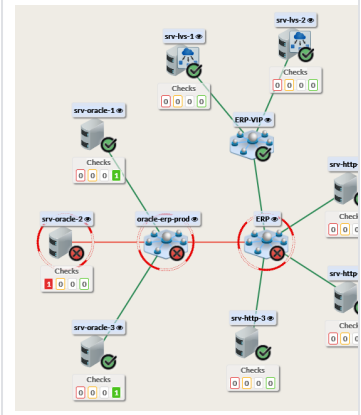
Dépendances : réseau, entre un check et un hôte, entre un cluster et ses éléments

Sommaire

Concept



Dans Shinken, il existe des relations de dépendance entre les éléments (*Hôte, Cluster et Check*).

- Ces relations traduisent des situations dans lesquelles l'état d'un élément impacte celui d'un autre élément (*Par exemple, plusieurs serveurs peuvent dépendre d'un switch qui lui-même dépend d'un firewall*).
- Certaines dépendances sont créées automatiquement par Shinken, tandis que d'autres doivent être explicitement définies par l'utilisateur.

Les relations de dépendance affectent les fonctionnalités suivantes :

- L'envoi des notifications,
- La résolution du statut d'un élément,
- L'exécution des commandes par le gestionnaire d'évènement,
- Le calcul dynamique de l'impact métier.

Cela permet, dans le cas où pour être vérifiés par Shinken certains équipements sont dépendants les uns des autres, de limiter l'envoi des notifications et la remontée des alertes dans l'Interface de Visualisation à l'équipement bloquant, plutôt qu'à tous les équipements qui en dépendent.

Par exemple, dans le cas d'un serveur derrière un switch (*qui est en CRITIQUE*), la dépendance permettra à Shinken d'identifier que l'origine du problème sur le serveur est certainement le switch et d'attirer l'attention de tout utilisateur de l'interface sur ce point.

Créer des liens de dépendances dans Shinken permet de différencier la cause d'un problème, appelé **pro blème source** dans Shinken, de ses impacts et de limiter les alertes au problème source.

Les trois types de dépendance dans Shinken

Lien de dépendance réseau entre un hôte vers des hôtes ou des clusters

Un hôte peut dépendre d'un ou plusieurs autres hôtes et/ou clusters. On parle alors de **dépendances réseaux**.

Ces dépendances doivent être spécifiées par l'utilisateur :

- Soit via l'Interface de Configuration (*voir la page* Éditer un Hôte),
- Soit en utilisant la clé "parents" dans un fichier d'import .cfg (*voir la page Syntaxe des fichiers d'imports*).

La logique consiste à définir, pour un hôte, les hôtes et/ou clusters dont l'état va l'affecter.

Contrainte :

- Les hôtes et leurs dépendances réseaux doivent appartenir au même royaume.
- Une relation de dépendance ne peut pas être créée entre des éléments d'un royaume et de son sous-royaume, ou de deux sous-royaumes différents.

> Un seul Scheduler gérera l'ensemble des éléments liés entre eux par des dépendances réseaux. Il est donc important de faire attention lors de la spécification des liens de dépendance, notamment dans le cas de dépendances imbriquées pour éviter de faire pointer toutes les dépendances vers un même hôte et/ou cluster. Cela pourrait entraîner une mauvaise répartition de la charge entre les Schedulers.
>
> Ce point fera l'objet d'une évolution dans une prochaine version de Shinken, pour enlever cette limitation.

Lien de dépendance entre un check et un hôte

Dès qu'un **check** est créé dans Shinken, il dépend de **l'hôte** auquel il est associé :

- C'est la seule relation de dépendance d'un check dans Shinken.
- Ce lien est généré automatiquement par Shinken et ne peut pas être modifié un utilisateur.

Lien de dépendance entre un cluster et ses éléments

Dès qu'un **cluster** est créé dans **Shinken**, il dépend de tous les **éléments** qui le constituent. C'est la seule relation de dépendance d'un cluster dans Shinken.

- Ce lien est généré automatiquement par Shinken et ne peut pas être modifié ni par un utilisateur.

