

Dimensionnement des Pollers à l'aide de la commande `shinken-scheduler-export-data`

Sommaire

Connaître les consommation des checks afin de pouvoir dimensionner ses Pollers quand on rajoute des hôtes
Création du tableau récapitulatif sur la consommation totale des temps CPU par royaume
Consolidation des données: utilisation d'un tableau croisé dynamique
Création du tableau croisé dynamique
Obtenir la consommation CPU totale par royaume
Sélection des royaumes en tant que lignes de notre tableau
Sélection des "Valeurs": le `cpu_time`, le temps consommé par les checks
Passer du nombres de lignes avec `"cpu_time"` à une vraie somme des temps CPU
Passer du nombre total de temps CPU consommé au nombre de CPU nécessaires
Tableau final avec le nombres de CPUs utilisés par royaume
Analyse des résultats

Connaître les consommation des checks afin de pouvoir dimensionner ses Pollers quand on rajoute des hôtes

Traditionnellement dans un projet de supervision, on commence par mettre en supervision un parc représentatif de l'ensemble de son infrastructure. Une fois cet échantillon représentatif en place, il est utile de savoir quelle va être la consommation CPU une fois l'ensemble de son infrastructure supervisée.

Prenons comme exemple le cas où on supervise deux clients, partiellement déployés, avec chacun son royaume isolé :

- Royaume **customer-a**, déployé à **10%**
- Royaume **customer-b**, déployé à **20%**

On se pose la question du nombres de CPUs qui seront nécessaires une fois que **100%** du parc sera supervisé.

Chaque royaume étant isolé, chacun a ses propres Pollers, et donc sa propre consommation CPU.



Il est important de noter qu'ici qu'on extrapole en partant du principe qu'on rajoutera le même type d'éléments, dans les mêmes proportions.

Si le déploiement préliminaire était sur un type d'élément (*exemple: windows*) et que la suite est composée d'éléments totalement différents (*exemple: équipement réseaux*), alors l'extrapolation ne sera pas concluante car les checks de ces éléments ne sont pas les mêmes et leur consommation CPU également.

Pour cela, on va extraire des informations issues de la commande **shinken-scheduler-export-data** afin d'avoir :

- **le nombre de CPUs utilisés par les checks pour chaque royaume**

Nous allons avoir besoin d'une extraction de données avec en option une prévision de la charge sur une période représentative, disons par exemple 1 heure. Il suffit alors de lancer la commande comme ceci:

```
shinken-scheduler-export-data --export-type=sizing-pollers --simulate-scheduling-for-X-seconds=3600
```

L'importation du fichier .csv généré est décrit dans la page suivante : [shinken-scheduler-export-data - export des données du Scheduler](#)



Avec ce lancement les noms (hôte, check, commandes et royaumes) seront présents dans l'export. Il est tout à fait possible de faire l'analyse sur un export en **--anonymous**, des hash des noms des royaumes seront utilisés au lieu des noms finaux.

Pour retrouver le nom des royaumes, on peut prendre l'UUID d'un hôte qui a ce royaume, et trouver son nom via l'interface de visualisation ou de configuration. (Voir la page [TIPS - Récupérer l'UUID d'un élément \(Cluster / Hôte / Check \)](#))

Création du tableau récapitulatif sur la consommation totale des temps CPU par royaume

Consolidation des données: utilisation d'un tableau croisé dynamique

On va devoir procéder à une consolidation des nombreuses données de checks obtenues afin d'obtenir un résultat exploitable.

- Pour cela on va utiliser la fonctionnalité d'Excel : **le tableau croisé dynamique**.
- Un tableau croisé dynamique dans un tableur est un outil de analyse de données qui vous permet de créer une vue synthétique et facile à lire d'une grande quantité de données (*ici nos exécutions de checks*).

On y choisi les données à inclure, comment les organiser et comment les synthétiser, filtrer, classer et totaliser les données en fonction de nos besoins.

