Plugin Monitoring for GLPI

Contents

Preface

Introduction

Introduction

But de ce projet

Description des buts de ce projet

Le but de ce plugin est de :

- Manage the configuration of Shinken monitoring via GLPI
- Get state of all services in real time in GLPI
- Get history of checks with rrdtool graphs
- Create tickets in GLPI helpdesk directly on event (planned)

Pourquoi est-ce révolutionnaire?

une révolution?

Nous introduisons plusieurs concepts dans le monitoring :

- Finish time you add each services on a device (computers, servers, switch...) and/or add into templates... **We now use INVENTORY of these devices to set automatically services** (so without human intervention, except for configuration on begin to install plugin Monitoring). This is great with inventory system like FusionInventory).

  

  Note: For example, all computers with name begin by "SRV" and have Apache installed, will be set check_http and check_apachestatus.

- We have defined templates for RRDTOOL in JSON format and can be used by other monitoring systems. We have created 2 kind of templates :
  - to convert perf_data of checks to RRDTOOL files
  - for graph generation gased on these RRDTOOL files previously created and update with data

Liens

Comment contacter le projet

Site internet

Site internet du projet

Vous pouvez trouver les sitesinternet des projets à :

- *https://forge.indepnet.net/projects/monitoring/*

Canal IRC

Adresse du canal IRC

Voici les informations pour trouver les canaux IRC des projets :

- Serveur: *freenode*

- Canaux : *#glpi #shinken #fusioninventory*
- Demandez l'utilisateur *ddurieux*

Termes utilisés dans cette solution

Certains termes sont bien mieux définis

Nous avons redéfinis des termes plus faciles à utiliser. TODO WRITE

Part
I

Préparation

Topics:

Les éléments nécessaires pour préparer la solution

Ce chapitre détaille la préparation de l'installation et son utilisation

Outils indispensables

Voici la liste et la description des outils indispensables

Voici la liste des outils indispensables pour cette solution :

GLPI (Gestion Libre de Parc Informatique)

GLPI est un outil open source for la gestion d'actifs et d'assistance helpdesk

License: GPLv2

Site internet: *http://www.glpi-project.org/*

Version requise: 0.80.x (si possible, la dernière version disponible : 0.80.61)

Il y a plusieurs plugins disponibles qui étendent les fonctionnalités.

FusionInventory

FusionInventory est un outil qui s'occupe de l'inventaire des ordinateurs, serveurs, switchs, imprimantes réseau...

Site internet: *http://www.fusioninventory.org/*

FusionInventory se compose de 2 parties, un agent :

License: GPLv2

Version requise: 2.1.x (si possible, prendre la dernière version disponible : 2.1.13)

et un plugin pour GLPI (côté serveur)

License : AGPLv3

Version requise : 0.80+1.1

Plugin Webservices pour GLPI

C'est un plugin pour GLPI qui ajoute l'accès à GLPI par service web (XML-RPC et SOAP)

License: GPLv2

Site internet: *https://forge.indepnet.net/projects/webservices*

Version requise : 1.2.0

Shinken

Shinken est un outil de monitoring (fork de Nagios) écrit en Pyhton. Shinken va faire les crontrôles de ressources.

License : AGPLv3

Site internet: *http://www.shinken-monitoring.org/*

Version requise : 0.8.1

Plugin Monitoring pour GLPI

C'est ce plugin qui va piloter Shinken à partir de GLPI (et pour d'autres système de monitoring plus tard)

License : AGPLv3

Site internet: *https://forge.indepnet.net/projects/monitoring/*

Version requise : 0.80+1.0

Application Monitoring for GLPI pour Android

Comment cette solution fonctionne

Pour mieux appréhender l'utilisation, voici une description du fonctionnement de la solution

Légende :

- *[D]* = opération automatique
- *[M]* = opération manuelle
- *[DM]* = peut être dynamique ou manuelle

Voici la description des processus :

- *[M]* Un technicien installe un nouveau serveur
- *[DM]* Installation d'un agent FusionInventory afin d'avoir un inventaire complet de ce serveur
- *[D]* FusionInventory envoi un inventaire à GLPI
- *[D]* Le plugin Monitoring associe des ressouces à ce nouvel ordinateur qui sont définis par des règles
- *[DM]* Redémarrer le module Arbiter de Shinken afin de charger la nouvelle configuration
- *[D]* Shinken s'exécute avec les nouveaux contrôles
- *[D]* Shinken envoi les évènements à la base de donnée MySQL de GLPI
- *[D]* Une tâche planifié système met à jour les fichiers RRDTOOL avec des évènements et génère / met à jour les graphiques
- L'utilisateur peut voir les évènements dans GLPI ou avec une application sur son smartphone Android

Conclusion : Ajouter un nouvel équipement à monitorer est simple et ne requiert pas d'intervention humaine.

