



Guide de maintenance

**XXII CORE REAL TIME
V.2.0.0**

Version guide - 1.0

Auteurs & historique des modifications.

Contributeurs

Auteurs	Postes
Nathan LAFONTAINE	Technical Manager
Stanislas MAHIEUX	Directeur de la formation

Historique du document

Auteurs	Date	Modifs
Nathan LAFONTAINE	5 Janvier 2023	Création du document
Stanislas MAHIEUX	10 Janvier 2023	Création du document

Sommaire.

Auteurs & historique des modifications.	2
Contributeurs	2
Historique du document	2
1. Introduction.	6
2. Le processor de debug.	6
2.1. Qu'est ce que le processor de debug?	6
2.2. Activation du processor de debug.	6
3. Logiciel d'orchestration.	8
3.1. Connexion en ssh à la machine	8
3.1.1 Depuis un PC Windows	8
3.1.2 Depuis un PC Linux	9
3.1.3 Depuis un PC MAC	9
3.2. Utilisation de Grafana	9
3.2.1. Introduction	9
3.2.2 Accéder à Grafana	10
3.2.3. Liste des sources par défaut et celles de XXII	10
3.2.3.1. Prometheus	10
3.2.3.2. Dashboard	12
3.2.3.4. Alertes	13
3.2.3.5 Loki	14
3.2.4. Les tableaux	16
3.2.4.1. Nodes	16
3.2.4.2. Node Exporter Full	16
3.2.4.3. Kubernetes / Compute Resources / Namespace (Pods)	17
3.2.4.4. Kubernetes / Compute Resources / Pod	18
3.2.4.5. Prometheus Alerts	18
3.3. K9S	19
3.3.1. Introduction à K9S	19
3.3.2. Présentation de l'interface	19
3.3.3. Les différents modes	20

3.3.3.1. Le mode commande	20
3.3.3.2. Le mode filtre	22
3.3.3.3. Le panneau des raccourcis	23
3.3.3.4. Afficher les logs	24
3.3.3.5. Detruire un pod	25
3.3.3.6. Décrire / Editer une ressource	26
3.3.3.7. Vues speciales	26

1. Introduction.

Ce document à pour but de vous accompagner à la maintenance de la solution XXII CORE REAL TIME en version V.2.X.X.

Version compatible avec ce guide:

XXII CORE REAL TIME CITY	V.2.0.0	V.2.0.1	V.2.1.0
XXII CORE REAL TIME INFRA	V.2.0.0	V.2.0.1	V.2.1.0
XXII CORE REAL TIME QSR	V.2.0.0	V.2.0.1	V.2.1.0

2. Le processor de debug.

2.1. Qu'est ce que le processor de debug?

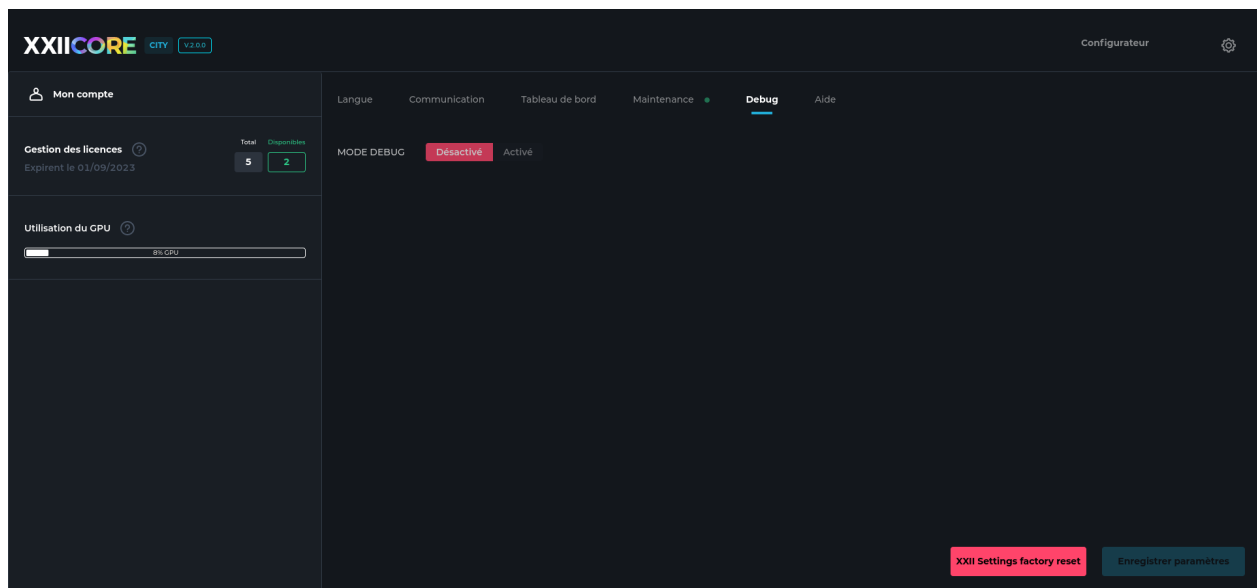
Le processor de debug est un composant interne à la plateforme permettant de les bounding box* sur la / les vidéo(s) en live sur une page dédiée.

** Elle permet de délimiter les objets reconnus par un rectangle sur l'image*

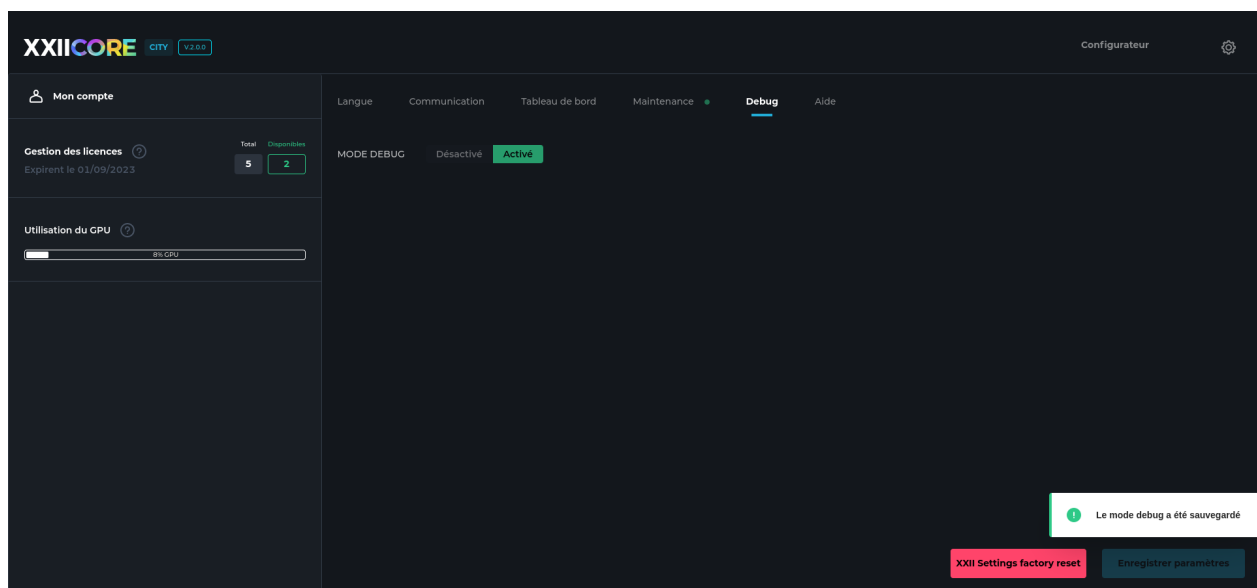
2.2. Activation du processor de debug.

Pour activer le processor de debug il faut dans un premier temps vous connecter à votre interface de configuration des compétences XXII en utilisant les liens suivants:

SMART CITY	SMART INFRA	QSR
http://smartcity.xxii-core.io/account/debug	http://smartinfra.xxii-core.io/account/debug	http://qsr.xxii-core.io/account/debug



Une fois sur cette page, cliquer alors sur **Activé** puis sur **Enregistrer paramètres**.