Détermination des problèmes sources à partir des liens de dépendances

Lorsqu'un élément a un statut **non-OK**, le problème source est calculé par Shinken pour donner une indication sur la cause du statut d'un élément : est-ce que le problème provient de l'élément lui-même, où est-ce seulement une conséquence ?

Shinken parcourt les liens de dépendances pour vérifier si un autre élément que lui-même possède un statut **non-OK** qui peut expliquer ce statut :

- Si c'est le cas, cet autre élément devient alors le problème source.
- Si le statut ne peut pas être expliqué par un autre élément, alors l'élément est considéré comme son propre problème source.
- Dans le cas de dépendances multiples, il est possible d'avoir plusieurs problèmes sources.

Définition d'un problème source

Un élément est considéré comme problème source si son statut **non-OK**, ne peut être expliqué que par l'état de l'élément lui-même et non par l'état d'une dépendance.

- Si son état est explicable par une dépendance, alors c'est cette dépendance qui est le problème source.

Dans Shinken, un élément est considéré comme problème source seulement lorsque son statut non-OK est confirmé (*HARD*) et qu'au moins une de ses dépendances est *OK*.

- Cela traduit que le statut de l'élément ne peut pas s'expliquer par l'état de ses dépendances.
- La seule exception concerne les éléments dépendants d'un cluster. Un cluster ne pouvant pas être un problème source, les checks de cluster ou les hôtes dépendants d'un cluster qui ont un statut non-OK confirmé seront toujours des problèmes sources.

Les dépendances permettent de définir les éléments dont il faut vérifier l'état avant de pouvoir confirmer si un élément est bien la cause de son problème.

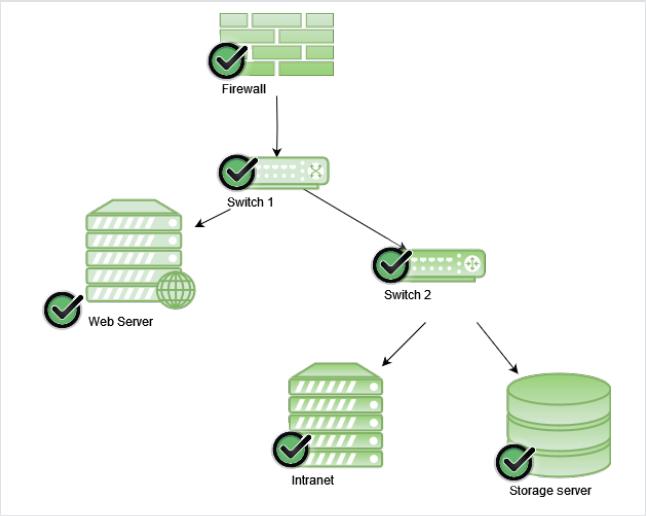
Lorsque le statut d'un élément devient non-OK, Shinken va automatiquement planifier une vérification de ses dépendances. Si une dépendance ne peut pas être considérée comme problème source, ses dépendances sont également vérifiées jusqu'à l'identification de tous les problèmes sources.

> Comme l'état d'un cluster dépend de l'état des éléments que le compose, Un cluster ne peut pas être un problème source.

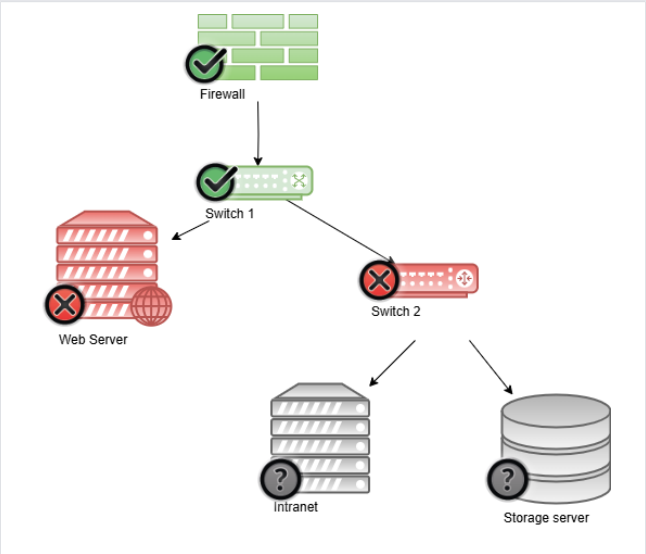
Exemple de problèmes sources

Dans cette architecture, Shinken doit passer par :

- "Switch 1" pour contacter "Web Server"
- "Switch 1" puis "Switch 2" pour accéder à "Intranet" et "Storage server".

