

Editer un check

Sommaire

- Onglet Général
- Onglet Données
- Onglet Supervision
 - Vérification du statut de l'élément
 - Actif
 - Passif
 - Duplicate for Each
 - Données stockées
 - Métrologie
 - SLA
- Onglet Notifications
- Onglet Expert
 - Paramétrage du Flapping
 - Modulations
 - Gestionnaire d'événements
 - Autre

Onglet Général

Staging > Check appliqué à l'hôte

Check appliqué à l'hôte >

Général *
Données [0]
Supervision *
Notifications
Expert

Propriété	Valeur	Venant des modèles
Nom du Check *		
Modèle de Check hérité	-- Par défaut [Aucun] --	
Attaché sur les groupes d'hôtes		
Attaché sur les hôtes	-- Par défaut [Aucun] --	
Impact métier	☒ Par défaut (Même comportement que son hôte)	
URL externe		
Liste des URL externes	+ Créer une Url Format fichier Cfg	
Activé	☒ Vrai ☐ Faux Par défaut [Vrai]	
Pack		

Nom du Check

- *Clé d'import*: service_description

 Les caractères ~!\$%^&*"'|<>?,()=/+ sont interdits dans ce champ.

— Un caractère non autorisé provoquera un compteur erreur

 Le nom doit être unique pour les modèles de checks.

  Ce champ ne peut pas rester vide

Modèles de Check hérité

- **Clé d'import:** use
Les modèles de checks qui sont attachés au check pour hériter de leurs propriétés et leurs données.
 - L'ordre des modèles est important.
 - Une valeur définie dans deux modèles prendra la valeur du modèle le plus proche du début de la liste.
 - Les valeurs définies pour une propriété ou une donnée, seront prioritaire sur celles héritées de modèles.

Utilisation d'expression logique

Il est possible d'utiliser des expressions logiques afin de mieux maîtriser l'attachement d'un check. Les propriétés qui acceptent les expressions logiques sont :

- **Attaché sur les groupes d'hôtes** pour les *Check appliqué à l'hôte* ou *Check appliqué au modèle d'hôte*.
- **Attaché sur les modèles d'hôte** pour un *Check appliqué au modèle d'hôte*.
- **Attaché sur les modèles de cluster** pour un *Check appliqué au modèle de cluster*.

Liste des opérateurs utilisable dans une expression logique dans leur ordre d'évaluation :

1. **!** : négation
2. **& ou +** : et logique
3. **| ou ,** : ou logique

Exemples d'utilisation d'expression logique :

(Les exemples sont donnés dans le cas de check appliqué à un modèle d'hôte, mais le comportement des expressions complexes est identique pour un check appliqué sur les groupes d'hôtes ou un check appliqué sur les modèles de cluster.)

```
linux!windows
```

Le check s'applique sur tous les hôtes héritant du modèle linux mais pas du modèle windows

```
pop3 & ( smtp | smtps )
```

Le check s'applique sur tous les hôtes héritant des modèles smtp et pop3, ou les modèles smtps et pop3



Une expression logique doit contenir au moins un modèle d'hôte sur lequel accrocher le check. L'expression suivante sera refusée :

```
!esx
```

Attaché sur les groupes d'hôtes

- **Clé d'import:** `hostgroup_name`
Cette propriété est utilisée pour attacher ce check sur un ou plusieurs groupes d'hôtes. Utilisez la virgule (,) pour spécifier plusieurs groupes d'hôtes. Tous les hôtes appartenant à un des groupes aura le check accroché. Il est possible d'utiliser des expressions logiques.

Attaché sur les hôtes (pour les Checks appliqués à l'hôte) ou Attaché sur les clusters

- **Clé d'import:** `host_name`
Cette propriété est utilisée pour attacher ce check sur un ou plusieurs hôtes (ou clusters). Utilisez le menu déroulant pour spécifier un ou plusieurs hôtes (ou clusters).

Attaché sur les modèles d'hôtes (pour les Checks appliqués aux modèles d'hôtes) ou Attaché sur les modèles de clusters

- **Clé d'import:** `host_name`
 Cette propriété est utilisée pour attacher ce check sur un ou plusieurs modèles d'hôtes (*ou clusters*). Utiliser les virgules pour spécifier plusieurs modèles. Tous les hôtes (*ou clusters*) héritant d'un des modèles spécifiés auront le check accroché.

