Валидация конфигурации или "Всё правильно сделал"

Требования

Валидация конфигурации оборудования может оказаться крайне полезной. Помогает избежать опечаток, простых ошибок и сохранить единство принципов конфигураций на сети, в случае, если настройка производится разными людьми. Принципы работы следующие:

1. Нок получает конфигурацию с оборудования.
2. Специально написанный парсер проходит по конфигурации (это происходит по последней актуальной конфигации из базы НОКа).
3. Результатом работы парсера становится набор фактов вида: версия ssh 2, на интерфейсу присвоен такой-то адрес и т.д.
4. Далее мы добавляем правила, которым должен соответствовать факт или набор фактов.
5. Объединяем правила в политику.
6. Привязываем политику к оборудованию.

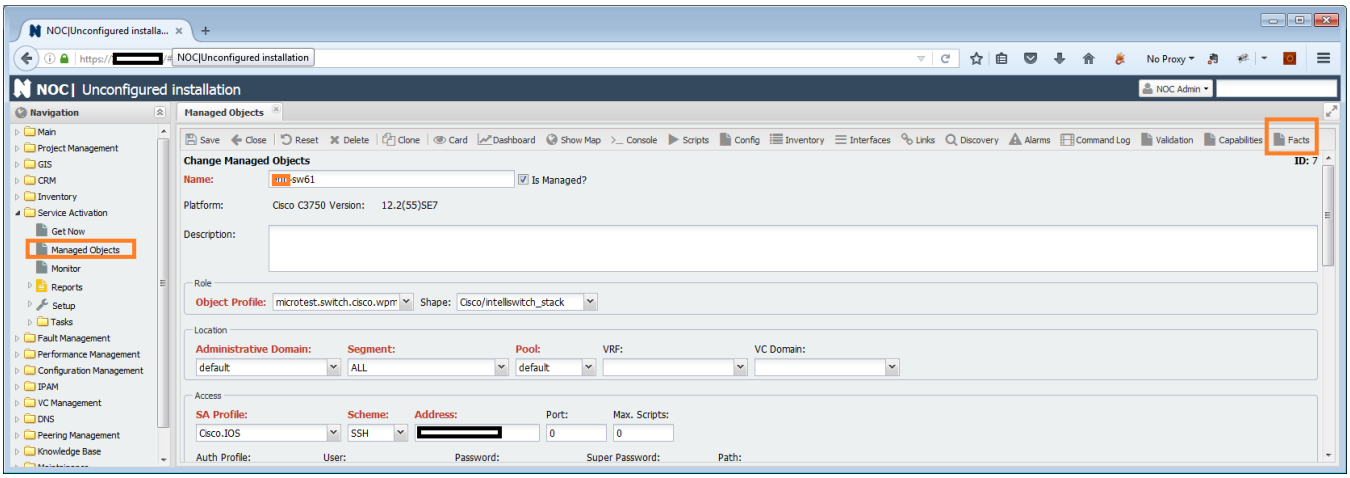
Теперь, после того как в НОКе обновится конфигурация оборудования, произойдёт автоматическая проверка правил и в случае срабатывания создастся Предупреждение.

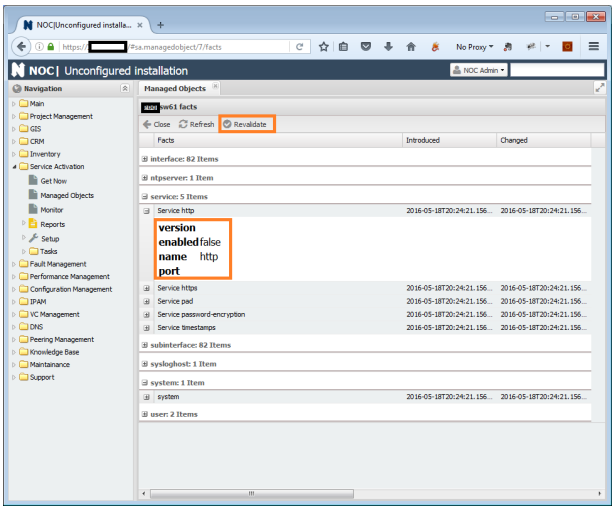
Исходя из описания, для работы валидации необходимо:

1. Написанный парсер, превращающий конфигурацию оборудования в набор фактов.
2. Правила, на основе фактов, контролирующие правильность конфигурации.
3. Политики привязки правил к оборудованию.

