UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

université PARIS-SACLAY

#11

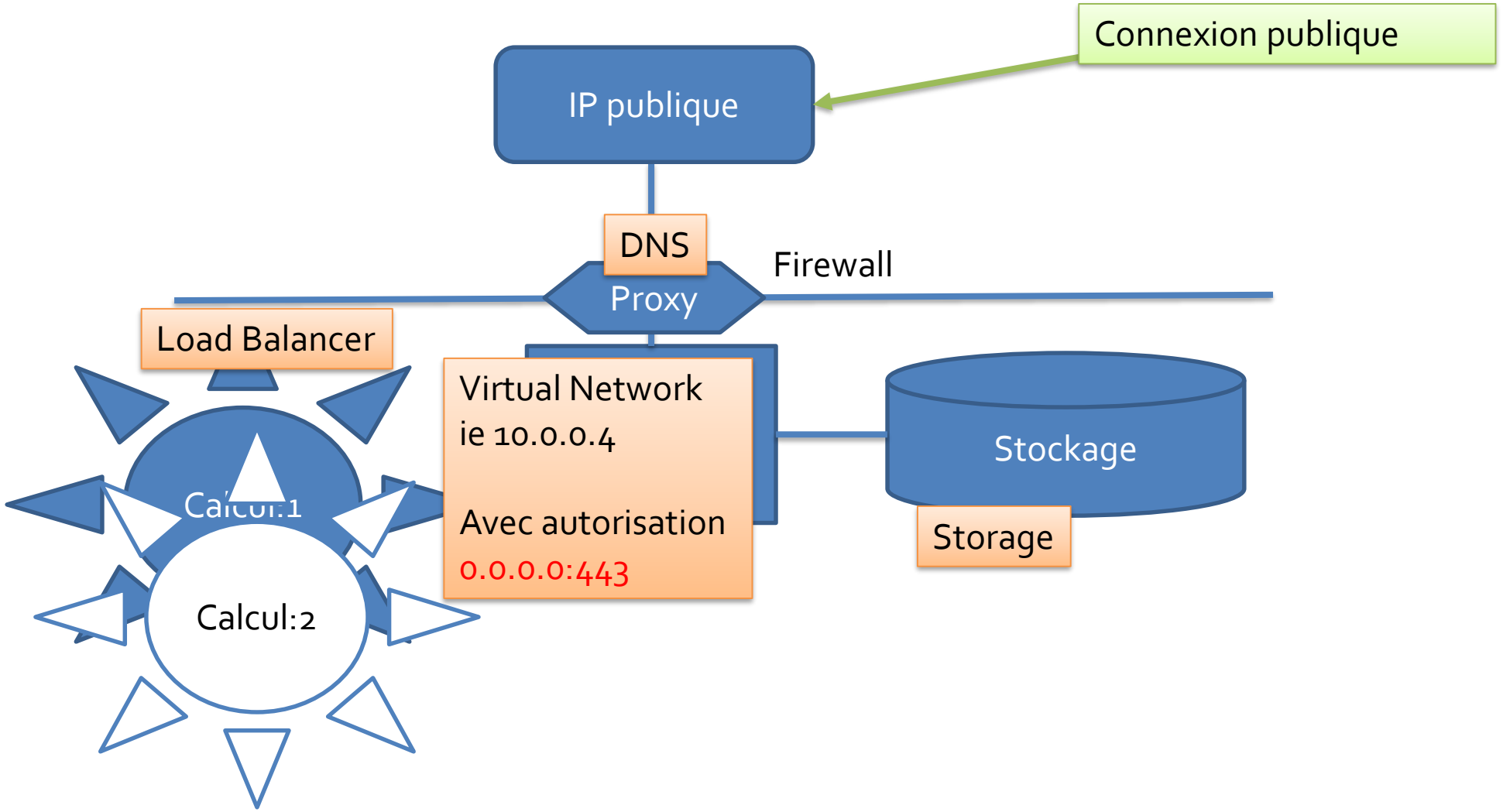
20/02/2026

jean-michel.batto@cea.fr

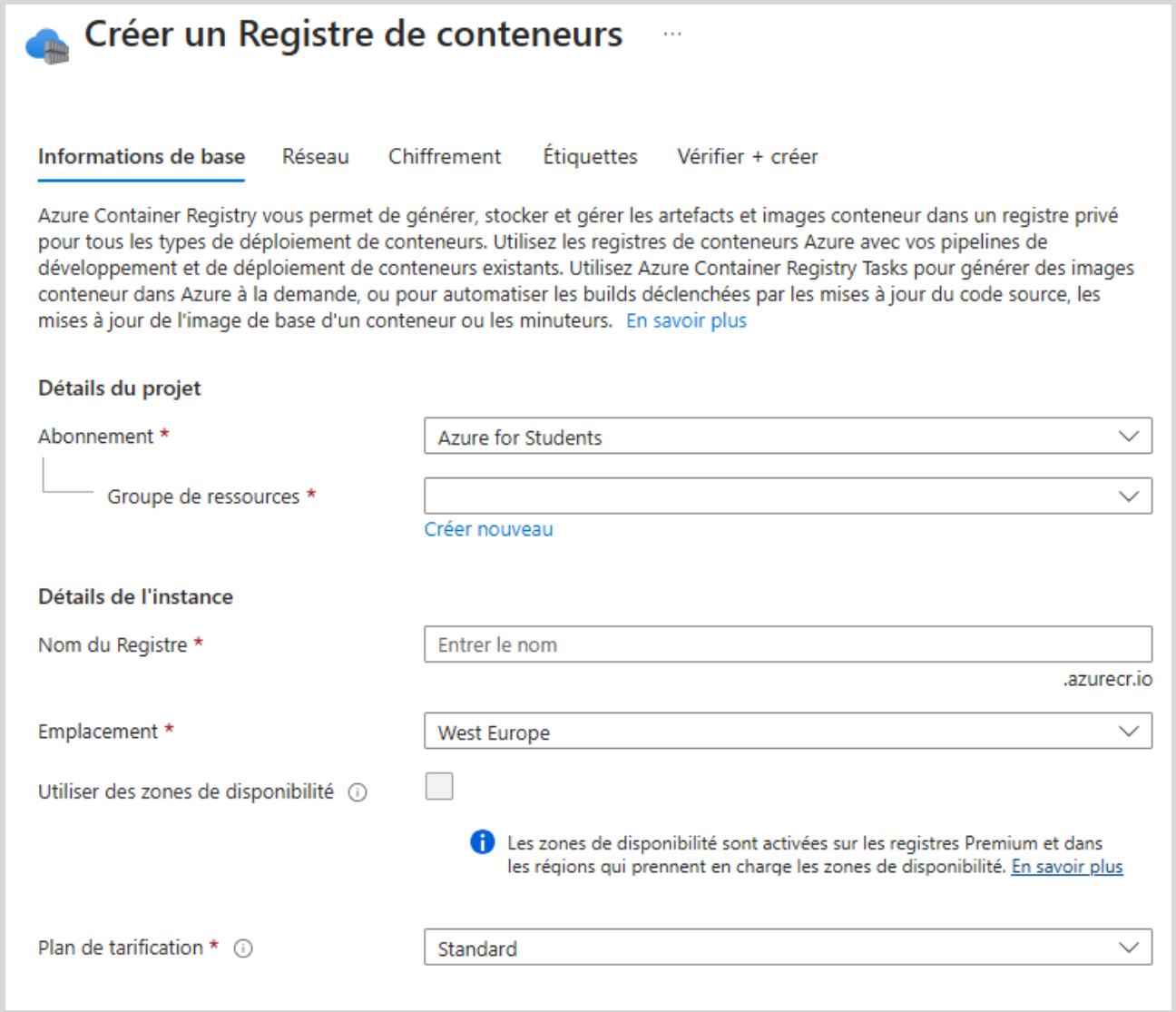
cea

https://gogs.eldarsoft.com/M2_IHPS

Le cloud : les services indispensables



Azure registry



Construire une image dans la registry Azure

1/vous avez une registry tenetreg (le nom de votre choix…) attention choisir l'option

2/vous avez installé le client azure sur votre machine

```
apt-get install azure-cli
az login
```

3/on doit se loguer dans sa registry sinon pas de push

```
az acr login -n tenetreg
```

4/on récupère un repo d'intérêt

```
git clone https://github.com/philhawthorne/docker-influxdb-grafana.git
```