La page [Logique des modèles](#) décrit l'utilité des modèles d'hôtes (*ou clusters*) et décrit l'utilisation de cette propriété.

Cliquer sur le champ (1) pour faire apparaître le menu déroulant (4) afin de sélectionner un ou plusieurs modèles d'hôtes (*ou clusters*). L'affichage du menu déroulant (4) s'adapte à la saisie en cours (1).

L'entête du menu déroulant (2) affiche le filtre actif (*la saisie en cours*) mais également le nombre d'éléments filtrés par la saisie sur le nombre d'éléments total (3).

Il est possible d'utiliser des expressions logiques.

Impact métier

- **Clé d'import:** `business_impact`
 Cette propriété permet de définir l'importance que l'on donne à un check pour le business. (*valeur numérique entre 0 et 5 dans les fichiers d'imports, mais affichée dans l'interface avec des étoiles ou des dollars suivant le niveau*) :
 - du moins important (0 => *)
 - (1 => **)
 - (2 => ***)
 - (3 => \$\$\$\$)
 - (4 => \$\$\$\$\$)
 - au plus important (5 => \$\$\$\$\$\$)
 La valeur par défaut est 2 (***).

URL externe

- **Clé d'import :** `notes_url`
 Cette propriété est utilisée pour définir une URL qui peut être utilisée pour fournir plus d'information sur ce check. N'importe quelle URL valide peut être mise, commençant par `http://` ou `https://`

Cette URL sera disponible dans l'interface de visualisation dans le volet détail présentant le check.

Cela peut être pratique pour faire une description écrite détaillée du check, une procédure de reprise sur panne... qui sera visible pour les autres membres de l'équipe.

Une URL qui ne commence pas par `http://` ou `https://` provoquera un compteur d'erreur et il sera impossible d'enregistrer le check

Les caractères " " et les espaces sont interdits dans ce champ .

Un caractère non autorisé provoquera un compteur d'erreur et il sera impossible d'enregistrer le check

Liste des URL externes

- **Clé d'import :** `notes_multi_url`
 Cette propriété est utilisée pour définir plusieurs URL qui peuvent être utilisées pour fournir plus d'informations sur ce check.

(1) Nom de l'URL (*label pour l'interface de visualisation*), Le nom à afficher est optionnel (*mais son absence montrera toujours un warning dans l'interface*)

Les caractères `<>` et `'` sont interdits dans ce champ .

Un caractère non autorisé provoquera un compteur d'erreur et il sera impossible d'enregistrer le check

(2) Adresse de l'URL, N'importe quelle URL valide peut être utilisée, commençant par `http://` or `https://`

Les caractères " " et les espaces sont interdits dans ce champ .

Un caractère non autorisé provoquera un compteur d'erreur et il sera impossible d'enregistrer le check

- (3) Icône pour l'UI de visualisation
- (4) Type d'affichage dans l'interface de Visualisation
- Nouvel onglet,
 - Pop-up de différente taille (30%, 40%, 50%, 60%, 70%, 80% de la taille de l'écran)
 - Caché
- (5) Valeur à copier-coller dans un fichier .cfg pour retrouver cette propriété dans l'état
- (6) Passer la souris en survol du champ format fichier .cfg pour faire apparaître les boutons de copie du format (*gauche*) et d'édition du format (*droite*).
- Le bouton d'édition permet d'éditer la propriété URL externe soit en collant un format précédemment copié ou bien en écrivant le format.

Liste des URL externes

Label	URL	Actions
Documentation	http://doc.shinken-solutions.com	Nouvel onglet
Bug track	http://bug-tracker.com	Pop-up 80%

Format fichier Cfg

```
notes_multi_url
Documentation~=http://doc.shinken-solutions.com~=documentation~=OPEN_IN_NEW_TAB~#~Bug track~=http://bug-tracker.com~
=bug~=POPUP(80)
```

Une URL externe peut contenir le mot clé ##USER## qui sera remplacé par le nom de l'utilisateur courant dans l'interface de Visualisation

Important

Dans cette propriété, des données (*macro*) peuvent être utilisées dans l'URL.