Смотрим факты

Посмотреть собранные факты можно нажав на кнопку Facts на странице МО.



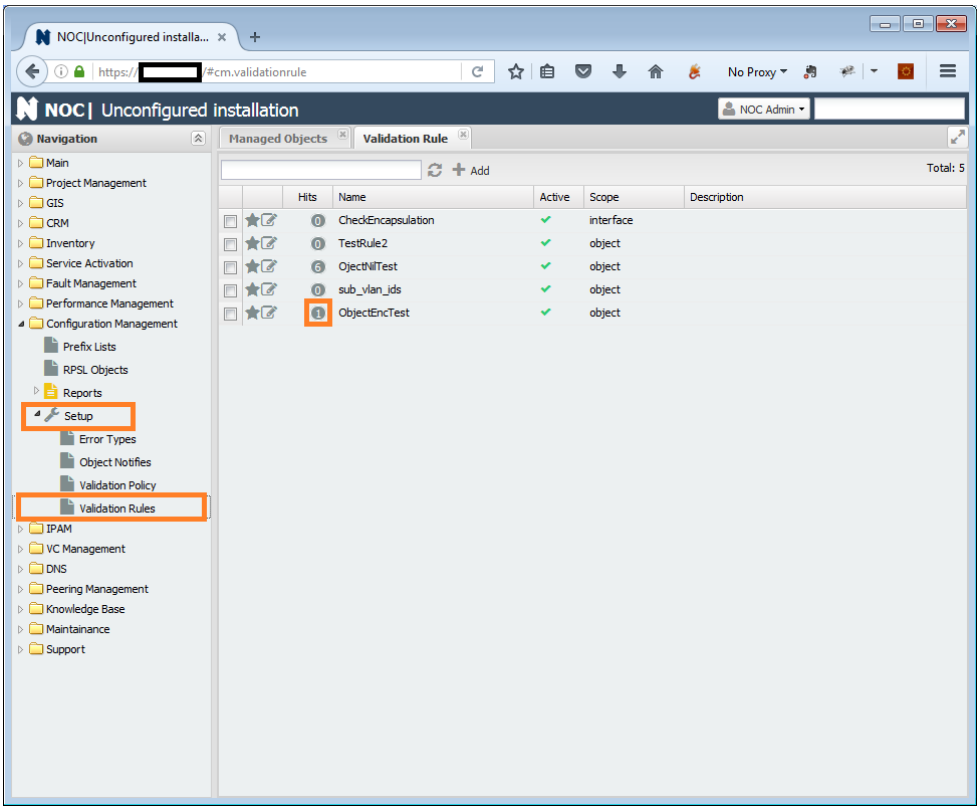


В открывшемся окне можно посмотреть собранные факты. Кнопка Revalidate позволяет запустить поцесс сбора фактов вручную.

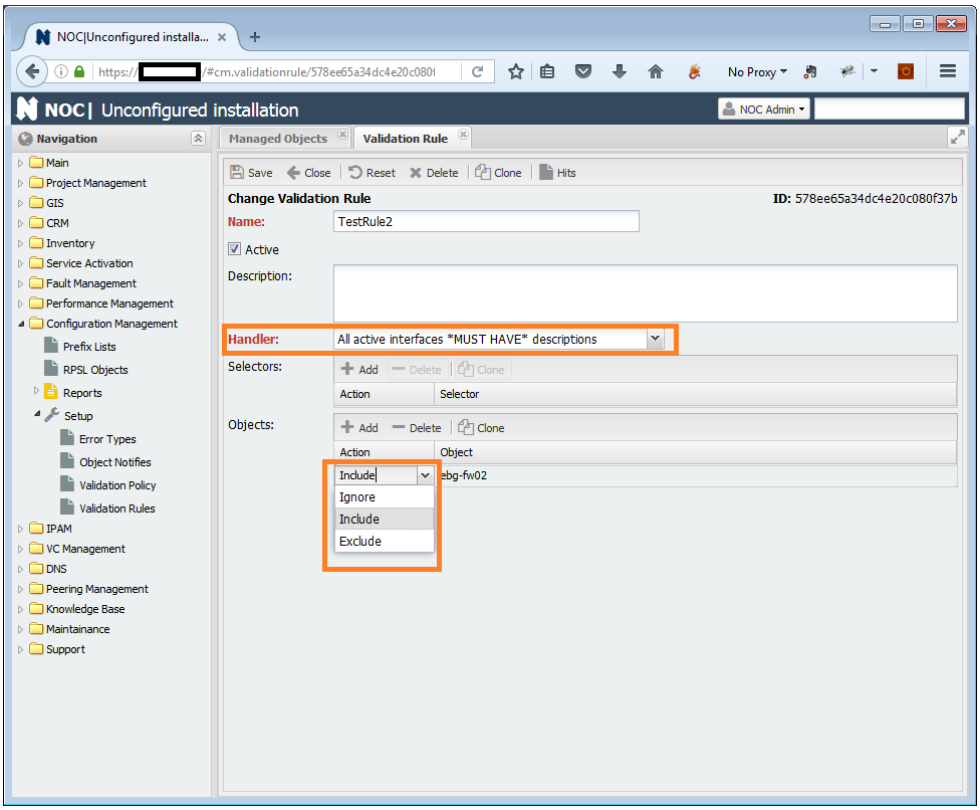
⚠ Если окно facts пустое, это означает что парсер для данного профиля не написан.

