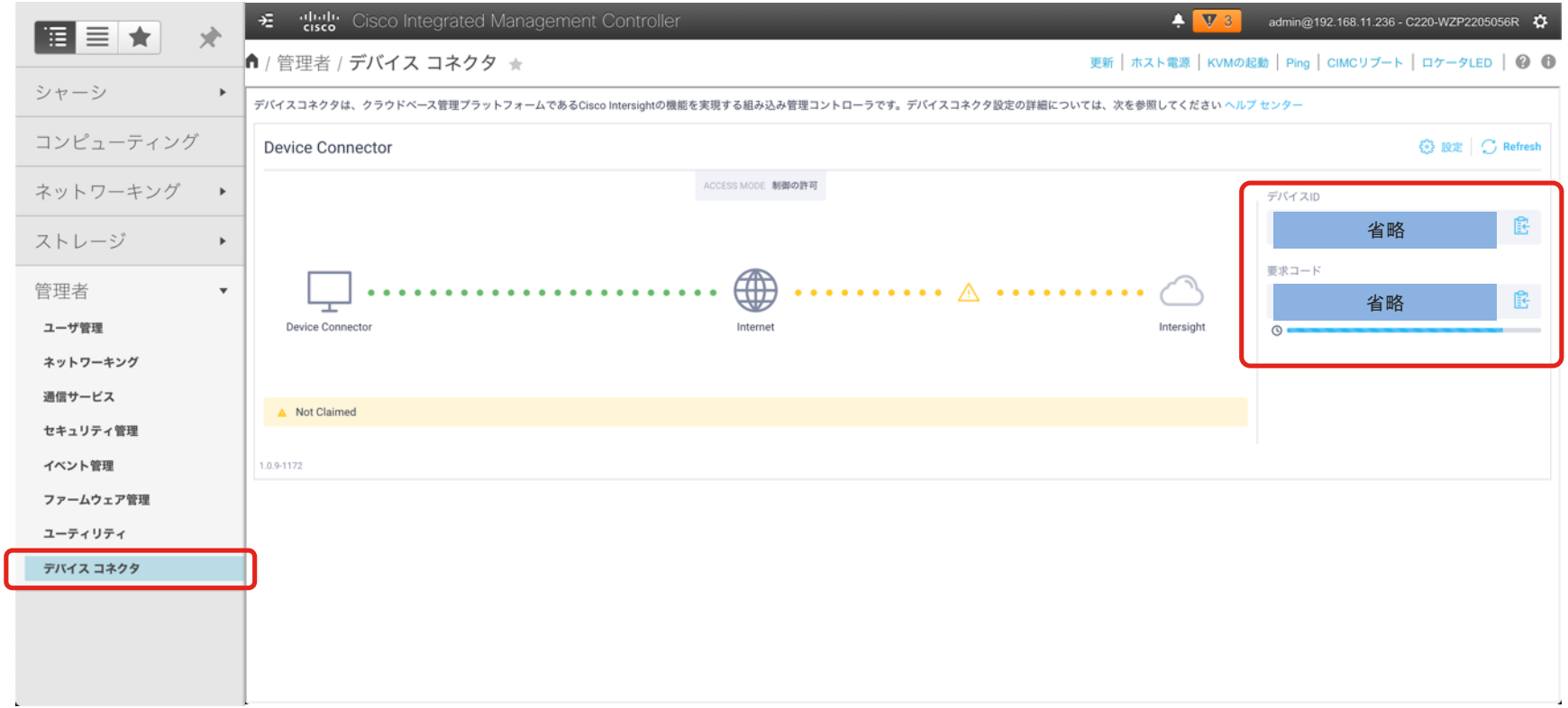
デバイス通信要件

- Long Lived Web Socket (HTTPS 443)
- デバイス側から開始する通信のみ
- HTTPS Proxy 環境もサポート
- DNS 必須
  'svc.intersight.com' (推奨) もしくは 'svc.ucs-connect.com' (将来非推奨となる予定) の解決が必要



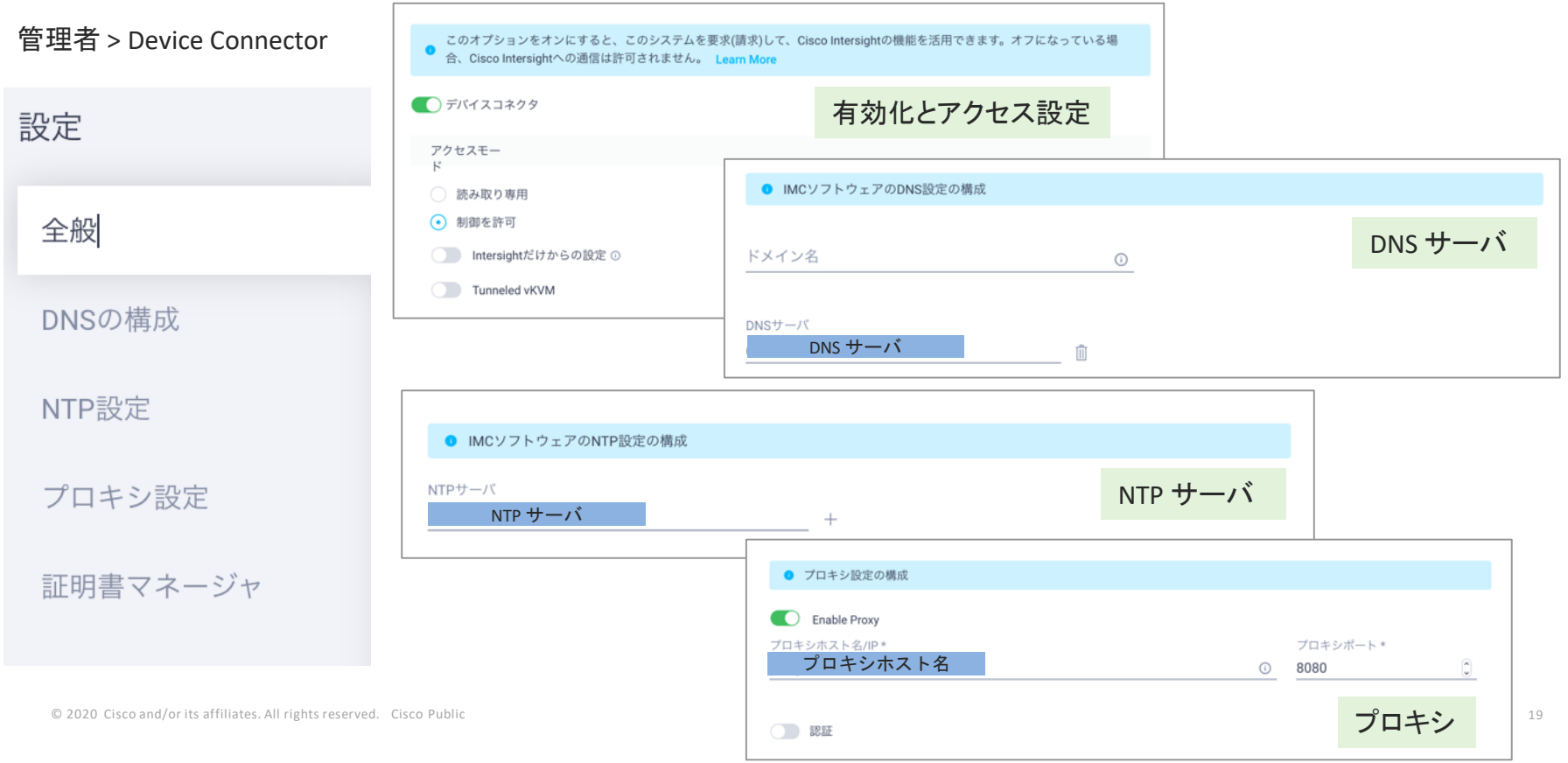
デバイス ID・要求コードの取得方法 (Cisco IMC)

管理者 > Device Connector



デバイスコネクタの設定項目 - Cisco IMC

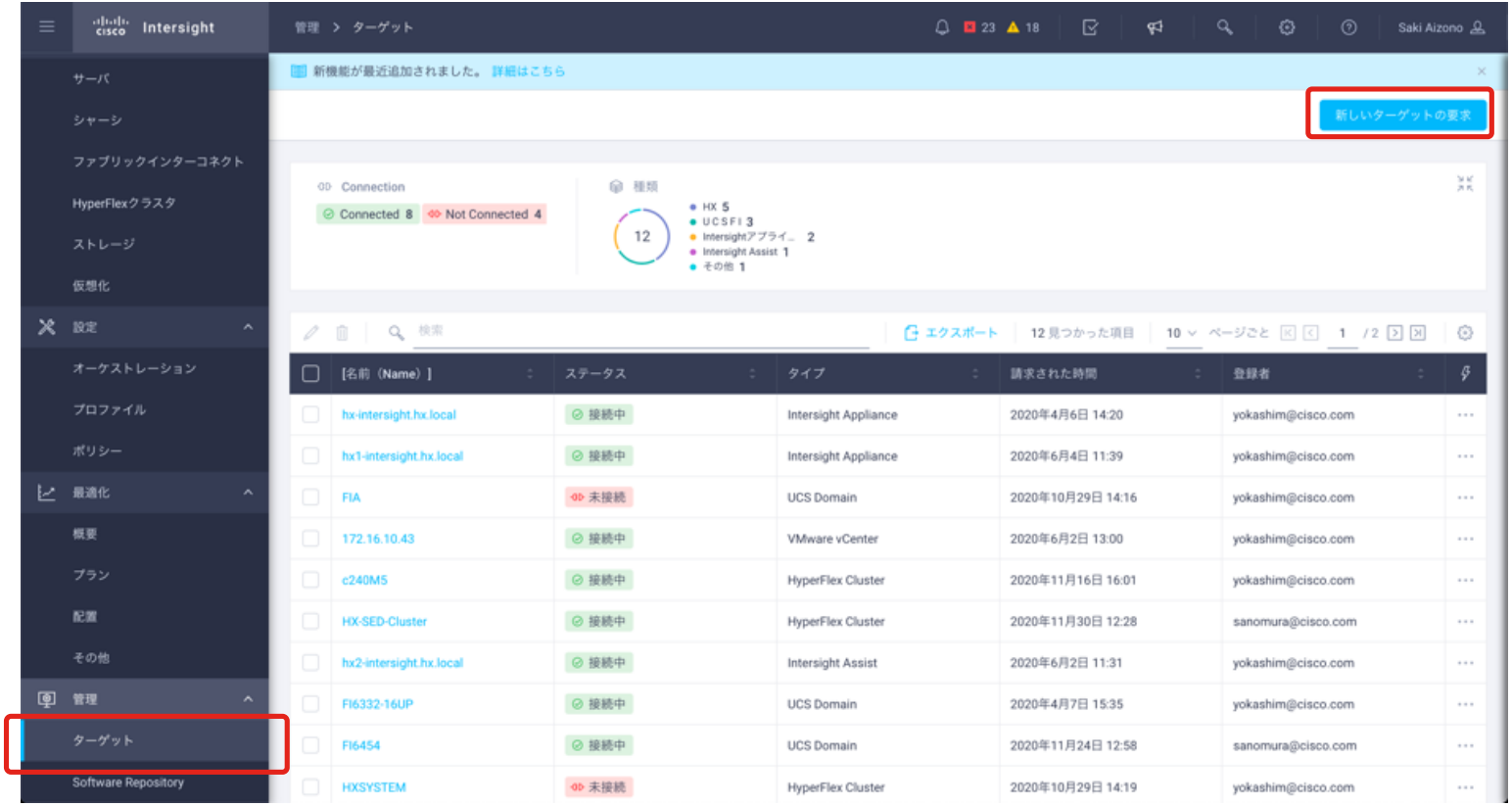
管理者 > Device Connector



デバイス ID・要求コードの取得方法 (UCS Manager)

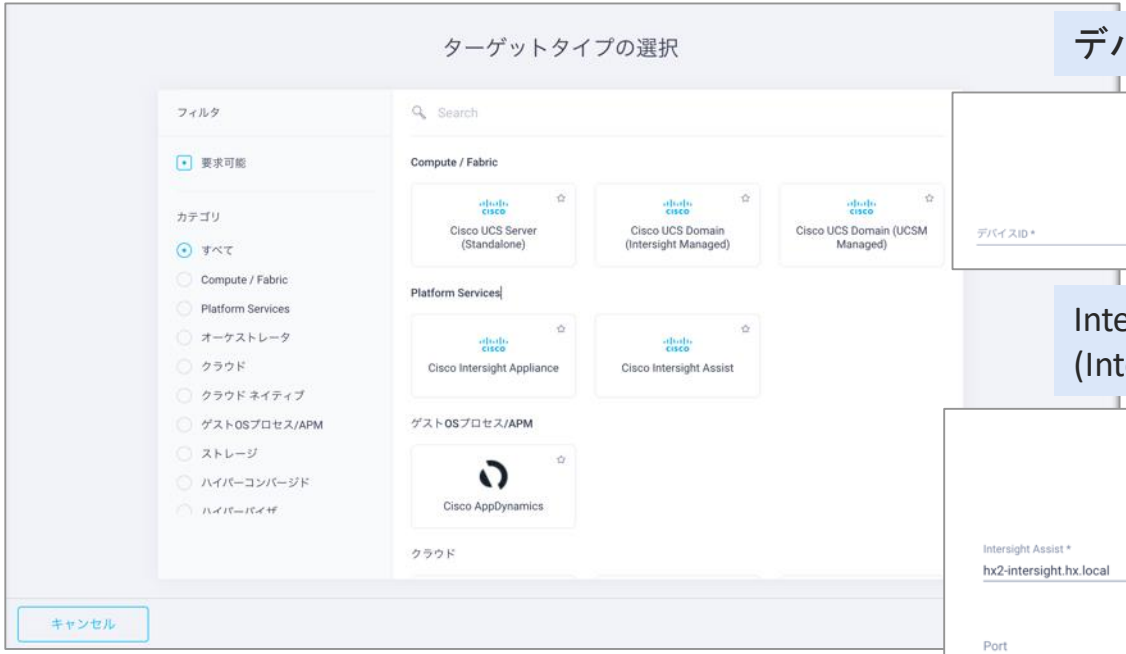
管理者アイコン>Device Connector

1. ターゲットメニューから “新しいターゲットの要求” をクリック



※ Intersight Workload Optimizer のリリースに伴い、”デバイス” タブから “ターゲット” タブに名前が変更されているのでご注意ください。

2. ターゲットタイプを選択し、デバイス ID と要求コードを入力



ターゲットタイプから該当するものを選択

デバイスコネクタによる登録の例

Intersight Assist による登録の例
(Intersight Assist 登録後に利用可能)

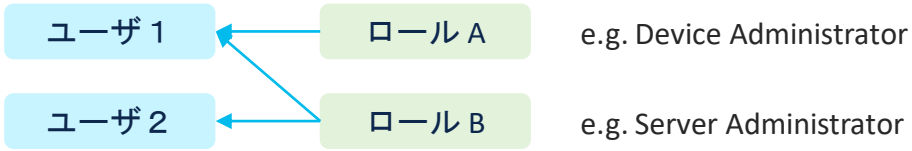
Intersight 初期設定

- アカウント作成
- デバイス登録
- ユーザ、ロール設定

ユーザーとロール

- Intersight では、ロールベースアクセス制御 (RBAC) をサポート
- ID プロバイダ (IdP) を利用してユーザ、グループの追加が可能
- 外部 ID プロバイダの利用も可能 (SSO)
- ユーザおよびグループには、複数ロールを設定可能
- システム定義のロールに加え、ユーザがカスタムロールを作成可能

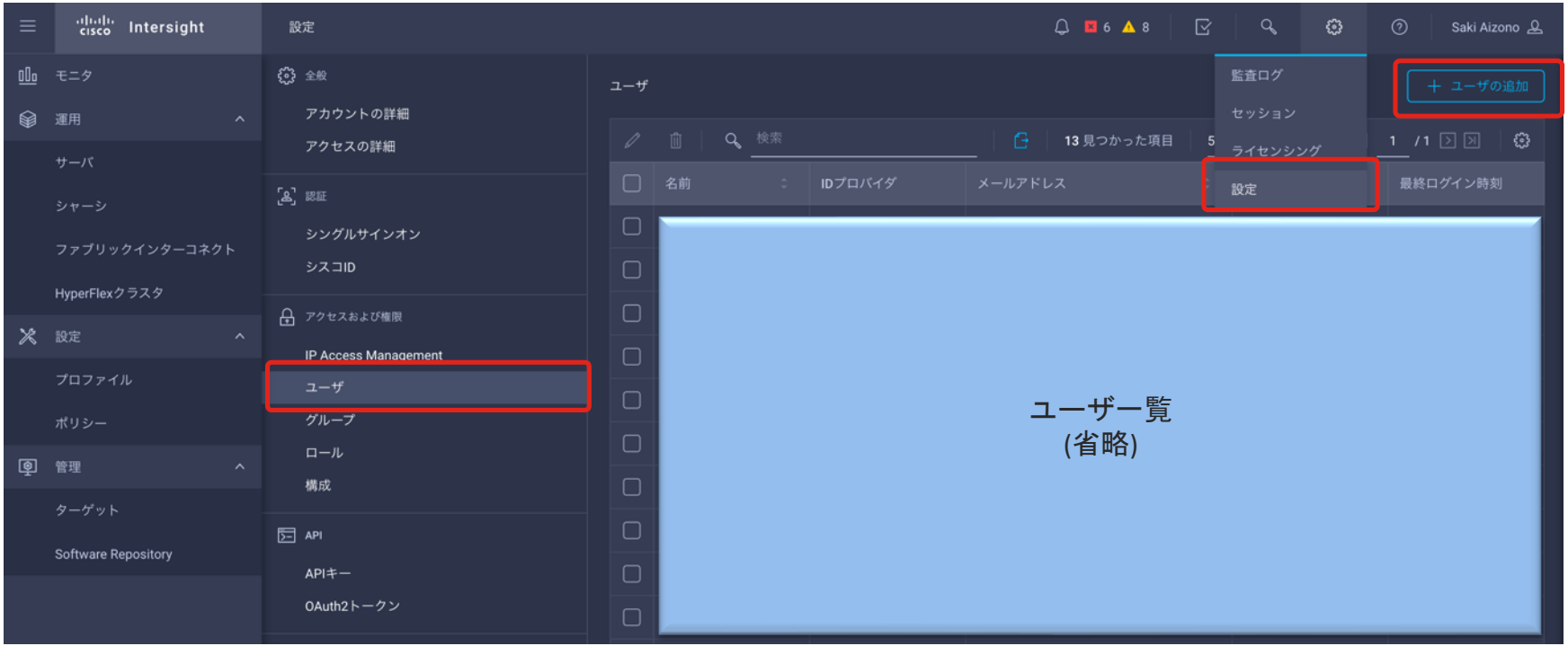
ロールベースアクセス制御 (RBAC) イメージ :



(参考) システム定義ロール一覧

ロール	説明
Account Administrator	Intersight アカウント内の全てのサービスとリソースへの完全なアクセスが可能。
Device Administrator	デバイスクレームとアンクレーム、デバイス詳細やライセンス状態、クレームされたデバイスリストの閲覧、API キーの生成が可能。その他の管理や管理者タスクは出来ない。
User Access Adinistrator	ユーザとグループの追加管理が可能。アカウント詳細や Audit ログの閲覧、Identity Provider、ロール、セッション、Account Administrator 以外のユーザ向け API キーの管理が可能。デバイスクレームや運用管理タスクは実行できない。また、Account Administrator ロールのユーザの追加、管理は出来ない。
Server Administrator	サーバと Fabric Interconnect の管理および閲覧、サーバと FI に関するダッシュボードウィジェットの閲覧、サーバアクションの実行、サーバ詳細の閲覧、管理コンソールおよび CLI の起動、サーバポリシーとプロファイルの作成と展開、API キーの管理が可能。デバイスクレームは出来ない。
HyperFlex Cluster Administrator	HyperFlex ポリシー、プロファイルの作成、クラスタの展開や管理、HX クラスタダッシュボードウィジェットの閲覧、HX Connect の起動が可能。デバイスクレームは出来ない。
Device Technician	デバイスクレーム、デバイス詳細やライセンス状態、クレームされたデバイスリストの閲覧、API キーの生成が可能。その他の管理や管理者タスクは出来ない。
Read-Only	ダッシュボード、管理対象デバイスのテーブル閲覧、ユーザ設定の変更、API キーの生成が可能。デバイスクレーム、ユーザの追加・削除、Identity Provider の設定やサーバアクションは出来ない。

ユーザの追加手順（１）
設定 > ユーザ 画面から、”ユーザの追加” を選択



ユーザの追加手順（２）
ID プロバイダ、メールアドレス、付与したいロールを選んで保存

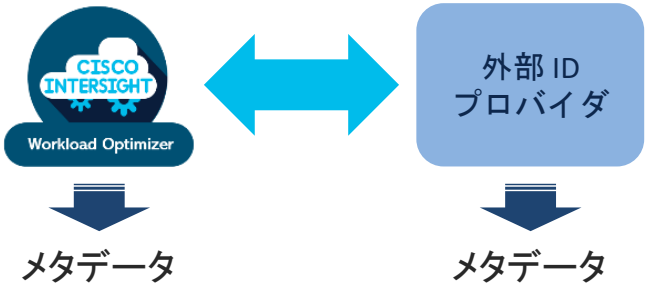
ID プロバイダをドロップダウンより選択
- デフォルトでは Cisco が設定されているので、シスコ ID を持ったユーザのみが登録可能
- 外部の ID プロバイダの追加も可能
(次ページ参照)

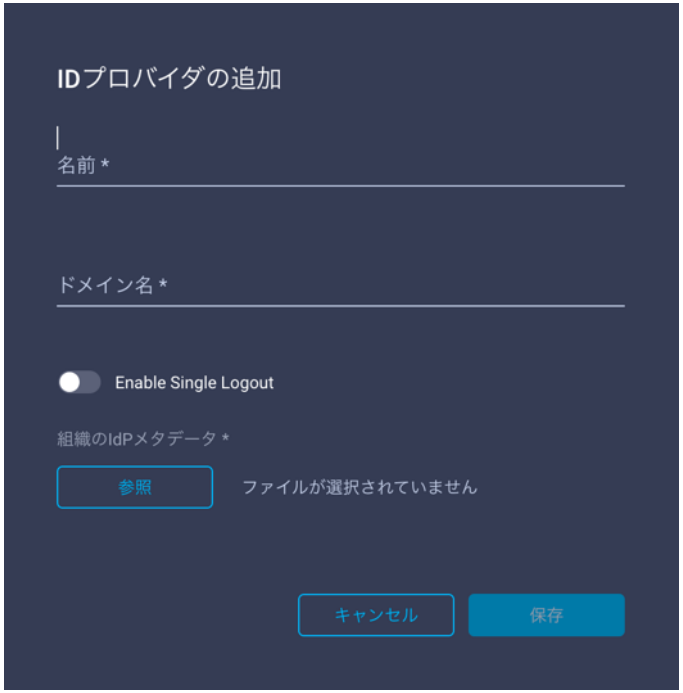
追加したいユーザのメールアドレスを入力

割り当てたいロールを設定
- 複数のロールを割り当てることが可能

外部の ID プロバイダの追加 (SSO)