- La référence à ces données se fait via le principe décrit ici : [Les Variables \(Remplacement dynamique de contenu - Anciennement les Macros \)](#).
- Pour intégrer une donnée dans l'URL, il suffit d'ajouter \$ _SERVICEDATA1\$

Les données (*macros*) qui peuvent être utilisées sont les suivantes :

- Données globales
- Données accrochées à l'hôte (*ou cluster*) présentes dans l'onglet Données
- et les propriétés (*données statiques*) suivantes :
 - SERVICEUUID
 - SERVICEDESC
 - SERVICEDISPLAYNAME

Activé

- Clé d'import : enabled

Permettre que cet élément soit pris en compte lors du passage de **Staging** en **Production**.

Pack

- Clé d'import : pack

Permet de donner un nom commun à un ensemble d'éléments de Shinken. Il sera ainsi possible de filtrer par ce nom commun dans les listes des éléments pour retrouver l'ensemble des éléments.

Cette propriété n'est pas héritable.

Les caractères < > & " ' / sont interdits dans ce champ.

Un caractère non autorisé provoquera un compteur d'erreur et il sera impossible d'enregistrer le check

Onglet Données

Cet onglet définit des données qui pourront être utilisées par des variables (voir la page [Les Variables \(Remplacement dynamique de contenu - Anciennement les Macros \)](#)), notamment à l'utilisation de la commande (voir la page [Les commandes](#)) de check. Consultez ces pages pour plus d'info



Après avoir donné un nom de donnée protégée à l'une des données, il n'est plus possible de modifier ce nom par la suite.

- Il faudra le supprimer puis le recréer pour le renommer.

Cette modification est interdite afin d'éviter qu'une donnée protégée devienne visible, en changeant son nom.

La valeur de la données pouvant être longue, il est possible d'agrandir le champ de la valeur afin d'améliorer la lisibilité du champs.

Pour agrandir le champ de la valeur, il faut maintenir le clic sur l'icône  située en bas à droite du champ et réajuster verticalement .



Onglet Supervision

Staging > Check appliqué à l'hôte

Check appliqué à l'hôte >

Général

Données [0]

Supervision

Notifications

Expert

Propriété	Valeur	Venant des modèles
Période de maintenance planifiée	-- Par défaut [Même comportement que son hôte] --	

Vérification du statut de l'élément (ACTIF et PASSIF peuvent être combiné)

Actif (Les commandes de vérifications sont ordonnancées et lancées par Shinken)

Actif activé

?

Vrai

Faux

Par défaut [Vrai]

Commande de vérification *

?

-- Par défaut [Aucun] --

Tag de Poller

?

-- Par défaut [Même comportement que son hôte] --

Période de vérification

?

-- Par défaut [Même comportement que son hôte] --

Nb maximum de tentatives de confirmation du statut du check. *

?

Intervalle entre les vérifications (minutes) *

?

Intervalle de nouvelles tentatives de confirmations d'état (minutes) *

?

Temps maximum d'exécution d'un check (secondes)

?

Par défaut [Même comportement que son hôte]

Seuil d'alerte de l'utilisation CPU (secondes)

?

Par défaut [Même comportement que son hôte]

Passif (Shinken accepte les états reçus depuis des outils externes pour cet élément)

Passif activé

?

Vrai

Faux

Par défaut [Vrai]

Vérification que l'état reçu des outils externes ne soit pas expiré

?

Vrai

Faux

Par défaut [Faux]

Seuil d'expiration des états reçus des outils externes (secondes)

?

Données stockées

Métrologie

Lire et stocker les métriques

? ☐ Vrai ☐ Faux ☒ Par défaut [Vrai]

SLA

Seuil d'avertissement

? Même comportement % ☒ Par défaut [Même comportement que son hôte]

Seuil critique

? Même comportement % ☒ Par défaut [Même comportement que son hôte]

Période de maintenance planifiée

- **Clé d'import :** maintenance_period

Cette propriété permet de définir une période de temps pendant laquelle le check sera en maintenance.