Пишем правила

После того как мы просмотрели необходимые факты - приступаем к написанию проверок. Это делается из пункта меню "Configuration Management" -> "Setup" -> "Validation Rules".



В нём представлены написанные правила, цифра в кружочке означает количество совпадений. Добавить новое правило можно кнопкой "Add" в верхней строке.



Структура окна добавления/редактирования правила следующая:

- Поле Name (Имя) - имя, просто имя.
- Выпадающий список Handlers (обработчик) - отвечает за проверку, которую будет производить правило
- Списки Selectors и Objects служат для фильтрации устройств (МО) на которые подействует правило. Доступны 3 действия:
  - Ignore - игнорировать значение в данной строчке (т.е. не проводить проверку на влючение/исключение)
  - Include - при совпадении включить в список на применение правила
  - Exclude - при совпадении исключить из списка на применение правила

Правило не начнёт работать, пока не будет добавлено в политику (Validation Policy), прикреплённую к конкретному устройству.

Проверки (Handlers)

Основной функционал правил реализуется в Handlers (обработчиках). Условно, их можно поделить на 2 вида:

- Python-based
- CLIPS-based

Python-based находятся по пути "<noc_folder>/cm/validators": папки config, interface, object, service. Они представляют собой модули, написанные на Python'е которым передаются объекты, внутри себя они их проверяют и создают ошибку. Если позволяет знание Python'а можно реализовать свои модули-валидаторы.

CLIPS-based проверки основаны на специальном языке экспертной оценке CLIPS. Их можно реализовать как во внешних модулях (аналогично Python-based), так и вставкой прямо в правило (для этого есть специальные Handlers: "Object CLIPS Rules", "Interface CLIPS Rules").

Описание языка CLIPS можно посмотреть: Википедия

С НОКом поставляется набор встроенных правил, которые можно использовать для стандартных задач. Если требуются специализированные проверки - то необходимо осваивать CLIPS. В папке "<noc_folder>/cm/validators" можно посмотреть код встроенных валидаторов.

Приведём пример валидатора, которые проверяет - назначен ли созданному подинтерфейсу VLAN. Для случая Cisco.IOS настройка выглядит так:

interface FastEthernet0/1.200

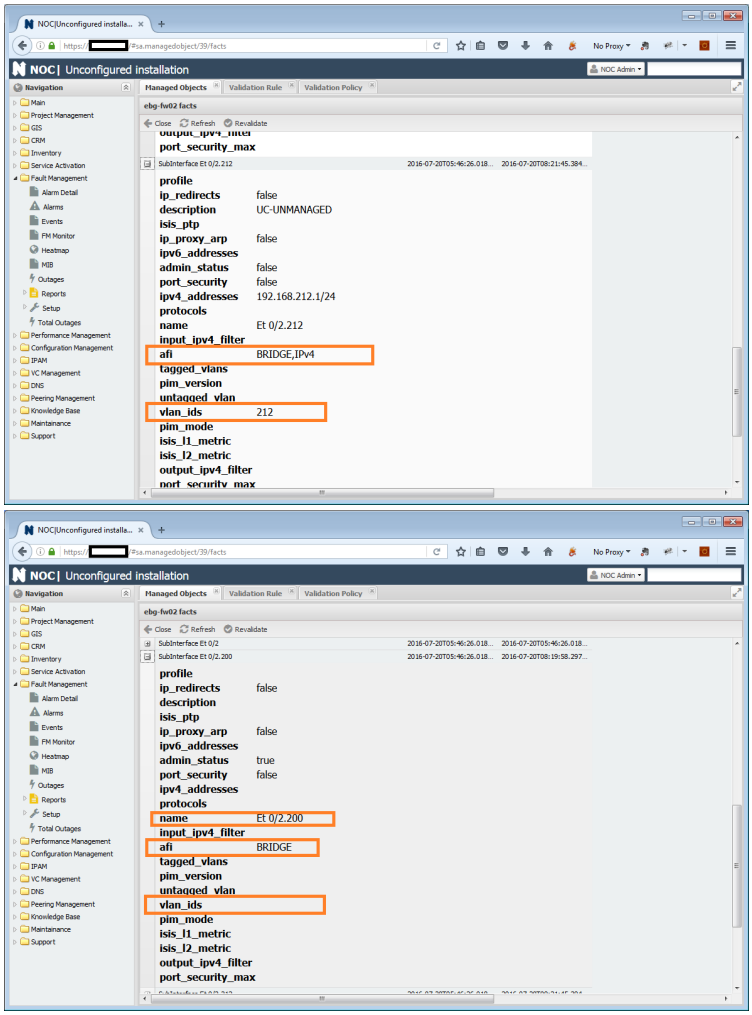
encapsulation dot1q 200

В случае если в конфиге будет отсутствовать команда "encapsulation dot1q 200" должна выдаваться ошибка.

Перефразируя вопрос - в фактах интерфейсов присутствует поля:

- vlan_ids - оно заполняется номером влана, который назначен подинтерфейсу
- afi - в случае подинтерфейса в нём будет присутствовать слово BRIDGED

Необходимо написать правило, которое в случае отсутствия в случае пустого поля vlan_ids и наличия BRIDGED в afi выдавало ошибку. На левой картинке показаны факты с правильно настроенного интерфейса, а на правой - настроенного с ошибкой.



Для данного случае стандартные правила нам не подходят. Напишем своё:

```
(defrule {{RULENAME}}
        (subinterface (vlan_ids) (afi $? "BRIDGE" $?) (name ?n))
        =>
        (assert
            (error (type "Interface | MPLS Without ISIS")
                   (obj ?n)))
)
```

Данное правило написано на CLIPS и выполнит задачу. Сделаем некоторые пояснения.

1. {{RULENAME}} - при выполнении данного правила в логах данное поле заменится строкой из Имени правила. Не возбраняется использовать любую текстовую строку
2. `subinterface -  ,  .  Facts.`

    a. `(vlan_ids) -  vlan_ids`

b. (afi $? "BRIDGE" $?) - afi ( ) "BRIDGE"