- SAML 2.0 をサポートする ID プロバイダを Intersight に登録可能
- Video demo of SSO setup および intersight.com/help をご確認ください。

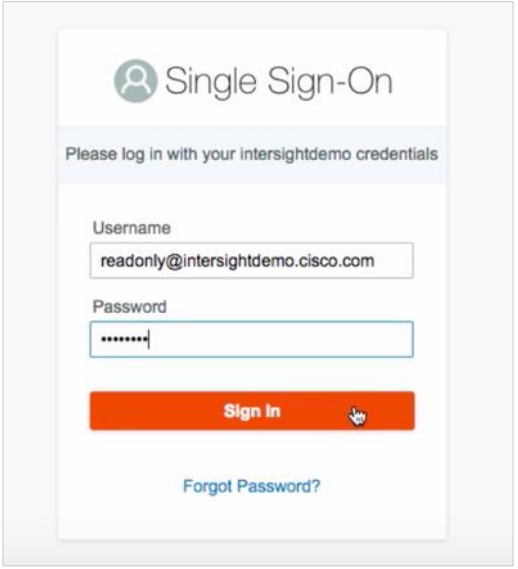




外部の ID プロバイダを利用した Intersight へのログイン

ログイン画面にて、シングルサインオン (SSO) を選択

外部 ID プロバイダを利用して認証が行われる。

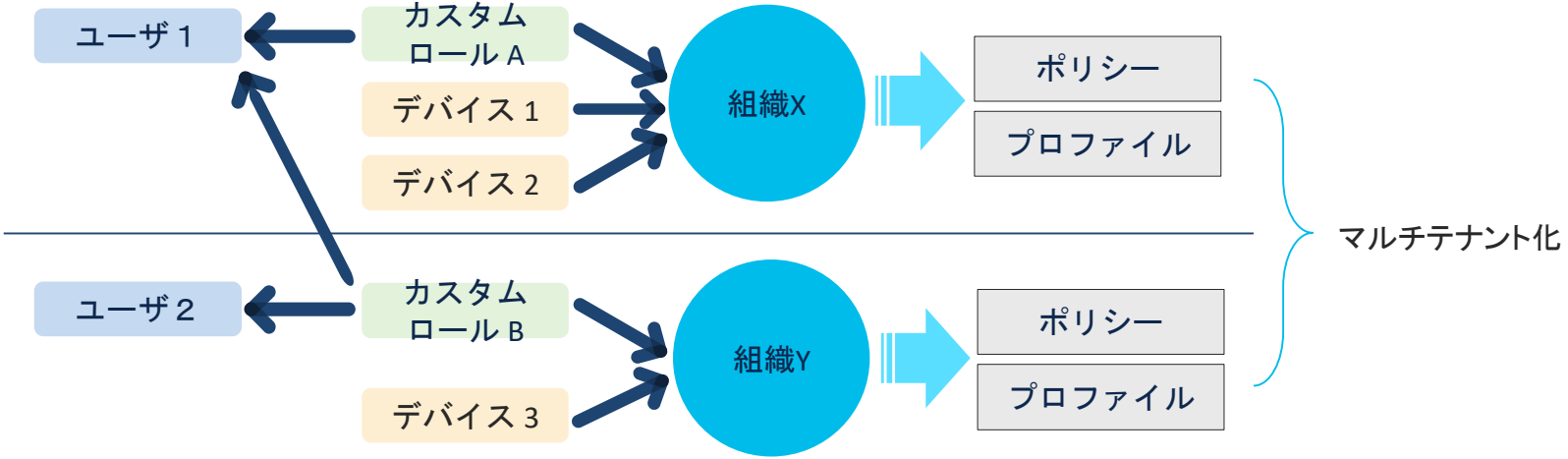


(AWS SSO サービスの例)

組織によるアクセス制御

- デバイスを論理的なグループ内に配置し、マルチテナント化
- デバイスへのアクセスだけでなく、ポリシーやプロファイルも制限
- ”default” の組織には、全てのリソースが紐付いている。

組織とユーザ、ロール、デバイスの関係イメージ



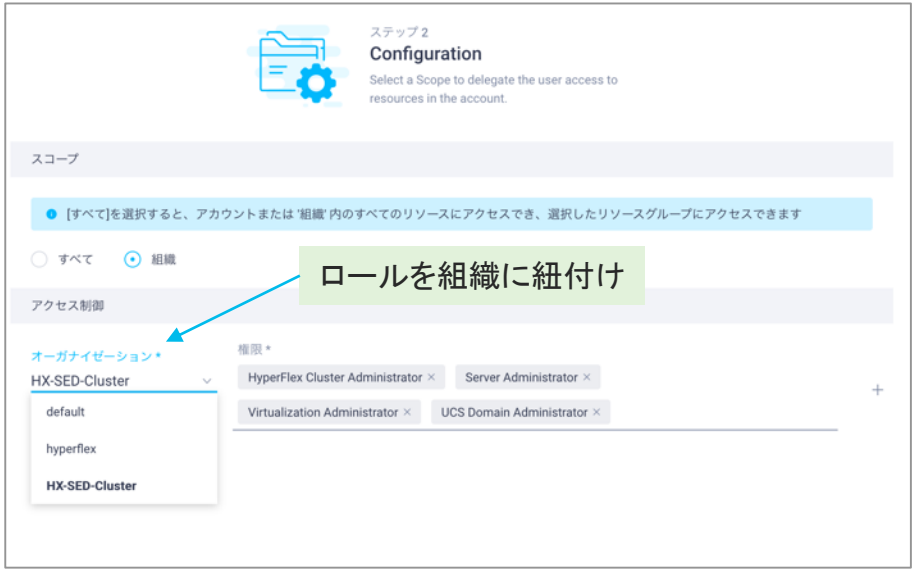
組織設定方法

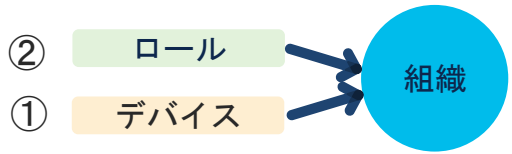
① 組織を作成し、デバイスを紐付ける

設定 > アクセスおよび権限 > 構成 > Create Organization

② カスタムロールを作成し、組織に紐付ける

設定 > アクセスおよび権限 > ロール > ロールを作成する





Intersight による サーバ運用管理

- Firmware アップグレード
- ポリシーベース設定

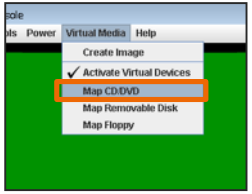
Intersight による サーバ運用管理

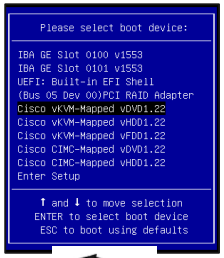
- Firmware アップグレード
- ポリシーベース設定

Intersight によるファームウェアアップグレード

- 従来手動で行っていた以下の作業を自動化し、作業手順を簡素化
  - Cisco Software Central からのファームウェアイメージをダウンロード、およびデバイスへのアップロード
  - ファームウェアイメージのサーバへのマウント

Cisco IMC でのファームウェアアップグレードイメージ





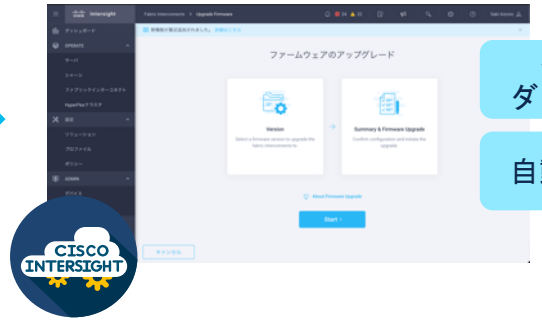

イメージダウンロード


Cisco IMC - KVM
CD/DVD Mapping


Boot 制御

分離された複雑なステップ、手動での実施

Intersight による自動化



イメージ
ダウンロード

自動マウント

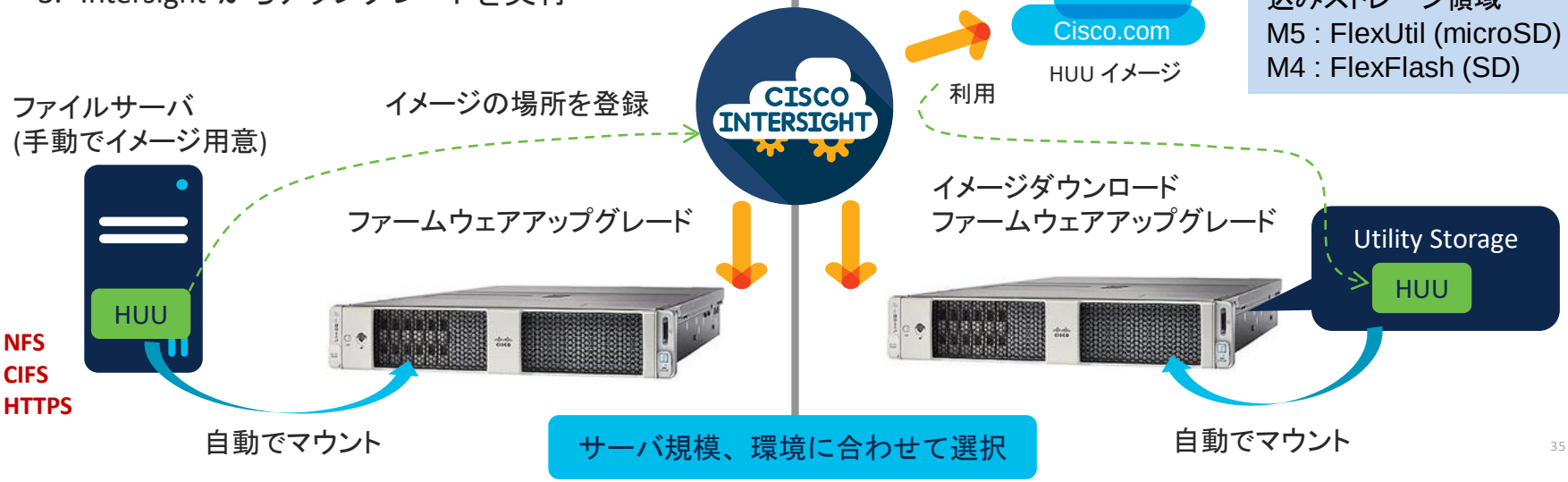
定義されたワークフローにより簡単に実施

Standalone UCS のファームウェアアップグレード

- Utility Storage がない場合
  1. データセンター内にファイルサーバを用意し、HUU イメージを置いておく
  2. Intersight にイメージへのアクセス情報を登録
  3. Intersight からアップグレードを実行

- Utility Storage がある場合
  1. Intersight からアップグレードを実行

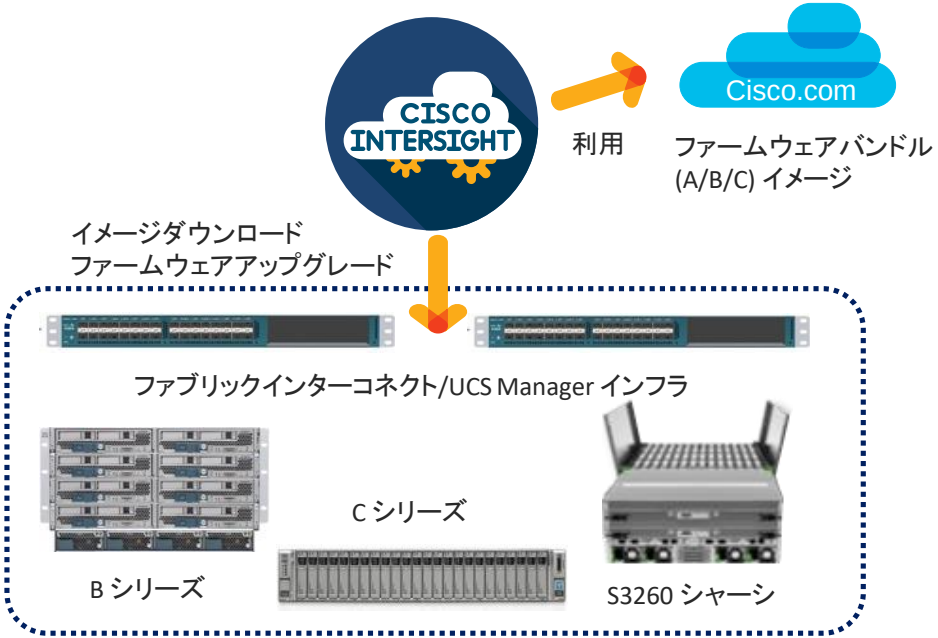
Utility Storage
メンテナンスのための組み込みストレージ領域
M5 : FlexUtil (microSD)
M4 : FlexFlash (SD)



サーバ規模、環境に合わせて選択

FI Integrated UCS のファームウェアアップグレード

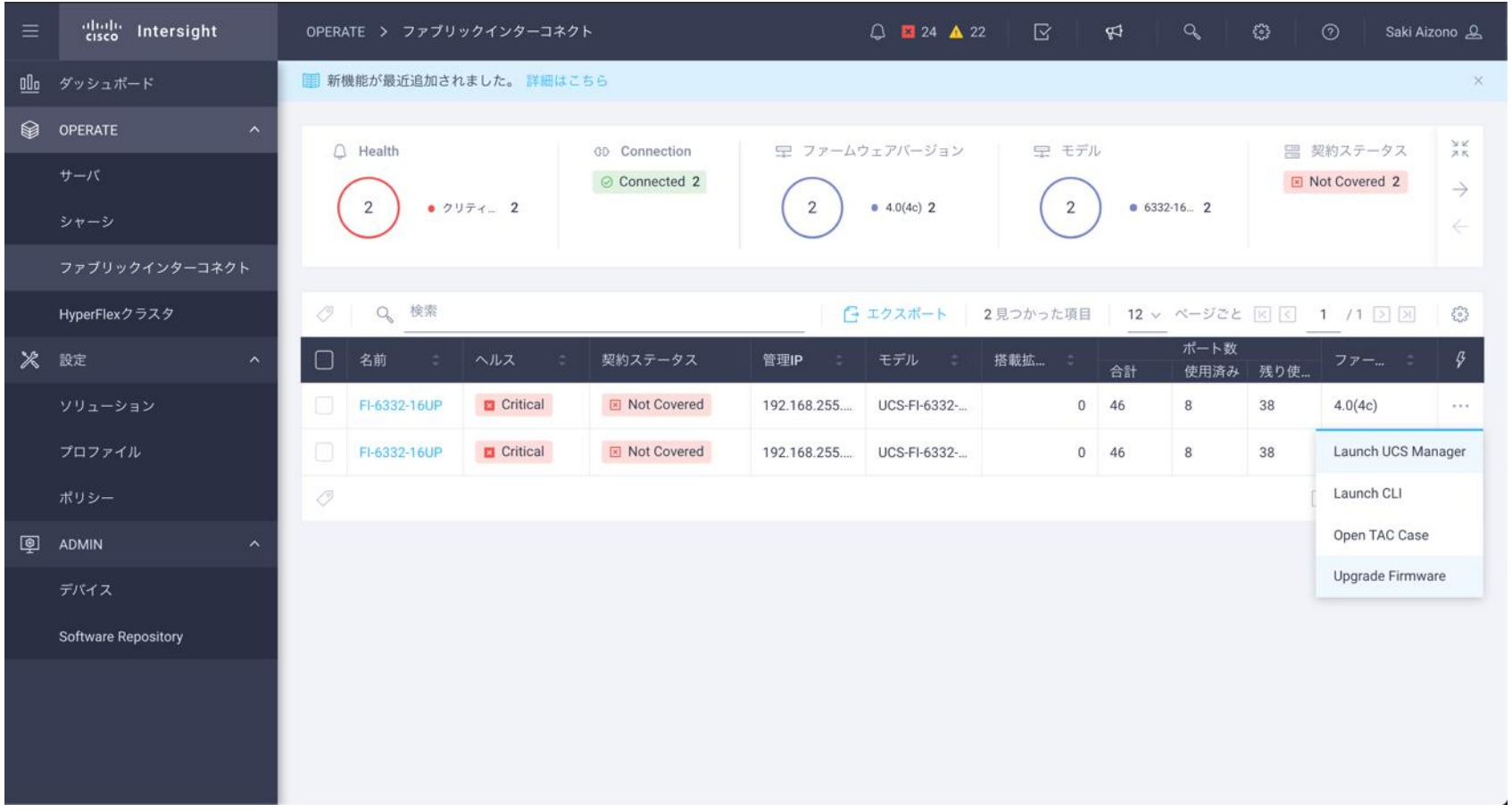
- FI にイメージをダウンロードし、自動でアップグレード



- FI およびサーバのファームウェアをアップグレード
- デフォルトで Fabric Evacuation を選択
- Associated 状態のサーバのみサポート
- アップデートテンプレート利用サーバは非サポート