En maintenance, les checks de l'hôte (*ou cluster*) seront toujours exécutés (*donc il n'y aura pas de manques dans les métriques ou l'historique de l'hôte*), mais les notifications et les commandes du gestionnaire d'événement ne seront pas lancées.

Vérification du statut de l'élément

Actif

Actif activé

- **Clé d'import :** active_checks_enabled

Cette propriété permet de définir si Shinken ordonnance et lance la commande de vérification pour déterminer l'état de ce check.

Ses valeurs peuvent être:

- True: Shinken va ordonnancer et lancer la commande de vérification
- False: Shinken ne va pas ordonnancer et lancer la commande de vérification

Commande de vérification

- **Clé d'import :** check_command

Cette propriété permet de spécifier la commande qui est employée pour vérifier si le check est dans un statut en OK ou CRITIQUE.

Si la propriété est vide, la commande de contrôle par défaut sera employée.

Pour le passage de plusieurs arguments, le séparateur est !

Exemple : mavaleur1!mavaleur2

Pour Duplicate For Each : \$VALUE1!\$VALUE2\$



★ Ce champ ne peut pas rester vide

Tag de poller

- **Clé d'import :** poller_tag

Cette propriété permet de définir le poller_tag du check.

Le check ne sera exécuté que par les Pollers ayant cette valeur dans la liste des Tags définie dans leur paramètre poller_tags.

Par défaut, la valeur de Tag de Poller est "même comportement que son parent (*hôte ou cluster*)".

Si la valeur est "non tagué" :

- Tous les Pollers n'ayant aucun poller_tag seront utilisés pour l'exécution
- Ou tous les Pollers spécifiant qu'ils prennent aussi les checks non tagués

Période de vérification

- **Clé d'import :** check_period

Cette propriété permet de choisir la période de supervision où sont faites les vérifications de l'état du check par Shinken.

Nombre maximum de tentatives de confirmation du statut du check

- *Clé d'import* : `max_check_attempts`

Cette propriété permet de définir le nombre de tentatives où Shinken recommencera la commande de vérification d'un check si son résultat est autre chose que le statut OK.

Mettre cette valeur à 1 impliquera que Shinken générera une alerte dès qu'il n'est plus en état OK, sans réessayer.

Pour que Shinken ne fasse plus de vérification de statut de ce check, laissez cette propriété à 1, mais la commande de vérification doit être vide.



— Une valeur non numérique provoquera un compteur erreur.

Intervalle entre les vérifications (*minutes*)

- *Clé d'import* : `check_interval`

Cette propriété permet de définir le temps entre deux vérifications de l'état du check.



La limite est fixée à 2630880 (*soit cinq ans*).



— Une valeur non numérique provoquera un compteur erreur.

Intervalle de nouvelles tentatives de vérification d'état (*minutes*)

- *Clé d'import* : `retry_interval`

Cette propriété permet de définir le temps d'attente avant de relancer la commande de vérification du check après un premier état différent de OK.

Ce nombre est en minutes.

Une fois que le nombre de relances de commandes de vérifications a atteint la valeur **max_check_attempts**, le dernier état récupéré est validé et la cadence des vérifications revient à la valeur normale, définie dans la propriété **Intervalle entre les vérifications**.



La limite est fixée à 2630880 (*soit cinq ans*).



— Une valeur non numérique provoquera un compteur erreur.

Temps maximum d'exécution d'un check (*secondes*)

- *Clé d'import* : `check_running_timeout`

Cette propriété permet de définir la valeur **en secondes** du temps maximum permis pour le lancement d'une commande de supervision. Passé ce délai, la commande sera terminée. Si le check ou la commande définit également ce paramètre, leur valeur sera prioritaire par rapport à celle de l'hôte (*ou cluster*).

- Aucune valeur ou une valeur a -1, indique qu'il n'y a pas de Temps maximum défini.

Si ni l'hôte (*ou clusters*), ni le check, ni la commande ne définissent ce paramètre, alors la valeur définie globalement pour Shinken (*via le fichier shinken.cfg et le paramètre check_running_timeout*) sera utilisée à la place.