c. (name ?n) - name ?n

3. =>

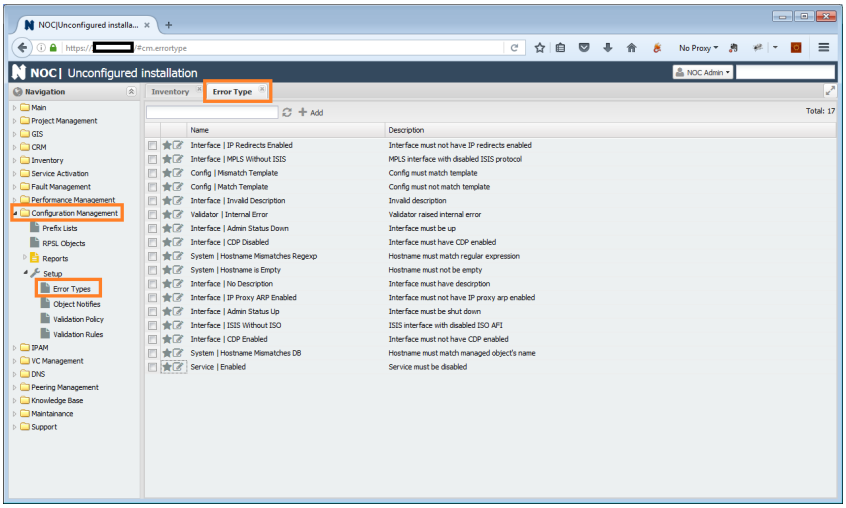
4. assert - ,

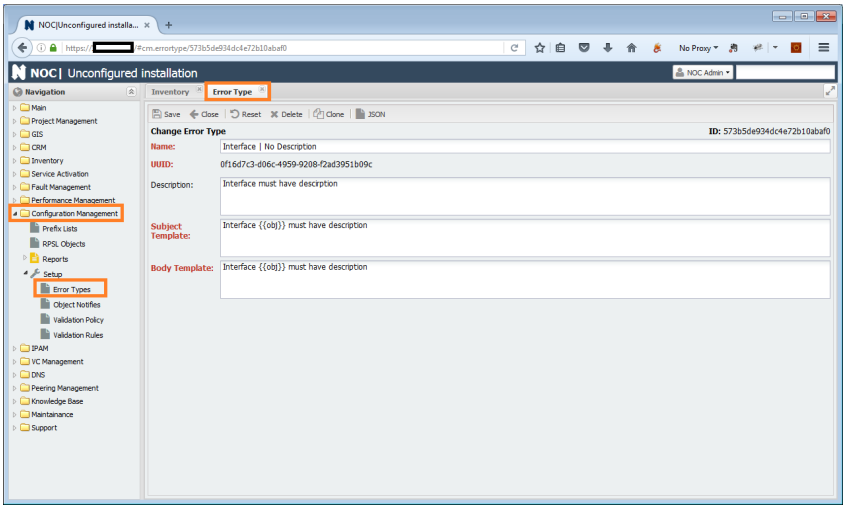
5. error - , . "Configuration Management"->Setup->"Error Types"

6. obj ?n -

Ошибки.

При написании собственных правил возникает вопрос создания собственных типов ошибок. Как вариант, дополнения существующих новой информацией. Типы ошибок доступны в меню "Configuration Management"->Setup->"Error Types".





Для измения доступны поля "Subject Body" и "Body Template". Они применяются при создании аварии, в случае провала проверки.

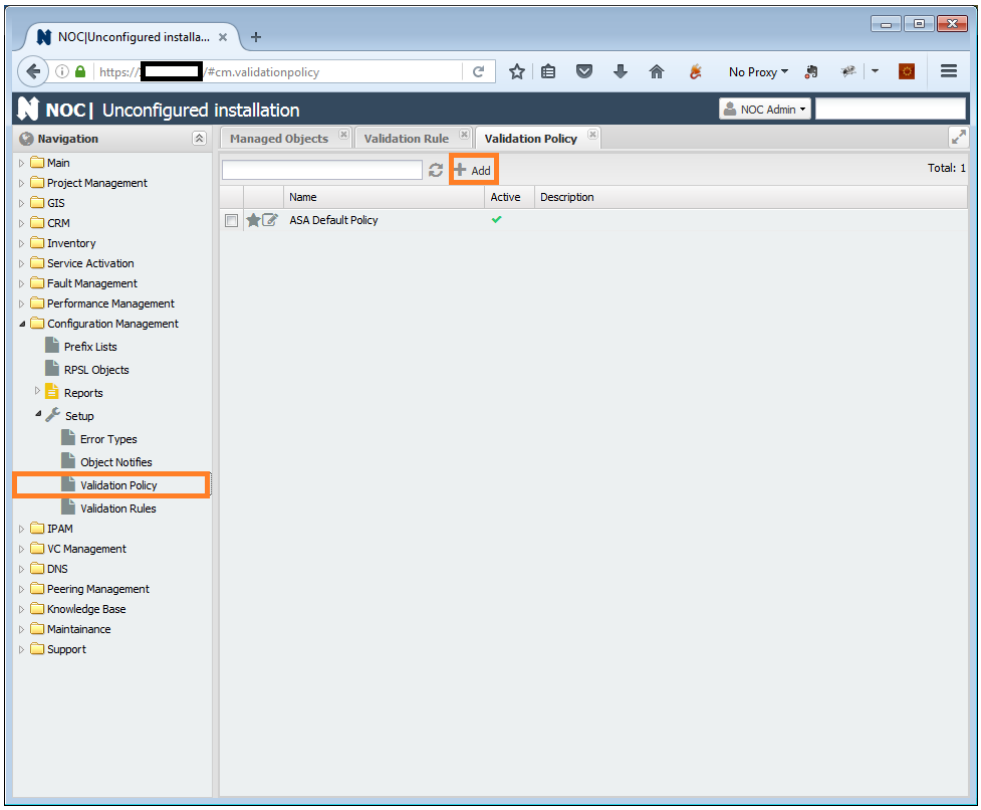
Встроенные обработчики снабжены хорошими комментариями. Поэтому приведём примеры наиболее востребованных:

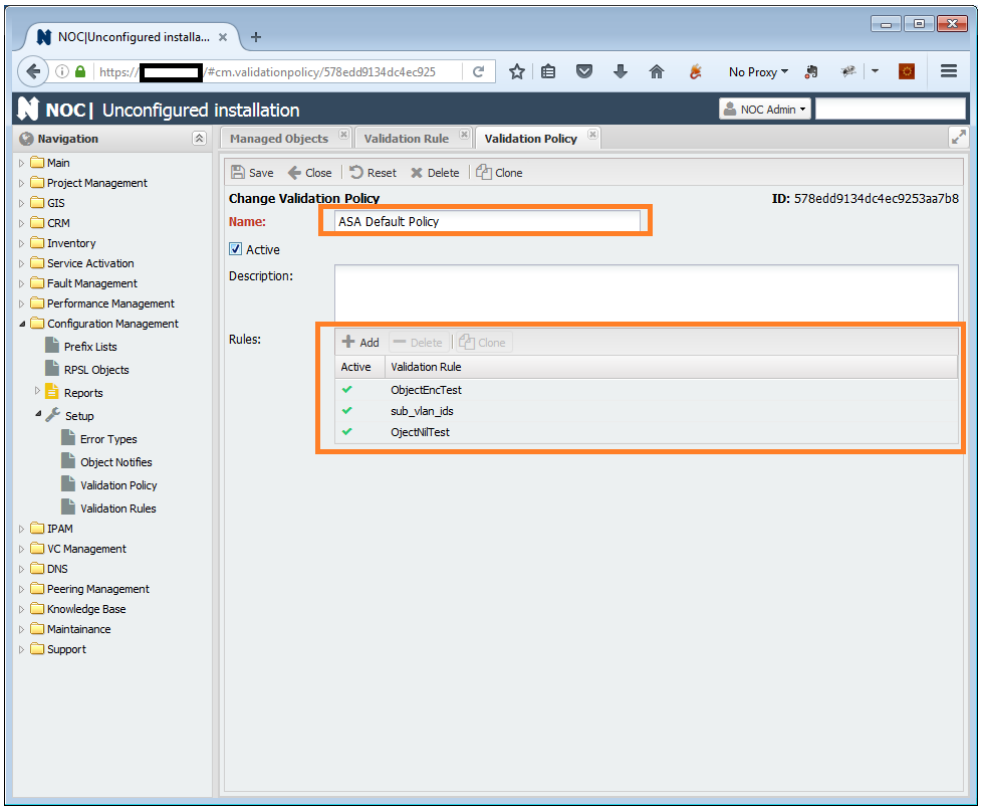
Имя	Описание	Опции	Пример
Config *MUST* match string	В конфигурации должна присутствовать строка.	строка, которую необходимо искать в конфигурации	
Hostname *MUST* match regexp	Проверяет hostname на соответствие регулярному выражению	регулярное выражение в формате Python	
Interface CLIPS Rules	Позволяет задавать свои CLIPS-based правила, применяются к блоку интерфейсов	правило в формате CLIPS	

Object CLIPS Rules	Позволяет задавать свои CLIPS-based правила, прменяются ко всем блокам	правило в формате CLIPS	

Формируем политики.