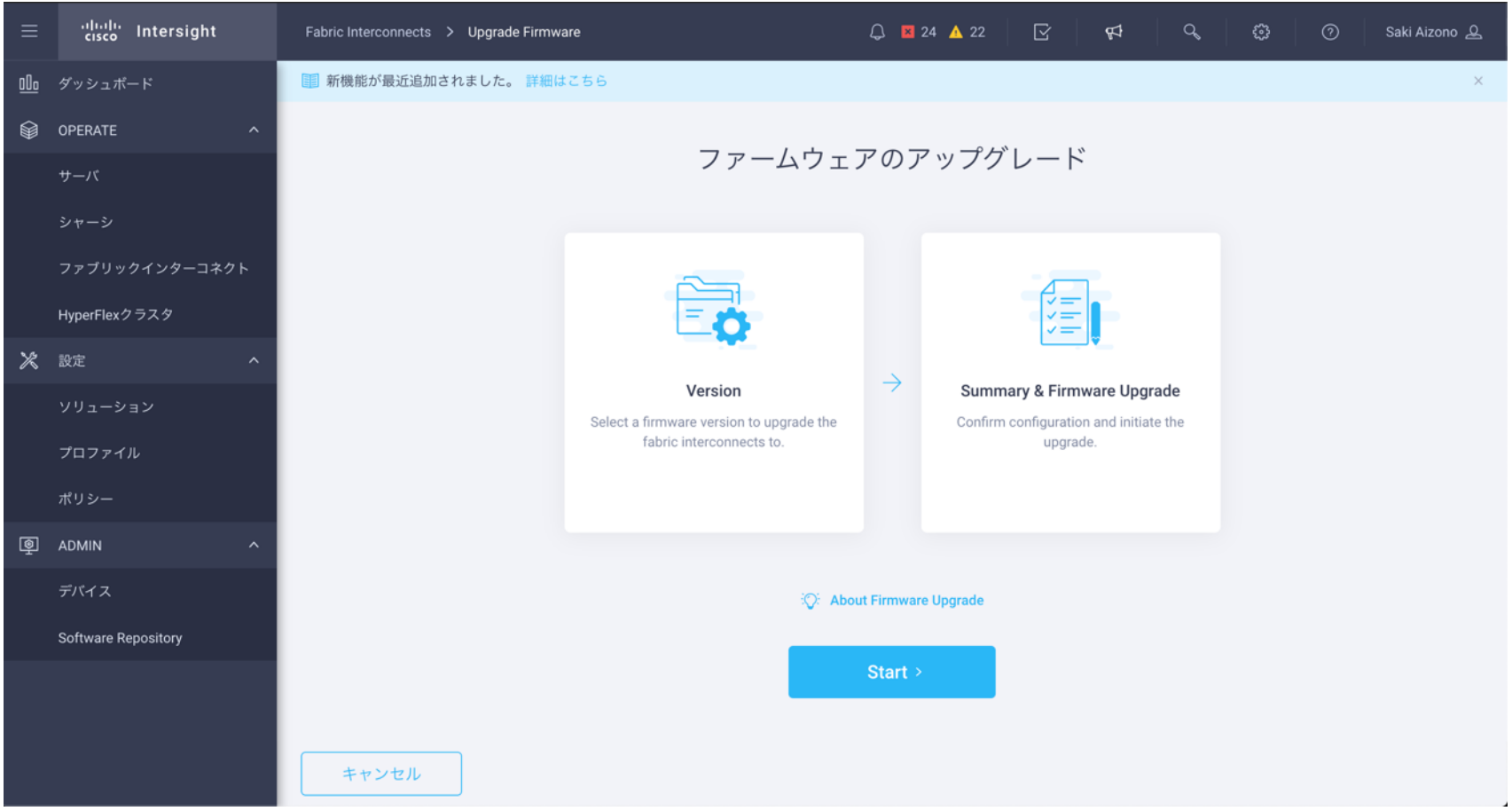
FI / UCSM インフラのファームウェアアップグレードフロー（１）

ファブリックインターコネクトのメニューから Upgrade Firmware を選択



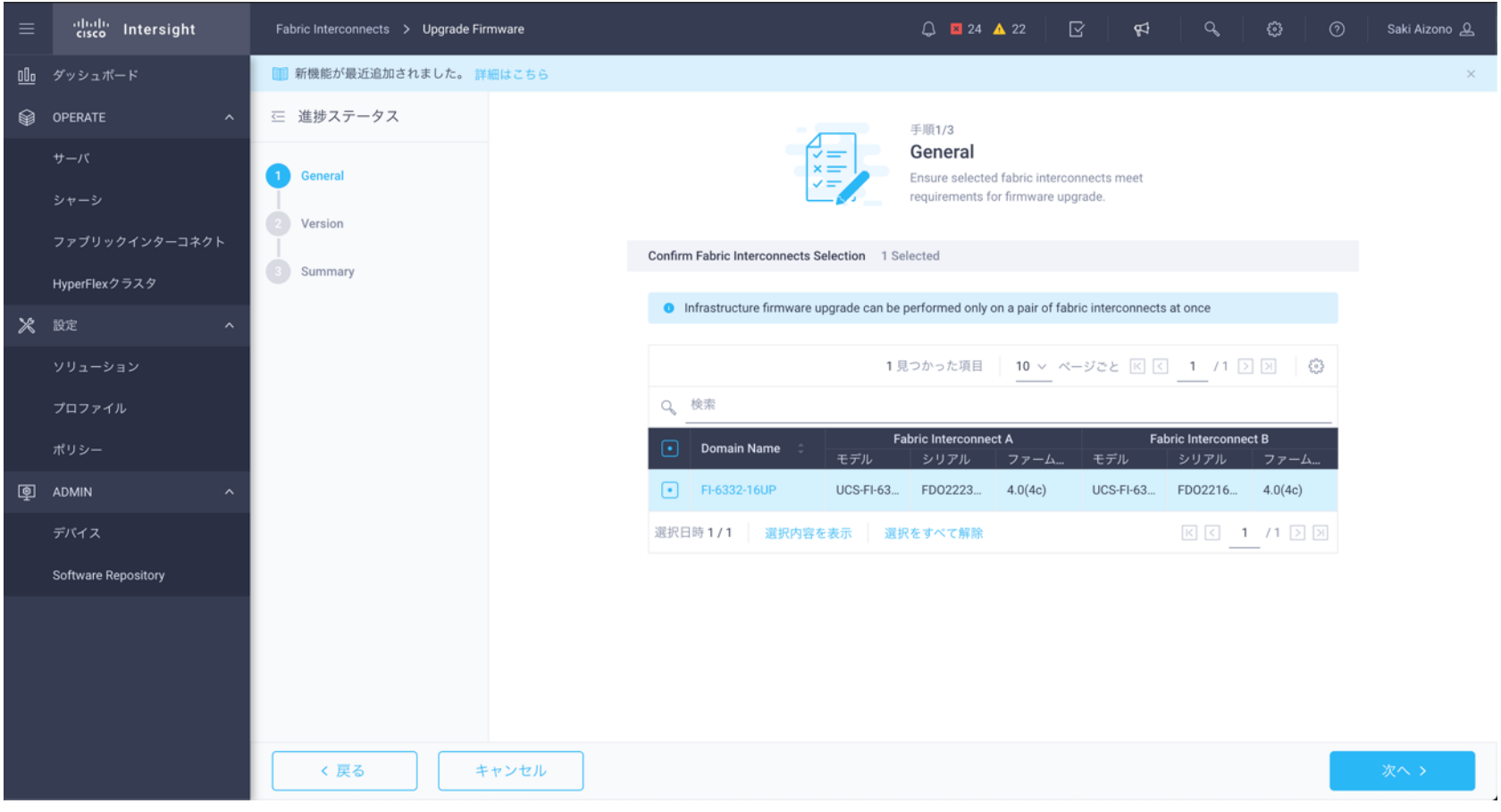
FI / UCSM インフラのファームウェアアップグレードフロー（２）

ファームウェアアップグレードのフローを確認して開始



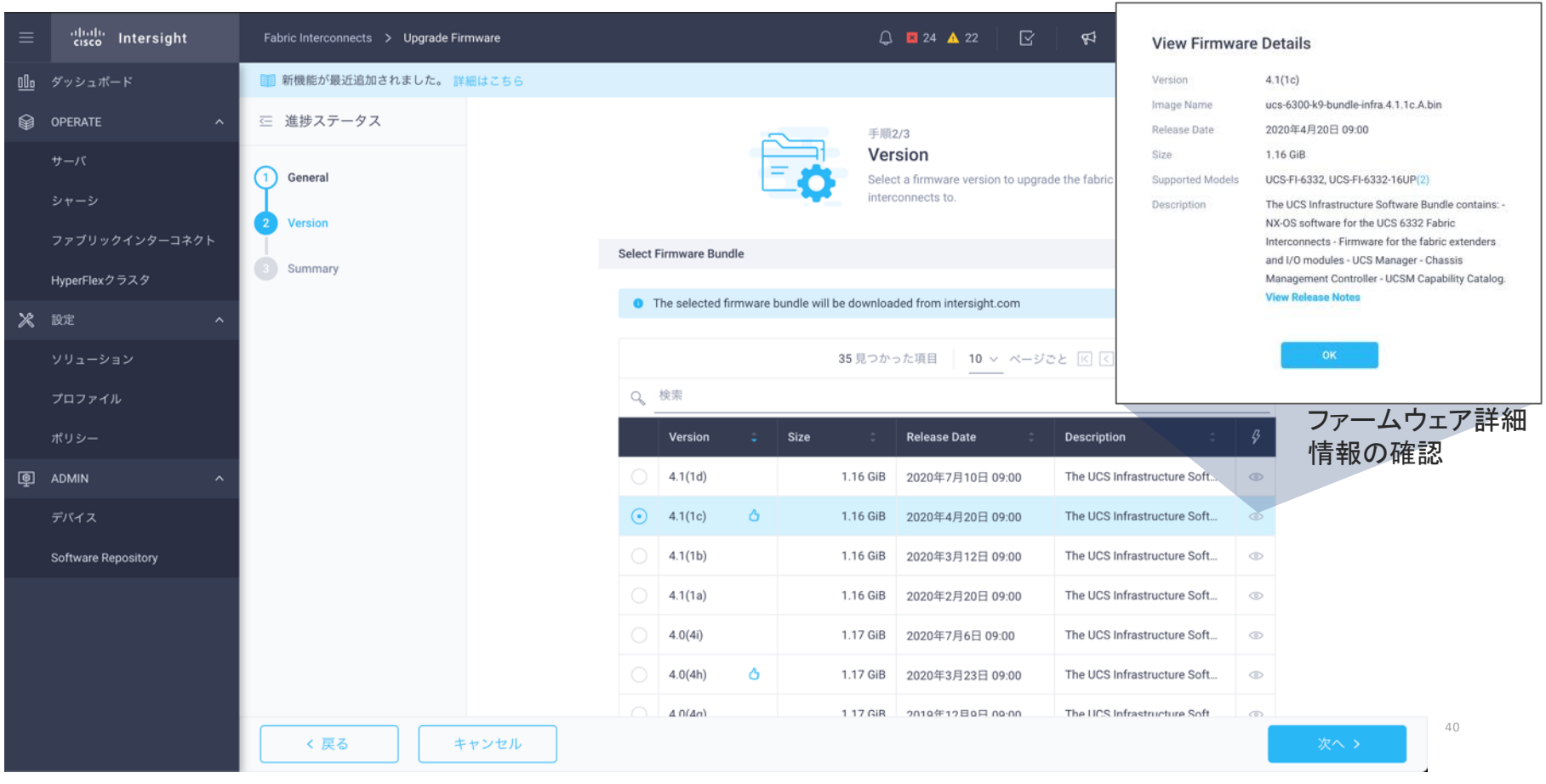
FI / UCSM インフラのファームウェアアップグレードフロー（３）

アップグレードしたいファブリックインターコネクトを選択



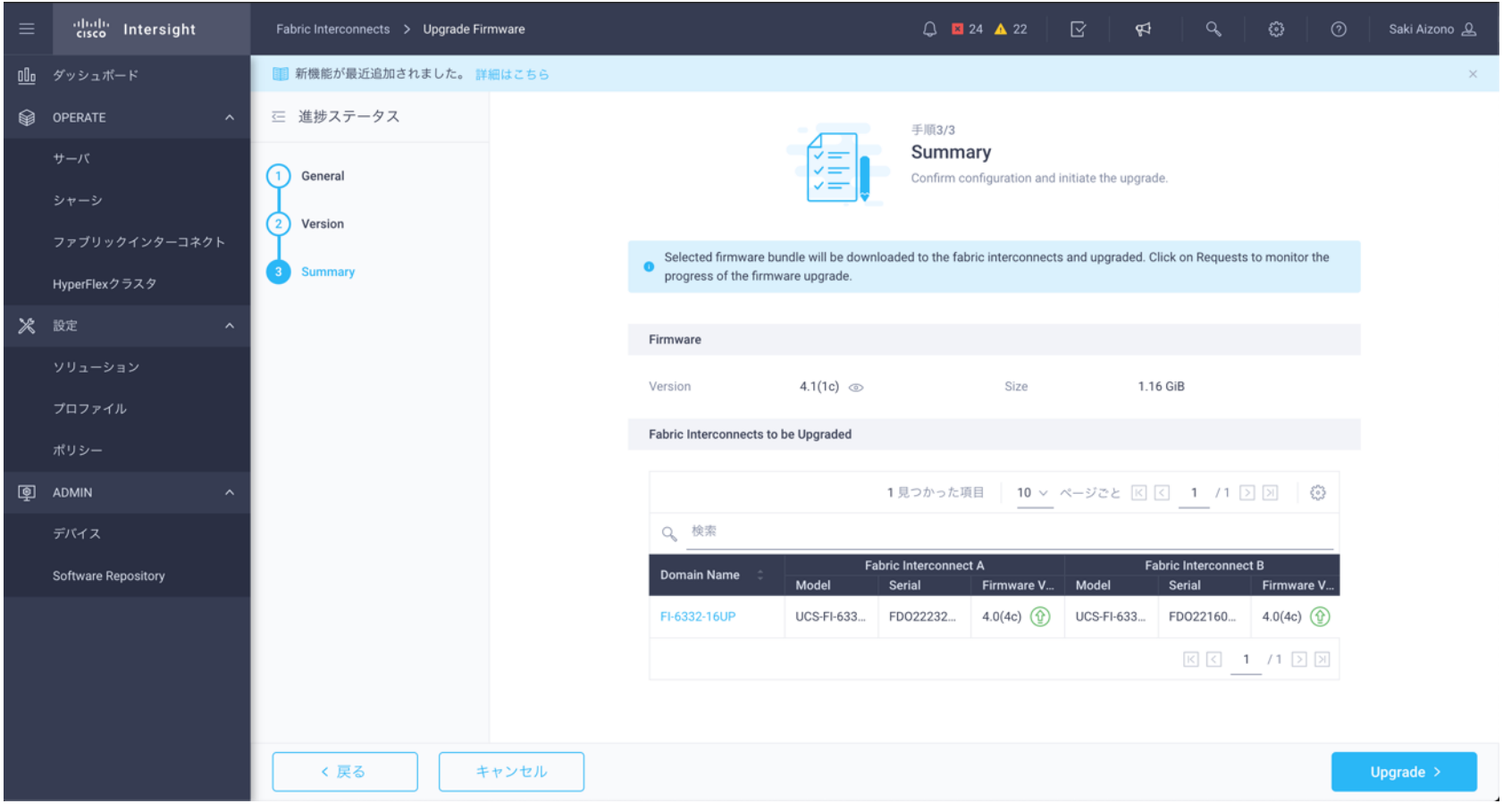
FI / UCSM インフラのファームウェアアップグレードフロー（４）

アップグレード先のファームウェアバージョンを選択



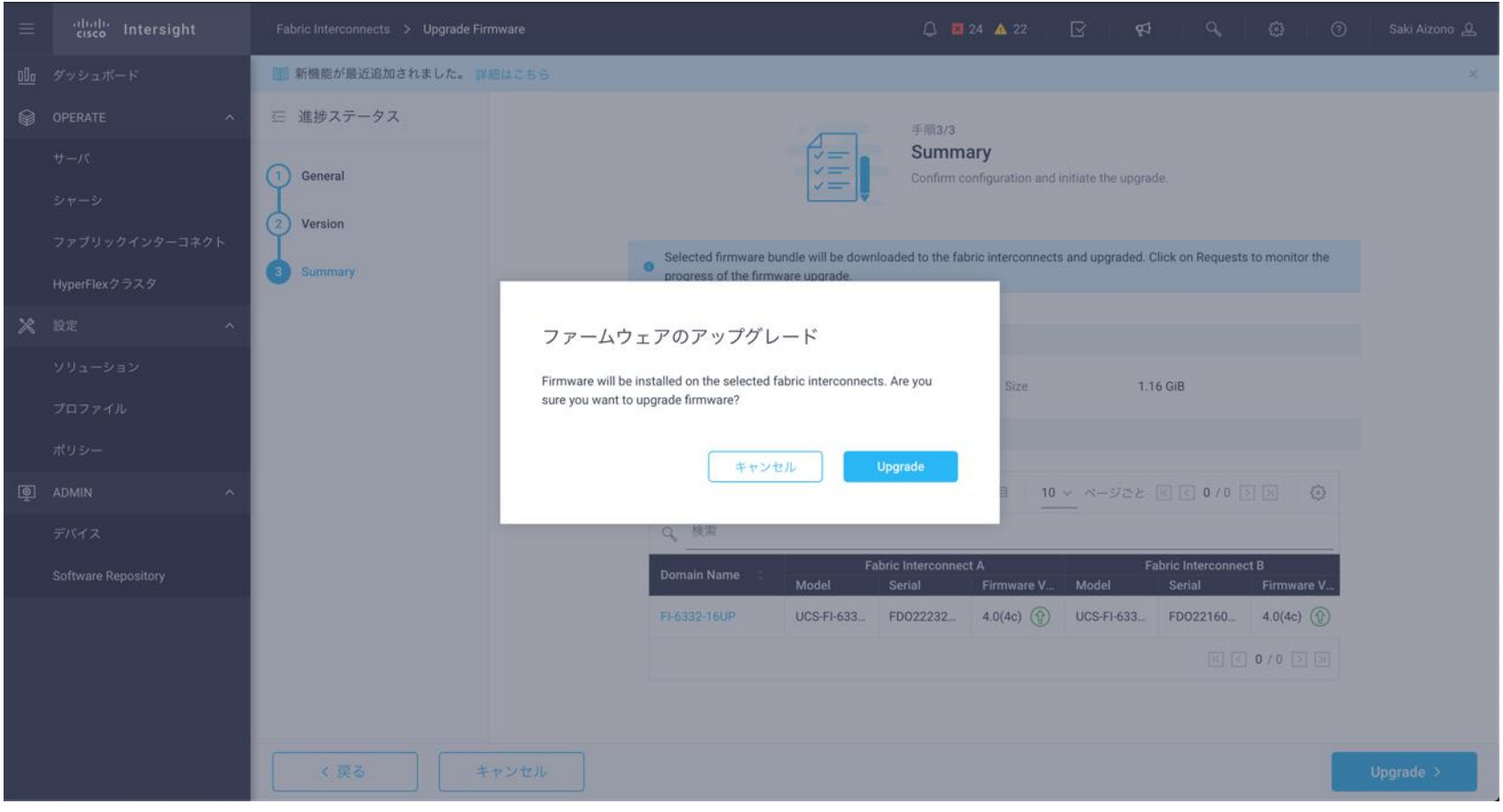
FI / UCSM インフラのファームウェアアップグレードフロー（５）

アップグレードのサマリを確認し、問題なければアップグレードボタンをクリック



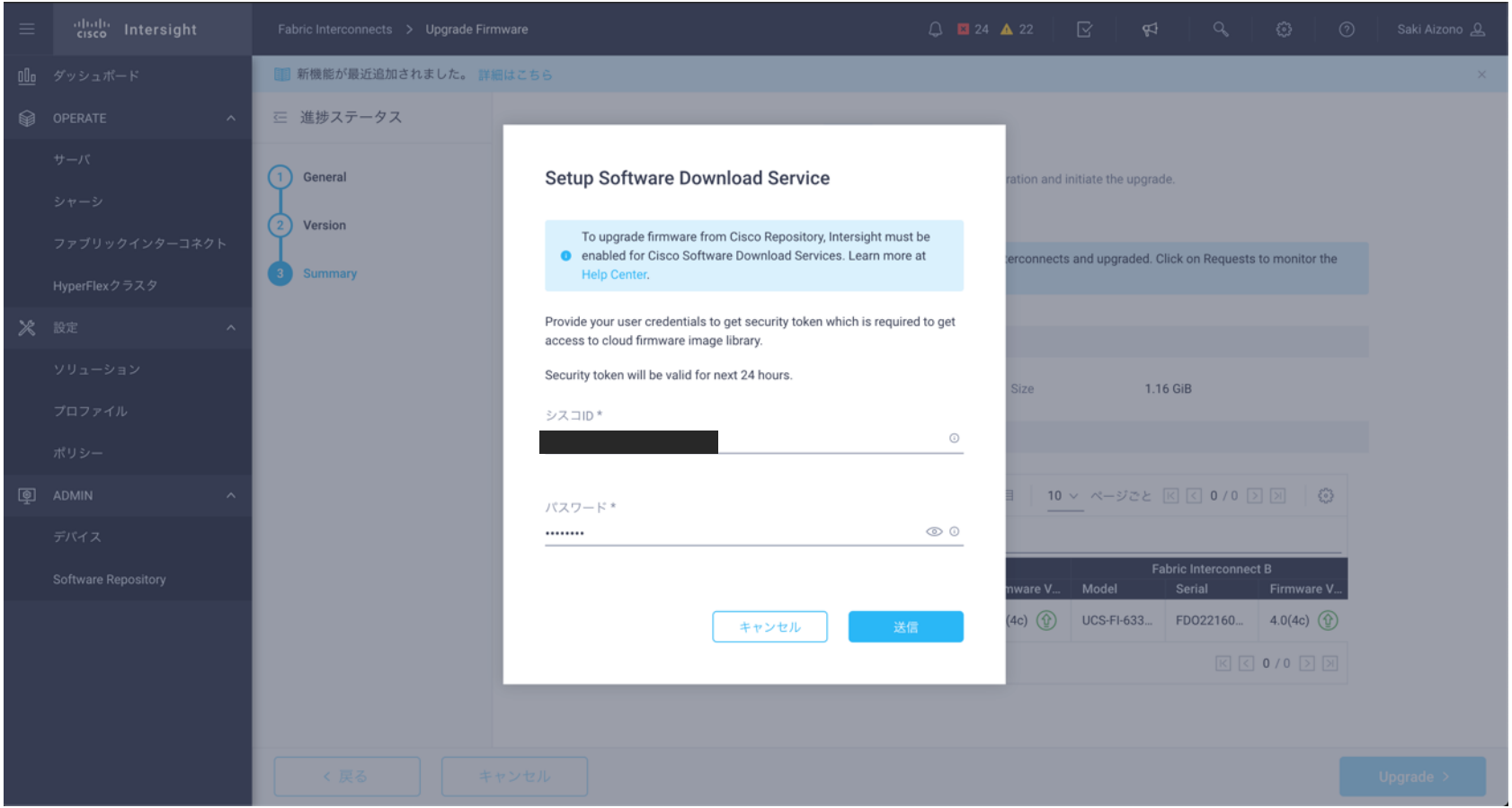
FI / UCSM インフラのファームウェアアップグレードフロー（６）

本当にアップグレードするかどうかの確認



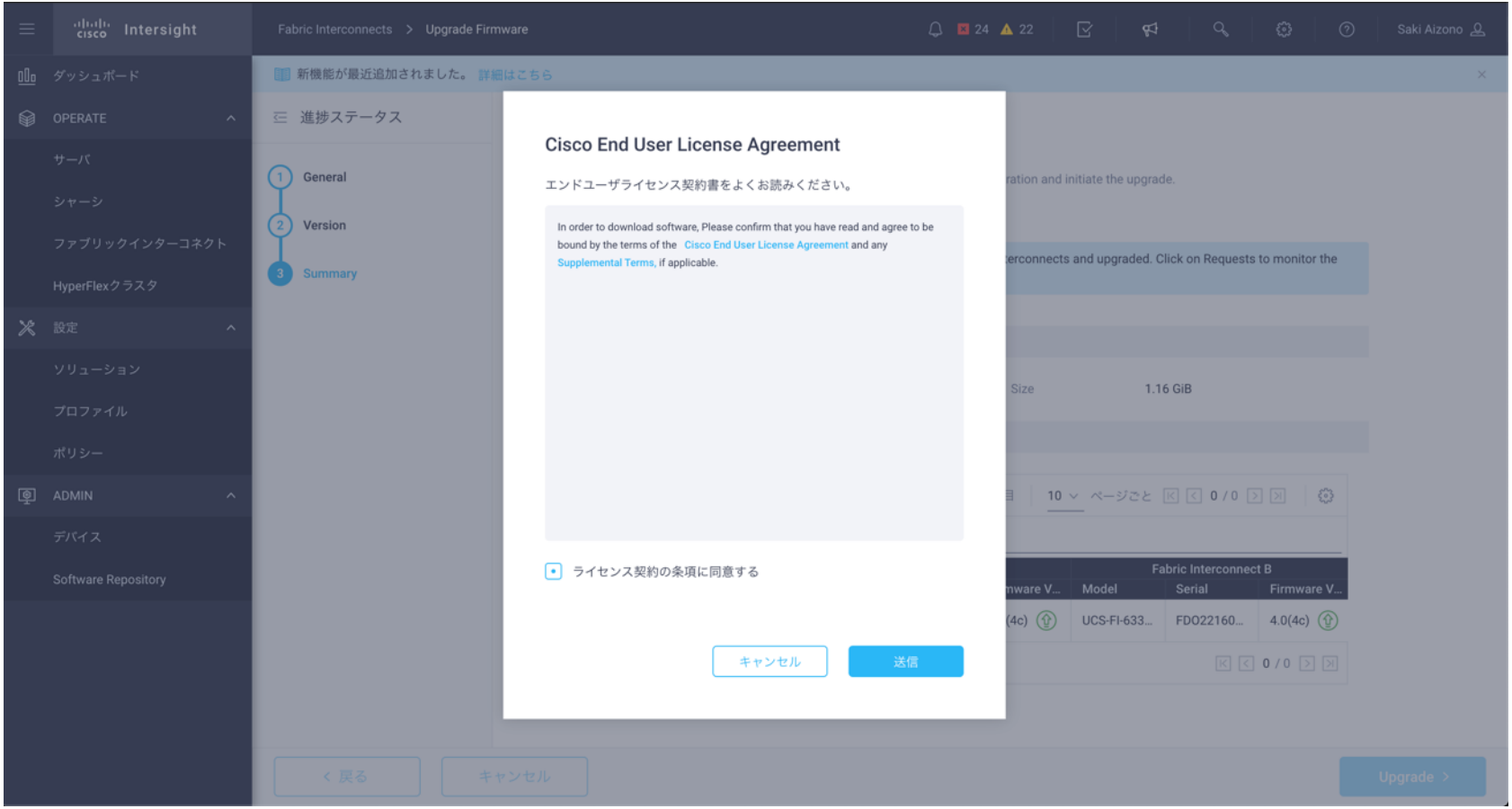
FI / UCSM インフラのファームウェアアップグレードフロー（７）

Intersight がソフトウェアを Cisco 側から引き出すために、Cisco ID とパスワードを提供



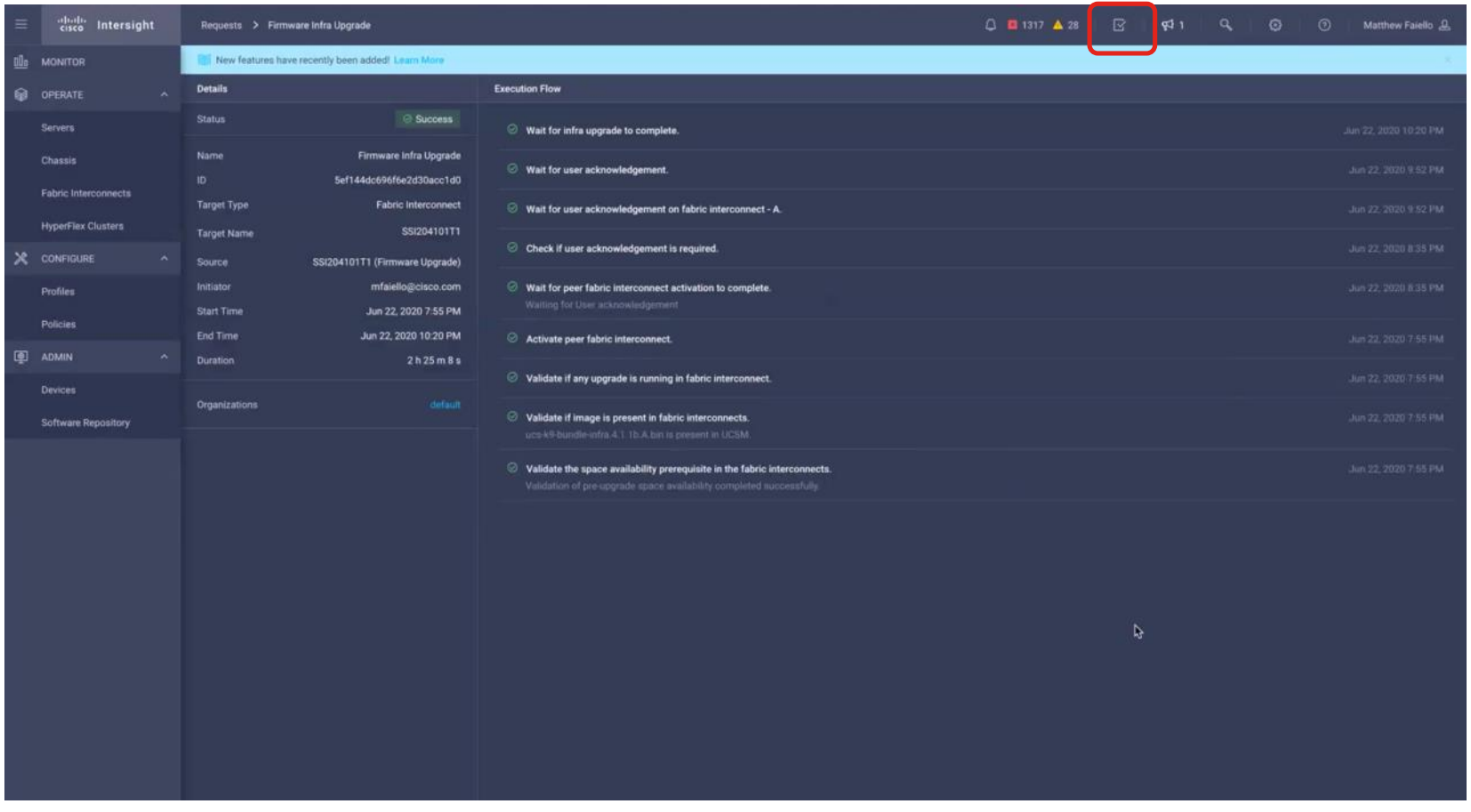
FI / UCSM インフラのファームウェアアップグレードフロー（８）

ライセンス契約の条項に同意すると、インストールが始まる



FI / UCSM インフラファームウェアアップグレード成功時の実行フロー

Intersight 上で進行中の実行フローを確認



Intersight による サーバ運用管理

- Firmware アップグレード
- ポリシーベース設定

UCS Manager - サービスプロファイルによる設定抽象化

- UUID・WWN・MAC・BIOS 設定などハードウェア固有情報を抽象化
- サーバに依存した設定をなくすことでサーバ移行を簡単にできる

サーバ管理者のメリット

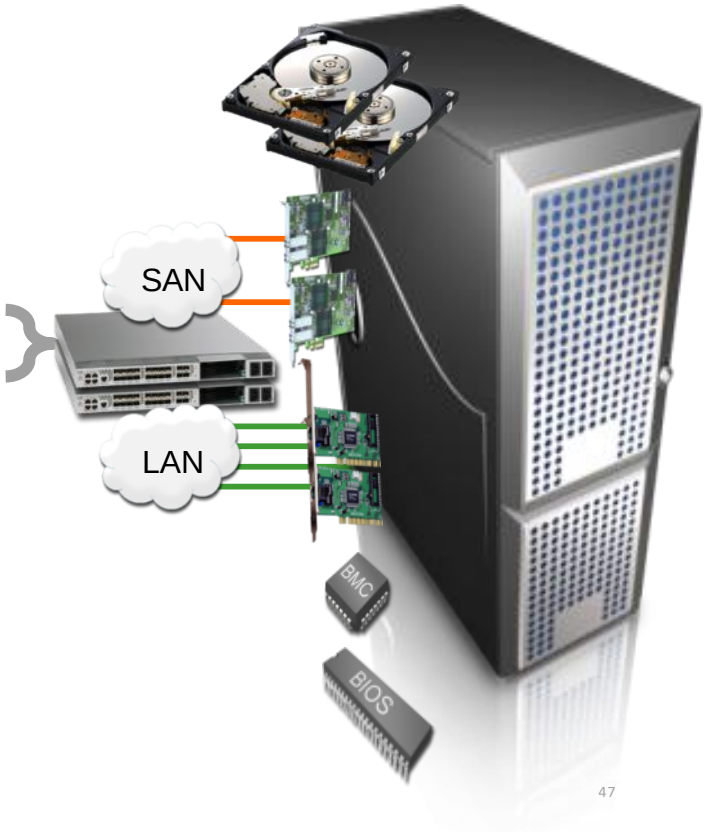
サーバ機種変更に伴うスイッチ・ストレージ変更依頼を一切行うことなく、既存環境を新規サーバに容易に移行可能

ストレージ管理者のメリット

ストレージの初期設定変更を行うことなくサーバ移行することが可能

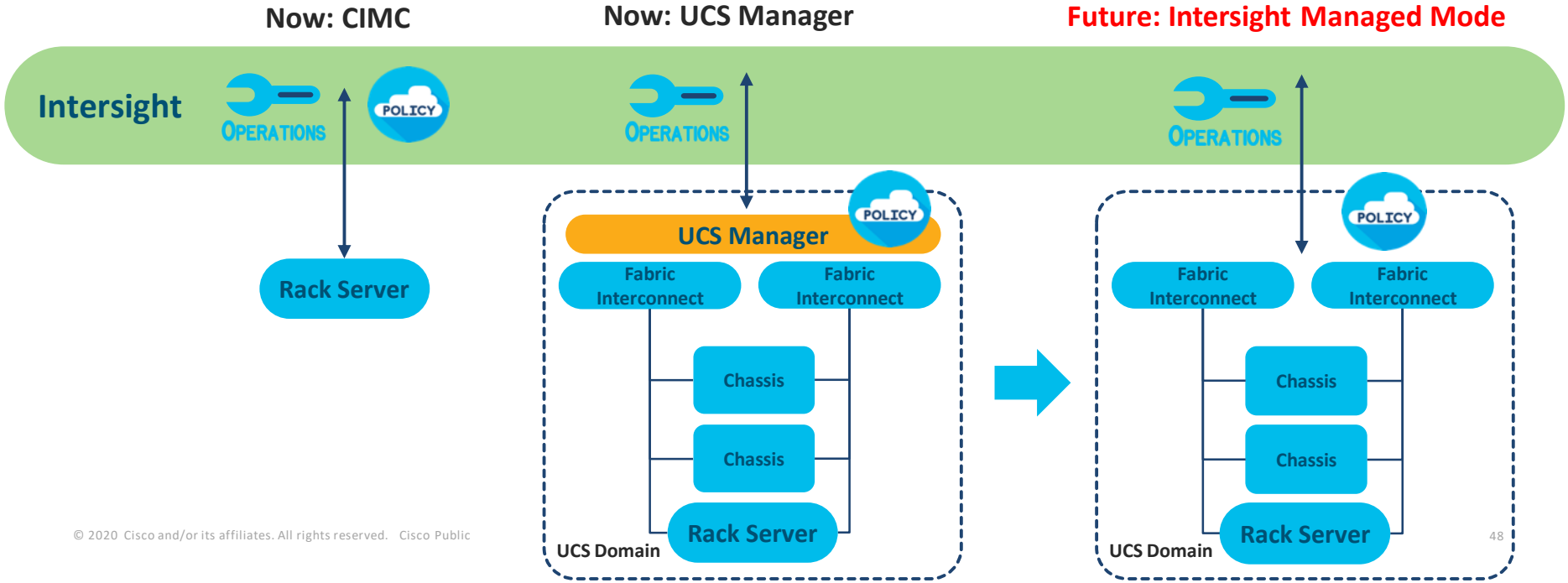
Cisco UCS サービスプロファイル

NIC Existence
NIC MACs
HBA Existence
HBA WWPNs
Server UUID
BIOS Settings
RAID Settings
SED/KMIP Settings
VLAN Assignments
VLAN Tagging
FC Fabrics Assign.
FC/iSCSI Boot Settings
Boot order
PXE settings
IPMI Settings
QoS
Template Association
Org & Sub Org Assoc.
Server Pool Assoc.
Statistic Thresholds
BIOS scrub actions
Disk scrub actions
BIOS firmware
Adapter firmware
BMC firmware
Advanced NIC settings
Serial over LAN settings



Intersight による UCS サーバのポリシーベース設定

- Standalone UCS は Intersight に接続することで、ポリシーベース設定が可能
- FI Integrated UCS は、現在は UCS Manager でポリシーの管理が可能だが、将来的に Intersight から管理が可能になる

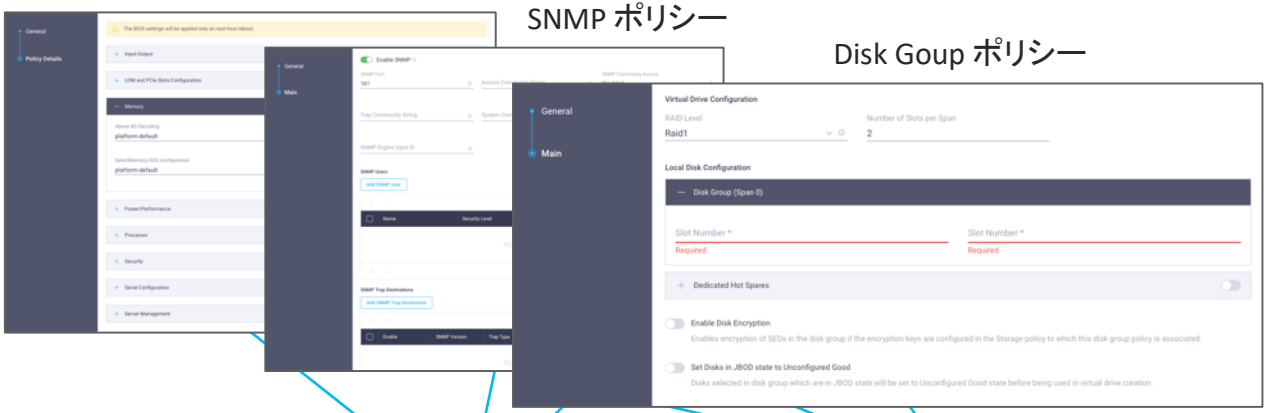


スタンドアロン UCS のポリシーベース設定

BIOS ポリシー

SNMP ポリシー

Disk Goup ポリシー

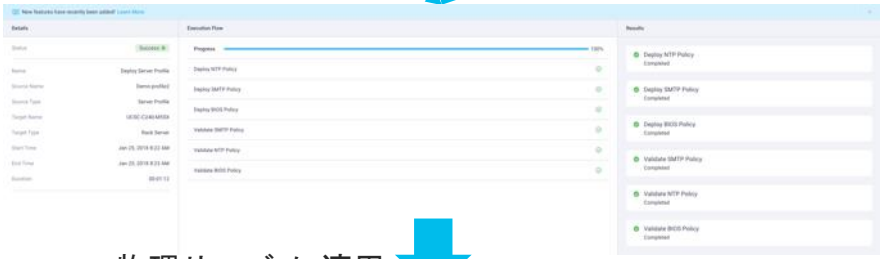


サーバポリシー

- ✓ NTP 設定
- ✓ Disk Group 設定
- ✓ DNS 設定
- ✓ DHCP 設定
- ✓ vKVM 設定
- ✓ SMTP 設定
- ✓ SNMP 設定
- ✓ SSH 設定
- ✓ Local User 設定
- ✓ Boot Order 設定

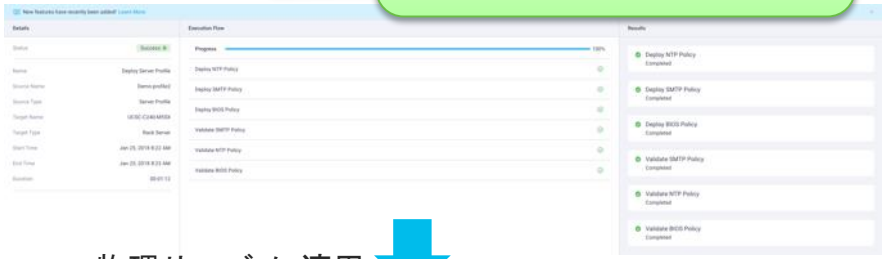
複数のサーバに同じポリシーを適用して、ベースラインのコンフィグが可能。

プロファイル1



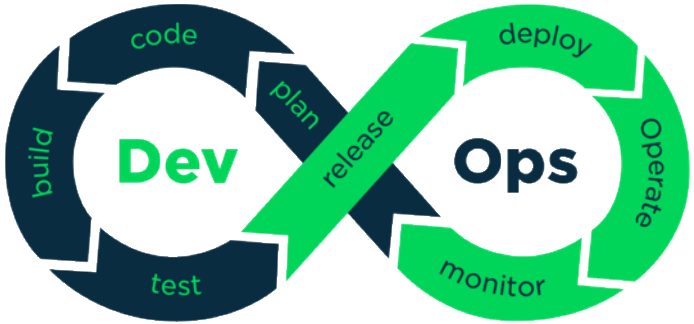
物理サーバ1に適用

プロファイル2



物理サーバ2に適用

Ansible 自動化



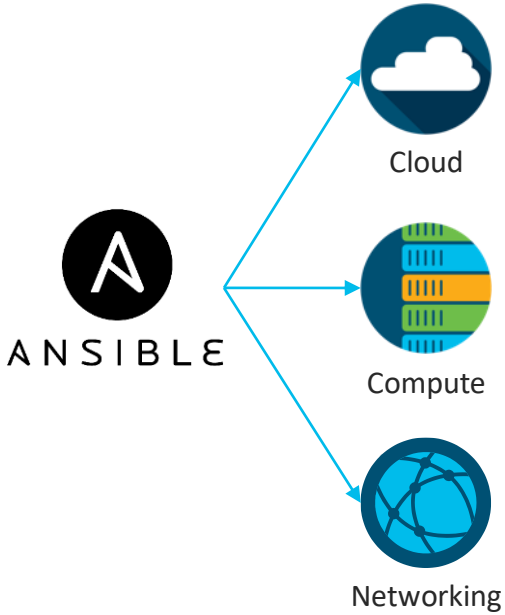
*"**Infrastructure as code** (IaC) is the process of managing and provisioning computer through machine-readable definition files* […]"

Wikipedia - https://en.wikipedia.org/wiki/Infrastructure_as_Code

Infrastructure as code を実現する様々なツール

Ansible 紹介

- シンプルな IT 自動化プラットフォーム
- 多くの種類のタスクを処理可能 :
