Les règles de découvertes du scan réseau (discovery-import)

Sommaire

Présentation

Le but des règles de découvertes est de pouvoir qualifier les hôtes à partir des champs collectés depuis nmap. Cette qualification permet :

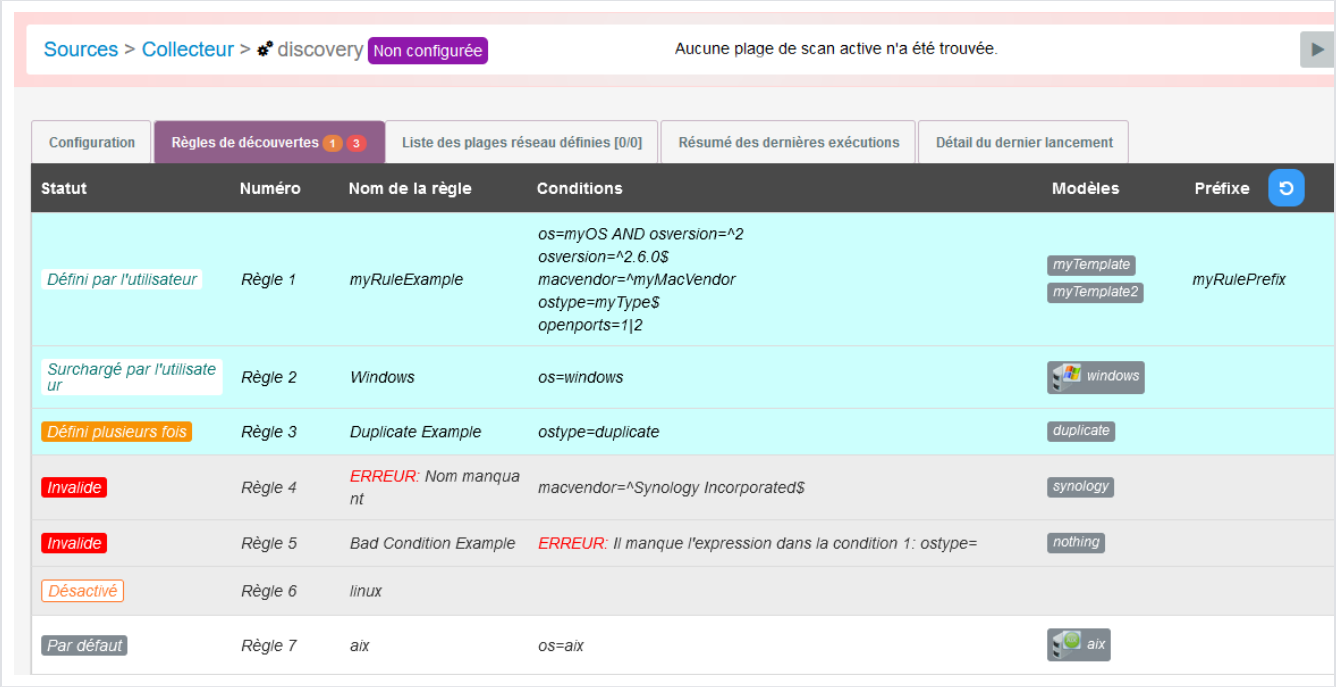
- D'ajouter un modèle en fonction des valeurs des champs collectés depuis nmap :
- D'ajouter un préfixe sur le nom de l'élément

Plusieurs règles peuvent s'appliquer sur un même élément. L'élément peut donc avoir plusieurs préfixes ou plusieurs modèles d'hôtes, en respectant l'ordre des règles appliquées.

Visualiser les règles de découvertes (Onglet "Règles de découvertes")

Les règles sont affichées sous forme de la liste :

- Triée par ordre de priorité. Exemple : Lorsque les règles 1 et 2 s'appliquent, la règle 1 s'applique avant la règle 2 (ajout du préfixe et des modèles d'hôtes)
- La couleur de fond de chaque ligne indique le type de règle :
  - **Blanc** : règle par défaut
  - **Bleu** : règle définie par l'utilisateur
  - **Gris** : règle affichée dans la liste, mais sans effet (*désactivée volontairement, syntaxe incorrecte ...*).