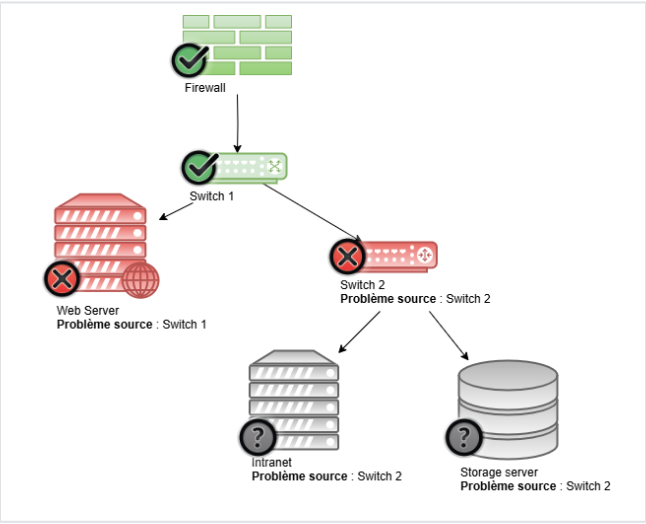


Maintenant, un problème survient sur "Web Server" et "Switch 2" qui deviennent hors service :



Suite au fait que "Switch 2" est hors service, Shinken ne peut plus joindre ni "Intranet" ni "Storage server" :

1. Par conséquent, lorsque Shinken essayera de vérifier si ces éléments sont en ligne, ils ne seront pas joignables à cause de ce problème de routeur et leurs commandes de vérification retourneront "**CRITICAL**"( ).
2. Cependant, il est tout à fait possible que ces éléments fonctionnent correctement. Dans ce cas, si pour "Intranet" et "Storage server" est indiqué une dépendance réseau vers "Switch 2", Shinken considérera que leur état est une conséquence de ce dernier. "Switch 2" sera considéré comme leur "Problème Source".
3. Contrairement à "Intranet" et "Storage server", la machine "Web Server" n'a aucune cause extérieure pouvant expliquer que Shinken ne puisse pas la contacter. "Web Server" est donc également considérée comme "Problème source".



L'impact des liens de dépendance dans Shinken

Héritage des contextes

Un élément peut hériter du contexte de ses dépendances (*Prise en compte, Période de maintenance, etc*). L'héritage des contextes est différent entre les checks, les hôtes et les clusters :

- Un **hôte** hérite uniquement de la prise en compte de ses dépendances,
- Un **check** hérite de la prise en compte et de la période de maintenance de l'hôte,
- Un **cluster** hérite de tous les contextes des éléments qui le constituent.

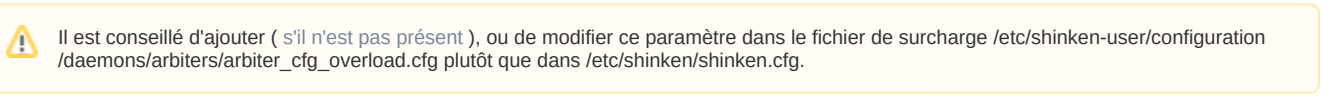
(*voir la page* Statut & Contexte)

Influence sur le statut

Par défaut, lorsqu'un élément devient problème source, l'ensemble des éléments qui en dépendent changent de statut (***UNREACHABLE*** *ou* ***INCONNU***).

Le paramètre de configuration **enable_problem_impacts_states_change** permet d'activer ou de désactiver ce mécanisme (*voir la page* Fichier de configuration (shinken.cfg)).