  - システムの構成 / 構成管理
  - ソフトウェアの展開 / ソフトウェアパッケージの管理
  - 様々なシステムのプロビジョニング
  - オーケストレーション
- 完全なオープンソース (開発元は Red Hat)



Ansible Collection

- ベンダーモジュールは Ansible に含まれなくなる
  - Ansible 2.10 では Ansible Collection のインストールが必要
  - ベンダーモジュールの追加・アップデートは Ansible Core リリースに依存しない
- ベンダーモジュールは Ansible Galaxy より入手可能
  - 必要なモジュールのみのインストール
  - 必要なモジュールのバージョンをインストール
- Intersight Ansible Collection
  - https://galaxy.ansible.com/cisco/intersight

免責事項:
Cisco は利便性と情報提供のみを目的として Ansible モジュールの提供を行っています。サポートについては提供しておりませんので、予めご了承ください。

Ansible Intersight Collection の使い方概要

- Python, Ansible 実行環境を準備
- Ansible Galaxy より Intersight Collection をインストール
- Intersight API キーの生成
- インベントリファイル、Playbook を編集して利用

今回利用した実行環境

- MacOS
- Python 3.7.5

```
$ python --version
Python 3.7.5
```

- Ansible 2.10.3

```
$ ansible --version
ansible 2.10.3
  config file = None
  configured module search path = ['/Users/sanomura/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location =
/Users/sanomura/dev/20201209_intersight_ansible_webinar/test/lib/python3.7/site-packages/ansible
  executable location = /Users/sanomura/dev/20201209_intersight_ansible_webinar/test/bin/ansible
  python version = 3.7.5 (v3.7.5:5c02a39a0b, Oct 14 2019, 18:49:57) [Clang 6.0 (clang-600.0.57)]
```

Intersight Collection のインストール by Ansible Galaxy

- Cisco Intersight Collection in Ansible Galaxy
  https://galaxy.ansible.com/cisco/intersight
- ansible-galaxy コマンドによる Collection のインストール

```
$ ansible-galaxy collection install cisco.intersight
Process install dependency map
Starting collection install process
Installing 'cisco.intersight:1.0.8' to
'/Users/sanomura/.ansible/collections/ansible_collections/cisco/intersight'
```

Intersight Collection のインストールコンテンツ確認

- インストール先のディレクトリのコンテンツを確認

```
$ cd /Users/sanomura/.ansible/collections/ansible_collections/cisco/intersight
$ ls
Development.md   LICENSE.txt      README.md  misc       plugins
FILES.json MANIFEST.json    docs       playbooks  roles
```

module, module_units, documentation が含まれる

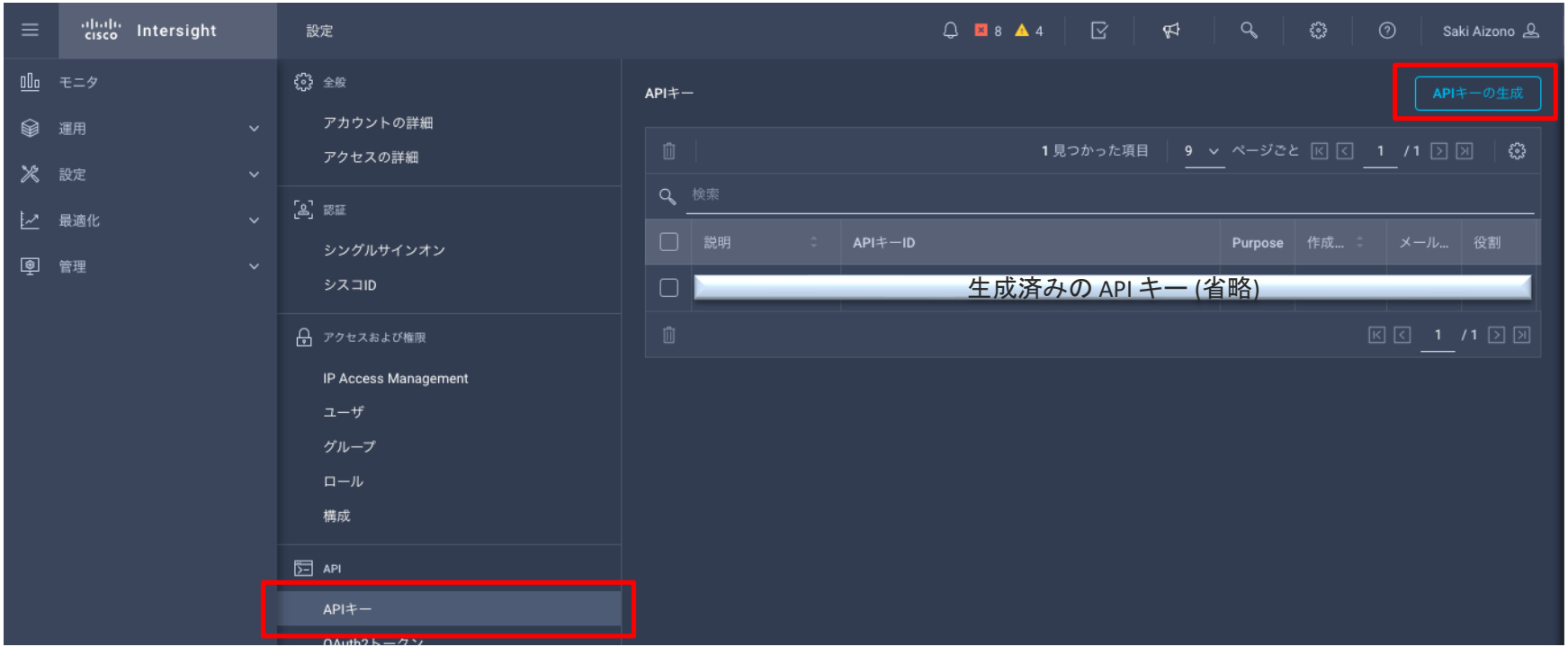
- Playbook ディレクトリの内容

サンプル Playbook が含まれる

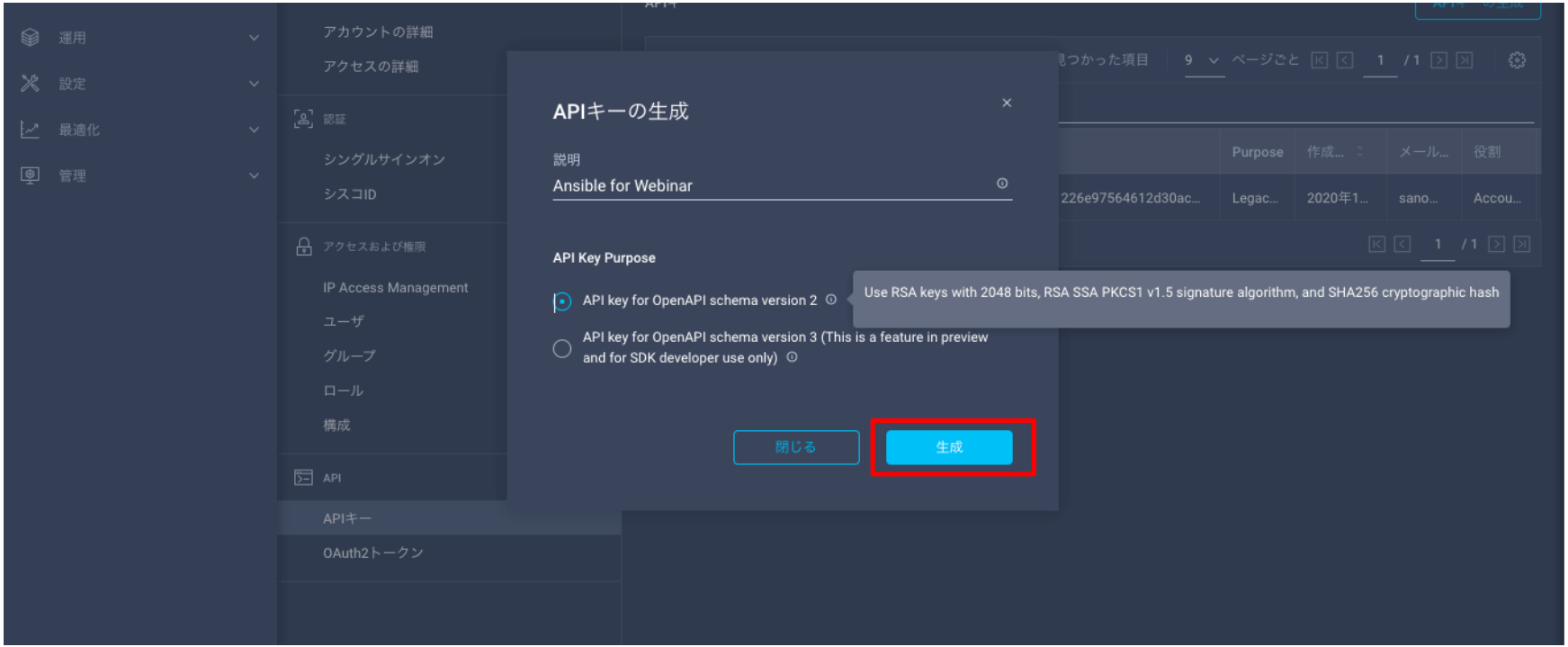
```
$ cd playbooks/
$ ls
claim_device.yml                    intersight_ntp_policy.yml
cos_server_policies_and_profiles.yml  intersight_server_profile.yml
deploy_server_profiles.yml          intersight_virtual_media_policy.yml
devnet_inventory                    ova_workflow.yml
example_inventory                   roles
firmware_direct_download.yml              server_actions.yml
hcl_status.yml                      server_firmware.yml
intersight_boot_order_policy.yml    staging_inventory
intersight_imc_access_policy.yml    update_all_inventory.yml
intersight_local_user_policy.yml    update_standalone_inventory.yml
```