Part
II

Installation

Topics:

Installation de tous les élements/outils

Suivez ce tutorial pour avoir la solution fonctionnelle :

Shinken

Installation de Shinken

Pour une installation simple en 10 minutes de Shinken, voir la documentation officielle à l'adresse :*http://www.shinken-monitoring.org/wiki/shinken_10min_start*

GLPI

Installation de GLPI

Pour l'installation de GLPI dans votre environnement, se reporter à la documentation officielle à l'adresse : h*ttps://forge.indepnet.net/projects/glpidoc/files*

FusionInventory

Installation de FusionInventory

Il faut installer et configurer l'environnement FusionInventory (serveur et agent). Pour celà, se reporter à la documentation de FusionInventory disponible à l'adresse : *http://fusioninventory.org/wordpress/documentation/*

Plugin Webservices pour GLPI

Installation du plugin Webservices

- Récupérez la dernière version disponible de GLPI : *https://forge.indepnet.net/projects/webservices/files*
- Décompresser l'archive téléchargée dans <glpiFolder>/plugins/
- Se connecter au serveur GLPI (web)
- Aller dans *Configuration* > *Plugins*
- Cliquer sur *installer* pour intégrer le plugin Webservices à GLPI
- Cliquer sur *activer* pour activer le plugin WebServices
- Le plugin est installé

Plugin Monitoring pour GLPI

installation du plugin Monitoring

- Récupérez la dernière version disponible de GLPI : *https://forge.indepnet.net/projects/monitoring/files*
- Décompresser l'archive téléchargée dans <glpiFolder>/plugins/
- Se connecter au serveur GLPI (web)
- Aller dans *Configuration* > *Plugins*
- Cliquer sur *installer* pour intégrer le plugin Monitoring dans GLPI
- Cliquer sur *activer* pour activer le plugin Monitoring
- Le plugin est installé

Android : application glpi_monitoring (optionnel)

Installation de l'application glpi_monitoring pour Android

- Téléchargez le fichier APK avec le navigateur d'Android : *https://forge.indepnet.net/projects/monitoring/files*
- Ouvrir le package APK et installer l'application.
- Cette application est installée

Part
III

Configuration

Topics:

COnfiguration des éléments/outils

Ce chapitre dérit comment configurer les éléments de contrôle

Création des comptes GLPI

Créer un compte GLPI

Il faut créer des comptes utilisateur GLPI :

- Shinken needs a GLPI account to connect via the WebServices plugin to get its configuration. Create a local GLPI account via *Administration* > *User*.

  For example:

  - Login: *shinken*
  - Password: *passshinken*
- Normal user account(s): Don't use GLPI's default super-admin, create your own account (or import from LDAP, see the GLPI documentation for this).

Webservice de GLPI

Configurer le webservice de GLPI

pour Shinken

Configurer les services web de GLPI pour Shinken

Dans GLPI, aller dans le menu *Plugins* > *Web Services*, et ajouter un nouveau service web. Mettre ces valeurs :

- Name: *Shinken*
- Enabled Services: *Yes*
- Compression enabled: *No* (not tested with compression enabled)
- Log connections: *No* (activate it if you want to keep track of connections)
- Debug: *No* (enable for debugging)
- SQL pattern for services: .*
- IP address range: set IP range, in this case set twice the IP of your server where Shinken is installed
- User name: (keep empty in this case)
- Password: (keep empty in this case)

Cliquer sur le bouton ajouter. Le service web pour Shinken est bien créé et donne accès à Shinken.

Pour l'application Android

onfigurer le webservice de GLPI pour l'application Android

Configurer uniquement si vous souhaitez utiliser l'application Android

Dans GLPI, aller dans le menu *Plugins* > *Web Services*, et ajouter un nouveau service web. Définir les valeurs :

- Name: *Android glpi_monitoring*
- Enabled services: *Yes*
- Compression enabled: *No* (not tested with compression enabled)
- Log connections: *No* (activate it if you want to keep track of connections)
- Debug: *No* (enable for debugging)
- SQL pattern for services: *monitoring.**
- IP address range: if you have a fixed IP on your Android phone (not likely), set this IP in the 2 inputs, else you can set a big range like *1.0.0.1* to *254.254.254.254*
- User name: (keep empty in this case)
- Password: (keep empty in this case)

Cliquer sur le bouton ajouter : le service web pour l'application Android est créé et permet d'accéder à GLPI.