- Par défaut, ce mécanisme est activé.

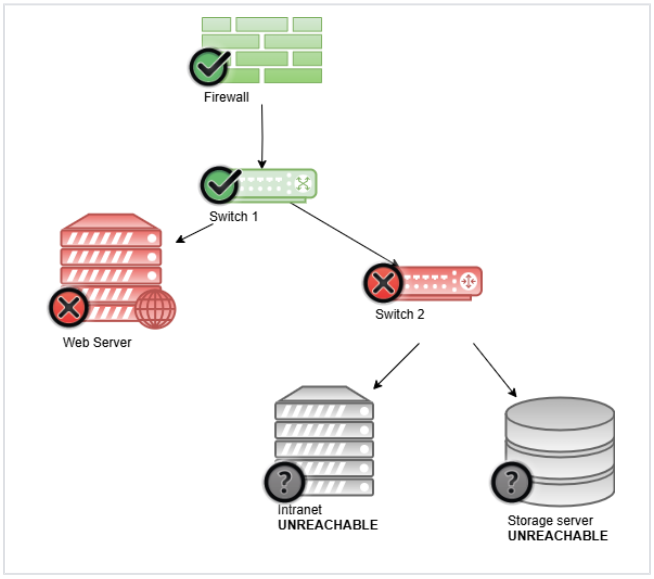


Il est conseillé d'ajouter (s'il n'est pas présent), ou de modifier ce paramètre dans le fichier de surcharge /etc/shinken-user/configuration /daemons/arbiters/arbiter_cfg_overload.cfg plutôt que dans /etc/shinken/shinken.cfg.

sur les hôtes (UNREACHABLE)

Seuls les hôtes peuvent avoir le statut **UNREACHABLE** (*traduit en* ***INCONNU*** *dans l'Interface de Visualisation et les SLA*).

- Un hôte a le statut **UNREACHABLE** s'il est dans un état **non-OK** ( ,  , ), mais qu'il n'est pas problème source. Autrement dit, si un hôte est dans un état **non-OK** et que l'ensemble de ses dépendances sont également dans un état **non-OK,** alors le statut de l'hôte sera **UNREACHABLE**. Il conservera ce statut tant que son état reste **non-OK**.
- Le statut **UNREACHABLE** se traduit dans l'interface de Visualisation et pour les SLA par le statut **INCONNU** ().
- La variable $HOSTSTATE$, utilisable dans les notifications ou dans les commandes exécutées par le gestionnaire d'évènement, retournera quant à elle le statut **UNREACHABLE**.



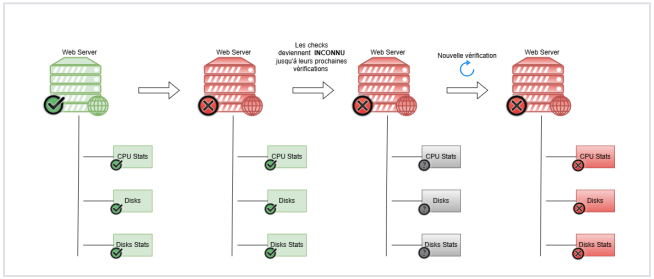
Par exemple, dans la situation suivante :

- Les hôtes "**Intranet**" et "**Storage server**" prendront le statut **UNREACHABLE**

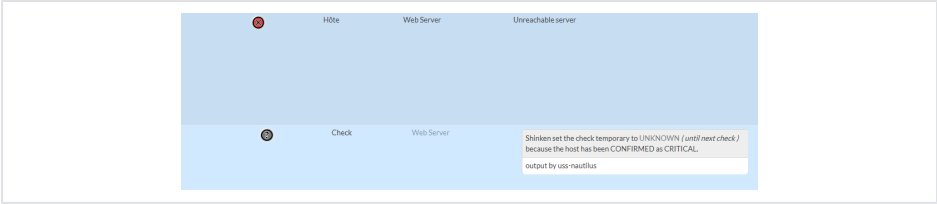
sur les checks (INCONNU)

Comme expliqué plus haut, un check dépend de l'hôte auquel il est attaché.