Création du tableau croisé dynamique

La création du tableau récapitulatif passe par la création d'un Tableau croisé dynamique. Depuis votre Feuille d'importation des données, il faut cliquer sur **Insertion Tableau croisé dynamique**, et valider :

The screenshot shows the Excel interface with the 'Créer un tableau croisé dynamique' dialog box open. The dialog box has the following settings:

- Tableau/Plage:** Feuil2!\$A\$1:\$O\$356
- Choisissez l'emplacement de votre rapport de tableau croisé dynamique:**
 - ☒ Nouvelle feuille de calcul
 - ☐ Feuille de calcul existante
- Emplacement:** Feuil2
- ☐ Ajouter ces données au modèle de données

The background data table has the following columns:

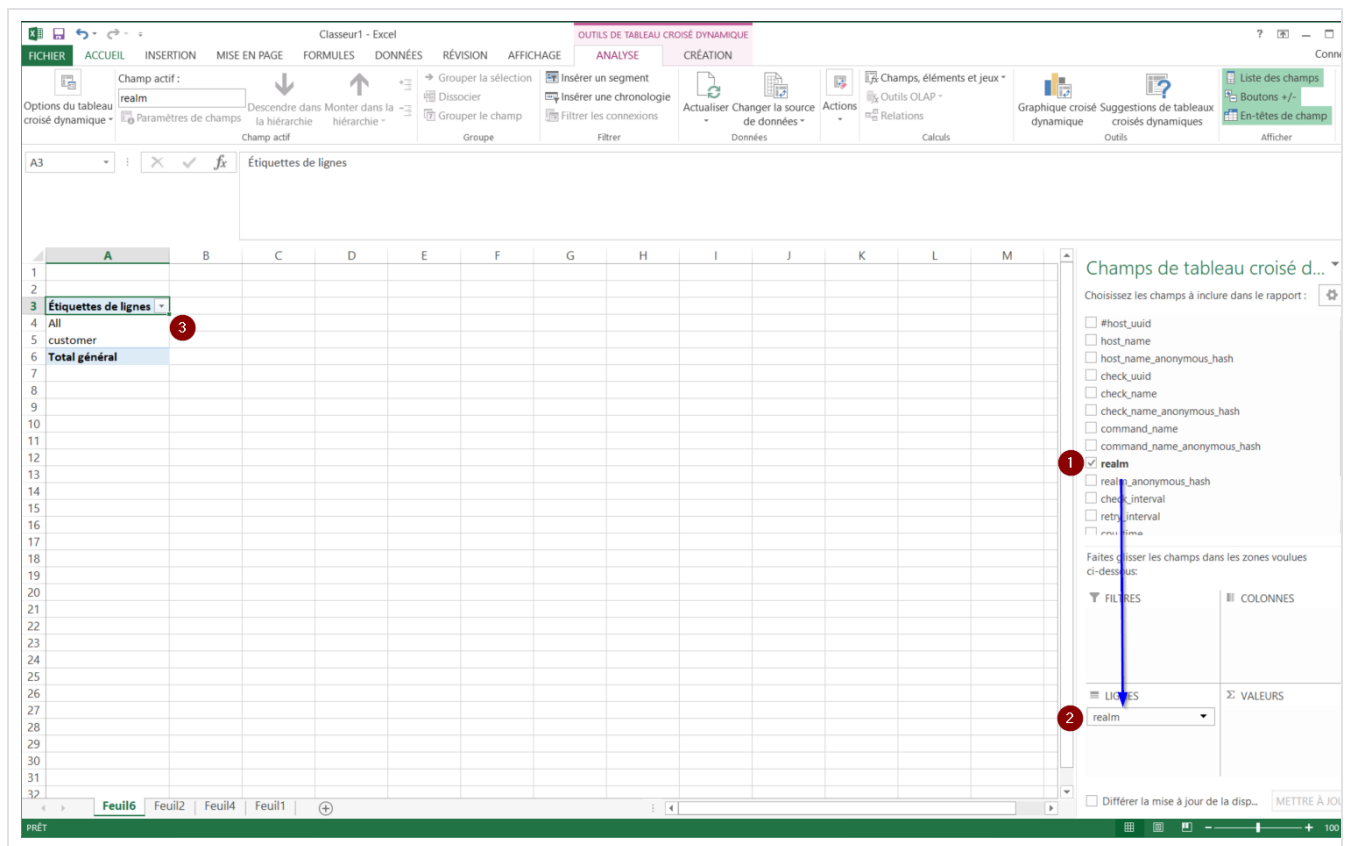
	A	B	C	D	E	F	G	H
1	#host_uid	host_name	host_name_anonymous_hash	check_uid	check_name	check_name_anonymous_hash	command_name	command_name_anonymous_hash
2	dfe65e40072b41b8950ed69d1329cc48	srv-15	anonymous-hash-0f628613ec	0f31a3424df44ae41053d5c72dbc7eda	dump-check-example-2-3	anonymous-hash-ab3dec090e	dump-check-example-2	anonymous-hash-aa36432587
3	1b7ce868df547038603eeca61616cae2	srv-7	anonymous-hash-eb645d6b77	24e86f31b7f133bda96ac03a67ff0f14	dump-check-example-2-13	anonymous-hash-a6eb5d73e9	dump-check-example-2	anonymous-hash-aa36432587
4	8bc339da0a311eda85d080027940ca8	srv-2	anonymous-hash-6fbb0b8226	bb64373c94c329a13609798292c7db56	dump-check-example-2-14	anonymous-hash-19848ce407	dump-check-example-2	anonymous-hash-aa36432587
5	8bc91c8a0a311eda99d080027940ca8	srv-8	anonymous-hash-3			anonymous-hash-2e9e104669	dump-check-example-2	anonymous-hash-aa36432587
6	8bd19548a0a311eda8e0f080027940ca8	srv-10	anonymous-hash-a			anonymous-hash-19848ce407	dump-check-example-2	anonymous-hash-aa36432587
7	8ae43305c6b6490d8129c92d58ca8082	srv-1	anonymous-hash-9			anonymous-hash-688a5ae71d	dump-check-example-2	anonymous-hash-aa36432587
8	1884e097f2a74ee7b82f1a8479852ea1	srv-5	anonymous-hash-4			anonymous-hash-ff61e06169	dump-check-example-2	anonymous-hash-aa36432587
9	f4f1c894dcea4c938ca2f86acfe7a2c0	srv-17	anonymous-hash-a			anonymous-hash-ab3dec090e	dump-check-example-2	anonymous-hash-aa36432587
10	8bdc4e2aa0a311eda148080027940ca8	srv-16	anonymous-hash-b			anonymous-hash-19848ce407	dump-check-example-2	anonymous-hash-aa36432587
11	8bd19548a0a311eda8e0f080027940ca8	srv-10	anonymous-hash-a			anonymous-hash-2e9e104669	dump-check-example-2	anonymous-hash-aa36432587
12	8be631e2a0a311eda99d080027940ca8	srv-20	anonymous-hash-4			anonymous-hash-5f8ecad9f97	dump-check-example-2	anonymous-hash-aa36432587
13	340f7d2fcaab43ccb394e0f6d6427469	srv-13	anonymous-hash-2			check-host-alive	anonymous-hash-2d6a3ff550	All
14	dfe65e40072b41b8950ed69d1329cc48	srv-15	anonymous-hash-0			anonymous-hash-688a5ae71d	dump-check-example-2	anonymous-hash-aa36432587
15	d26e317996a2441e955f04f759f93c18	srv-9	anonymous-hash-8			anonymous-hash-b5242c09dc	dump-check-example-2	anonymous-hash-aa36432587