Une fois que le processor de debug est activé, il est important de désactiver TOUTES les compétences actives sur la plateforme, puis re-activer la compétence souhaité

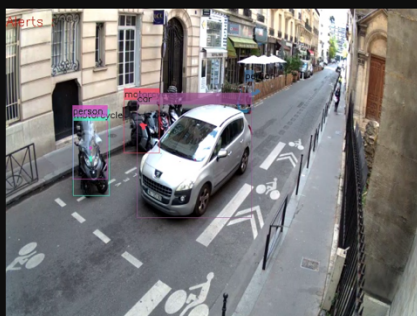
Une fois ces manipulations effectués vous pouvez alors vous rendre sur l'url dédiée au processor de debug :

SMART CITY	SMART INFRA	QSR
http://smartcity.debug.xxii-core.io/	http://smartinfra.debug.xxii-core.io/	http://qsr.debug.xxii-core.io/

Running tasks list

• 638673149a677d11010967 DETECTION : ☒ TRACKING : ☒ RULES : ☒ POST : ☒ FIRE_DETECTION : ☒ default

Vous arrivez alors sur cette page, vous voyez qu'une compétence tourne, si vous cliquez sur le lien, vous pouvez alors voir l'algo tourner en temps réel.



Une fois que vous avez fini d'utiliser le processor de debug, il est important de désactiver le processor de debug et d'activer et désactiver à nouveau la compétence.

3. Logiciel d'orchestration.

3.1. Connexion en ssh à la machine

3.1.1 Depuis un PC Windows

Pour se connecter en ssh depuis un PC sous Windows, ouvrez un terminal, puis utiliser la commande `ssh` suivie de l'adresse IP ou du nom d'hôte de l'ordinateur auquel vous voulez vous connecter, ainsi que votre nom d'utilisateur et votre mot de passe.

Par exemple:

```
ssh username@hostname
```

Vous serez alors invité à entrer votre mot de passe. Une fois connecté, vous serez en mesure de naviguer et de travailler sur l'ordinateur distant comme s'il s'agissait de votre propre ordinateur.

3.1.2 Depuis un PC Linux

Pour se connecter en ssh depuis un PC sous Linux, ouvrez un terminal, puis utiliser la commande `ssh` suivie de l'adresse IP ou du nom d'hôte de l'ordinateur auquel vous voulez vous connecter, ainsi que votre nom d'utilisateur et votre mot de passe.

Par exemple:

```
ssh username@hostname
```

Vous serez alors invité à entrer votre mot de passe. Une fois connecté, vous serez en mesure de naviguer et de travailler sur l'ordinateur distant comme s'il s'agissait de votre propre ordinateur.

3.1.3 Depuis un PC MAC

Pour se connecter en ssh depuis un PC sous MAC, ouvrez un terminal, puis utiliser la commande `ssh` suivie de l'adresse IP ou du nom d'hôte de l'ordinateur auquel vous voulez vous connecter, ainsi que votre nom d'utilisateur et votre mot de passe.

Par exemple:

```
ssh username@hostname
```

Vous serez alors invité à entrer votre mot de passe. Une fois connecté, vous serez en mesure de naviguer et de travailler sur l'ordinateur distant comme s'il s'agissait de votre propre ordinateur.

3.2. Utilisation de Grafana

3.2.1. Introduction

Grafana est un logiciel utilisé pour afficher les métriques qui sont stockées par d'autres logiciels. Il permet d'avoir une vue globale d'un cluster, d'un nœud, d'un pod.

Les explications suivantes sont valables pour *XXII CORE REAL TIME V.2.X.X*.

Cette page n'explique pas en détail comment créer un tableau de bord, mais vous donne des astuces pour vous aider.

3.2.2 Accéder à Grafana

Sur un de nos cluster, vous pouvez vous connecter à grafana avec le lien suivant (en adaptant votre fqdn) et en utilisant les identifiants suivant.

SMART CITY		SMART INFRA	QSR
http://smartcity.grafana.xxii-core.io		http://smartinfra.grafana.xxii-core.io	http://qsr.grafana.xxii-core.io
Identifiant:	admin		
Mot de passe:	prom-operator		

3.2.3. Liste des sources par défaut et celles de XXII

Dans notre solution nous utilisons actuellement:

- Prometheus (solution pour stocker les métriques et calculer les alertes).
- Loki (un scraper de log)

3.2.3.1. Prometheus

Prometheus est utilisé comme une base de données qui va stocker les métriques calculées par les applications déployées dans notre système. Il peut s'agir d'un composant spécifique (appelé "exportateur"), qui récupère des données et les stocke dans Prometheus pour permettre de créer un tableau de bord Grafana (Nvidia GPU, HiveMQ, MongoDB, Redis, ...), ou d'un composant interne qui stocke les données dans Prometheus.

Actuellement, les *Processors* et *MediaManager* stockent des données sur les images par seconde et la latence pour générer un tableau de bord.

Pour voir ce que prometheus stocke, cliquer sur les liens suivants:

SMARTCITY	SMARTINFRA
http://smartcity.prometheus.xxii-core.io/graph	http://smartinfra.prometheus.xxii-core.io/graph

Sur cette page, saisissez des requêtes pour avoir accès à des informations attendu.

Ici nous allons écrire « container_cpu_user_seconds_total » pour avoir des informations sur le CPU. Cliquer sur "Execute" pour afficher les informations souhaitées.

Enable query history

Try experimental React

container_cpu_user_seconds_total

Execute - insert metric at cursor -

Graph Console

◀ Moment ▶

Element

```

container_cpu_user_seconds_total{endpoint="https-metrics",id="7",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",node="node1",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="docker20e467468ec809be9817cd380d5bedde14367da92532105f7ecdb28d926900",image="nvidia/nvidia",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",name="nvidia",node="node1",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",node="node1",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods/besteffort",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",node="node1",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods/besteffort/pod18b08fca-4ad9-44c5-9841-d45211d8c4ac",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",namespace="monitoring",node="node1",pod="monitoring-prometheus-adapter-d8566d79-7w4z7",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods/besteffort/pod24b8c1c-34b6-47de-a3e5-52d11101a6fe",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",namespace="monitoring",node="node1",pod="monitoring-prometheus-operator-kube-state-metrics-4-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods/besteffort/pod4896366-d876-4333-8d6c-6218919cd244",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",namespace="monitoring",node="node1",pod="monitoring-metrics-server-59b5b8dd-mvw88",service="monitoring-prometheus-oper-kubelet"}
container_cpu_user_seconds_total{endpoint="https-metrics",id="kubepods/besteffort/pod4e266341-a351-4acb-9a5d-bd1a1e36d6fe",instance="192.168.30.31:10250",job="kubernetes",metrics_path="/metrics/cadvisor",namespace="kube-system",node="node1",pod="kubemeter-metrics-scraper-54fb4d595-kbpg4",service="monitoring-prometheus-oper-kubelet"}

```

Vous pouvez voir quelques informations de la requête.

Maintenant, vous pouvez afficher uniquement les informations sur l'espace de nom "default":

```
container_cpu_user_seconds_total{namespace="default"}
```

Pour le nom du pod contenant la plateforme:

```
container_cpu_user_seconds_total{namespace="default", pod=~"platform.*"}
```

En voyant les données, vous pouvez trouver quelles sont les clés disponibles pour la requête et adapter la regex:

```

container_cpu_user_seconds_total{container="POD",
endpoint="https-metrics",
id="/kubepods/burstable/pod0700a190-aa37-43a3-93ca-4c616e613adc/9a0d365da2ddb0a2ef07e92b04f0e3d1ace7fd18042de5e1c429afe9b0f94e86",
image="k8s.gcr.io/pause:3.2",
instance="192.168.30.31:10250",
job="kubernetes",
metrics_path="/metrics/cadvisor",
name="k8s_POD_platform-web-gateway-5ff9b97c4d-sffdb_default_0700a190-aa37-43a3-93ca-4c616e613adc_0",
namespace="default",
node="node1",
pod="platform-web-gateway-5ff9b97c4d-sffdb",
service="monitoring-prometheus-oper-kubelet"}

```

[RE2](#) est utilisé pour les regex.