La limite est fixée à 2630880 (*soit cinq ans*).



— Une valeur non numérique provoquera un compteur erreur.

Seuil d'alerte de l'utilisation CPU (*secondes*)

- *Clé d'import* : `warning_threshold_cpu_usage`

Nombre de secondes que peut consommer une commande avant d'apparaître en Warning dans le check 'Scheduler - Performance'.



La valeur sera récupérée dans l'ordre, par la commande, par l'élément Check, par l'élément Hôte (*ou clusters*), ou par la valeur par défaut (*shinken.cfg:warning_threshold_cpu_usage*)

  La limite est fixée à 2630880 (*soit 5 ans*).

  Une valeur non numérique provoquera un compteur erreur.

Passif

Passif activé

- *Clé d'import* : `passive_checks_enabled`
Cette propriété permet de définir si Shinken accepte les résultats de commandes de checks depuis des outils externes pour cet hôte (*ou clusters*)
- Les résultats externes seront reçu via le démon Receiver de Shinken.
Les valeurs possibles sont:
 - True: accepte les résultats de commandes de checks externes
 - False: n'accepte pas ces résultats

Vérification que l'état reçu des outils externes ne soit pas expiré

- *Clé d'import* : `check_freshness`
Cette propriété permet de définir si Shinken doit vérifier que la date de réception du dernier état reçu des outils externes ne soit pas expirée (*trop vieille*). Si la date est plus antérieure à la période d'expiration, alors Shinken va forcer l'exécution de la commande de check.
Ses valeurs peuvent être:
 - Activé: Shinken va vérifier l'expiration
 - Désactivé: Shinken ne vérifiera pas l'expiration

Seuil d'expiration des états reçus des outils externes (*secondes*)

- *Clé d'import* : `freshness_threshold`
Cette propriété permet de définir quand Shinken doit forcer la commande pour check si le dernier état reçu des outils externes est antérieur à ce seuil.
Cette valeur est en secondes.

  La limite est fixée à 2630880 (*soit 5 ans*).

  Une valeur non numérique provoquera un compteur erreur.

Duplicate for Each

Duplicate for Each

Nom de la donnée sur l'hôte qui
contiendra les valeurs utilisées pour la
duplication des checks



Valeur par défaut des arguments des
valeurs utilisées pour la duplication des
checks



Nom de la donnée sur l'hôte (*ou cluster*) qui contiendra les valeurs utilisées pour la duplication des checks

- *Clé d'import* : `duplicate_foreach`
Cette propriété est utilisée pour générer plusieurs checks dans l'Arbiter avec une seule définition de check dans le Synchronizer. Shinken comprend cette propriété comme : Crée un check pour chaque valeur de cette propriété.

Le nom du check doit contenir un \$KEY\$. En créant un check pour chaque valeur, Shinken remplacera le \$KEY\$ par cette Valeur. \$KEY\$ est aussi utilisable dans la check command si nécessaire, pour son exécution. Des paramètres peuvent être fournis aux checks (*appelé \$VALUE\$ si il y en a qu'un ou \$VALUEx\$, avec x valant de 1 à n*).





— Une valeur non numérique provoquera un compteur erreur.

Valeur par défaut des arguments des valeurs utilisées pour la duplication des checks

- *Clé d'import* : default_value
Il est possible de fournir plusieurs arguments pour chaque valeur définie dans la donnée de l'hôte (*ou cluster*) utilisé par la fonctionnalité Duplicate for Each.
- Mais si la liste des arguments devient grande et plusieurs fois la même, il est inutile de les répéter.

Prenons le cas d'une base de données avec un port par défaut et une connexion sécurisée en ssl:
 - La donnée DATABASE contient DB1, DB2, DB3
 - Si on définit les arguments sur chaque base, on aurait dans la donnée DATABASE : DB1\$(1521)\$(ssl)\$,DB2\$(1521)\$(ssl)\$,DB3\$(1521)\$(ssl)\$
 - Définir un port particulier sur DB2 donnerait DB1\$(1521)\$(ssl)\$,DB2\$(20000)\$(ssl)\$,DB3\$(1521)\$(ssl)\$
 - Mais plus simplement, il est possible de définir dans cette propriété les valeurs par défaut des arguments **\$(1521)\$(ssl)\$**, alors la propriété DATABASE sur l'hôte (*ou cluster*) pourra avoir cette forme simplifiée DB1,DB2\$(20000)\$(ssl)\$,DB3



— Un format non valide provoquera un compteur erreur

Données stockées

Données stockées

Métrologie

Lire et stocker les métriques

?