Créer des comptes MySQL

Créer un compte MySQL

I faut créer un compte MySQL car le module *Broker* de Shinken va ajouter et mettre à jour des évènements dans la base de donées.

Par exemple créer :

- Login: *shinkenbroker*
- Password: *passshinken*
- Host: IP of your server where Shinken is installed

Vous pouvez mettre des droits à toute la base GLPI pour ce compte mais, pour des raisons de sécurité, donnez lui uniquement les droits à ces tables

- glpi database\glpi_plugin_monitoring_services (SELECT and UPDATE only)
- glpi database\glpi_plugin_monitoring_serviceevents (INSERT only)

Shinken

Configurer Shinken

Module Arbiter

Configurer le module Arbiter de Shinken

Le module *Arbiter* est utilisé pour charger la configuration de Shinken. Ouvrir le fichier de configuration *shinken-specific.cfg* et modifiez le module :

```
define module{ module_name GLPI module_type glpi uri http://localhost/glpi/
plugins/webservices/xmlrpc.php login_name glpi login_password glpi }
```

Les valeurs à modifier dans notre environnement sont :

- uri: url of GLPI, finish always by */plugins/webservices/xmlrpc.php*
- login_name: GLPI account created in "Create GLPI accounts" (in our example, it's *shinken*)
- login_password: password of the GLPI account (in our example, it's *passshinken*)

Le module est désormais configuré, il faut le rajouter dans la configuration du module Arbiter :

```
define arbiter{ arbiter_name Arbiter-Master # host_name node1 ;result of the
 hostname command under Unix address localhost ;IP or DNS adress port 7770
 spare 0 # uncomment the line below if you want to use some modules for your
 arbiter # modules CommandFile, Mongodb, NSCA, VMWare_auto_linking # List
 of interesting modules : # CommandFile : open the named pipe nagios.cmd
 # Mongodb : load hosts from a mongodb database # PickleRetentionArbiter :
 save data before exiting # NSCA : NSCA server # VMWare_auto_linking :
 lookup at Vphere server for dependencies # GLPI : import hosts from GLPI #
 TSCA : TSCA server [...]
```

Donc il faut ajouter le module et nous allons avoir :

```
define arbiter{ arbiter_name Arbiter-Master # host_name node1 ;result of the
 hostname command under Unix address localhost ;IP or DNS adress port 7770
 spare 0 # uncomment the line below if you want to use some modules for your
 arbiter # modules CommandFile, Mongodb, NSCA, VMWare_auto_linking modules
 GLPI # List of interesting modules : # CommandFile : open the named
 pipe nagios.cmd # Mongodb : load hosts from a mongodb database #
```

```
 PickleRetentionArbiter : save data before exiting # NSCA : NSCA server #
 VMWare_auto_linking : lookup at Vphere server for dependencies # GLPI :
 import hosts from GLPI # TSCA : TSCA server [...]
```

Module Broker

Configurer le module Broker de Shinken

Le module Broker envoi les évènements. Ouvrir le fichier de configuration *shinken-specific.cfg* et modifier le module :

```
# send into GLPI DB # ================ Work with Plugin Monitoring of GLPI
 ================ define module{ module_name glpidb module_type glpidb
 database glpi ; database name user root ; database user password root ;
 must be changed host localhost ; host to connect to }
```

Les valeurs à modifier ici sont :

- database: name of GLPI MySQL database
- user: MySQL account created on chapterTODO (in our example, it's *shinkenbroker*)
- password: password of GLPI account (in our example, it's *passshinken*)
- host: IP or name of server where MySQL server is installed.

Le module est défini, désormais ajoutons le dans la configuration du module Arbiter :

```
#The broker manages data export (in flat file or in database) #with
 its modules #Here just log files and status.dat file modules define
 broker{ broker_name broker-1 address localhost port 7772 spare 0 # Which
 modules to load? LiveSatus and logs by default. modules Livestatus, Simple-
log, WebUI # Other interesting modules to add : # PickleRetentionBroker :
 save data when quitting # ToNdodb_Mysql : NDO database support # NPCDMOD :
 Use the PNP addon # Graphite-Perfdata : Use teh Graphite backend for
 perfdata # WebUI : Shinken Web interface [...]
```

Donc il faut ajouter le module et nous allons avoir :

```
#The broker manages data export (in flat file or in database) #with
 its modules #Here just log files and status.dat file modules define
 broker{ broker_name broker-1 address localhost port 7772 spare 0 # Which
 modules to load? LiveSatus and logs by default. modules Livestatus,
 Simple-log, WebUI, glpidb # Other interesting modules to add : #
 PickleRetentionBroker : save data when quitting # ToNdodb_Mysql : NDO
 database support # NPCDMOD : Use the PNP addon # Graphite-Perfdata : Use
 teh Graphite backend for perfdata # WebUI : Shinken Web interface [...]
```

Pour une implémentation simplifiée, les modules Livestatus, Simple-log et WebUI peuvent être supprimés.