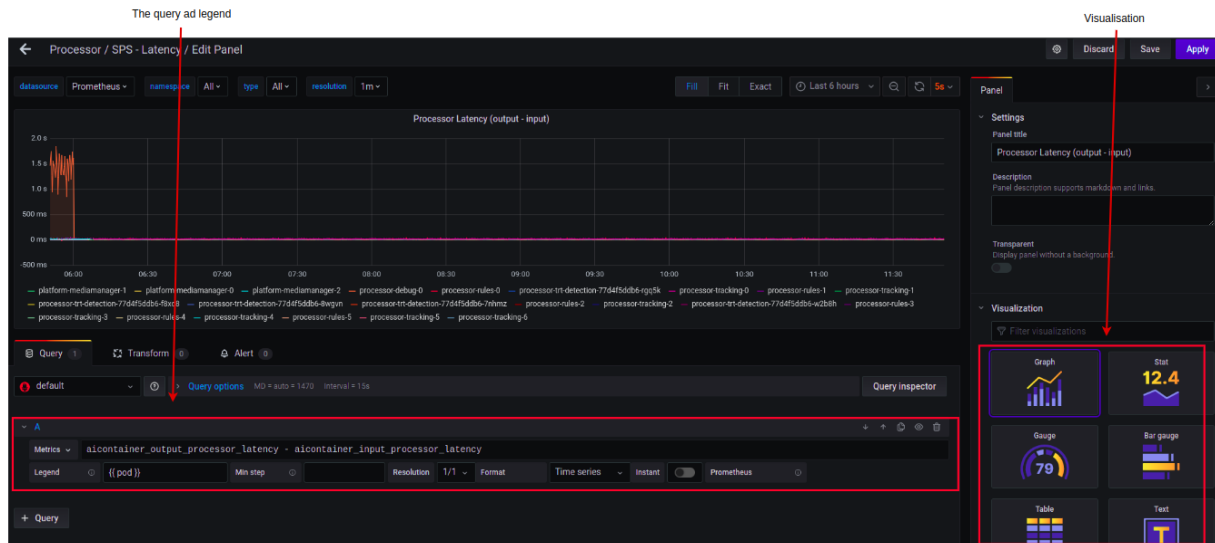
Sur chaque champ, vous pouvez effectuer une opération ou une fonction (taux, somme, ...) Pour plus d'informations, cliquez ici :

- <https://prometheus.io/docs/prometheus/latest/querying/basics/>
- <https://prometheus.io/docs/prometheus/latest/querying/examples/>
- <https://prometheus.io/docs/prometheus/latest/querying/functions/>

3.2.3.2. Dashboard

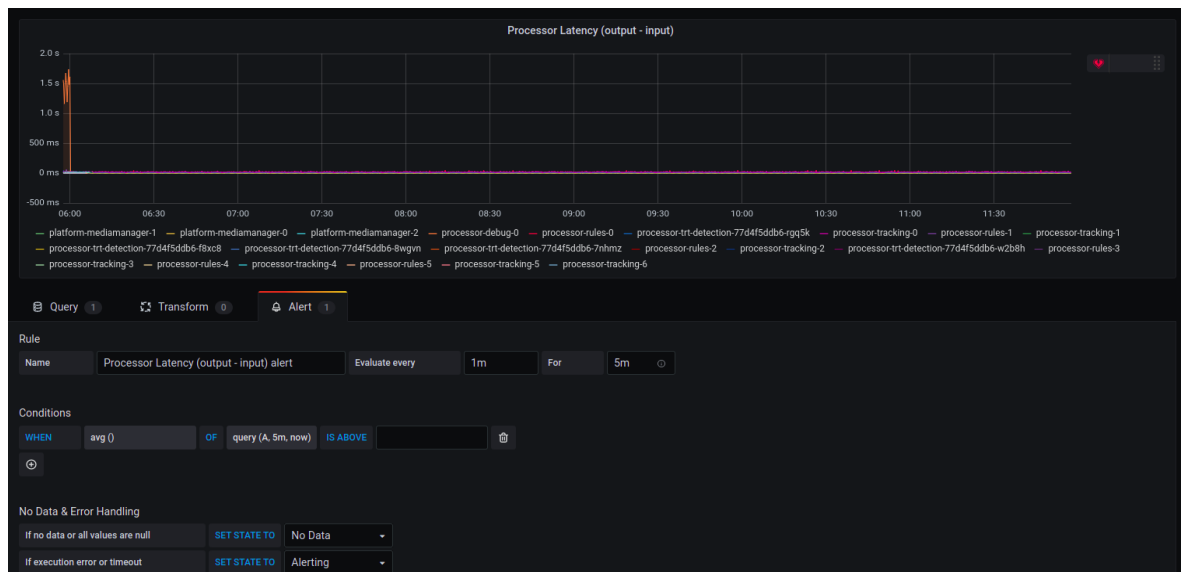


- Voir les panneaux affichés
 - Possibilité de sélectionner un élément en cliquant dessus dans la légende du panneau
 - Possibilité de sélectionner plusieurs éléments en utilisant CTRL
 - Possibilité de supprimer un élément en utilisant la touche CTRL si tous les éléments sont actuellement sélectionnés.
- Sélectionnez l'intervalle de temps
- Modifier les variables pour afficher ce qui vous intéresse
 - Ou ajouter des variables
<https://grafana.com/docs/grafana/v9.0/variables/variable-types/global-variables/>
- Modifier un panneau (sera réinitialisé à chaque F5, sauf si vous le sauvegardez, le mettez dans GIT et déployez une nouvelle version)
- Ajouter un panneau
- Modifier les paramètres (changer les noms, ajouter des variables, ...)
- Modifier un panneau (en cliquant sur son nom et ensuite sur modifier)



- Vous pouvez faire tous les tests que vous voulez, les appliquer car aucune modification ne sera sauvegardée sauf si un autre déploiement est effectué.

3.2.3.4. Alertes



Les alertes peuvent être définies dans un panneau

Vous pouvez définir une alerte, basée sur une requête. Ainsi, si vous souhaitez créer des alertes spécifiques, vous devez d'abord créer le panneau, puis ajouter les alertes.

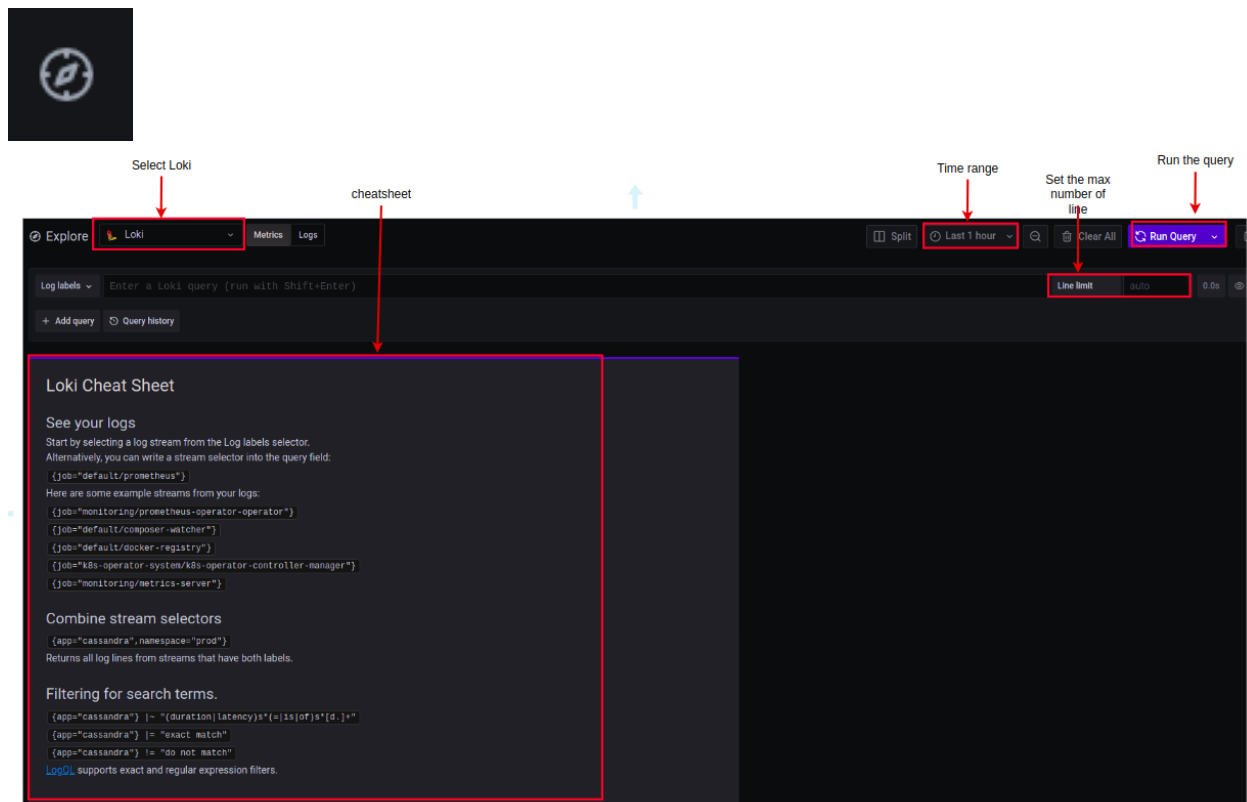
Certaines limitations existent, car vous pouvez utiliser des variables si vous voulez créer une alerte, donc parfois, vous devrez créer un panneau pour l'affichage, et un autre spécifiquement pour l'alerte.