Vrai

Faux

Par défaut [Vrai]

SLA

Seuil d'avertissement

?

Même comportement%

Par défaut [Même comportement que son hôte]

Seuil critique

?

Même comportement%

Par défaut [Même comportement que son hôte]

Métrologie

Lire et stocker les métriques

- *Clé d'import* : process_perf_data

Cette propriété permet de définir si Shinken doit lire et stocker les métriques issues de la commande de vérification du check.
Les valeurs possibles sont:
 - Vrai: Shinken va lire et stocker les métriques issues de la commande de vérification
 - Faux: Shinken ne va pas lire et stocker les métriques issues de la commande de vérification

SLA

Seuil d'avertissement

- *Clé d'import* : sla_warning_threshold

Cette propriété permet de définir un seuil d'avertissement pour le module SLA.
La valeur doit être un nombre décimal entre 0 (*inclus*) et 100 (*inclus*) avec jusqu'à 3 chiffres après la virgule.

Seuil critique

- *Clé d'import* : sla_critical_threshold

Cette propriété permet de définir un seuil critique pour le module SLA.
La valeur doit être un nombre décimal entre 0 (*inclus*) et 100 (*inclus*) avec jusqu'à 3 chiffres après la virgule.

Onglet Notifications

Staging > Check appliqué à l'hôte

Check appliqué à l'hôte >

Général *	Propriété	Valeur		Venant des modèles
Données [0]	Notifications activées	?	☐ Vrai ☐ Faux ☒ Par défaut [Même comportement que son hôte]	
Supervision *	Utilisateurs	?	☒ -- Par défaut [Même comportement que son hôte] --	
Notifications	Groupes d'utilisateurs	?	☒ -- Par défaut [Même comportement que son hôte] --	
Expert	Intervalle de temps entre les notifications (minutes)	?	Par défaut [Même comportement que son hôte]	
	Période de temps de notification	?	☐ -- Par défaut [Même comportement que son hôte] --	
	Options de notification	?	Par défaut [Même comportement que son hôte]	
	Décalage de la première notification (minutes)	?	Par défaut [Même comportement que son hôte]	
	Escalades	?	☒ -- Par défaut [Même comportement que son hôte] --	

Notifications activées

- Clé d'import :** notifications_enabled
 Cette propriété permet de définir si Shinken va envoyer des notifications aux utilisateurs listés comme cibles (*Utilisateurs ou Groupes d'utilisateurs*) sur ce check.
 Les notifications sont envoyées:
 - Premièrement après la fin de la période de nouvelle tentative, une fois que l'état CRITICAL est confirmé.
 - Ensuite après chaque **intervalle de notification**, exprimé en minutes
 - Quand le check redevient OK

Les valeurs possibles sont:

- Vrai: Shinken va envoyer les notifications
- Faux: Shinken n'enverra pas les notifications

Utilisateurs

- Clé d'import :** contacts
 Cette propriété est utilisée pour définir quels sont les utilisateurs à notifier.

Groupes d'utilisateurs

- Clé d'import :** contacts_groups
 Cette propriété est utilisée pour définir quels sont les groupes d'utilisateurs à notifier.

Intervalle de temps entre les notifications (minutes)

- Clé d'import :** notification_interval
 Cette propriété permet de définir les intervalles de temps entre les notifications.
 Cette valeur est en minutes.

Note: la valeur 0 sur cette propriété signifie que seule la première notification sera envoyée.

La limite est fixée à 2630880 (soit 5 ans).

Une valeur non numérique provoquera un compteur erreur.

Période de temps de notification

- *Clé d'import* : notification_period
Cette propriété permet de définir la période de temps durant laquelle les notifications sont autorisées. En dehors de cette période, aucune notification ne sera envoyée.