Intersight API キー取得方法（１）

Intersight API キー取得方法（２）



Intersight API キー取得方法（３）



Intersight API キー取得方法（４）

Intersight API キー取得方法（５）

インベントリファイル内 API 変数のカスタマイズ

- Intersight Collection Playbook ディレクトリの ’example_inventory’ ファイルに API キー情報が含まれており、今回はこれを利用する。
- example_inventory をコピーして新しいインベントリファイルを作成

```
$ cp example_inventory inventory
```

- インベントリファイル内の [Intersight:vars] セクションが、API キー認証を含んでおり、これを Playbook から参照する形をとる。

インベントリファイル内 API 変数のカスタマイズ

- 生成した API キーを利用するようインベントリファイルを編集

example_inventory ファイル ：

```
[Intersight_HX]
# Note: at least one host (e.g., sjc07-r13-501) must be present for
update_*_inventory.yml to work
sjc07-r13-501
sjc07-r13-503

[Intersight_Servers]

[Intersight:children]
Intersight_HX
Intersight_Servers

[Intersight:vars]
api_private_key=~/Downloads/SecretKey.txt
api_key_id=596cc79e5d91b400010d15ad/5f0ce0ad7564612d3311a1f3/5f0ea8eb7564612d334ccb5a
```

前段はマニュアルでの修正は不要

保存した秘密鍵のパスを指定

API キー ID を入力

インベントリ情報をアップデートする

- Playbook “update_standalone_inventory.yml” を利用して Intersight から情報を取得し、スタンドアロンサーバのインベントリファイルをアップデートする。

```
$ ansible-playbook -i inventory update_standalone_inventory.yml
[WARNING]: running playbook inside collection cisco.intersight

PLAY [Intersight] ************************************************************

TASK [cisco.intersight.intersight_rest_api] ****************(snip)
```

実行結果例 :

通常は将来的な Collection のアップデートによる上書き等を考慮して Collection ディレクトリ外に Playbook を配置するが、ラボ環境ではOK

```
$ ansible-playbook -i inventory update_standalone_inventory.yml
[WARNING]: running playbook inside collection cisco.intersight

PLAY [Intersight] ************************************************************

TASK [cisco.intersight.intersight_rest_api] ***********************************
ok: [sjc07-r13-501]

TASK [debug] *****************************************************************
ok: [sjc07-r13-501] => {
    "msg": "Inventory filepath
¥"/Users/sanomura/.ansible/collections/ansible_collections/cisco/intersight/playbooks/inventory¥""
}

TASK [lineinfile] ************************************************************
ok: [sjc07-r13-501]

TASK [lineinfile] ************************************************************
changed: [sjc07-r13-501] => (item=UCS-PBST-C-C220M4-3)
changed: [sjc07-r13-501] => (item=UCS-PBST-C-C220M4-4)
changed: [sjc07-r13-501] => (item=UCS-PBST-C-C220M4-2)
changed: [sjc07-r13-501] => (item=UCS-PBST-C-C220M4-1)

PLAY RECAP *******************************************************************
sjc07-r13-501              : ok=4    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

インベントリファイルが自動的にアップデートされる。

実行前：

```
[Intersight_Servers]
```

（空白）

実行後：

```
[Intersight_Servers]
UCS-PBST-C-C220M4-1 server_moid=5e33c03a6176752d33676f96 model=UCSC-C220-M4S
UCS-PBST-C-C220M4-2 server_moid=5e33be156176752d3366f948 model=UCSC-C220-M4S
UCS-PBST-C-C220M4-4 server_moid=5e18050b6176752d339e56f3 model=UCSC-C220-M4S
UCS-PBST-C-C220M4-3 server_moid=5e16e2966176752d3366afdd model=UCSC-C220-M4S
```

サーバ名

サーバの Moid

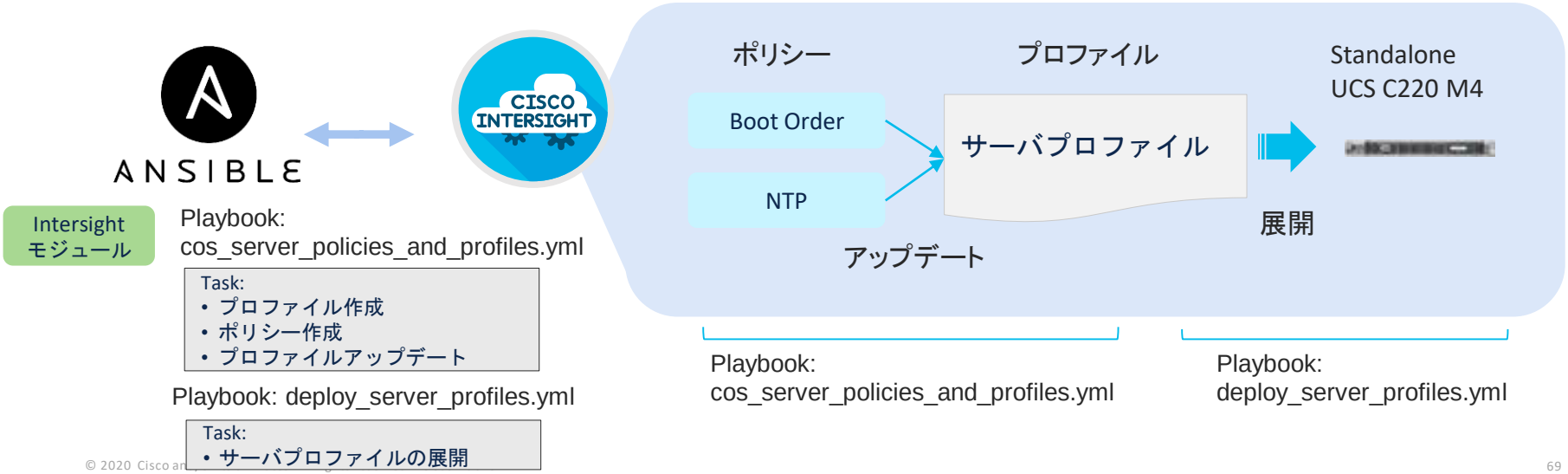
サーバモデル

※ Moid = Managed Object Identifier
REST リソース (Managed Object) が
作成されたときに割り当てられる固有の ID

Intersight + Ansible を利用した運用例

Scenario1 : サーバのポリシー/プロファイル設定

- Standalone UCS サーバのポリシー/プロファイルを作成し、サーバに展開 (設定) する。



ポリシー/プロファイル設定 Playbook

“cos_server_policies_and_profiles.yml” の中身抜粋

```
- hosts: "{{ group | default('Intersight_Servers') }}"
  connection: local
  collections:
    - cisco.intersight
...
  tasks:
    # Get the Organization Moid used by all profiles and policies
    - name: "Get Organization {{ org_name }} Moid"
      intersight_rest_api:
        <<: *api_info
        resource_path: /organization/Organizations
        query_params:
          $filter: "Name eq '{{ org_name }}'"
      register: org_resp
      delegate_to: localhost
      tags: always
    #
    # Configure profiles specific to server (run for each server in the inventory)
    # Server Profiles role will register a profile_resp and profile_resp list (from all hosts) can
be used by policy tasks
    #
    - name: "Configure {{ profile_name }} Server Profile"
      intersight_rest_api:
...
```

Collction ステートメント
各タスクでのモジュールの完全修飾が不要

ここからタスクを記載

組織の Moid を取得するタスク

サーバプロファイルを設定するタスク

ポリシー/プロファイル設定 Playbook - Boot Order Policy

“cos_server_policies_and_profiles.yml” - Task > Block 内の Boot Order Policy 設定

```
- block:
        # Boot Order policy
        - import_role:
            name: policies/server_policies
          vars:
            resource_path: /boot/PrecisionPolicies
            api_body: {
              "Name": "COS-Boot",
              "ConfiguredBootMode": "Legacy",
              "BootDevices": [
                {
                  "ObjectType": "boot.LocalDisk",
                  "Enabled": true,
                  "Name": "Disk",
                  "Slot": "MRAID"
                },
                {
                  "ObjectType": "boot.VirtualMedia",
                  "Enabled": true,
                  "Name": "VM",
                  "Subtype": "cimc-mapped-dvd"
                }
              ],
              "Organization": {
                "Moid": "{{ org_resp.api_response.Moid }}"
              }
            }
          tags: boot_order
```

Boot Order ポリシーの設定値

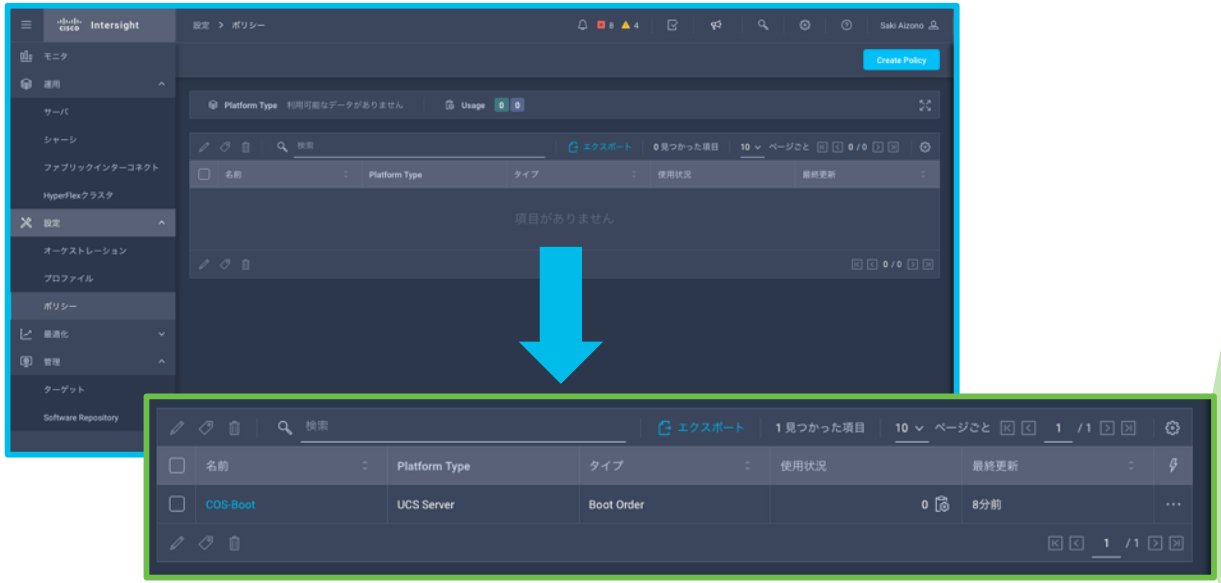
ポリシー名 : COS-Boot
設定されたブートモード : レガシー
ブートデバイス
- ローカルディスク
  有効 : Yes
  デバイス名 : Disk
  スロット : MRAID
- 仮想メディア
  有効 : Yes
  デバイス名 : VM
  サブタイプ : cimc-mapped-dvd

タグ : Playbook の特定の部分のみ実行できるようになる

ポリシー/プロファイル設定 Playbook - Boot Order Policy - タグを利用した部分実行 -

```
$ ansible-playbook -i inventory cos_server_policies_and_profiles.yml --tags boot_order
PLAY [Intersight_Servers] ******************************************************

TASK [Get Organization default Moid] *******************************************
ok: [UCS-PBST-C-C220M4-1]
ok: [UCS-PBST-C-C220M4-4]
ok: [UCS-PBST-C-C220M4-3]
ok: [UCS-PBST-C-C220M4-2]

TASK [policies/server_policies : Configure COS-NTP Server Policy] **************
changed: [UCS-PBST-C-C220M4-1]

TASK [policies/server_policies : set_fact] *************************************
skipping: [UCS-PBST-C-C220M4-1]

TASK [policies/server_policies : Update Server Profiles used by COS-NTP Server Policy (change may always be
reported)] ***
skipping: [UCS-PBST-C-C220M4-1]

PLAY RECAP *********************************************************************
UCS-PBST-C-C220M4-1        : ok=2    changed=1    unreachable=0    failed=0    skipped=2    rescued=0    ignored=0
UCS-PBST-C-C220M4-2        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-3        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-4        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Intersight GUI 結果画面



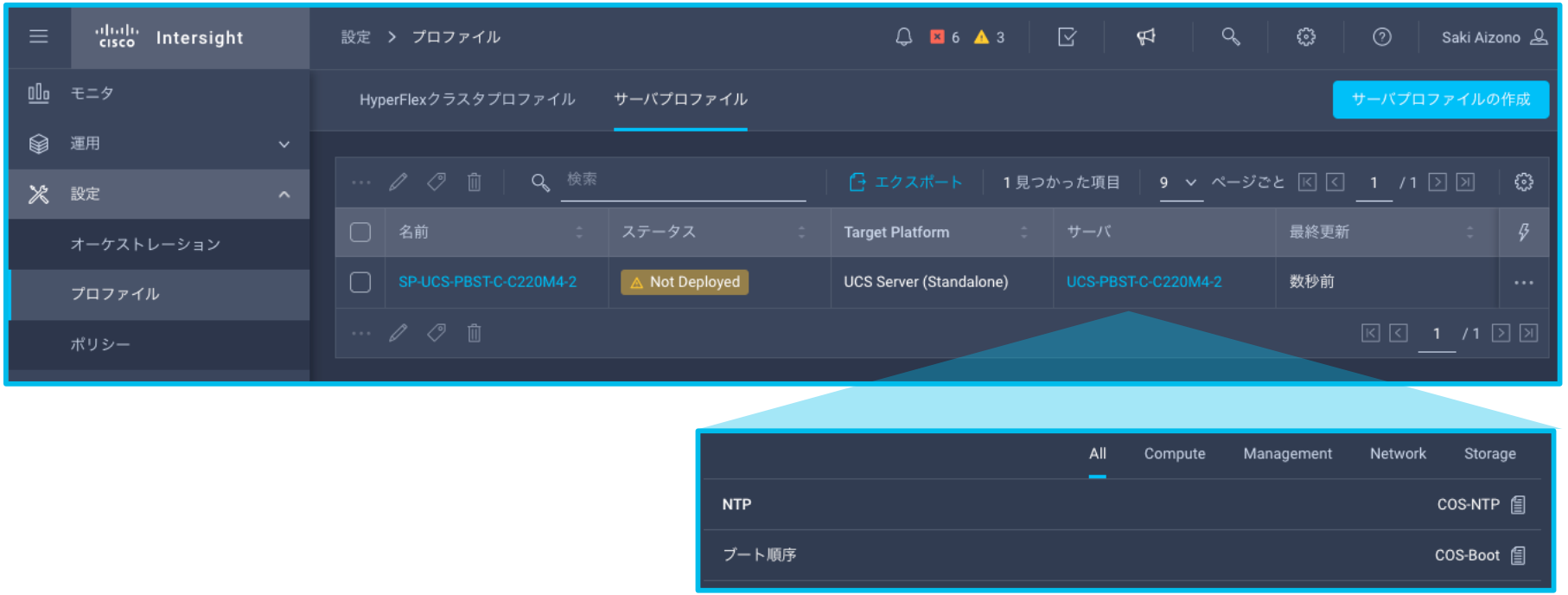
Playbook で指定したポリシーが作成される。

ポリシー/プロファイル設定 Playbook - 全体実行

カスタマイズをして実行した

```
$ ansible-playbook -i inventory cos_server_policies_and_profiles.yml
PLAY [Intersight_Servers] ******************************************************

TASK [Get Organization default Moid] *******************************************
ok: [UCS-PBST-C-C220M4-2]

TASK [Configure SP-UCS-PBST-C-C220M4-2 Server Profile] *************************
ok: [UCS-PBST-C-C220M4-2]

TASK [policies/server_policies : Configure COS-Boot Server Policy] *************
ok: [UCS-PBST-C-C220M4-2]

TASK [policies/server_policies : set_fact] *************************************
ok: [UCS-PBST-C-C220M4-2]

TASK [policies/server_policies : Update Server Profiles used by COS-Boot Server Policy (change may always be
reported)] ***
changed: [UCS-PBST-C-C220M4-2]

TASK [policies/server_policies : Configure COS-NTP Server Policy] **************
ok: [UCS-PBST-C-C220M4-2]
(...snip)

PLAY RECAP *********************************************************************
UCS-PBST-C-C220M4-2        : ok=16   changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Intersight GUI 結果画面



作成したプロファイルをサーバに展開する

- Playbook “deploy_server_profiles.yml” を利用してサーバにプロファイルを展開する。

```
$ ansible-playbook -i inventory deploy_server_profiles.yml
PLAY [Intersight_Servers] ******************************************************

TASK [Deploy (or user defined action) Server Profile] **************************
changed: [UCS-PBST-C-C220M4-2]

PLAY RECAP *********************************************************************
UCS-PBST-C-C220M4-2        : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Intersight GUI 結果画面

その他の Intersight + Ansible 運用例

Scenario :

2. サーバの HCL - Hardware Compatibility List 情報の取得
3. Intersight へのデバイスの登録 (参考情報リンク)
4. HyperFlex の構築 (参考情報リンク)

Scenario 2 : サーバの HCL 情報の取得

- Ansible Playbook “hcl_status.yml” により、各サーバの HCL リストを取得する。
- 取得した情報は csv 形式でローカルに保存する。

Playbook ”hcl_status.yml” タスク内容

```
tasks:
    # Get HclStatus
    - name: Get HCL Status for Server
      intersight_rest_api:
        <<: *api_info
        resource_path: /cond/HclStatuses
        query_params:
          $filter: "ManagedObject.Moid eq '{{ server_moid }}'"
      delegate_to: localhost
      register: hcl_resp
      when:
        - server_moid is defined
    # Create .csv file with version and status information
    - copy:
        content: |
          Name, FW version, OS vendor, OS version, HW status, SW status, Overall Status
          {% for host in hostvars %}
            {% set vars = hostvars[host|string] %}
            {% if vars.hcl_resp.api_response is defined %}
              {{ vars.inventory_hostname }}, {{ vars.hcl_resp.api_response.HclFirmwareVersion }}, {{ vars.hcl_resp.api_response.HclOsVendor
}}, {{ vars.hcl_resp.api_response.HclOsVersion }}, {{ vars.hcl_resp.api_response.HardwareStatus }}, {{
vars.hcl_resp.api_response.SoftwareStatus }}, {{ vars.hcl_resp.api_response.Status }} {{ vars.hcl_resp.api_response.ServerReason }}
            {% endif %}
          {% endfor %}
        dest: /tmp/hcl_status.csv
        backup: false
      run_once: true
      delegate_to: localhost
```

HCL ステータスを取得するタスク

.csv ファイルを作成するタスク

Playbook ”hcl_status.yml” の実行と結果

- Playbook の実行

```
$ ansible-playbook -i inventory hcl_status.yml
PLAY [Intersight_Servers] ******************************************************

TASK [Get HCL Status for Server] ***********************************************
ok: [UCS-PBST-C-C220M4-3]
ok: [UCS-PBST-C-C220M4-1]
ok: [UCS-PBST-C-C220M4-2]
ok: [UCS-PBST-C-C220M4-4]

TASK [copy] ********************************************************************
ok: [UCS-PBST-C-C220M4-1]

PLAY RECAP *********************************************************************
UCS-PBST-C-C220M4-1        : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-2        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-3        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-4        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Playbook ”hcl_status.yml” の実行と結果

- Playbook の実行

```
$ ansible-playbook -i inventory hcl_status.yml
PLAY [Intersight_Servers] ******************************************************

TASK [Get HCL Status for Server] ***********************************************
ok: [UCS-PBST-C-C220M4-3]
ok: [UCS-PBST-C-C220M4-1]
ok: [UCS-PBST-C-C220M4-2]
ok: [UCS-PBST-C-C220M4-4]

TASK [copy] ********************************************************************
ok: [UCS-PBST-C-C220M4-1]

PLAY RECAP *********************************************************************
UCS-PBST-C-C220M4-1        : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-2        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-3        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
UCS-PBST-C-C220M4-4        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

作成された .csv ファイル内容

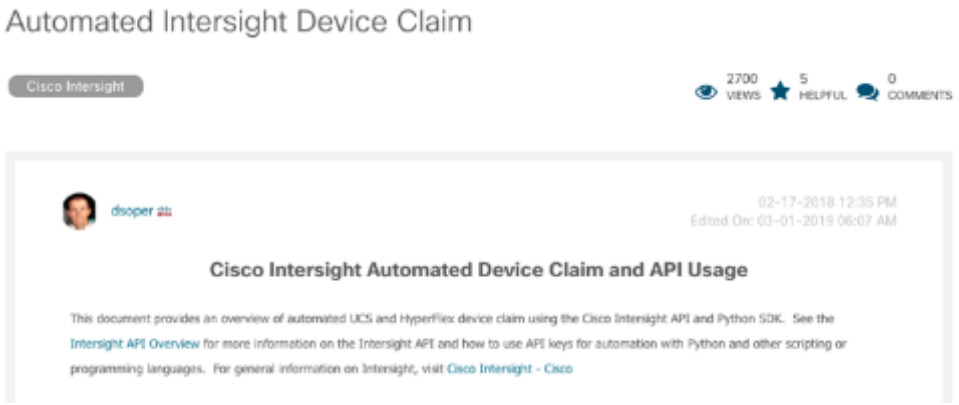
- “hcl_status.csv” ファイルが作成される。

hcl_status

Name	FW version	OS vendor	OS version	HW status	SW status	Overall Status
UCS-PBST-C-C220M4-1	4.0(2)	VMware	ESXi 6.5 U1	Validated	Validated	Not-Listed Incompatible-Components
UCS-PBST-C-C220M4-2	4.0(1)	VMware	ESXi 6.5 U1	Validated	Validated	Not-Listed Incompatible-Components
UCS-PBST-C-C220M4-3	4.0(2)	VMware	ESXi 6.5 U1	Validated	Validated	Not-Listed Incompatible-Components
UCS-PBST-C-C220M4-4	4.0(1)			Validated	Incomplete	Incomplete Missing-Os-Driver-Info