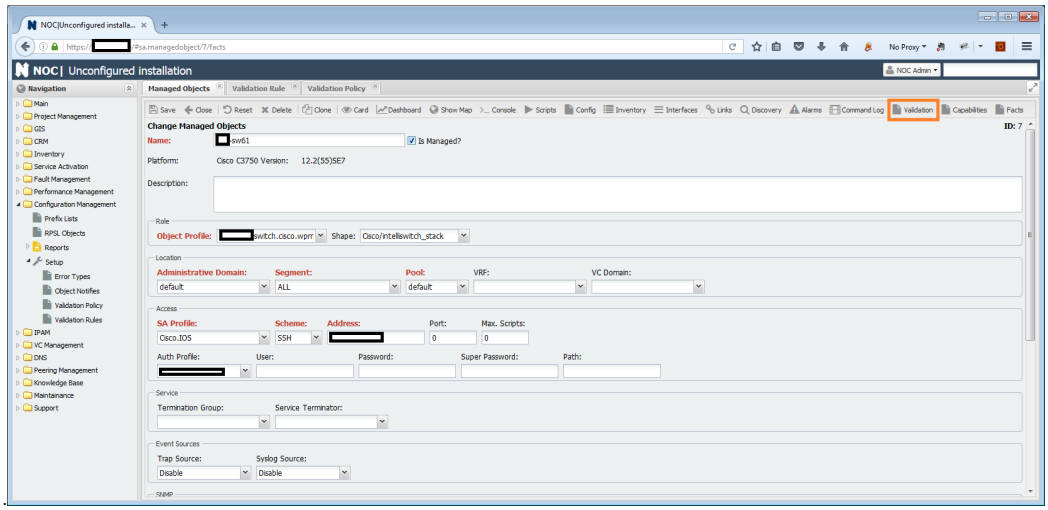
После того как написаны правила - необходимо объединить их в политики и привязать к оборудованию (MO). Политики настраиваются в меню "Configuration Management" -> "Setup" -> "Validation Policy".

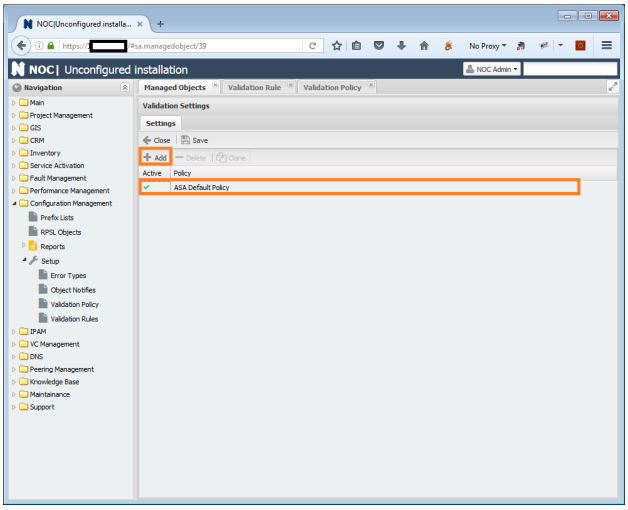




После нажатия кнопки Add, откроется окно добавления политики. В нём необходимо задать имя и список правил, по которым будет проходить проверка. Порядок правил не влияет.

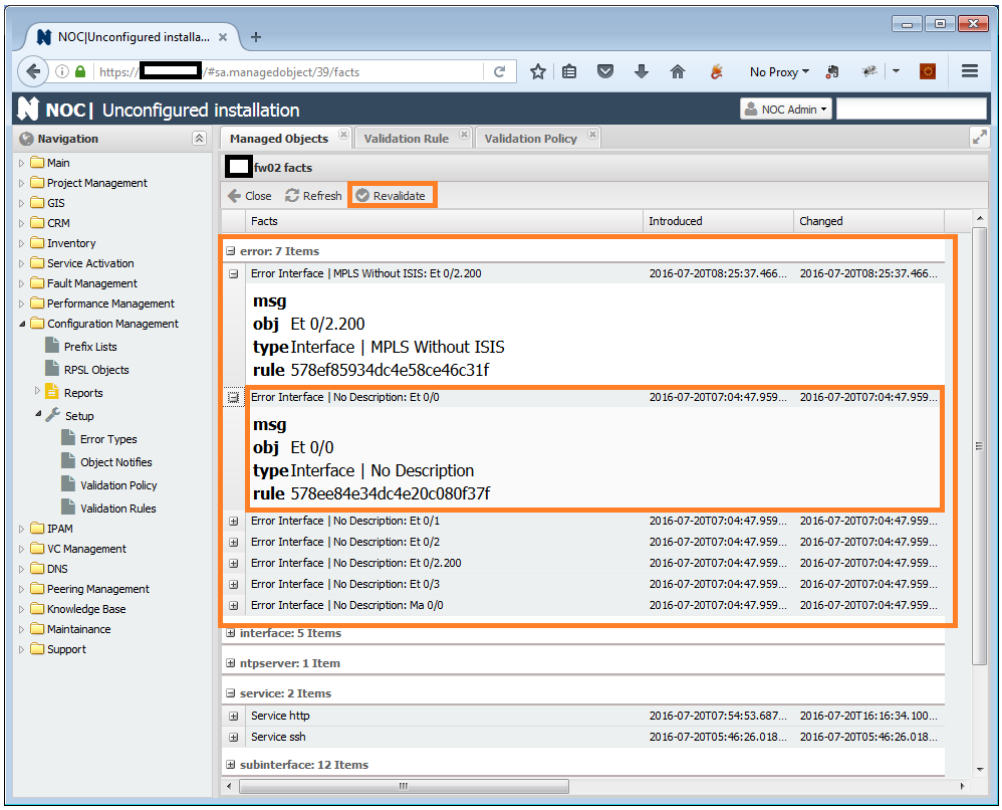
После того как политика валидации (Validation Policy) создана, необходимо привязать её к устройству. Делается это через меню "Service Activation" -> "Management Object" -> Validation. В нём необходимо добавить настроенную политику. Для её применения необходимо или дождаться обновления конфига или воспользоваться кнопкой Revalidate на экране Facts.