Les alertes peuvent être :

- Affichées dans un panneau sous forme de liste (solution proposée)

3.2.3.5 Loki

Vous pouvez accéder à la vue de l'explorateur pour consulter les journaux en cliquant sur le logo de l'explorateur.

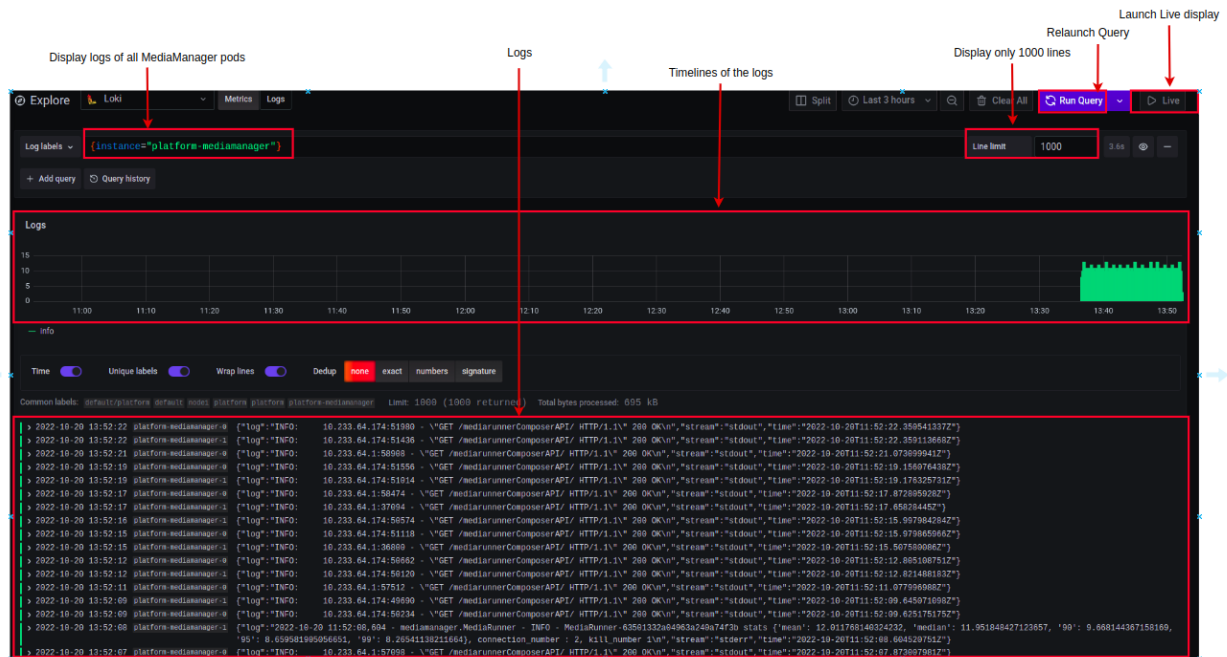


Vous pouvez sélectionner les journaux que vous voulez voir en cliquant sur "Log Labels".

- un pod : pod=xxx
- un service (groupe de pods en même temps): instance=xxx

Seul un nombre maximum de lignes sera affiché (basé sur les valeurs max, ou vos paramètres)

Pour le *MediaManager*:



Ici vous pouvez voir 1000 lignes, à cause du maximum réglé.

Un mode live existe pour afficher les nouveaux logs quand ils se produisent.

Vous pouvez zoomer sur une période spécifique en zoomant dans la ligne de temps.

Chaque log contient des métadonnées, vous pouvez cliquer dessus pour les afficher.

L'horodatage affiché n'est pas l'horodatage des logs mais l'horodatage du scrap. Si vous voulez voir l'horodatage du log, vous devez regarder dans le message directement.

Si vous voulez voir seulement la ligne avec le mot clé "stats" :

```
{job=~"default/platform"} |= "stats" | logfmt
```

Il est également possible de créer un tableau de bord affichant l'enregistrement dans le temps.

```
sum(count_over_time({job=~"default.+"}[5m])) by (instance)
```

3.2.4. Les tableaux

3.2.4.1. Nodes

Ce tableau de bord permet d'avoir une vision globale d'un serveur sur les éléments suivants :