Objets communs

onfigurer les objects communs de Shinken

Par défaut, Shinken charge les object communs comme les commandes, timeperiod, hôtes, services. Si vous souhaiter charger uniquement la configuration avec GLPI, vous devez modifier le fichier *nagios.cfg* :

```
#Configuration files with common objects like commands, timeperiods, #or
 templates that are used by the host/service/contacts #cfg_file=commons.cfg
 cfg_file=commands.cfg cfg_file=timeperiods.cfg cfg_file=escalations.cfg
 cfg_file=dependencies.cfg cfg_file=contacts.cfg #cfg_dir=/opt/omd/sites/
pilot/etc/nagios/conf.d/pilot/dynamic #Now templates of hosts, services
 and contacts cfg_file=templates.cfg #Now groups cfg_file=servicegroups.cfg
 cfg_file=hostgroups.cfg cfg_file=contactgroups.cfg #and now real hosts,
```

```
 services, packs and # discovered hosts cfg_dir=hosts cfg_dir=services
 cfg_dir=packs cfg_dir=objects/discovery # Un comment this for a sample
 configuration #cfg_dir=sample [...]
```

Par

```
#Configuration files with common objects like commands, timeperiods,
 #or templates that are used by the host/service/contacts
 #cfg_file=commons.cfg #cfg_file=commands.cfg #cfg_file=timeperiods.cfg #cfg_file=escala
 #cfg_dir=/opt/omd/sites/pilot/etc/nagios/conf.d/pilot/dynamic #Now
 templates of hosts, services and contacts #cfg_file=templates.cfg #Now
 groups #cfg_file=servicegroups.cfg #cfg_file=hostgroups.cfg #cfg_file=contactgroups.cfg
 #and now real hosts, services, packs and # discovered
 hosts #cfg_dir=hosts #cfg_dir=services #cfg_dir=packs #cfg_dir=objects/
discovery # Un comment this for a sample configuration #cfg_dir=sample [...]
```

Plugin monitoring

Configurer le plugin Monitoring

Configurer les définitions de contrôle

COnfiguration des contrôles

Cette définition de contrôle est utilisée pour définir une ressource, quel type de contrôle (toutes les 5 minutes avec 3 essais, toutes les 15 minutes avec 2 essais...)

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Définition de contrôle*

Elements à configurer pour chaque definition de contrôle sont :

- *Name*: give a name
- *Time in minutes between 2 checks*: define the time in minutes between 2 checks
- *Max check attempts (number of retries)*: define number of retries when a state (OK, warning, critical...) change to be really this new state.
- *Time in minutes between 2 retry*: time between 2 retries

Definir les commandes

Configuration des commandes

Ces commandes sont utilisée pour les contrôles et détectent l'état des services à monitorer.

Plusieurs commandes sont ajoutées lors de l'installation du plugin.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Commandes*

Les éléments à configurer pour chaque commande sont :

- *Name*: give a name
- *Command name*: define command name, must be unique and without special characters
- *Command line*: command name of file to run (on server where Shinken is installed because it's Shinken will run this command). You can use Shinken Macro like *$PLUGINSDIR$*, *$MYSQLUSER$*... and you can use macro for dynamic arguments used into plugin_monitoring like *$ARG1$*, *$ARG2$*, *$ARG3$*...
- Arguments description: this is visible only when have *$ARG1$*, *$ARG2$*, *$ARG3$*... in *Command line* field. For each arguments, have an input with description of this argument. For exemple, if have in *Command line* the command *$PLUGINSDIR$/check_load -r -w $ARG1$ -c $ARG2$*, will have 2 input for each of the 2 arguments and in *ARG1* set what is this argument is for: *WARNING status if load average exceeds WLOADn (WLOAD1,WLOAD5,WLOAD15)*. Define these arguments is important for define components.

Défini les éléments de contrôles

Configuration des composants

Ces composants sont utilisés pour définir quoi monitorer pour chaque équipement.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Elements de contrôles*

Les élements à configurer pour chaque contrôle decomposant sont :

- *Name*: give a name
- *Command*: define command definied on chapterTODO. This field is required except if you define this component as a remote check.
- *Template (for graphs generation)*: select a template for graph generation (RRDTOOL).
- *Check definition*: define the definition check for this component (see chapterTODO).
- *Active checks*: is this component is an active check (Shinken run the command)
- *Passive check*: is this component is a passive check (remote host run the command and send event to Shinken only when change)
- *Check period*: set period (it's GLPI calendar) to define when check is running (24x7, not week end....)