- Si jamais l'hôte a un problème (*par exemple le serveur s'est éteint ou n'est plus joignable*) les checks qui lui sont attachés (*vérification de l'espace disque restant, de la mémoire vive disponible, etc*) ne peuvent plus retourner de résultat pertinent.
- Leur statuts devient **temporairement** donc **INCONNU** () et ils le **conserveront jusqu'à la prochaine vérification** par Shinken.



Dans l'interface, le retour sera :



Influence sur les notifications

Shinken Enterprise envoie des notifications uniquement pour les problèmes sources afin d'éviter de saturer les utilisateurs de notifications lorsqu'un équipement, dont beaucoup d'éléments dépendent (*par exemple un switch*), tombe.

Cas particulier des **Clusters** :

- Étant donné qu'un cluster ne peut pas être un problème source, la gestion des notifications est différente.
- Un cluster dont le statut passe en **non-OK** enverra toujours des notifications.

Influence sur le gestionnaire d'évènements

La commande du gestionnaire d'événement est toujours exécutée, que l'élément soit problème source, ou non (*voir la page* Gestionnaire d'événements).

Pour pouvoir utiliser les variables (*voir la page Les Variables (Remplacement dynamique de contenu - Anciennement les Macros)*) :

- $**SERVICEIS_ROOT_PROBLEM**$
- $**HOSTIS_ROOT_PROBLEM**$ (*pour les hôtes et les clusters*)
- $**HOSTSTATE**$ (*pour les hôtes et les clusters*)

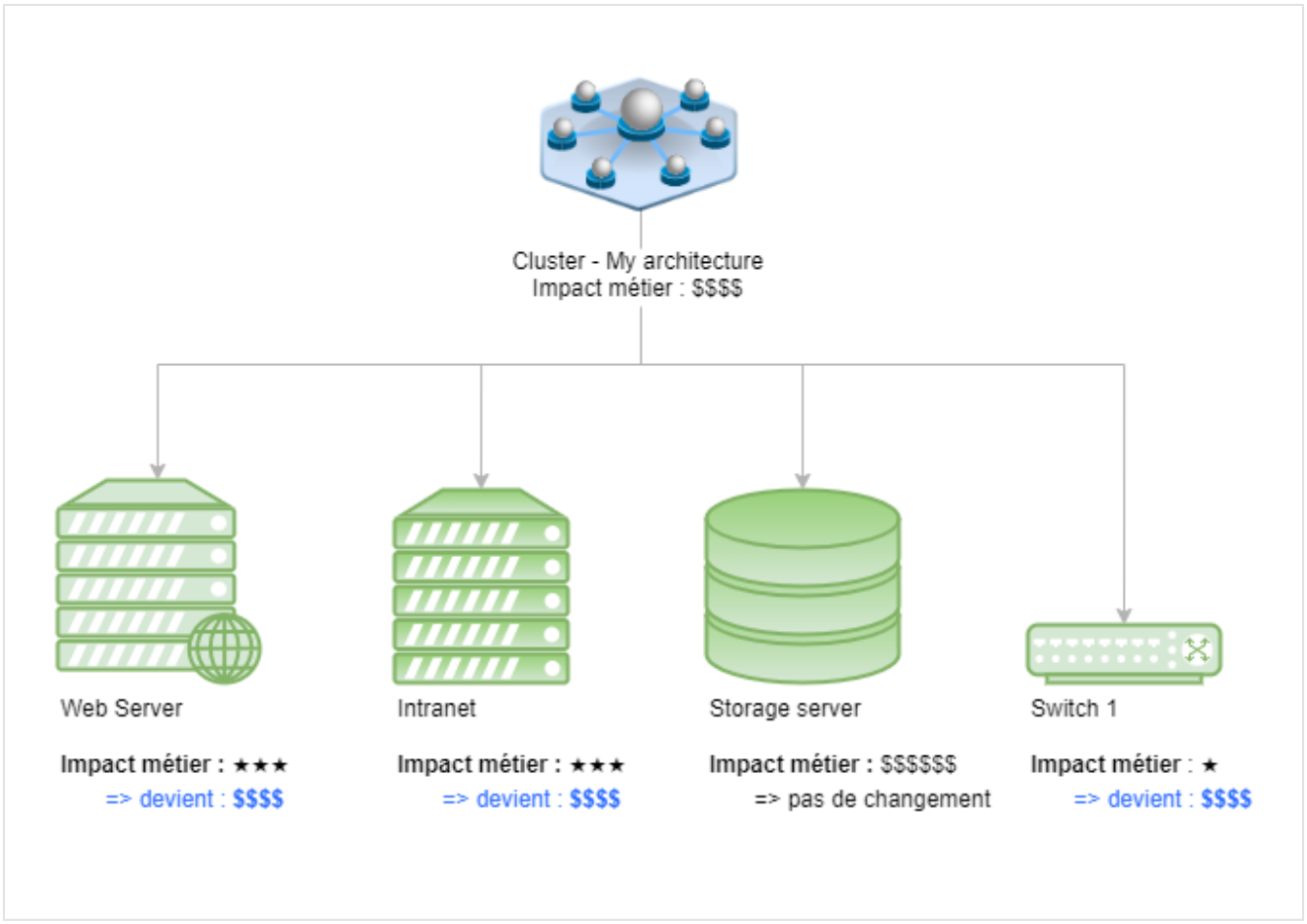
Shinken doit vérifier les dépendances afin de déterminer si un élément est un problème source ou non, ainsi que son statut :