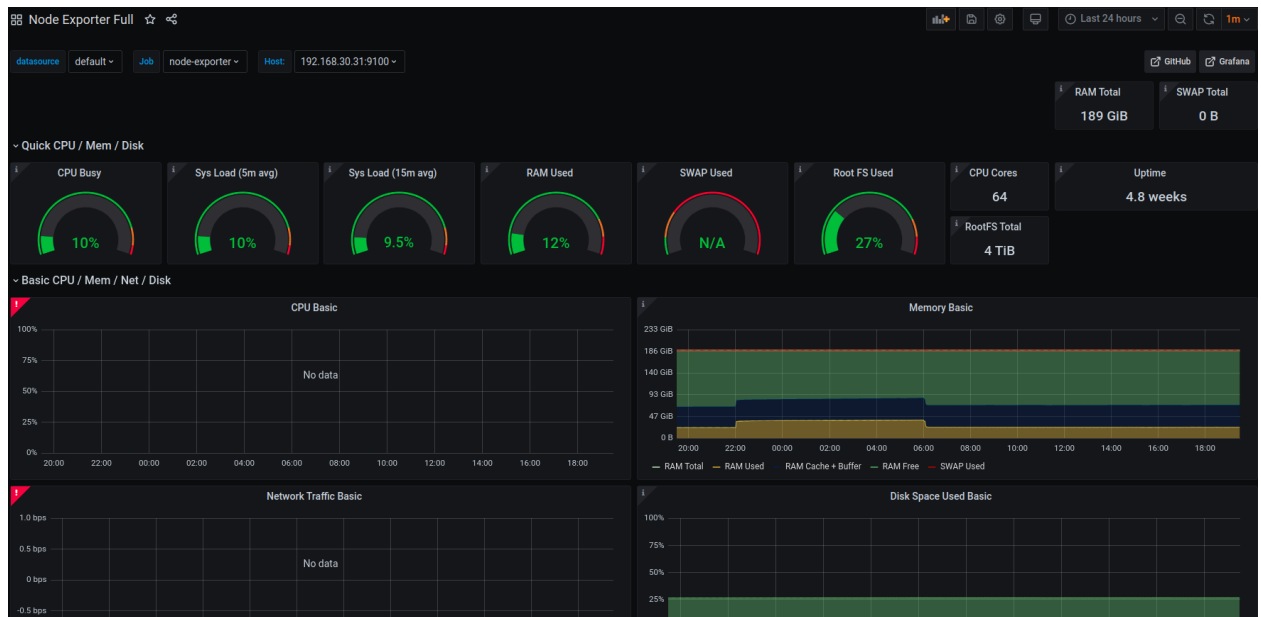
- CPU
- RAM
- Disques
- Réseau



3.2.4.2. Node Exporter Full

Pour ajouter : <https://grafana.com/grafana/dashboards/1860-node-exporter-full/>

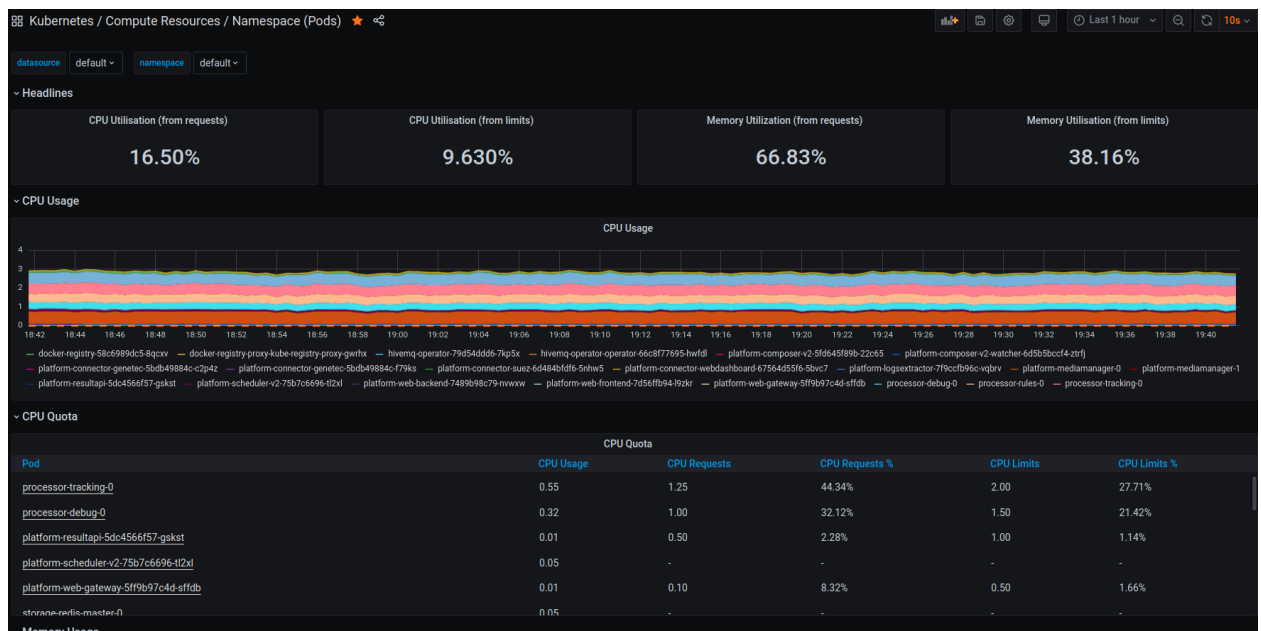
Afficher approximativement toutes les informations sur le serveur



3.2.4.3. Kubernetes / Compute Resources / Namespace (Pods)

Permet d'afficher des informations pour un espace de nom donné, et donc de sélectionner un ou plusieurs pods à afficher :

- CPU
- RAM
- Réseau



3.2.4.4. Kubernetes / Compute Resources / Pod

Permet de sélectionner un espace de nom et un pod.

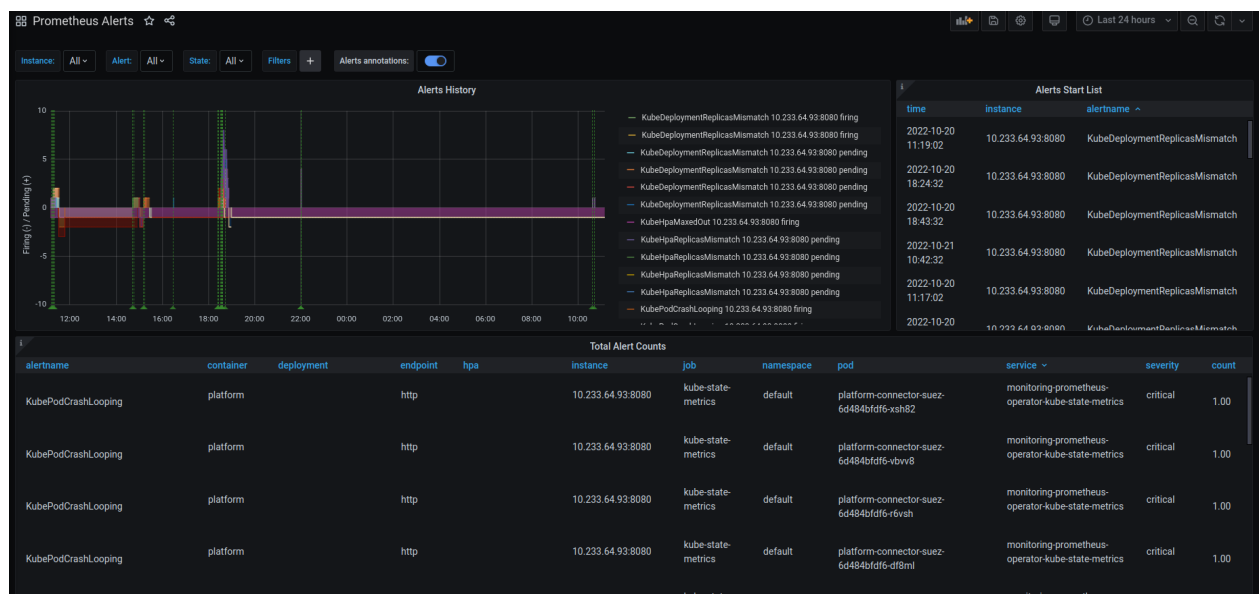
Vous pouvez voir :

- Consommation du CPU (seuils de demande et de limite)
- Restriction du CPU (si la restriction a lieu, la demande donnée n'est pas suffisante).
- Consommation de RAM (seuils de demande et de limite)
- Réseau



3.2.4.5. Prometheus Alerts

Afficher les alertes déclenchées dans prometheus pendant la période définie.



3.3. K9S

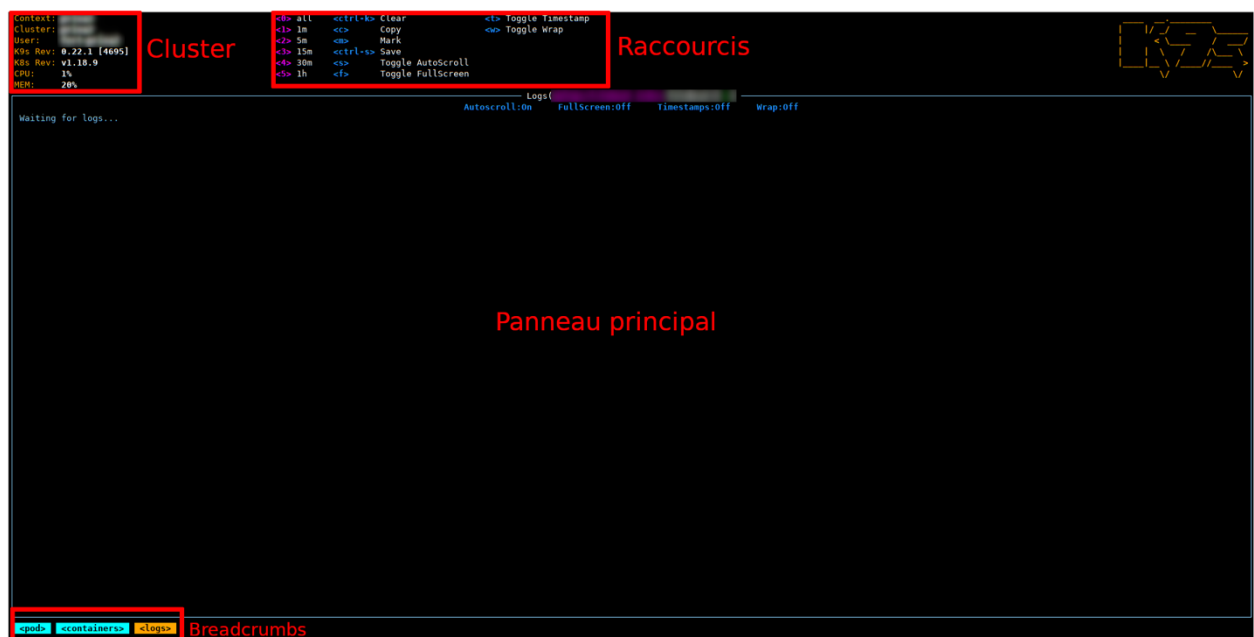
3.3.1. Introduction à K9S

K9S est un projet écrit en Go et publié en tant que binaire exécutable.

Pour interagir avec un cluster, *k9s* lit les informations du fichier `~/.kube/config`. Il vous suffit de lancer *k9s* une fois connecté en [ssh](#) depuis votre terminal et ce dernier utilisera automatiquement votre contexte Kubernetes courant.

3.3.2. Présentation de l'interface

Voici maintenant un aperçu de l'interface de *k9s* qui en résume les principales zones d'intérêt :



- En haut à gauche, quelques informations globales données par *k9s* telles que le nom du cluster auquel vous êtes connecté, le contexte actuellement utilisé, votre userid, votre version de *k9s*, etc.
- À droite, vous trouverez une liste de raccourcis utiles pour interagir avec *k9s*. Gardez dans un coin de votre tête que son contenu change en fonction de la vue active, nous y reviendrons plus tard. Notez aussi que la touche ? ouvrira l'aide complète à tout moment.
- Au milieu, on retrouve le panneau principal, c'est ici que se trouveront toutes les informations sur nos ressources. Son contenu changera lui aussi en fonction de la vue active et le nom de cette dernière vous sera rappelé dans le titre du panneau principal.