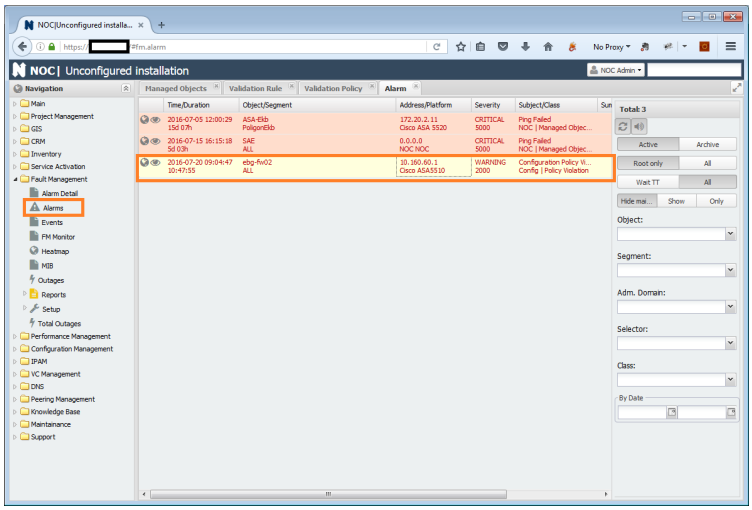




> ⚠ Для немедленного применения политики необходимо воспользоваться кнопкой **Revalidate** на экране **Facts**

Результаты применения политики отобразятся на том же экране Facts. Появится список ошибок. Также на экране правил (Rule) увеличится число в счётчике ошибок рядом с правилом и создастся Alarm.





Важное примечание.

После создания политики и её назначения на устройство НЕ происходит автоматической проверки конфигурации. Проверка пройдёт только после поступления в базу ИЗМЕНЕНИЙ в конфигурации (парсер фактов работает с базой конфигов НОКа).

Данное поведение связано с 2 моментами:

1. Действует принцип "Закон обратной силы не имеет". Т.е. считается, что, изначально, все конфиги соответствуют политике.
2. Мало приятного получить 2 сотни аварий за минуту 🙂.

Если очень хочется:

Если необходимо проверить устройства на политики сейчас

```
./noc shell

from noc.cm.engine import Engine
from noc.sa.models.managedobject import ManagedObject
for mo in ManagedObject.objects.filter(is_managed=True):
    e = Engine(mo)
    e.check()
```

Возможные проблемы

1. Окошко Facts пустое
    a. Для данного профиля не написан парсер. Проверить можно посмотрев содержимое папки "cm/parsers/".
    b. Парсер написан, но не работает. Ошибки необходимо искать в логах web.
2. Правило написано, но не обнаруживает ошибку
    a. Необходимо проверить что правило включено в политику, а политика привязана к устройству.
    b. Необходимо нажать кнопку Revalidate в окошке Facts.
    c. Проверить логи web на наличие ошибок (в нём отображается какие правила были применены и результат)