Par défaut, il n'y a pas de période de temps, et donc les notifications ne seront jamais bloquées.

Options de notification

- *Clé d'import* : notification_options
Cette propriété permet de définir quels états et contextes du check peuvent envoyer des notifications. La valeur est une liste d'options séparées par des virgules:
 - c: envoie une notification sur un état CRITIQUE
 - w: envoie une notification sur un état AVERTISSEMENT
 - u: envoie une notification sur un état INCONNU
 - d: envoie une notification sur un état CRITIQUE et AVERTISSEMENT
 - r: envoie une notification lors d'un rétablissement (*signifie que l'état redevient OK*)
 - f: envoie une notification lors de l'entrée ou la sortie en FLAPPING
 - s: envoie une notification lors de l'entrée ou la sortie en DOWNTIME
 - n: désactive toutes les notifications

Par défaut, tous les états permettent d'envoyer une notification.

  Une option non autorisée provoquera un compteur erreur

Décalage de la première notification (*minutes*)

- *Clé d'import* : first_notification_delay
Cette propriété permet de définir combien de minutes Shinken doit attendre avant d'envoyer la première notification. Ce temps additionnel peut être mis à profit par les utilisateurs pour prendre en compte le check depuis l'interface de visualisation avant que la notification ne soit envoyée.

Par défaut la valeur est 0, ceci signifie que la première notification sera envoyée sans attendre.

  La limite est fixée à 2630880 (*soit 5 ans*).

  Une valeur non numérique provoquera un compteur erreur.

Escalades

- *Clé d'import* : escalations
Cette propriété permet de lier ce check à une ou plusieurs définitions d'escalade.

Si, au bout d'un certain temps, le check n'est toujours pas revenu OK ou pas pris en compte (*Contexte ACKNOWLEDGE ou DOWNTIME*), la règle d'escalade sera appliquée.

Onglet Expert

	Propriété	Valeur		Venant des modèles
Général *				
Données [0]	Envoyer une notification à chaque retour en erreur de check ?	Vrai	Faux	Par défaut [Faux]
Supervision *				
Notifications				
Expert				

Paramétrage du Flapping

Détection du FLAPPING activée ?	Vrai	Faux	Par défaut [Vrai]
Options de détection du FLAPPING ?	Par défaut [o,w,c,u]		
Sortie du Contexte FLAPPING ?	<input type="range"/>		Par défaut [25%]
Entrée du Contexte FLAPPING ?	<input type="range"/>		Par défaut [50%]

Modulations

Modulations d'impact métier ?	☐	-- Par défaut [Même comportement que son hôte] --
Modulations de données ?	☐	-- Par défaut [Aucun] --
Modulations des résultats ?	☐	-- Par défaut [Même comportement que son hôte] --

Gestionnaire d'événements

Gestionnaire d'événements activé ?	Vrai	Faux	Par défaut [Faux]
Tag de Reactionner ? (Remarque: ce Tag n'est utile que pour la commande de Gestionnaire d'événements. Les notifications utilisent le tag défini sur la commande de notification.)	-- Par défaut [Même comportement que son hôte] --		
Commande lancée par le gestionnaire d'événement ?	-- Par défaut [Aucun] --		

Autre

Ordre de définition ?	Par défaut [100]
-----------------------	--------------------

Envoyer une notification à chaque retour en erreur du check

- **Clé d'import** : is_volatile
Cette propriété permet de forcer Shinken à envoyer une notification pour chaque retour en erreur d'un check.
 - Si non forcé, Shinken va vérifier l'état du check avant de lancer une notification (avec un *nombre maximum de confirmation du statut*), puis ne va envoyer des notifications que tous les **interval de notifications** (*typiquement une fois par jour par défaut*).
 - Dans le cas contraire cette option est activée, tous ces systèmes seront désactivés, et une notification sera envoyée pour chaque check retournant un statut en erreur.

Paramétrage du Flapping

Détection du flapping activée

- **Clé d'import** : flap_detection_enabled
Cette propriété permet de définir si la détection du Contexte FLAPPING est actif.
Peut être:
 - Vrai: active la détection du contexte FLAPPING.
 - Faux: désactive la détection du contexte FLAPPING.