- Enfin tout en bas à gauche, un *fil d'Ariane* (ou *breadcrumbs*) nous donne la hiérarchie de la vue active et indique vers quelle vue nous retournons lorsqu'on la quitte.

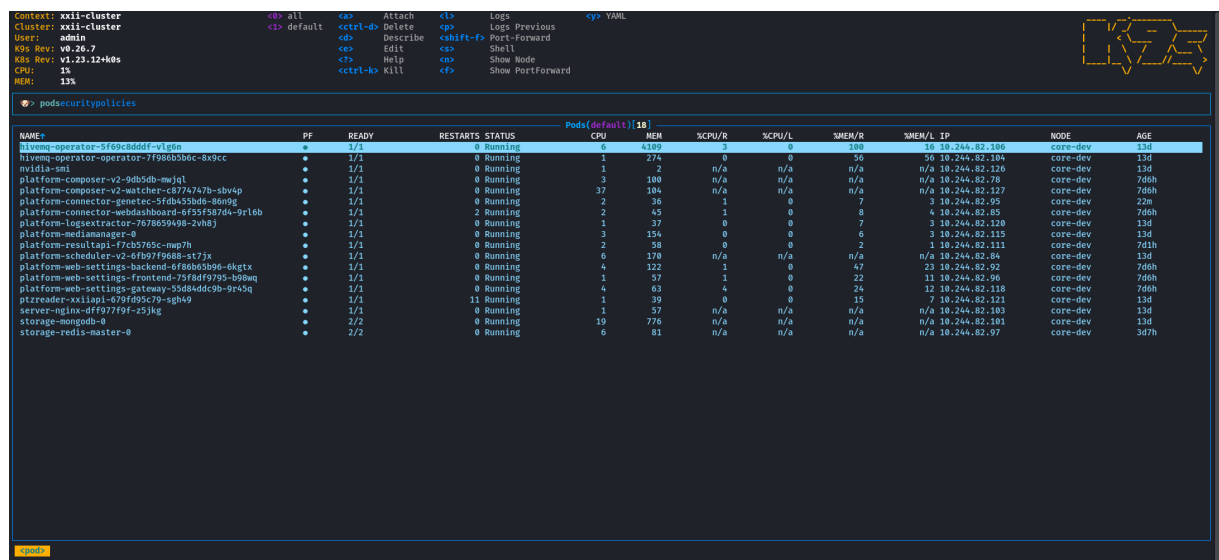
D'une manière générale, k9s reprend plusieurs principes de vim. La touche `esc` sert à revenir en arrière ou à annuler / fermer un prompt et on quitte le programme avec `:q` (ou `ctrl + c`).

Lorsque vous serez parfaitement à l'aise avec k9s, vous pourrez masquer le panneau principal en pressant `ctrl + e` afin d'économiser un peu de place, mais pour l'instant je vous invite à le laisser affiché.

3.3.3. Les différents modes

3.3.3.1. Le mode commande

La touche `:` ouvre un prompt chien (`>`) au-dessus du panneau principal. C'est le mode de k9s qui permet d'exécuter des commandes. Il existe une commande par ressource kubernetes plus quelques autres spécifiques que nous verrons plus tard. Par exemple, `:pod` bascule dans la vue *Pod* et ce sont désormais les informations sur les pods qui sont affichées dans le panneau principal.



k9s fait de l'autosuggestion pour le nom des commandes. Quand je tappe `pod`, il me propose la commande `poddisruptionbudget`.

- Les touches `↑` et `↓` permettent de naviguer circulairement entre les différentes propositions
- Les touches `→`, `tab` ou `ctrl + f` permettent de compléter automatiquement la suggestion dans le prompt (autocomplétion), sans toutefois exécuter la commande

- Les touches ctrl + u ou ctrl + w permettent d'effacer la ligne sans quitter le mode commande

La liste complète des commandes est disponible par le raccourci ctrl + a ou avec :aliases.

RESOURCE	COMMAND	ALIASES (ctrl + w)	API GROUP
Alertmanagers	alertmanager, alertmanagers, monitoring.coreos.com/v1/alertmanagers		monitoring.coreos.com
aliases	a, alias, aliases		
apiservices	apiregistration.k8s.io/v1/apiservices, apiservice, apiservices		apiregistration.k8s.io
apmservers	apm.k8s.elastic.co/v1/apmservers, apmserver, apmservers		apm.k8s.elastic.co
beats	beat, beat.k8s.elastic.co/v1beta1/beats, beats		beat.k8s.elastic.co
benchmarks	be, bench, benchmark, benchmarks		
bindings	binding, bindings, v1/bindings		
certificatesigningrequests	certificates.k8s.io/v1beta1/certificatesigningrequests, certificatesigningrequest, certificatesigningrequests, csr		certificates.k8s.io
ciliumclusterwideenetworkpolicies	cnpp, cilium.io/v2/ciliumclusterwideenetworkpolicies, ciliumclusterwideenetworkpolicies, ciliumclusterwideenetworkpolicy		cilium.io
ciliumendpoints	cep, cilium.io/v2/ciliumendpoints, ciliumendpoints, ciliumendpoints, ciliumep		cilium.io
ciliumidentities	cilium.io/v2/ciliumidentities, ciliumid, ciliumidentities, ciliumidentity		cilium.io
ciliumnetworkpolicies	cilium.io/v2/ciliumnetworkpolicies, ciliumnetworkpolicies, ciliumnetworkpolicy, ciliumnp, cnp		cilium.io
ciliumnodes	cilium.io/v2/ciliumnodes, cilium, ciliumnode, ciliumnodes, cn		cilium.io
clusterrolebindings	clusterrolebinding, clusterrolebindings, crb, rbac.authorization.k8s.io/v1/clusterrolebindings		rbac.authorization.k8s.io
clusterroles	clusterrole, clusterroles, cr, rbac.authorization.k8s.io/v1/clusterroles		rbac.authorization.k8s.io
clusterworkflowtemplates	argoproj.io/v1alpha1/clusterworkflowtemplates, clusterwftmpl, clusterworkflowtemplate, clusterworkflowtemplates, cwt		argoproj.io
componentstatuses	componentstatus, componentstatuses, cs, v1/componentstatuses		
configmaps	cm, configmap, configmaps, v1/configmaps		
contexts	context, contexts, ctx		
controllerrevisions	apps/v1/controllerrevisions, controllerrevision, controllerrevisions		apps
cronjobs	batch/v1beta1/cronjobs, cj, cronjob, cronjobs		batch
cronworkflows	argoproj.io/v1alpha1/cronworkflows, cronwf, cronworkflow, cronworkflows, cwf		argoproj.io

On notera que toutes les commandes disposent d'un alias plus court qui permet d'accélérer la navigation (e.g. :a au lieu de :aliases)

Si vous entrez une commande qui n'existe pas k9s vous le fera savoir avec une cowsay qui malheureusement ne partira pas avec la touche esc. Pour sortir de cette vue, il faut rejouer une commande valide comme :pod ↵ par exemple.