- Cela retarde l'exécution de la commande par le gestionnaire d'événements par rapport au moment où Shinken détecte le besoin de le faire.
  - Il faut une seconde (*un tour de boucle du Scheduler*) à Shinken pour vérifier les dépendances.
  - Puis si le statut des dépendances n'est pas confirmé (*HARD*) Shinken relance une vérification de la dépendance.
  - Il faudra donc attendre le résultat de cette revérification pour exécuter la commande du gestionnaire d'événement.
  - Donc l'exécution la commande du gestionnaire d'événement pourra être retardé le temps de revérifier toutes ses dépendances.

Influence des clusters sur l'impact métier des hôtes

Lorsqu'un hôte est dans un ou de plusieurs **clusters,** son impact métier est recalculé : il **devient égal** à la valeur **la plus élevée** entre **son propre impact métier** et celui des clusters auxquels **il appartient**.

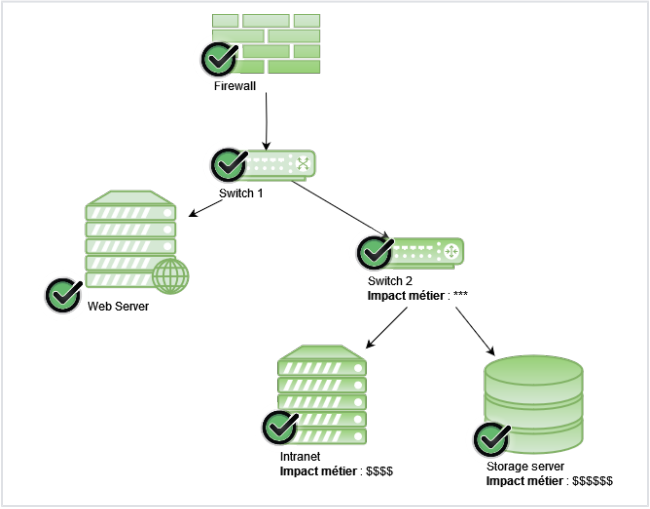
- L'interface de Configuration permet de définir l'impact métier de l'hôte et des clusters.
- Le calcul de la plus haute valeur entre l'impact métier de l'hôte et celui des clusters est réalisé par le **Scheduler**.
  - L'impact métier modifié n'est visible que dans l'**Interface de Visualisation**.



Influence des dépendances réseaux sur l'impact métier des hôtes

Lorsqu'un hôte devient problème source, son impact métier est modifié pour prendre la valeur de l'impact métier le plus élevé parmi tous les éléments qui dépendent de cet hôte et son propre impact métier.