Options de la détection du flapping

- **Clé d'import** : flap_detection_options
Cette propriété permet de définir quel statut d'un check sont pris en compte pour le calcul du % de FLAPPING.
C'est une combinaison de l'un ou de plusieurs valeurs:
 - o = statuts OK
 - w = statuts WARNING
 - c = statuts CRITICAL
 - u = statuts UNREACHABLE

Le séparateur est la virgule. La valeur **o** est obligatoire.
Exemple: o,w,u



— Une option non autorisée provoquera un compteur erreur

Sortie du Contexte FLAPPING

- *Clé d'import* : low_flap_threshold
Sur les 21 derniers statuts, chaque fois qu'un statut est différent du précédent (*de OK à Warning par exemple*), le % de FLAPPING augmente. Donc 10 changements représenteront un % de flapping de 50% et 20 représenteront 100%.
Si ce % calculé est **inférieur** au % **de sortie du Contexte FLAPPING**, alors le Contexte ne sera plus FLAPPING.

Entrée du Contexte FLAPPING

- *Clé d'import* : high_flap_threshold
Sur les 21 derniers statuts, chaque fois qu'un statut est différent du précédent (*de OK à Warning par exemple*), le % de FLAPPING augmente. Donc pour 10 changements, cela représentera un % de FLAPPING de 50% et pour 20, cela représentera 100%.
Si ce % calculé est **supérieur** au % **d'entrée dans le Contexte FLAPPING**, alors le Contexte deviendra FLAPPING.
Il sortira de ce Contexte quand ce pourcentage calculé sera **inférieur** au % de sortie du Contexte FLAPPING.

Modulations

Modulations d'impact métier

- *Clé d'import* : business_impact_modulations
Cette propriété permet de définir une ou plusieurs modulations d'impact métier. Les modulations ont une période de temps durant laquelle elles sont actives.
Pendant cette période, la valeur d'impact métier sera changée par celle de la modulation.

Modulations de données

- *Clé d'import* : macromodulations
Cette propriété permet de définir une ou plusieurs modulations de données. Les modulations ont une période de temps durant laquelle elles sont actives.
Pendant cette période, les données du check seront changées par celle de la modulation.

Modulations des résultats

- *Clé d'import* : resultmodulations
Cette propriété permet de définir une ou plusieurs modulations de résultats. (*maximum 4*)
Les modulations de résultats redéfinissent le statut de sortie d'un check, en fonction de son statut initial, d'une période de temps, ou de sa sortie.

Gestionnaire d'événements

Gestionnaire d'événements activé

- *Clé d'import* : event_handler_enabled
Cette propriété permet de définir si Shinken va lancer une commande (*définie par le paramètre **commande de gestionnaire d'évènements***) à des étapes spécifiques du statut du check:
 - quand le check passe d'un état OK à un état à problème (*CRITICAL ou UNKNOWN*)
 - à chaque vérification additionnelle, donc un nombre **maximum de confirmation du statut**

Si la propriété est:

- Vrai: va lancer la commande quand nécessaire
- Faux: ne va pas lancer cette commande

Tag de reactionner

- *Clé d'import* : reactionner_tag
Cette propriété permet de définir le reactionner_tag du check.
La commande du gestionnaire d'événement définie sur ce check sera exécutée uniquement par les Reactionners qui ont dans leur paramètre **reactionner_tags** la valeur définie par cette propriété. Remarque : ce Tag n'est utile que pour la commande de Gestionnaire d'événements.
Les notifications utilisent le tag défini sur la commande de notification.

Par défaut, la valeur de **Tag de Reactionner** est **non tagué**, donc les Reactionners n'ayant aucun reactionner_tag gèrent les événements des checks d'un hôte (*ou cluster*) non tagué, car la valeur par défaut pour les Reactionners est aussi **non tagué**.

Commande lancée par le gestionnaire d'événements

- *Clé d'import* : event_handler
Cette propriété permet de définir la commande que lancera le gestionnaire d'événements.

Autre

Ordre de définition

- *Clé d'import* : definition_order
Ordre de définition



Une valeur non numérique provoquera un compteur erreur