Context: xxii-cluster
Cluster: xxii-cluster
User: admin
K9s Rev: v0.26.7
K9s Rev: v1.23.12+k8s
CPU: 1%
MEM: 13%


<= all <= Attach <= Logs <= YARL
<= default <= Delete <= Logs Previous
<= Describe <= Port-Forward
<= Edit <= Shell
<= Help <= Show Node
<= Kill <= Show Portforward

NAME	PF	READY	RESTARTS	STATUS	CPU	MEM	%CPU/R	%CPU/L	%MEM/R	%MEM/L	IP	NODE	AGE
hiveq-operator-5f69c8dddf-vl6gn	•	1/1	0	Running	6	4108	3	0	100	16	10.244.82.186	core-dev	13d
hiveq-operator-5f69c8dddf-vl6gn	•	1/1	0	Running	1	274	0	0	56	56	10.244.82.184	core-dev	14d
nvidia-smi	•	1/1	0	Running	1	2	n/a	n/a	n/a	n/a	10.244.82.126	core-dev	13d
platform-composer-v2-9db5db-mjq1	•	1/1	0	Running	4	100	n/a	n/a	n/a	n/a	10.244.82.78	core-dev	7d7h
platform-composer-v2-watcher-c877474b-sb4p	•	1/1	0	Running	28	104	n/a	n/a	n/a	n/a	10.244.82.127	core-dev	7d7h
platform-connector-genetec-5f6b455bdc-86n9g	•	1/1	0	Running	2	36	1	0	7	3	10.244.82.95	core-dev	24m
platform-connector-webdashboard-6f5f387d4-9rl6b	•	1/1	2	Running	1	45	0	0	8	4	10.244.82.85	core-dev	7d7h
platform-logextractor-7878859498-2vbnj	•	1/1	0	Running	1	37	0	0	7	3	10.244.82.128	core-dev	13d
platform-mediananager-0	•	1/1	0	Running	3	154	0	0	6	3	10.244.82.115	core-dev	13d
platform-resultapi-f7cb5765c-mp7h	•	1/1	0	Running	3	58	0	0	2	1	10.244.82.111	core-dev	7d1h
platform-scheduler-v2-6f897f9688-st7jk	•	1/1	0	Running	1	1	0	0	n/a	n/a	10.244.82.84	core-dev	13d
platform-web-settings-backend-6f86b5b96-8kgtx	•	1/1	0	Running	1	1	0	0	67	23	10.244.82.92	core-dev	7d6h
platform-web-settings-frontend-75f8df9793-b98wq	•	1/1	0	Running	1	1	0	0	22	11	10.244.82.96	core-dev	7d6h
platform-web-settings-gateway-5d84ddc9b-9r45q	•	1/1	0	Running	1	1	0	0	24	12	10.244.82.118	core-dev	7d6h
ptreader-xxiiapi-079f4d98c79-gh49	•	1/1	0	Running	1	1	0	0	15	7	10.244.82.121	core-dev	13d
server-nginx-diff977f9f-25jkg	•	1/1	0	Running	1	1	0	0	n/a	n/a	10.244.82.103	core-dev	14d
storage-mongodb-0	•	2/2	0	Running	1	1	0	0	n/a	n/a	10.244.82.101	core-dev	14d
storage-redis-master-0	•	2/2	0	Running	1	1	0	0	n/a	n/a	10.244.82.97	core-dev	3d7h

< Ruroh? 'xxii' command not found >

Dismiss

3.3.3.2. Le mode filtre

La touche / ouvre un prompt caniche  qui permet d'appliquer un filtre sur la liste d'éléments affichés dans le panneau. C'est le mode filtre de k9s.

En reprenant notre exemple, si l'on est dans la vue pod (:pod) et que l'on tape /processor on n'affichera que les *Pods* qui contiennent processor dans leur nom.

```
Context: xxii-cluster
Cluster: xxii-cluster
User: admin
K9s Rev: v0.26.7
K8s Rev: v1.23.12+k8s
CPU: 1%
MEM: 13%

<u>all</u>  <u>Attach</u>  <u>Logs</u>  <u>YAML</u>
<u>default</u>  <u>Delete</u>  <u>Logs Previous</u>
               <u>Describe</u>  <u>Port-Forward</u>
               <u>Edit</u>  <u>Shell</u>
               <u>Help</u>  <u>Show Node</u>
               <u>Kill</u>  <u>Show PortForward</u>

processor

Pod(s)all|1|<u>processor</u>
NAMESPACE+  NAME  PF  READY  RESTARTS STATUS  CPU  MEM  %CPU/R  %CPU/L  %MEM/R  %MEM/L IP  NODE  AGE
Default  platform-mediananager-0  *  1/1  0 Running  3  154  0  0  6  3  10.244.82.115  core-dev  13d
```

On peut aussi filtrer les ressources qui portent un label kubernetes avec la syntaxe `/-l <nom_du_label>=<valeur_du_label>`. Dans notre exemple `/-l k8s-app=xxii` liste tous les pods qui portent le label `k8s-app=xxii` (notez bien qu'on filtre uniquement sur l'existence de label ici et pas sur le nom des pods.)

Ce prompt fonctionne dans *toutes* les vues de *k9s* et applique le filtre sur tout ce qui est affiché dans le panneau principal. Il n'est pas réservé aux vues des ressources Kubernetes. Voyez un exemple dans la vue aliases.

3.3.3.3. Le panneau des raccourcis

Intéressons-nous de plus près au panneau des raccourcis. Il contient tout ce qu'il vous faut pour vos opérations courantes. Rappelez-vous que son contenu change en fonction de la vue dans laquelle vous êtes, alors pensez à vous y référer régulièrement 💡. Voici par exemple le contenu des raccourcis pour la vue pod :

<0>	all	<a>	Attach	<l>	Logs
<1>	ppinfra	<ctrl-d>	Delete	<shift-l>	Logs Previous
<2>	default	<d>	Describe	<shift-f>	Port-Forward
		<e>	Edit	<s>	Shell
		<?>	Help	<f>	Show PortForward
		<ctrl-k>	Kill	<y>	YAML

Vous pouvez à tout moment appuyer sur la touche ? pour afficher l'aide complète.

RESOURCE	GENERAL	NAVIGATION	HELP	Aliases
<0>	all	<j>	Down	
<1>	ppinfra	<shift-g>	Goto Bottom	Aliases
<2>	default	<q>	Goto Top	Help
<a>	Attach	<tab>	Left	
<ctrl-d>	Copy	<shift-tab>	Page Down	
<d>	Delete	</term>	Page Up	
<e>	Describe	<space>	Right	
<ctrl-k>	Edit	<ctrl-l>	Up	
<?>	Help	<ctrl-space>		
<l>	Kill	<q>		
<shift-l>	Logs	<ctrl-r>		
<shift-f>	Logs Previous	<ctrl-s>		
<ctrl-f>	Port Forward	<ctrl-o>		
<ctrl-r>	Refresh			
<s>	Shell			
<f>	Show PortForward			
<ctrl-y>	Sort %CPU (LIM)			
<shift-y>	Sort %CPU (REQ)			
<ctrl-p>	Sort %MEM (LIM)			
<shift-p>	Sort %MEM (REQ)			
<shift-a>	Sort Age			
<shift-c>	Sort CPU			
<shift-i>	Sort IP			
<shift-m>	Sort MEM			
<shift-n>	Sort Name			
<shift-p>	Sort Namespace			
<shift-q>	Sort Node			
<shift-r>	Sort Ready			
<shift-t>	Sort Restart			
<shift-u>	Sort Status			
<ctrl-e>	Toggle Faults			
<ctrl-w>	Toggle Wide			
<enter>	View			
<y>	YAML			

Globalement, un raccourci s'applique à la ligne du panneau principal en surbrillance et vous pouvez changer de ligne en utilisant les flèches de votre clavier.

NAMESPACE	NAME
kube-system	cilium-operator
kube-system	cluster-autoscaler
kube-system	coredns
kube-system	coredns-autoscaler
kube-system	dns-controller
kube-system	metrics-server
monitoring	blackboxexporter
monitoring	cloudwatch-exporter
monitoring	grafana
monitoring	kube-state-metrics
monitoring	prometheus-operator
monitoring	thanos-externalizer
monitoring	thanos-querier
observability	jaeger-collector
observability	jaeger-operator
observability	jaeger-query
observability	jaeger-storage
ppargo	argo-server

Les touches numériques affichées en magenta dans le panneau des raccourcis permettent de filtrer le contenu du panneau principal sur la base d'un paramètre qui change d'une vue à l'autre. Pour les pods par exemple, ces touches changent le namespace, mais pour les logs elles changent la plage temporelle à afficher. La valeur du paramètre est rappelée en magenta dans le titre du panneau principal

Dans l'exemple ci-dessus, mon écran se vide lorsque je sélectionne le namespace *default* (<2>) car je n'ai pas de pod sur ce namespace. Pensez donc à vérifier régulièrement la valeur de ces paramètres.

Vous avez maintenant toutes les billes pour commencer à faire des opérations sur vos ressources. Voici celles que j'utilise le plus fréquemment.

3.3.3.4. Afficher les logs

Dans la vue :pod, la touche l permet d'afficher des logs du pod sélectionné. Lorsque qu'un pod n'écrit pas beaucoup de logs, on est généralement confronté au message Waiting for logs....

Cette vue n'affiche en effet que les logs avec un timestamp vieux de maximum 1 minute et quand il n'y en a pas, elle attend alors qu'un log assez jeune arrive. Si vous cherchez des logs plus vieux, vous pouvez changer la plage temporelle avec les touches 0 à 5. La durée sélectionnée est rappelée dans le titre du panneau principal représenté par la flèche rouge dans l'image ci-dessous.