Il existe 6 statuts pour les règles de découvertes :

Cas	Statut
La règle est en un seul exemplaire dans votre fichier	Définie par l'utilisateur
Le nom de la règle existe déjà dans le fichier par défaut	Surchargée par l'utilisateur
Le nom de la règle est défini plusieurs fois dans le fichier utilisateur	Définie plusieurs fois
La règle comporte une erreur (comme une des clés obligatoires)	Invalide
La règle ne comporte ni de conditions, ni de modèles d'hôtes, ni de préfixe	Désactivé

Vous pouvez rafraîchir la liste des règles directement en appuyant sur le bouton de rafraîchissement en haut à droite  , ou en appuyant sur F5.

Définir des règles de découvertes

Le mécanisme de règles permet d'enrichir les équipements détectés.

- Par défaut, une installation fournit une liste de règles prédéfinies.
- Vous pouvez définir vos propres règles ou surcharger les règles prédéfinies.
  Vous devez pour cela éditer le fichier JSON :
  - */etc/shinken-user/configuration/daemons /synchronizers/sources/discovery/discovery_rules.json*
  - Un fichier d'exemple est disponible dans **/etc/shinken-user-example/configuration/daemons /synchronizers/sources/discovery**

Une règle de découverte est séparée en 4 parties:

- **name**: le nom et l'id de votre règle (*doit être unique*)
- **condition[1-9]**: représente une condition qui applique la règle si elle est remplie

(*il suffit qu'une seule condition soit bonne pour que la règle soit appliquée*)
- **prefix_name**: ce préfixe est ajouté au nom des éléments découverts par cette règle (*optionnel*)
- **use** : Les modèles d'hôtes ajoutés en cas d'application de la règle. Vous pouvez en mettre autant que vous voulez en les séparant d'une virgule.
    - Les modèles d'hôtes sont ajoutés à la suite de ceux dé jà présents sur l'hôte (*ajoutés par d'autres règles*)

Exemple de règle

```
{
        "rules":[
                        {
                        "name": "myRuleExample",
                        "condition1": "os=myOS AND
osversion=^2",
                        "condition2":
"osversion=^2.6.0$",
                        "condition3":
"macvendor=^myMacVendor",
                        "condition4":
"ostype=myType$",
                        "condition5":
"openports=1|2",
                        "prefix_name":
"myRulePrefix",
                        "use": "myTemplate,
myTemplate2"
                        }
        ]
}
```

Chemin du fichier de définition des règles d'application

Pour ajouter une règle utilisateur d'application des modèles, il faut éditer le chemin suivant :

```
/etc/shinken-user/configuration/daemons/synchronizers/sources/discovery/discovery_rules.json
```

Chemin de l'Exemple pour la source livrée par défaut :

```
/etc/shinken-user-example/configuration/daemons/synchronizers/sources/discovery
```

Définition d'une règle

Le mécanisme de condition utilise les données collectées par nmap pour modifier l'hôte à proposer au Synchronizer. Les clés du retour nmap sont utilisables pour vos conditions

Les champs collectés par nmap

Les scans réalisés par nmap remontent les clés suivantes :

Clé	Description	Exemple
fqdn	Nom de domaine complètement qualifié	DiskStation
mac	Adresse MAC de l'équipement	00:11:32:9F:09:44
macvendor	Nom du constructeur associé à l'adresse MAC (voir le chapitre suivant pour plus de détails sur la correspondance adresse MAC Constructeur)	Synology Incorporated
openports	Liste des ports identifiés comme ouverts	22,80,137,139,161,161,443,445,548,3261,5000,5001,5353
os	Famille du système d'exploitation détectée, par exemple *Windows*, *Linux*, *IOS* (routeurs Cisco), *Solaris* ou *OpenBSD* . Il y a des centaines d'autres familles de systèmes comme des routeurs, imprimantes ou autres systèmes propriétaires. Lorsque la famille du système d'exploitation ne peut pas être déterminée avec une confiance suffisante, la valeur *embedded* est utilisée.	Linux

ostype	Le type de système d'exploitation est une classification large selon l'usage prévu de ce système comme "*router*", "*printer*", ou "*game console*". Les systèmes d'exploitation universels tels que Linux et Windows, qui ont de nombreux cas d'utilisations sont classés en tant que "*general purpose*".	general purpose
osvendor	L'entreprise ou l'entité qui produisent le système d'exploitation ou équipement (par exemple *Apple*, *Cisco*, *Microsoft*, *Linksys*). Pour les projets communautaires comme Linux ou les différents BSD, la valeur de l'information "*os*" est répétée ici.	Linux
osversion	Version de l'os détectée	3.X
ip	Adresse de l'élément	192.168.1.52

Si nmap ne peux remplir une information, le message "Aucune valeur remontée" sera affiché dans la colonne valeur pour cette clé

Les conditions des règles

Il existe plusieurs possibilités pour les conditions de vos règles :

Commence par (=^...)