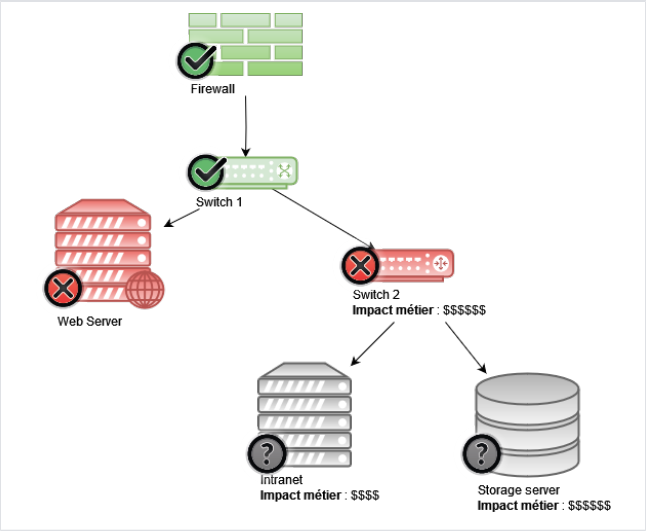
Par exemple, la configuration suivante :



Puis "Switch 2" tombe, ayant pour conséquence de rendre injoignable "Intranet" et "Storage server".

- De tous les hôtes dépendant de "Switch 2", "Storage server" a l'impact métier le plus élevé.
- L'impact métier de "Storage server" est plus élevé que celui de "Switch 2".

Ces deux conditions étant réunies, lorsque "Switch 2" tombe, son impact métier est modifié dynamiquement pour être remplacé par celui de Storage server.



Accéder à l'information dans Shinken

Visualiser les liens de dépendance dans l'interface

Depuis l'Interface de Visualisation, l'arbre de dépendance d'un hôte ou d'un cluster (*c'est-à-dire les dépendances directes et les dépendances des dépendances*), peut être visualisé depuis trois endroits :

1. **Les widgets** : Deux widgets permettent de visualiser les liens de dépendance d'un élément :
    a. Arbre de Dépendances : fournit une vue 2D des dépendances d'un élément (*voir la page Widget Graphe de Dépendance*),
    b. 360 : Fournit une vue 3D des dépendances d'un élément (*voir la page Widget 360°*).
2. **La synthèse :** Fournit un accès rapide vers deux vues fournissant les mêmes schémas que les deux widgets mentionnés au-dessus :
    a. Applications Clés : Équivalent du widget Arbre de Dépendances,
    b. 360 : Équivalent du widget 360.
3. **L'onglet Arbre de Dépendances** : permet également d'afficher la vue 2D des dépendances d'un élément (*voir la page* Onglet Arbre de Dépendance).

Connaitre les problèmes sources

Avec l'interface

La WebUI fournit différentes manière de visualiser et d'identifier les problèmes sources.

- Dans un tableau de bord, il est possible de placer un widget "Problème Sources" qui liste tous les problèmes source d'un hôte ou d'un cluster (*voir la page Widget Problèmes Sources*).
- Il est également possible de lister tous les éléments étant problème source grâce à la liste "Problèmes Sources" (*voir la page Vue - Les Listes (Tous les éléments ou Problèmes sources)*).

Avec les Variables (Remplacement dynamique de contenu)

Sur les hôtes et les services, deux variables permettent de savoir si l'hôte ou le service est un problème source. Ces variables peuvent ensuite être utilisées dans les checks ou commandes :

- **$SERVICEIS_ROOT_PROBLEM$** pour les checks (*pour les hôtes et les clusters*)
- **$HOSTIS_ROOT_PROBLEM$** pour les hôtes (*pour les hôtes et les clusters*)

Ces deux variables sont des booléens qui contiendront (*en toutes lettres*) "True" ou "False" selon si l'hôte ou le service est un problème source (*voir la page Les Variables (Remplacement dynamique de contenu - Anciennement les Macros)*).