N'oubliez pas également que > fonctionne ici également.



```

Context: xali-cluster
Cluster: xali-cluster
User: admin
Rev: v0.26.7
Hss Rev: V1.23.12-hss
GPU: 1%
Node: 1%

```

```

<> all
<> default
<> Describe
<> Edit
<> Help
<> Kill

```

```

Alcnd
Delete
Log Previous
Port-Forward
Shell
Show Node
Show Port-Forward

```

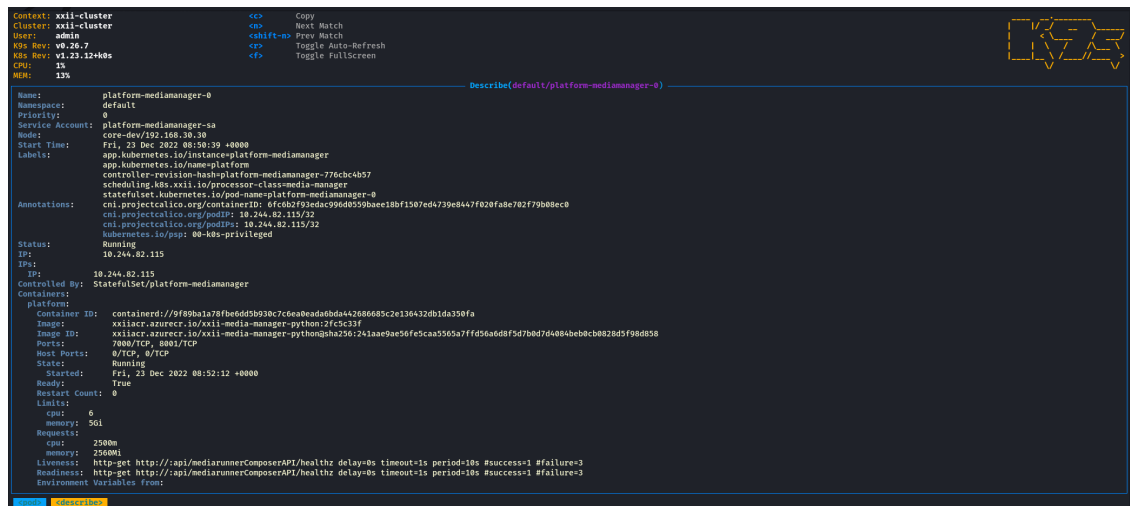
PODS (all) | 581

NAME	IP	READY	RESTARTS	STATUS	CPU	MEM	%CPU	%MEM	SHM/P	IP	NODE	AGE
cattle-fleet-local-system	fleet-agent-0bdc8699de-zm58	● 1/1	0	Running	2	79	n/a	n/a	n/a	n/a	10.244.82.94	core-dev 14d
cattle-fleet-system	fleet-controller-7bd989d4f4-zjrf5	● 1/1	0	Running	14	63	n/a	n/a	n/a	n/a	10.244.82.90	core-dev 14d
cattle-fleet-system	sls-lb-6dbb7b49-gvrc	● 1/1	0	Running	1	6	n/a	n/a	n/a	n/a	10.244.82.69	core-dev 14d
cattle-system	rancher-785dd4dc78-h8sc	● 1/1	0	Running	31	888	n/a	n/a	n/a	n/a	10.244.82.82	core-dev 14d
cert-manager	rancher-webhook-8f8853f4-7xtq	● 1/1	0	Running	8	52	n/a	n/a	n/a	n/a	10.244.82.99	core-dev 14d
cert-manager	cert-manager-0799dc0c-w33n	● 1/1	0	Running	1	6	n/a	n/a	n/a	n/a	10.244.82.79	core-dev 14d
cert-manager	cert-manager-cainjector-38c9955dc-5vbkj	● 1/1	0	Running	1	66	n/a	n/a	n/a	n/a	10.244.82.80	core-dev 14d
cert-manager	cert-manager-webhook-469d4885-tvqc	● 1/1	0	Running	4	59	n/a	n/a	n/a	n/a	10.244.82.83	core-dev 14d
default	hivey-operator-5f69c6d6f-vlgo	● 1/1	0	Running	7	409	3	0	300	10	10.244.82.110	core-dev 14d
default	hivey-operator-79786b0bc-8xvc	● 1/1	0	Running	1	274	0	0	56	56	10.244.82.104	core-dev 14d
default	nvidia-sm	● 1/1	0	Running	1	2	n/a	n/a	n/a	n/a	10.244.82.126	core-dev 14d
default	platform-composer-2-0db0db-mjql	● 1/1	0	Running	1	2	n/a	n/a	n/a	n/a	10.244.82.77	core-dev 767h
default	platform-composer-2-watcher-c8774747b-shvq	● 1/1	0	Running	1	n/a	n/a	n/a	n/a	n/a	10.244.82.127	core-dev 767h
default	platform-connector-guest-3f4b5b0dc-hong	● 1/1	0	Running	0	0	0	0	0	0	10.244.82.95	core-dev 767h
default	platform-connector-webdavshard-6f5f587d4-gr	● 1/1	0	Running	0	0	0	0	0	0	10.244.82.85	core-dev 767h
default	platform-logextractor-7d7885b0c-zvnh	● 1/1	0	Running	2	1	7	3	124	124	10.244.82.120	core-dev 14d
default	platform-mediawiki	● 1/1	0	Running	2	3	13	24	124	124	10.244.82.113	core-dev 767h
default	platform-realmapi-7fcb576c-mwpb	● 1/1	0	Running	0	0	0	0	0	0	10.244.82.111	core-dev 761h
default	platform-scheduler-2-4a9f008c-3tj3	● 1/1	0	Running	2	0	0	0	0	0	10.244.82.86	core-dev 767h
default	platform-web-settings-backend-6f8b05b96-ekgt	● 1/1	0	Running	2	0	0	0	0	0	23.10.244.82.92	core-dev 766h
default	platform-web-settings-frontend-75fdff975-b9b	● 1/1	0	Running	1	37	n/a	n/a	n/a	n/a	10.244.82.93	core-dev 766h
default	platform-web-settings-gateway-3a0ddc9b-9v4q	● 1/1	0	Running	1	39	0	0	15	7	10.244.82.123	core-dev 14d
default	ptzserver-xaliapi-679f69c79-gsh4u	● 1/1	11	Running	1	39	0	0	15	7	10.244.82.123	core-dev 14d
default	server-nginx-df077f9f-z5jk	● 1/1	0	Running	1	37	n/a	n/a	n/a	n/a	10.244.82.103	core-dev 14d
default	storage-mongodb-8	● 1/1	22	Running	22	776	n/a	n/a	n/a	n/a	10.244.82.97	core-dev 3d7h
default	storage-redis-master-0	● 1/1	0	Running	0	82	n/a	n/a	n/a	n/a	10.244.82.97	core-dev 14d
gpu-operator	gpu-feature-discovery-dm6c	● 1/1	0	Running	1	45	n/a	n/a	n/a	n/a	10.244.82.77	core-dev 14d
gpu-operator	gpu-operator-77787507c-f54mg	● 1/1	0	Running								

2023 - All rights reserved to XXII GROUP. 25

3.3.3.6. Décrire / Editer une ressource

Il est parfois intéressant de lire la description d'une ressource pour vérifier sa liste de labels par exemple. Pour cela on peut soit utiliser la touche d qui retourne la description formatée par *k9s*, soit la touche y pour obtenir le fichier YAML brut. Dans ces vues, lorsque l'on utilise le mode filtre et qu'il correspond à plusieurs résultats, on pourra naviguer vers l'occurrence suivante avec la touche n et l'occurrence précédente avec shift+n (pensez bien à valider le filtre avec ↵ avant d'appuyer sur ces touches).



Il est également possible d'éditer le fichier YAML correspondant à une ressource en appuyant avec e. Cela ouvrira votre éditeur de texte par défaut et appliquera les modifications une fois le fichier sauvegardé.

3.3.3.7. Vues speciales

Pulses est un dashboard global qui permet de voir l'état de santé de votre cluster. Il affiche le nombre de ressources OK / KO au cours du temps et permet via les touches numériques ou tab de passer rapidement dans la vue la plus pertinente pour surveiller un paramètre. On active sa vue avec :pulses