Il est possible de définir un contrôle à distance, le serveur distant va exécuter lui-même la commande et renvoyer le résultat à Shinken). Voici les champs pour faire celà :

- *Utilitaire utilisé pour les contrôles à distance*: choisir le système à utiliser : *byssh*, *nrpe* or *nsca*.
- *Utiliser des arguments (seulement pour NRPE)*: si *nrpe* est choisi, il est possible de définir la commande à exécuter, donc défini ici et pas dans le fichier de configuration nrpe.cfg du système distant. **Activer peut être dangereux pour la sécurité**
- *Commande alias si c'est requis (seulement pour NRPE)*: défini la commande and le fichier de config nrpe.cfg sur les serveurs distants.

Définir les catalogues de composants

Configuration des catalogues de composants

Ces catalogues de composants sont utilisés pour définir quels hôtes le système de monitoring va contrôler les ressources.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Catalogue de composants*

Elements de contrôles

Définir les composants de *catalogue de composants*

Dans cet onglet, il y a une liste de composants pour les hôtes de ce catalogue de composants. Ajouter tout simplement des composants.

Exemple : pour un *catalogue de composants* nommé 'serveurs http', il peut y avoir les composants check_http et check_apachestatus

Hôtes

Ajouter des hôtes à *catalogue de composants*

Il y a 2 méthodes pour ajouter des hôtes dans ce catalogue de composants.

Hôtes statiques

Ajouter des hôtes statiques à *catalogue de composants*

La première méthode est d'ajouter des hôtes manuellement par un utilisateur. Cette méthode est appelé 'hôtes statiques' et disponibles dans l'onglet portant le même nom.

Hôtes dynamiques

Configurer les hôtes dynamiques à *catalogue de composants*

La deuxième méthode pour ajouter des hôtes est de définir des rèles (moteur de recherche). Cette méthode est appelée 'hôtes dynamiques' et disponible dans les onglets *règles* et *hôtes dynamiques*.

Dans l'onglet *règles*, il est possible d'ajouter **seulement une règle** par type de matériel. Ces règles utilisent le moteur de recherche de GLPI. Si une règle d'ordinateur est définie par *Système d'exploitation* contient *Linux*, tous les ordinateur qui matchent cette recherche vont être ajoutés automatiquement dans l'onglet *hôtes dynamiques*.

Note: Quand on ajoute ou modifie un type de matériel (comme un ordinateur dans notre exemple), la règle va se rejouer et mettre à jour la liste des hôtes dynamiques sans intervention humaine. Donc quant on ajoute ou modifie un inventaire avec un outil automatique comme FusionInventory, il va mettre à jour les hôtes et des composants affectés.

Part
IV

Utilisation quotidienne

Topics:

Utilisation quotidienne du système de monitoring

Ce chapitre dérit comment configurer les éléments de contrôle

Redémarrer Shinken

Redémarrer Shinken

Quant on ajoute ou modifie un équipement (configuration d'un équipement), sa configuration est modifié dans GLPI. Redémarer le module Arbiter de Shinken est indispensable pour recharger la nouvelle configuration. Celà peut être fait par un utilisateur avec la commande de redémarrage ou peut être avec un script. Nous travaillons avec l'équipe de Shinken pour avoir la possibilité de recharger cette configuration de GLPI manuellement ou automatiquement quand la configuration change.

Afficher les évènements

Afficher les évènements

Où visualiser les évènements?

Dans GLPI

Dans GLPI

Les évènements peuvent être visualisés dans 2 endroits dans GLPI :

- Tableau de bord général dans le menu plugins > Monitoring > Tableau de bord. Il y a plusieurs vues :
  - *Components catalog*: display state of all catalogs
  - *All resources*: display all resources and you can use search engine to find a resource or a type of resource.
- Dans chaque fiche de matériel, un onglet *Monitoring-resources* affiche les ressources et leur état pour cet équipement.

Si un gabarit est défini pour les composants, les graphiques pourront être affichés.

Dans l'application Android

Dans l'application Android

Affiche le nombre d'équipements dans l'état Critical, warning ou OK (types de vues *Catalogue de composants* ou *Toutes les ressources* peuvent être paramétrées dans le menu *options*

Quand vous cliquez sur le boutton de couleur (vert, orange ou rouge), une nouvelle fenêtre apparait et affiche la liste des ressources/hôtes qui ont cet état.