Si l'expression commence par '^', la condition signifie que le résultat attendu **doit COMMENCER** par l'expression.

```
macvendor=^myMacVendor
```

Termine par (=...$)

Si l'expression termine par '$', la condition signifie que le résultat attendu **doit TERMINER** par l'expression.

```
ostype=myType$
```

Est égal (=^...$)

Si l'expression commence par '^' ET termine par '$', la condition signifie que le résultat attendu **doit être l'expression EXACTE**.

```
osversion=^2.6.0$
```

Contient (=...)

Si l'expression ne contient aucun des paramètres précédents, la condition signifie que le résultat attendu **doit CONTENIR** l'expression

```
os=myOS
```

Condition_1 ET condition_2 (condition_1 AND condition_2)

Si la condition AND est équivalente à la porte logique AND. Cela signifie que **tout** ce qui est dans cette condition doit être respecté pour que la règle soit appliquée.

```
os=myOS AND osversion=^2
```

Cas spécifique des openports (X|X)

L'écriture d'une condition pour la propriété openports est un cas spécifique.

```
openports=1|2
```

Sur cette propriété, les conditions de type "contient, commence par ou termine par" ne peuvent pas être utilisées.

- Il faut donc rentrer le port exact.
- La présence des caractères '^' et '$' sera donc considérée comme une erreur.

Pour faire un OU logique, il faut mettre un '|' entre chaque port.
Exemple: 80|8080

Liste des règles par défaut

Lors de l'installation, Shinken livre un certain nombre de règles par défaut pour la détection des objets via le collecteur discovery.

Ces règles par défaut sont les suivantes :

Règle	Condition	Modèle d'hôte appliqué
aix	`os=aix`	aix
cisco	`os=cisco`	cisco
dns	`openports=53`	dns
ftp	`openports=21`	ftp
HPAsm	`macvendor=hewlett packard AND openports=2301`	hp-asm
HPBladeChassis	`os=embedded AND ostype=remote management AND osvendor=hp`	hp-blade-chassis
HPPrinterState	`openports=631 AND openports=9100`	printer-hp
HpUx	`os=hp-ux`	hpux
Http	`openports=80`	http
Https	`openports=443`	https
Imap	`openports=143`	imap
Imaps	`openports=993`	imaps
Ldap	`openports=389`	ldap
Ldaps	`openports=636`	ldaps
linux	`os=linux`	linux
mongodb	`openports=27017`	mongodb
mssql	`openports=1433`	mssql
mysql	`openports=3306`	mysql
Oracle	`openports=1521|1526`	oracle
pop3	`openports=110`	pop3
pop3s	`openports=995`	pop3s
smtp	`openports=25`	smtp
smtps	`openports=465`	smtps
ssh	`openports=22`	ssh
Shinken-synchronizer	`openports=7765|7766`	shinken-synchronizer
Shinken-broker	`openports=7767|7772`	shinken-broker
Shinken-scheduler	openports=7768	shinken-scheduler
Shinken-reactionner	openports=7769	shinken-reactionner
Shinken-arbiter	openports=7770	shinken-arbiter

Shinken-poller	openports=7771	shinken-poller
Shinken-receiver	openports=7773	shinken-receiver
switch	`ostype=switch`	switch
ESX	`isesxhost=1`	esx
VMware-VM	`isesxvm=1`	vmware-vm
Windows	`os=windows`	windows
Windows 2000	`os=windows AND osversion=2000`	windows2000
Windows 2003	`os=windows AND osversion=2003`	windows2003
Windows 2008	`os=windows AND osversion=vista`	windows2008
Windows 2008r2	`os=windows AND osversion=7`	windows2008,windows2008r2
Windows 2012	`os=windows AND osversion=2012`	windows2012
Windows 2016	`os=windows AND osversion=2016`	windows2016

Le fichier des règles par défaut est le suivant : discovery_rules.json