16	1b7ce868df547038603eeca61616cae2	srv-7	anonymous-hash-e			anonymous-hash-ab3dec090e	dump-check-example-2	anonymous-hash-aa36432587
17	8bc1735ca0a311eda148080027940ca8	srv-4	anonymous-hash-9			anonymous-hash-2e9e104669	dump-check-example-2	anonymous-hash-aa36432587
18	8bd9ad28a0a311eda85d080027940ca8	srv-14	anonymous-hash-7			anonymous-hash-ab3dec090e	dump-check-example-2	anonymous-hash-aa36432587
19	8bd624b4a0a311eda8635080027940ca8	srv-12	anonymous-hash-0			anonymous-hash-2e9e104669	dump-check-example-2	anonymous-hash-aa36432587
20	2d833e2aeebc48838510043026bedda	srv-19	anonymous-hash-5			anonymous-hash-19848ce407	dump-check-example-2	anonymous-hash-aa36432587
21	340f7d2fcaab43ccb394e0f6d6427469	srv-13	anonymous-hash-2			anonymous-hash-8ac2db040f	dump-check-example-2	anonymous-hash-aa36432587
22	340f7d2fcaab43ccb394e0f6d6427469	srv-13	anonymous-hash-20686577d2	512071c5e2b49849ed7d72ae5f657578	dump-check-example-2-7	anonymous-hash-176d0d8c03	dump-check-example-2	anonymous-hash-aa36432587
23	8bc7135ca0a311eda148080027940ca8	srv-4	anonymous-hash-954836fc25	424c8690e6ff9da09f892f383b49de1	dump-check-example-2-15	anonymous-hash-80e21d30ac	dump-check-example-2	anonymous-hash-aa36432587
24	1884e097f2a74ee7b82f1a8479852ea1	srv-5	anonymous-hash-44610da532			check-host-alive	anonymous-hash-2d6a3ff550	All
25	8bd9ad28a0a311eda85d080027940ca8	srv-14	anonymous-hash-72169f192f	b4e36da700f3755d237eacd9fbee332e	dump-check-example-2-6	anonymous-hash-8ac2db040f	dump-check-example-2	anonymous-hash-aa36432587
26	8bd19548a0a311eda8e0f080027940ca8	srv-10	anonymous-hash-a89f7681d1			check-host-alive	anonymous-hash-2d6a3ff550	cur
27	8bc339da0a311eda85d080027940ca8	srv-2	anonymous-hash-6fb60b8226	0f31a3424df44ae41053d5c72dbc7eda	dump-check-example-2-3	anonymous-hash-ab3dec090e	dump-check-example-2	anonymous-hash-aa36432587
28	340f7d2fcaab43ccb394e0f6d6427469	srv-13	anonymous-hash-20686577d2	1769d9759e8be6d3680ff5739f431752	dump-check-example-2-12	anonymous-hash-bb4ef5977d	dump-check-example-2	anonymous-hash-aa36432587
29	340f7d2fcaab43ccb394e0f6d6427469	srv-13	anonymous-hash-20686577d2	424c8690e6ff9da09f892f383b49de1	dump-check-example-2-15	anonymous-hash-80e21d30ac	dump-check-example-2	anonymous-hash-aa36432587
30	8be20806a0a311eda9182080027940ca8	srv-8	anonymous-hash-c7cf9a1e80	1769d9759e8be6d3680ff5739f431752	dump-check-example-2-12	anonymous-hash-bb4ef5977d	dump-check-example-2	anonymous-hash-aa36432587
31	8bcd91c8a0a311eda99d080027940ca8	srv-18	anonymous-hash-35a25c2308	6582a351a934f732b7096e3c8ddaf10a	dump-check-example-2-8	anonymous-hash-3395da4466	dump-check-example-2	anonymous-hash-aa36432587
32	8bdc4e2aa0a311eda148080027940ca8	srv-16	anonymous-hash-bb7dc70c8e	512071c5e2b49849ed7d72ae5f657578	dump-check-example-2-7	anonymous-hash-176d0d8c03	dump-check-example-2	anonymous-hash-aa36432587