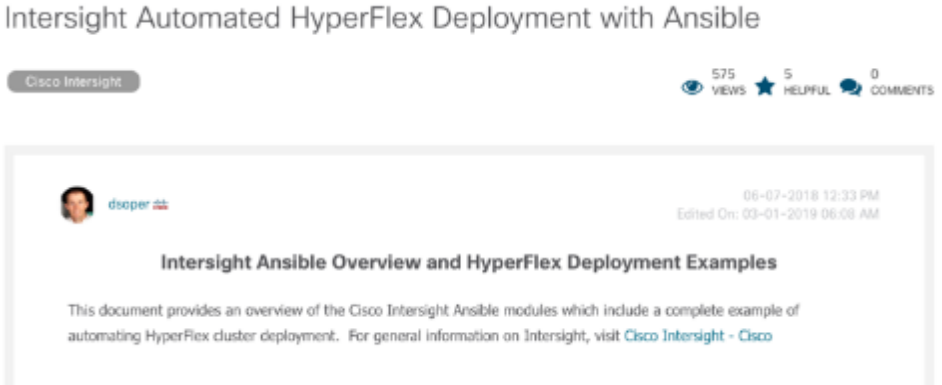
Scenario 3 : Intersight へのデバイス登録

- Python もしくは Ansible によりデバイスクレームプロセスの自動化が可能
- スクリプトにより、デバイスにアクセスし、デバイス情報と要求コードを自動で取得
- Video demo of automated device claim



Scenario 4 : HyperFlex の構築

- Intersight API を利用して、HyperFlex のポリシー/プロファイル作成およびクラスタの展開実施可能
  - Intersight Ansible モジュールを利用
  - 並行して複数のクラスタをインストール可能
- Video demo of HX automation



Connected TAC

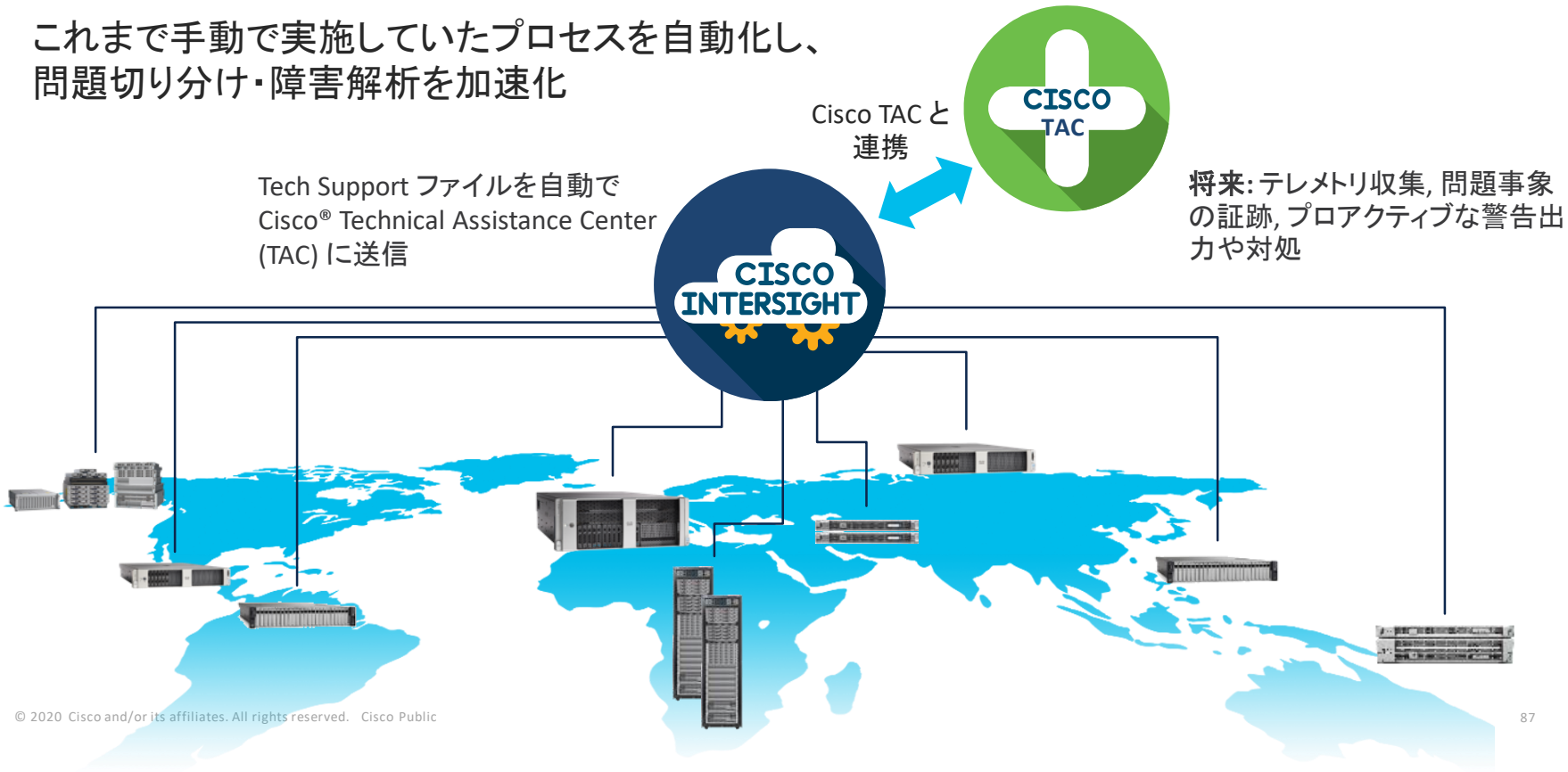
Connected TAC

これまで手動で実施していたプロセスを自動化し、
問題切り分け・障害解析を加速化

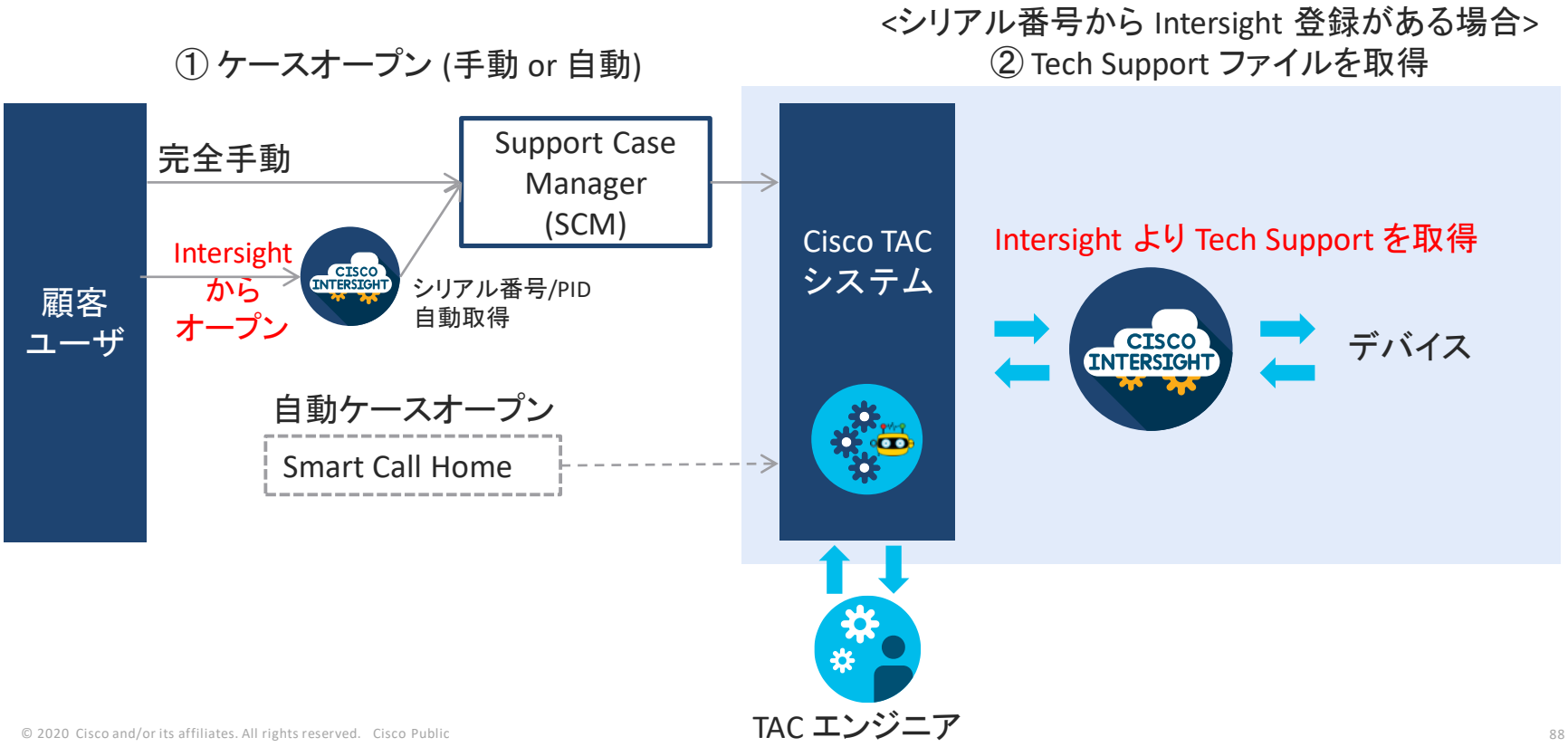
Cisco TAC と
連携

Tech Support ファイルを自動で
Cisco® Technical Assistance Center
(TAC) に送信

将来: テレメトリ収集, 問題事象の証跡, プロアクティブな警告出力や対処



Connected TAC の仕組み



① Intersight からの TAC ケースオープン

該当デバイスのアクションから「TAC ケースを開く」を選択
- サーバ
- ファブリックインターコネクト

Support Case Manager (SCM) が開かれ、シリアル番号などが自動的に入力される

シリアル番号等を自動的に取得
「続ける」を選択

② Intersight からの Tech Support ファイル取得

シリアル番号から Intersight 登録を判断登録があれば、Tech Support を取得

- Tech Support を取得・閲覧できるのは TAC ケースに関する Cisco エンジニアのみ (顧客、パートナーの実施は不可)
- Tech Support 取得は顧客に通知されない

デバイス種別	取得する Tech Support 種別
B-Series Motherboard	Chassis Tech Support
B-Series VIC/Adapter	Chassis Tech Support
IOM	Chassis Tech Support
Chassis	Chassis Tech Support
Fabric Interconnect	UCSM Tech Support
C-Series Motherboard	CIMC Tech Support
C-Series VIC/Adapter	CIMC Tech Support
FEX	FEX Tech Support

Connected TAC の効果 (顧客視点)

実際の TAC ケースの例

TAC ケースオープン:
XXXXXX244

Tech Support ファイルを
自動的に収集

分析結果を
自動的に生成

既知不具合に一致しており、ファームウェアアップグレードとの見解を提供

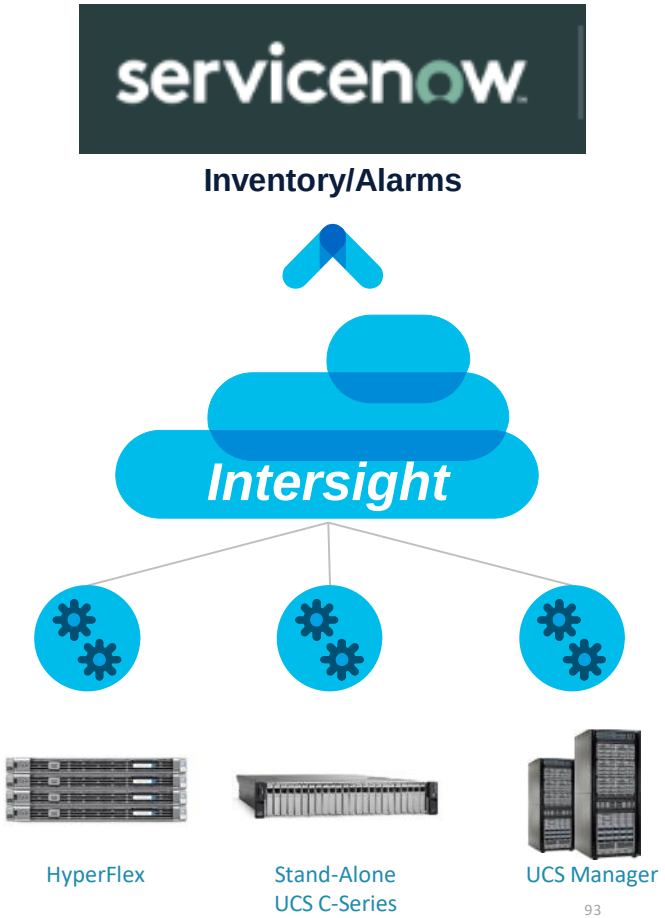
2018-12-DD 01:25 → 12 分後 → 16 分後

問題を診断するまでに要した時間:
合計 28 分

ServiceNow 連携

Intersight - ServiceNow 連携

- ServiceNow プラグインを提供
  - Cisco Intersight ITSM Plugin
- Read-Only の API アクセスで連携可能
- プラグインの機能
  - デバイスインベントリ情報の同期
  - インシデント管理
    - Intersight から取得したアラーム情報を利用
  - ServiceNow による分析レポートとダッシュボード
- Intersight Essentials ライセンス以上が必要



Cisco Intersight ITSM Plugin

https://store.servicenow.com/sn_appstore_store.do#!/store/application/baef2ff5db132300d7b59235ca9619e3

Cisco Intersight ITSM Plugin

now Certified App

Visibility and monitoring for Cisco UCS and HyperFlex.

Cisco Systems Inc

Compatibility: Paris, Orlando | **Other App Versions**

ServiceNow 互換性

Pricing

Free

☆☆☆☆☆ No Reviews

Share With

Product Details ☆ Ratings and Reviews

Summary

Cisco Intersight is a Software-as-a-Service (SaaS) infrastructure management platform that is augmented by other intelligent systems. It provides global management of the Cisco Unified Computing System™ (Cisco UCS®) and Cisco HyperFlex® hyperconverged infrastructure anywhere. Intersight provides a holistic approach to managing distributed computing environments from the core to the edge. The Cisco Intersight virtual appliance provides customers with deployment options while still

Get

Contact Seller → プラグインの入手

Type
Integration

Version
1.3.0
Other App Versions

Dependencies and Licensing
View Dependencies and Licensing Requirements

Compatibility
Paris
Orlando

Supporting Links and Docs
ServiceNow Store Terms Of Use
Vendor App Subscription Terms and Conditions
Installation Guide
Cisco Intersight

ドキュメントの入手

Cisco Intersight ITSM Plugin リリース

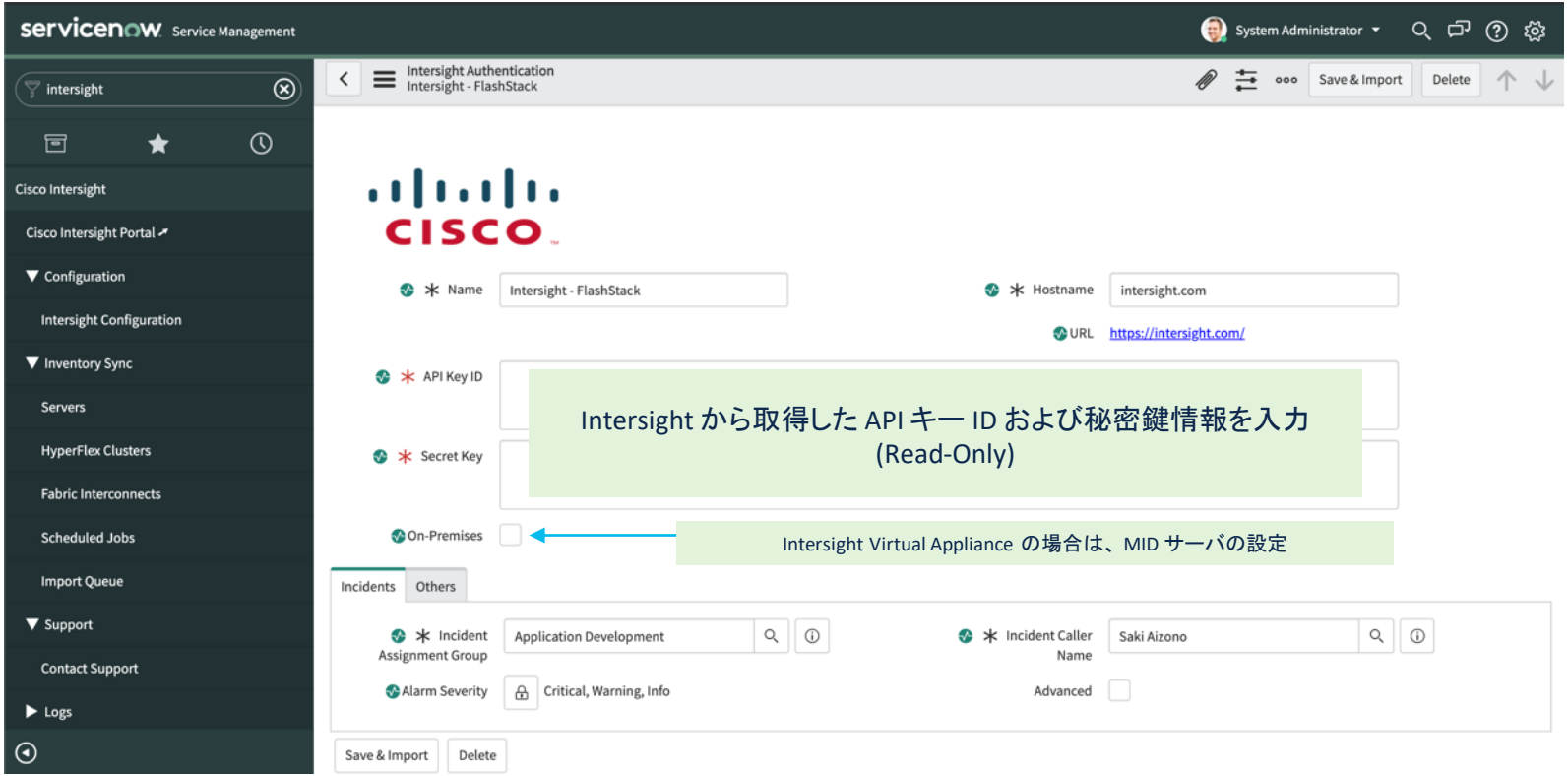
- 1.0.1 : NewYork
- 1.2.0 : Orlando
- 1.3.0 : Paris, Orlando

本日のデモは Paris の developer instance を利用

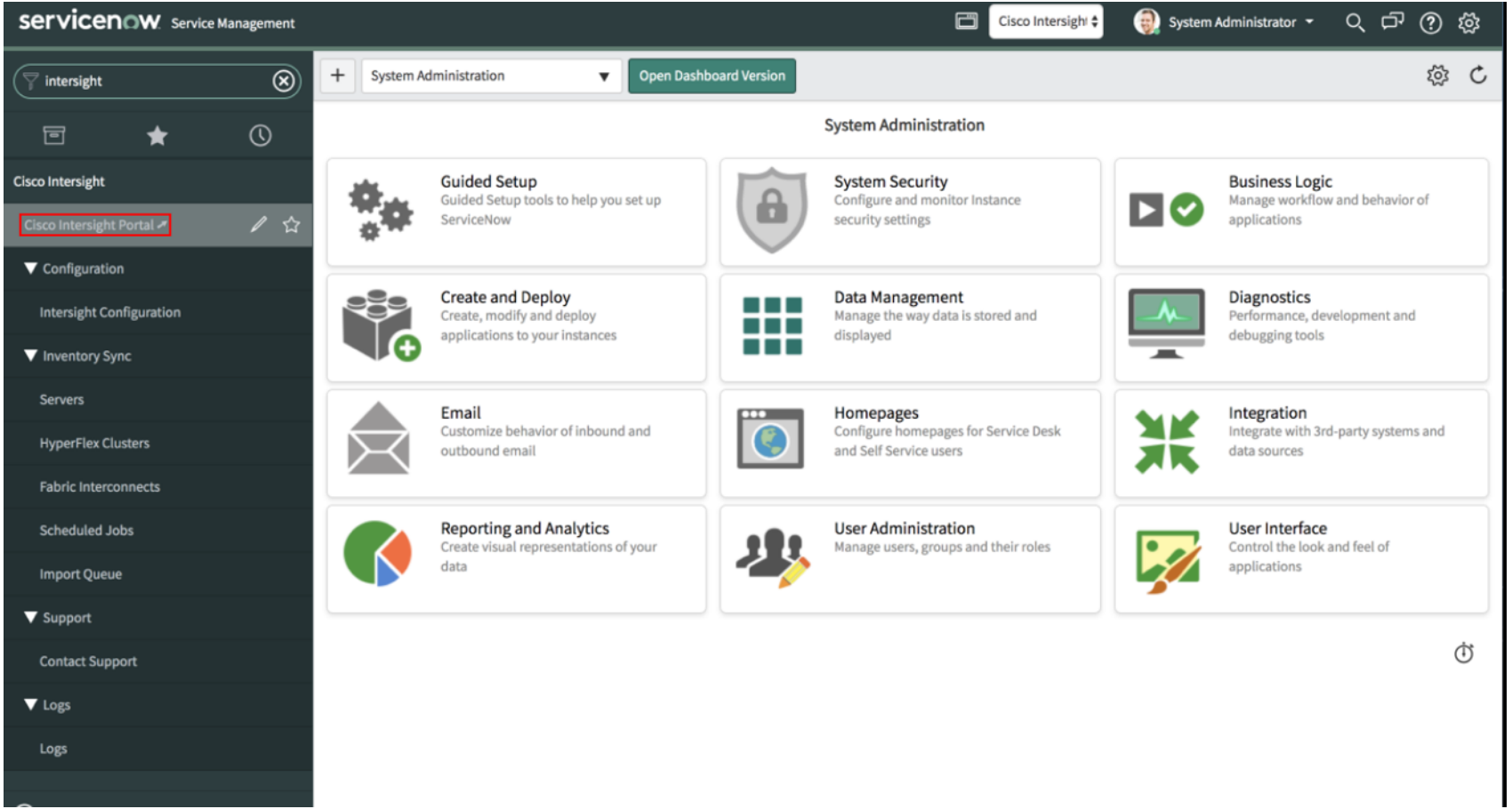
Cisco Intersight ITSM Plugin インストール手順

1. Performance Analytics and Reporting Plugin のアクティベート
   - Cisco Intersight ITSM Plugin は、ダッシュボードウィジェットを表示するために Performance Analytics and Reporting plugin を利用
2. (Intersight Virtual Appliance 利用の場合) MID サーバの設定
3. Cisco Intersight ITSM Plugin のインストール