Obtenir la consommation CPU totale par royaume

Sélection des royaumes en tant que lignes de notre tableau

Arrivé sur la nouvelle feuille, Excel nous propose de sélectionner les lignes de notre nouveau tableau, dans le bloc de droite "**Champs de tableau croisé dynamique**".

Dans notre cas, ce sera nos royaumes. Il faut donc faire glisser le champ **realm** (ou bien **realm_anonymous_hash** si on a une version anonyme de l'export) vers le bloc "**Lignes**" :



On obtient alors la base de notre tableau, avec des données qui seront désormais organisées par royaume.

Sélection des "Valeurs": le `cpu_time`, le temps consommé par les checks

A chaque ligne/royaume, il faut lui assigner une (*ou plusieurs*) "Valeurs". Pour cela, on fait glisser le champ "`cpu_time`" vers le bloc "Valeurs" afin d'avoir pour chaque royaume son champ `cpu_time` associé.

Par contre, par défaut, Excel prends par défaut le nombre d'occurrences du champ `cpu_time` comme "Valeur", ce qui n'est pas ce qui est souhaité :

The screenshot shows an Excel window with a PivotTable. The PivotTable is located in the range A3:M6. The row labels are 'Étiquettes de lignes' and the values are 'Nombre de cpu_time'. The data is as follows:

Étiquettes de lignes	Nombre de cpu_time
All	168
customer	169
Total général	337

The PivotTable Fields task pane on the right shows the following configuration:

- FILTRES:** realm
- VALEURS:** Nombre de cpu_time

Red circles and arrows highlight the field names and the value field name.

Avec ce choix par défaut, Excel nous montre que nous avons prévu de lancer **5250** checks dans l'heure sur le royaume **customer-a**, et **15750** sur le **customer-b**.

Ceci n'est pas utile pour notre analyse, car ne nous apprend pas quelle est leur consommation CPU réelle. On doit donc passer d'un nombre d'occurrences à une **"Somme"** des valeurs pour chaque royaume.

Passer du nombres de lignes avec "cpu_time" à une vrai somme des temps CPU

Une modification des **"Paramètres des champs de valeurs"** est nécessaire sur **"Nombre de cpu_time"** :

- il faut changer le type de calcul de **"Nombre"** à **"Somme"**
- puis dans un second temps on en profite pour changer le titre de la colonne en **"Somme de temps CPUS utilisés par royaume"** via **"Nom personnalisé"**

The screenshot shows an Excel spreadsheet with a PivotTable. The PivotTable has 'Étiquettes de lignes' (Row Labels) and 'Nombre de cpu_time' (Sum of cpu_time). The data is as follows:

Étiquettes de lignes	Nombre de cpu_time
All	168
customer	169
Total général	337

A dialog box titled 'Paramètres des champs de valeurs' (Value Field Settings) is open. It shows 'Nom de la source' as 'cpu_time' and 'Nom personnalisé' as 'Somme de cpu_time'. Under 'Synthèse des valeurs par', the 'Afficher les valeurs' tab is selected. The 'Résumer le champ de valeur par' section shows a list of aggregation functions: 'Somme' (highlighted with a red circle 2), 'Nombre', 'Moyenne', 'Max', 'Min', and 'Produit'. The 'OK' button is highlighted with a red circle 3.

On the right, the 'Champs de tableau croisé d...' (PivotTable Fields) task pane is visible. It shows a list of fields including 'realm', 'cpu_time', and others. 'realm' is in the 'LIGNES' (Rows) area, and 'Nombre de cpu_time' is in the 'VALEURS' (Values) area.

On obtient ainsi pour chaque royaume la somme de temps CPU utilisé pour l'ensemble des checks, sur la période voulue (*ici une heure*).

Passer du nombre total de temps CPU consommé au nombre de CPU nécessaires

Les valeurs de temps CPU cumulés par royaume n'est pas encore facilement lisible pour notre analyse.

Afin d'avoir le nombre de CPU nécessaires, il faut revenir à l'échelle d'une seconde, et donc rajouter une nouvelle colonne avec comme valeur la "**Somme de temps CPUS utilisés par royaume**" divisée par 3600 (*secondes, soit une heure correspondant à notre extraction*), et ce pour chaque ligne :

Classeur1 - Excel

Formule dans C4: `=LIREDONNEESTABRCROISDYNAMIQUE("cpu_time";$A$3;"realm";"All")/3600`

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															
2															
3	Étiquettes de lignes	Somme de cpu_time	Nombres de CPU utilisés												
4	All	39,798	0,011055												
5	customer	40,249	0,011180278												
6	Total général	80,047	0,02235278												
7															
8															
9															
10															
11															
12															
13															
14															
15															
16															
17															
18															
19															
20															
21															
22															
23															
24															
25															
26															
27															
28															
29															
30															
31															
32															

Feuil6 | Feuil2 | Feuil4 | Feuil1

Le **Total général** des CPUs utilisés est utile à titre informatif, mais ce sont surtout les valeurs par royaumes qui sont intéressantes.

Tableau final avec le nombres de CPUs utilisés par royaume

Une fois le calcul effectué, voici le tableau final obtenu :

Classeur1 - Excel

Formule dans C7:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1																		
2																		
3	Étiquettes de lignes	Somme de temps CPUS utilisés par royaume	Nombre de CPUS utilisés par royaume															
4	customer-a	1322,142	0,367261667															
5	customer-b	3985,799	1,107166389															
6	Total général	5307,941	1,474428056															
7																		
8																		
9																		
10																		
11																		
12																		
13																		
14																		
15																		
16																		
17																		
18																		
19																		
20																		
21																		
22																		
23																		
24																		
25																		
26																		
27																		
28																		
29																		
30																		
31																		
32																		
33																		
34																		
41																		
42																		
43																		

Feuil2 | Feuil1

Il indique le nombre de CPUs utilisés par royaume pour notre plateforme.

Analyse des résultats

Le tableau final obtenu, on peut procéder à notre analyse.

Sur le tableau, on peut voir clairement notre consommation actuelle de CPUs pour chaque royaume:

- **0,36** CPUs pour le premier royaume (*customer-a*)
- **1,10** CPUs pour le second royaume (*customer-b*)

On était partis d'une hypothèse sur l'état d'avancement du remplissage des royaumes:

- Royaume **customer-a**, déployé à **10%**
- Royaume **customer-b**, déployé à **20%**

On peut alors extrapoler notre consommation finale une fois toute l'infrastructure supervisée.



Pour rappel, l'extrapolation n'est possible que si on rajoute le même type d'éléments (*dans les mêmes proportions*) dans le futur.

En partant du principe que l'on rajoute le même type d'éléments que ce que nous avons actuellement (*afin d'avoir les mêmes checks, et donc la même consommation CPU*), on arrive sur une consommation finale suivante:

- Royaume **customer-a**: **$0,36 \times 10 = 3.6$ CPUs**
- Royaume **customer-b**: **$1.10 \times 5 = 5.5$ CPUs**

On aura donc besoin au final par royaume d'une allocation de:

- **4** CPUs pour le (ou les) poller(s) du royaume **customer-a**
- **6** CPUs pour le (ou les) poller(s) du royaume **customer-b**