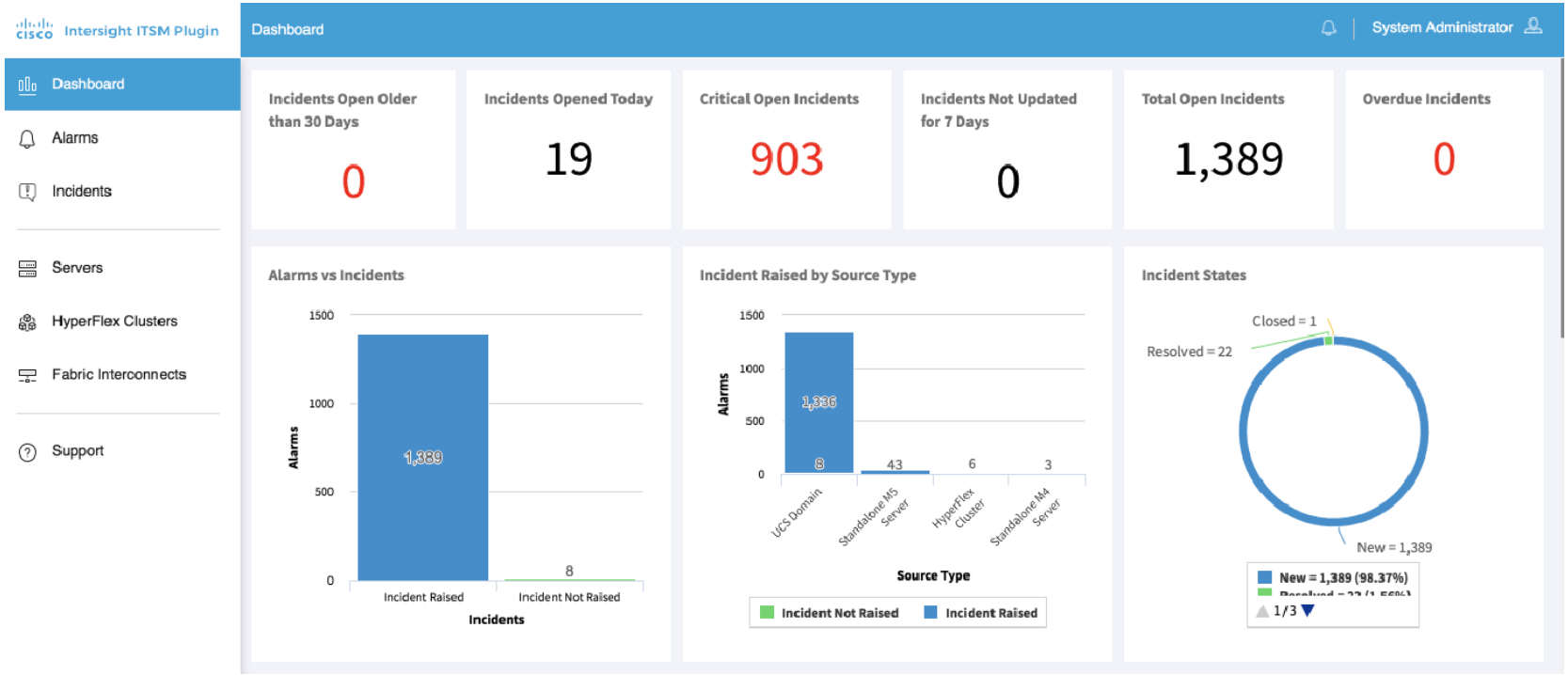
Cisco Intersight ITSM Plugin 設定



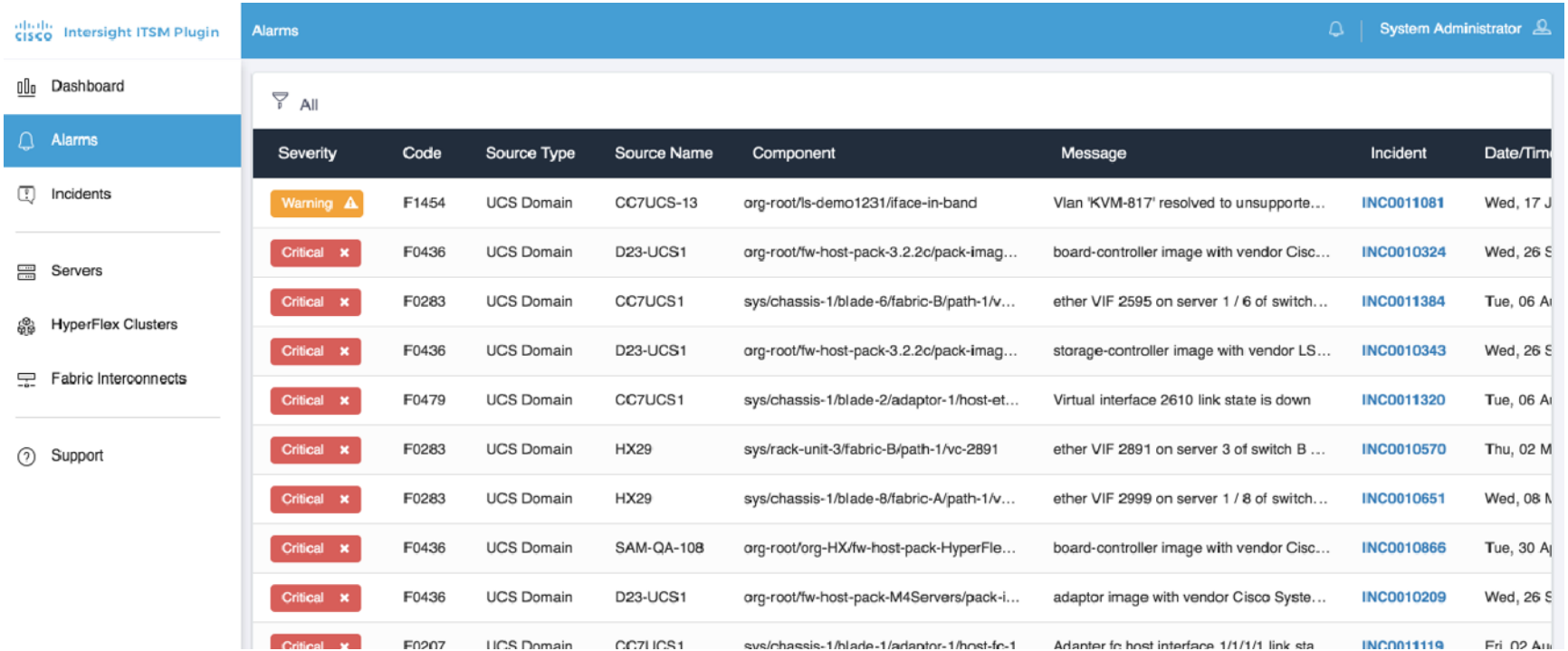
ServiceNow 連携サービスメニュー



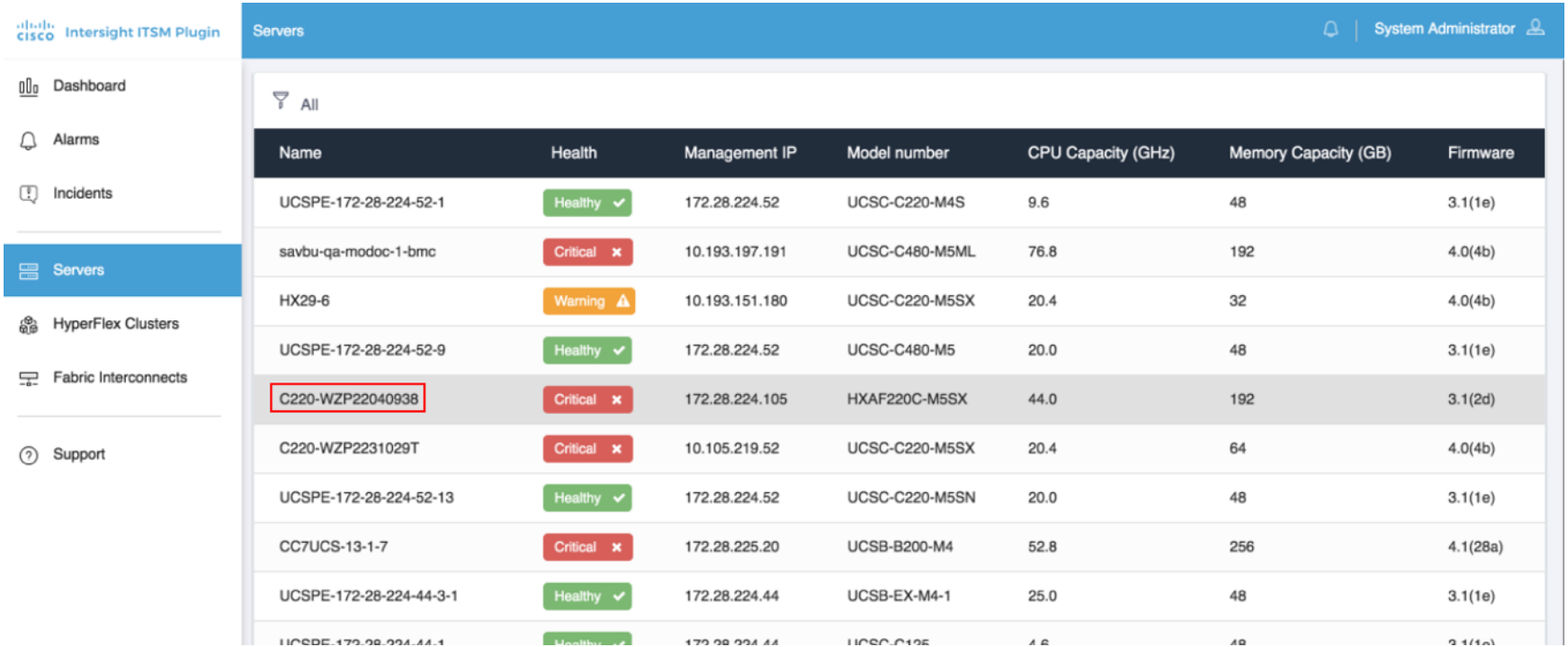
Cisco Intersight ITSM Plugin ダッシュボード



Cisco Intersight ITSM Plugin アラーム情報

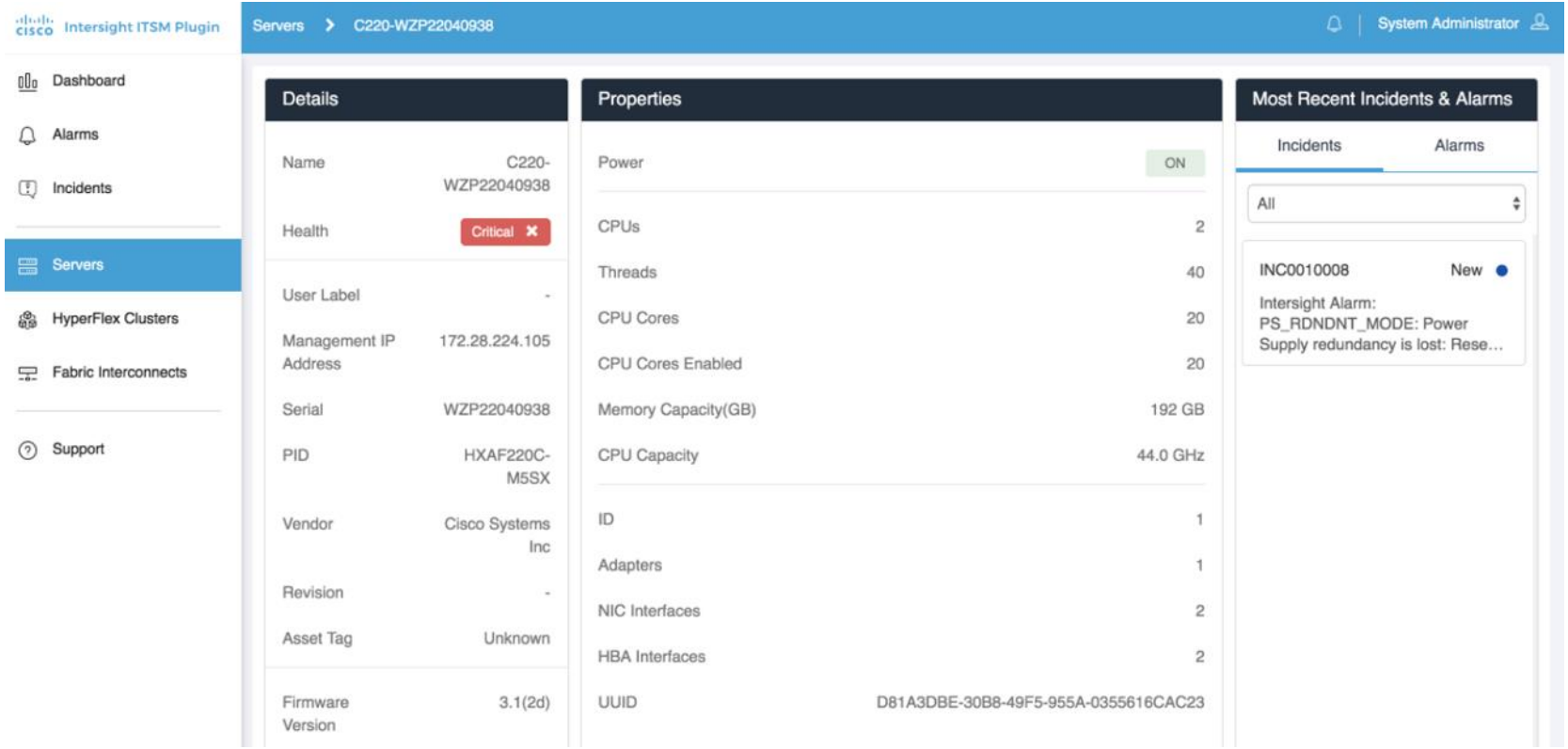


Cisco Intersight ITSM Plugin インベントリ情報一覧

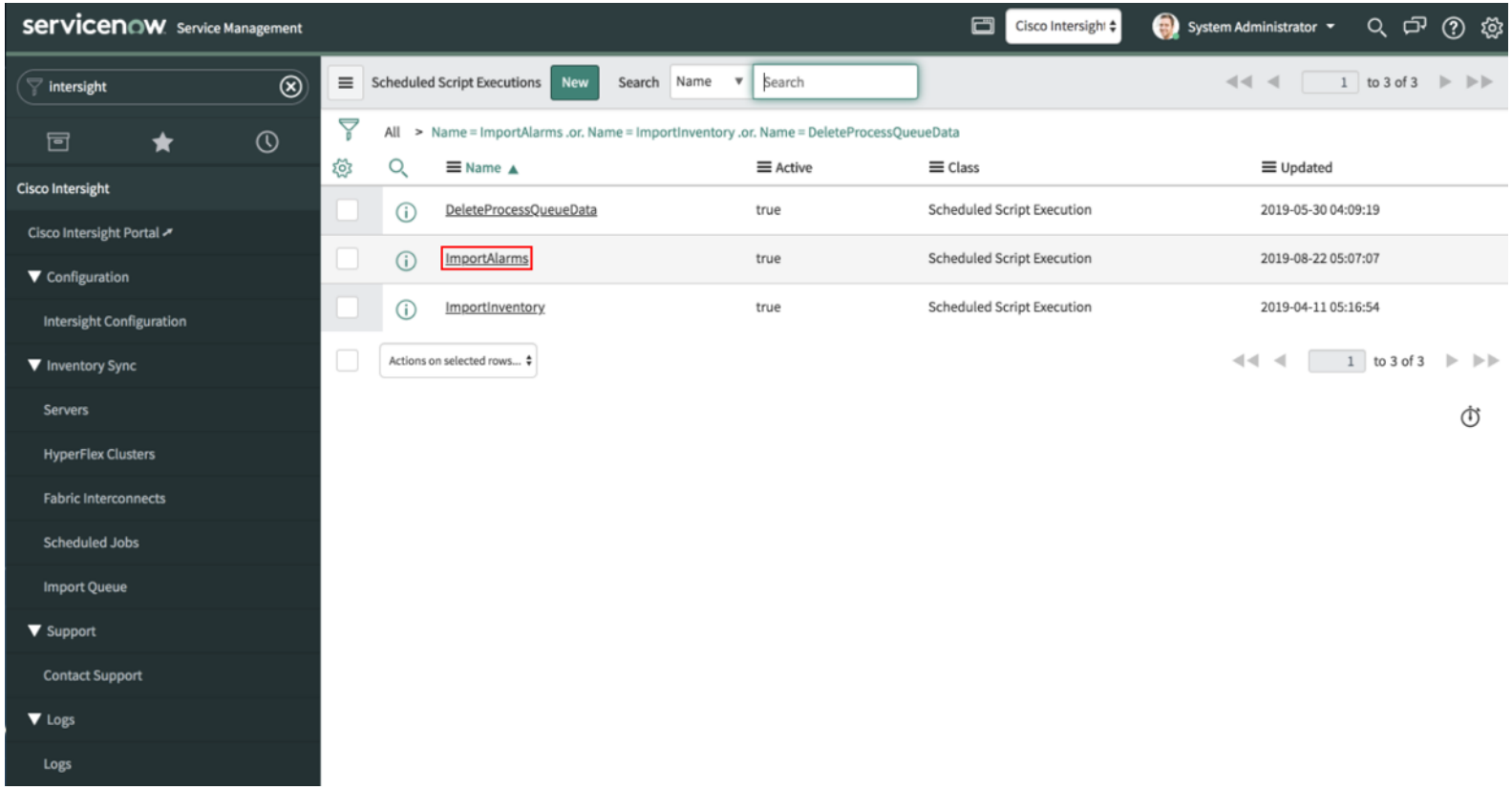


Name	Health	Management IP	Model number	CPU Capacity (GHz)	Memory Capacity (GB)	Firmware
UCSPE-172-28-224-52-1	Healthy	172.28.224.52	UCSC-C220-M4S	9.6	48	3.1(1e)
savbu-qa-modoc-1-bmc	Critical	10.193.197.191	UCSC-C480-M5ML	76.8	192	4.0(4b)
HX29-6	Warning	10.193.151.180	UCSC-C220-M5SX	20.4	32	4.0(4b)
UCSPE-172-28-224-52-9	Healthy	172.28.224.52	UCSC-C480-M5	20.0	48	3.1(1e)
C220-WZP22040938	Critical	172.28.224.105	HXAF220C-M5SX	44.0	192	3.1(2d)
C220-WZP2231029T	Critical	10.105.219.52	UCSC-C220-M5SX	20.4	64	4.0(4b)
UCSPE-172-28-224-52-13	Healthy	172.28.224.52	UCSC-C220-M5SN	20.0	48	3.1(1e)
CC7UCS-13-1-7	Critical	172.28.225.20	UCSB-B200-M4	52.8	256	4.1(28a)
UCSPE-172-28-224-44-3-1	Healthy	172.28.224.44	UCSB-EX-M4-1	25.0	48	3.1(1e)

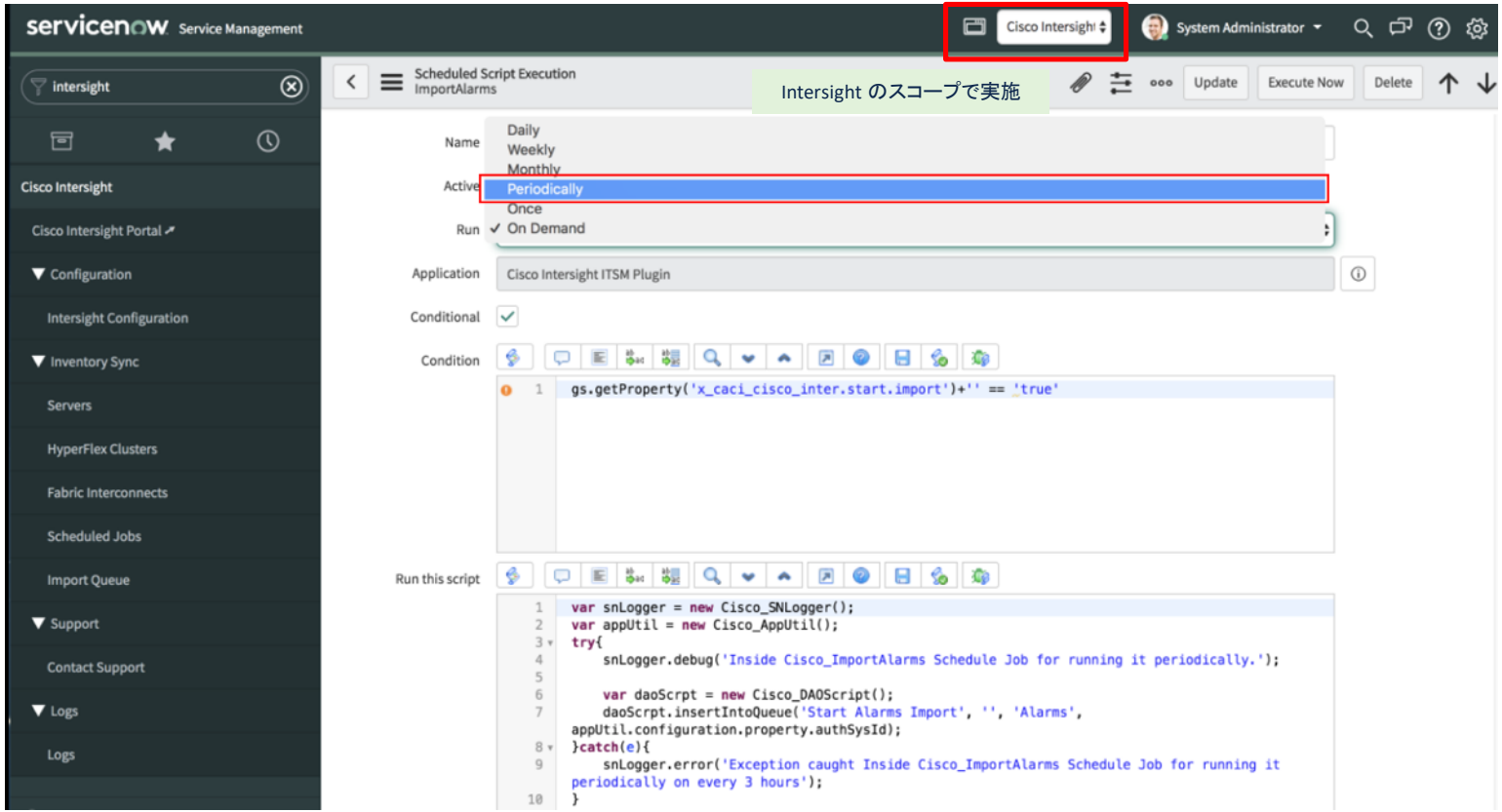
Cisco Intersight ITSM Plugin インベントリ情報詳細



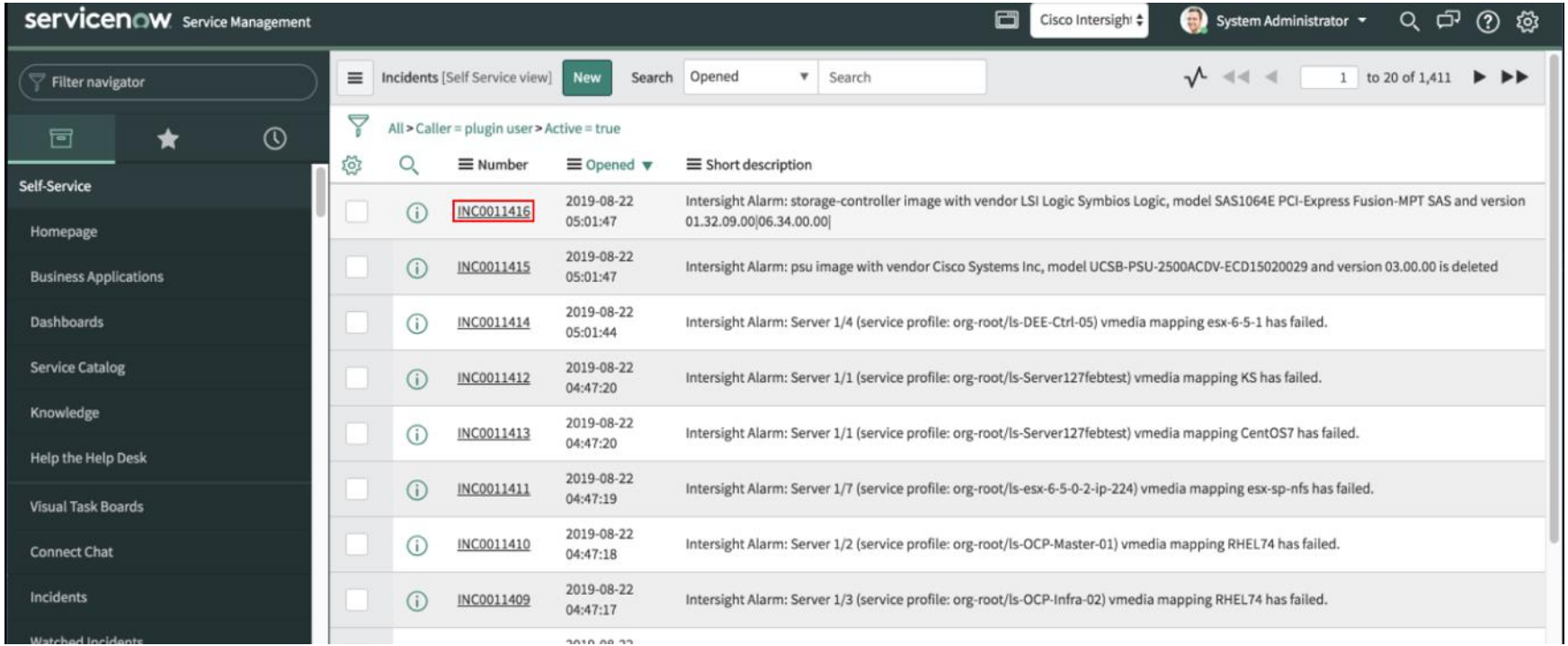
アラーム、インベントリインポートジョブ



Intersight アラームのインポート (実行間隔の設定)



インシデント管理



Intersight Workload Optimizer (IWO)

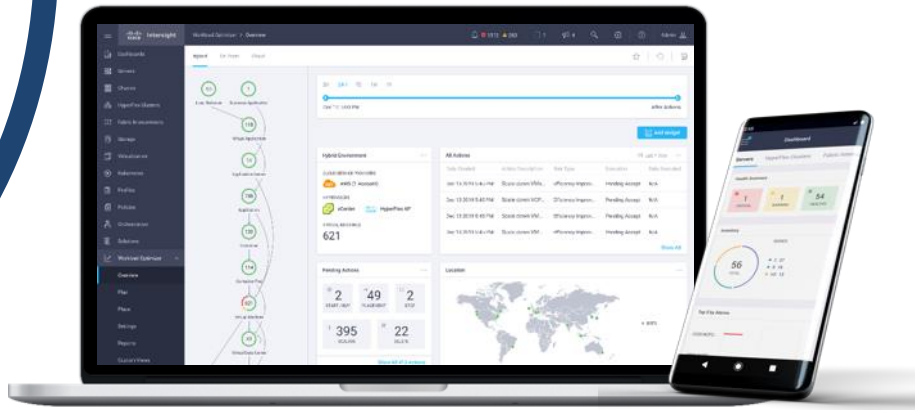
Cisco Intersight の開発コンセプトの拡大

クラウド運用モデル

クラウド

ワークロード

インフラストラクチャ



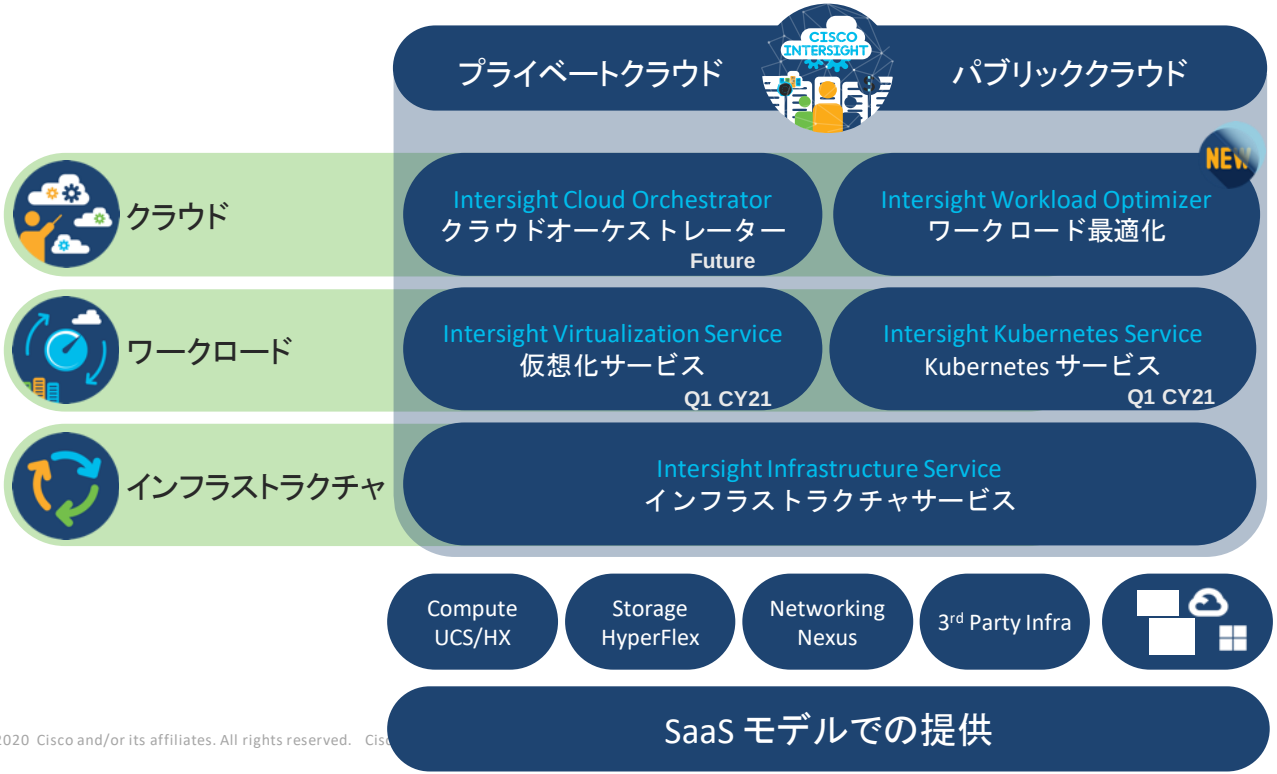
シンプルな運用

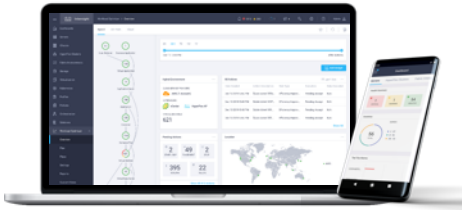
継続的な最適化

アジャイルによる提供

Cisco Intersight の機能モジュール

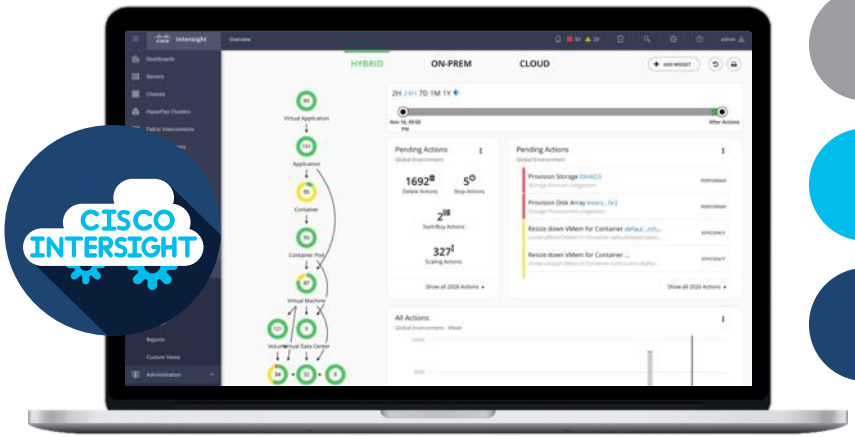
インフラストラクチャサービスに加えて、ワークロード、クラウドにも機能を拡張





2020/12/3 正式リリース

Intersight Workload Optimizer (IWO)



ハイブリッドクラウド環境を完全に視覚化
アプリケーションとインフラの依存関係をマッピング

リソースの需要と供給をリアルタイムに分析
パフォーマンス、コスト、コンプライアンス

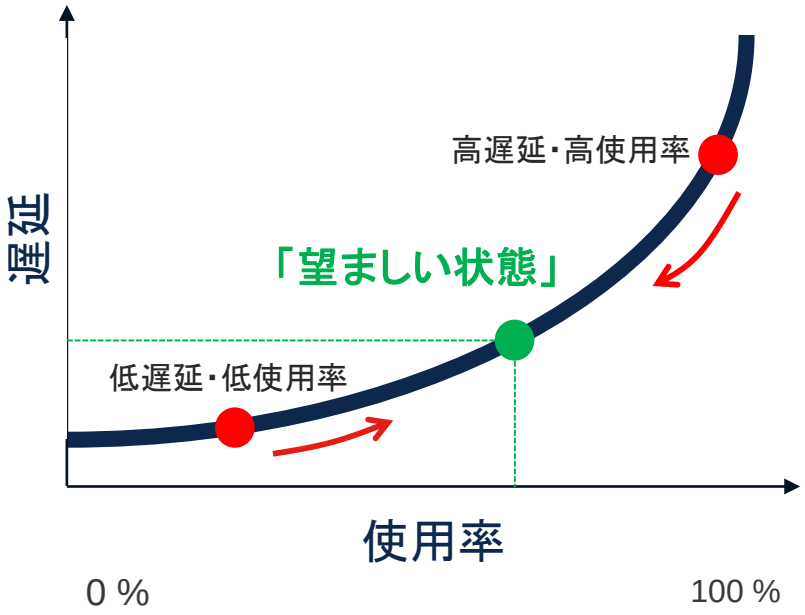
具体的なアクションを推奨
様々なターゲットタイプに対し、アクションをトリガー

マシンインテリジェンスを用いた
アプリケーションリソース管理の簡素化

オーバープロビジョニングを排する
ことによる投資保護と運用費の削減

アクション実行の即時性とキャパシテ計画
のシナリオモデル化によるリスク低減

IWO が定義するリソースの「望ましい状態」



- 低遅延・低使用率=リソースの非効率的な使用状態、リソース過多
  - リソース利用率の最適化
- 高遅延・高使用率=パフォーマンスが確保出来ない状態、リソース不足
  - パフォーマンスの最適化

「望ましい状態」とは、**インフラを効率的**に利用しながら**パフォーマンスを確保**している状態

IWO は「望ましい状態」に向かって運転する

パフォーマンスの保証

リソースを最も必要とするワークロードに
自動的にリソースを割り当てることで
パフォーマンスを最適化する

コンプライアンスの確保

ポリシーを常に遵守しながら、自動的に
ワークロードを配置、サイジング、
移行する

コストの削減

ワークロード密度とリソース使用率を
自動的に最大化し、インフラストラクチャ/
クラウドコストを最小限に抑える

常時バランスをとりながら
同時に解決する

1. パフォーマンスの向上
2. コンプライアンスの向上
3. コストの削減

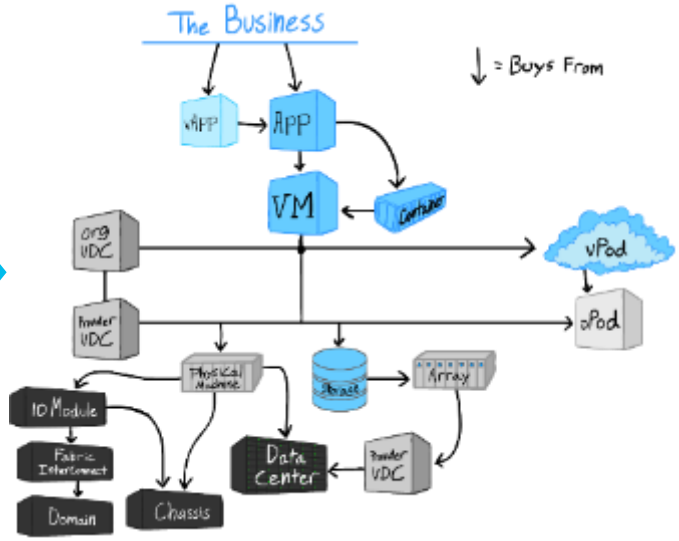
DC 内のフルスタックのリソース関係を把握

登録ターゲット
(管理ソフトウェア)

- APPLICATION PERFORMANCE MANAGEMENT
- PROVISIONING & ORCHESTRATION
- CONFIGURATION & MIGRATION
- CONTAINERS
- APPLICATION SERVERS
- DATABASES
- VIRTUALIZATION
- HYPERCONVERGED
- NETWORK
- STORAGE
- COMPUTE

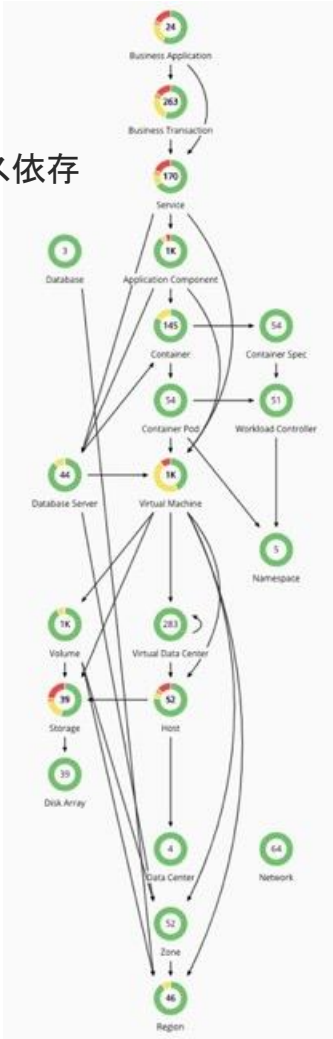
CISCO INTERSIGHT
Workload Optimizer

管理ソフトウェアから
情報を収集



データ・センター内のすべてを
リソース (CPU、メモリ、IO 等) を売買する
サプライチェーン市場に抽象化

1時間以内でリソース依存
関係を GUI に表示
(サプライチェーン)



登録可能なターゲット (2020/12/16 現在)

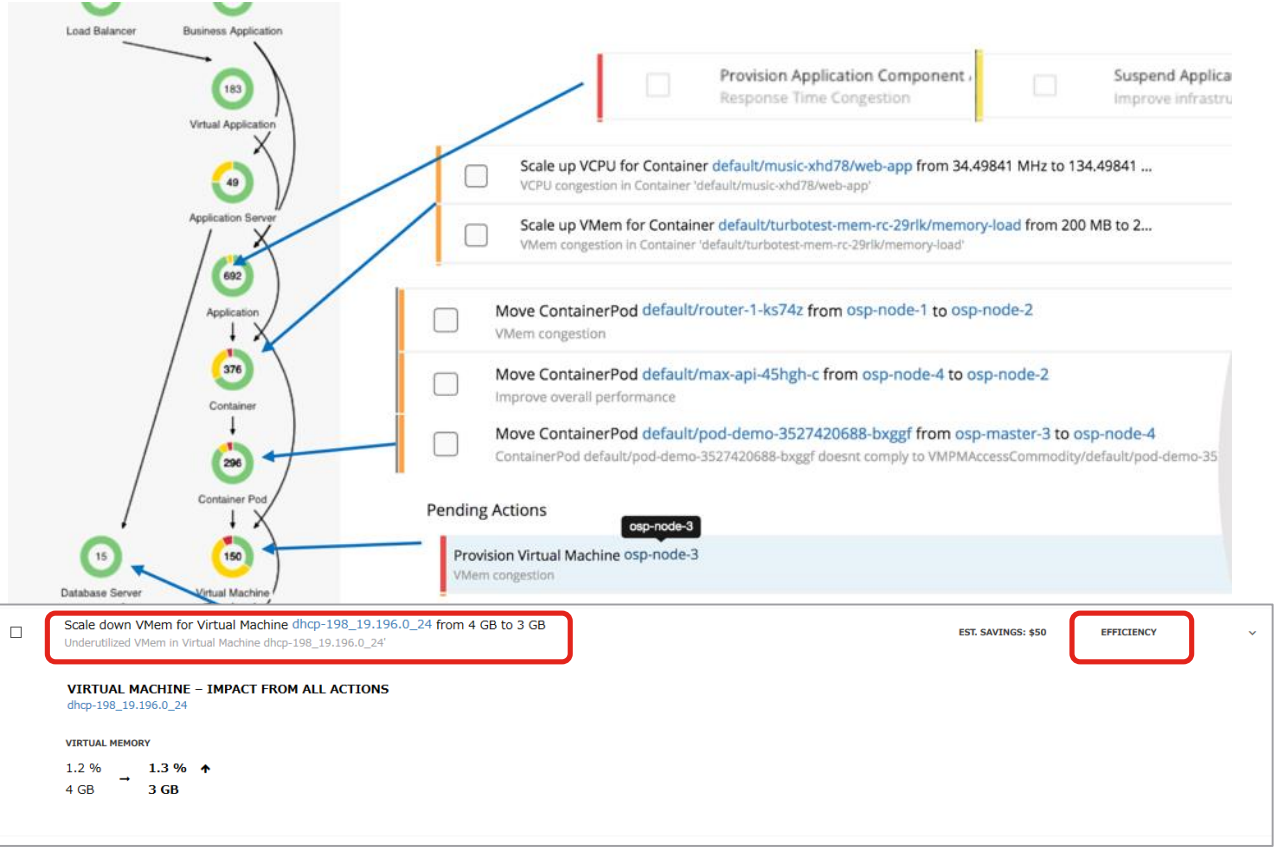
Category	Target Name	Minimum License Tier Required for Intersight Workload Optimizer	Intersight Assist Required
Cloud	Amazon Web Services	Intersight Workload Optimizer Essentials	No
	Amazon Web Services Billing	Intersight Workload Optimizer Essentials	No
	Microsoft Azure Enterprise Agreement	Intersight Workload Optimizer Essentials	No
	Microsoft Azure Service Principal	Intersight Workload Optimizer Essentials	No
Cloud Native	Kubernetes	Intersight Workload Optimizer Advantage	No
Compute / Fabric	Cisco UCS Server (Standalone)	Intersight Workload Optimizer Essentials	No
	Cisco UCS Domain (UCSM Managed)	Intersight Workload Optimizer Essentials	No
Application Performance Management (APM)	Cisco AppDynamics	Intersight Workload Optimizer Advantage	Yes
Hyperconverged	Cisco HyperFlex	Intersight Workload Optimizer Essentials	No
Hypervisor	VMware vCenter	Intersight Workload Optimizer Essentials	Yes
Storage	EMC VMAX	Intersight Workload Optimizer Essentials	Yes
	EMC XtremeIO	Intersight Workload Optimizer Essentials	Yes
	EMC ScaleIO	Intersight Workload Optimizer Essentials	Yes
	EMC VPLEX	Intersight Workload Optimizer Essentials	Yes
	NetApp	Intersight Workload Optimizer Essentials	Yes
	Pure Storage FlashArray	Intersight Workload Optimizer Essentials	Yes

- Intersight Assist VM
  - オンプレミスの Third Party 製品を接続する場合に必要
  - Cisco データセンター製品の場合は、デバイスコネクタで直接接続が可能なために不要
  - パブリッククラウドおよび Kuburnetes 環境は不要
- GA 後追加ソフトウェアをサポート予定

https://www.intersight.com/help/resources/cisco_intersight_workload_optimizer_getting_started#supported_targets

リアルタイム分析により IWO が提示するアクション

- アクションの種類
  - リソースの再配置
  - リソースの拡張・縮小
  - リソースの追加
  - リソースの停止・起動
- アクションによる効果
  - パフォーマンスの最適化
  - リソース利用の効率化
  - コンプライアンス遵守
- アクションの実行
  - 推奨するのみ
  - 手動
  - 自動 (ソフトウェアによる)



参考情報

Cisco Intersight 情報

- Intersight Online Help
  https://intersight.com/help
- Intersight How To (日本語のシスココミュニティサイト)
  https://community.cisco.com/t5/-/-/ta-p/4072043

Cisco dCloud デモ環境

- Getting Started with Cisco Intersight v1
  https://dcloud2-rtp.cisco.com/content/instantdemo/getting-started-with-cisco-intersight-v1?returnPathTitleKey=content-view
- Cisco UCS Management with Intersight v1
  https://dcloud2-rtp.cisco.com/content/demo/467125?returnPathTitleKey=content-view
- Cisco Intersight Workload Optimizer (Preview) v1
  https://dcloud2-sng.cisco.com/content/instantdemo/cisco-intersight-workload-optimizer-tech-preview-v1?returnPathTitleKey=content-view


CISCO