5/construction de l'image Docker ➔ attention cet exemple ne fonctionne plus…

```
docker build -t docker-influxdb-grafana:latest .
```

6/on change les noms des images

```
docker tag docker.io/library/docker-influxdb-grafana tenetreg.azurecr.io/docker-influxdb-grafana
docker tag docker.io/library/docker-influxdb-grafana:latest tenetreg.azurecr.io/docker-influxdb-grafana:latest
```

docker image ls ➔permet de vérifier

7/on le push vers notre registry

```
docker push tenetreg.azurecr.io/docker-influxdb-grafana:latest
```

8/on donne le droit de télécharger l'image sans login

```
az acr update --name tenetreg --anonymous-pull-enabled
```

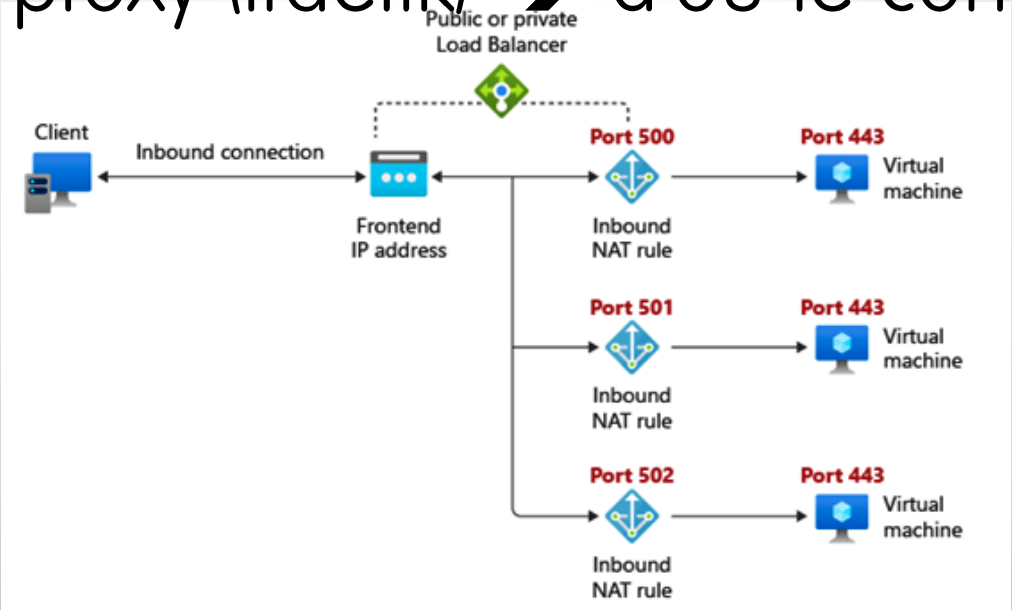
9/on visualise la liste des images sur sa registry

```
az acr repository list --name tenetreg --output table
```

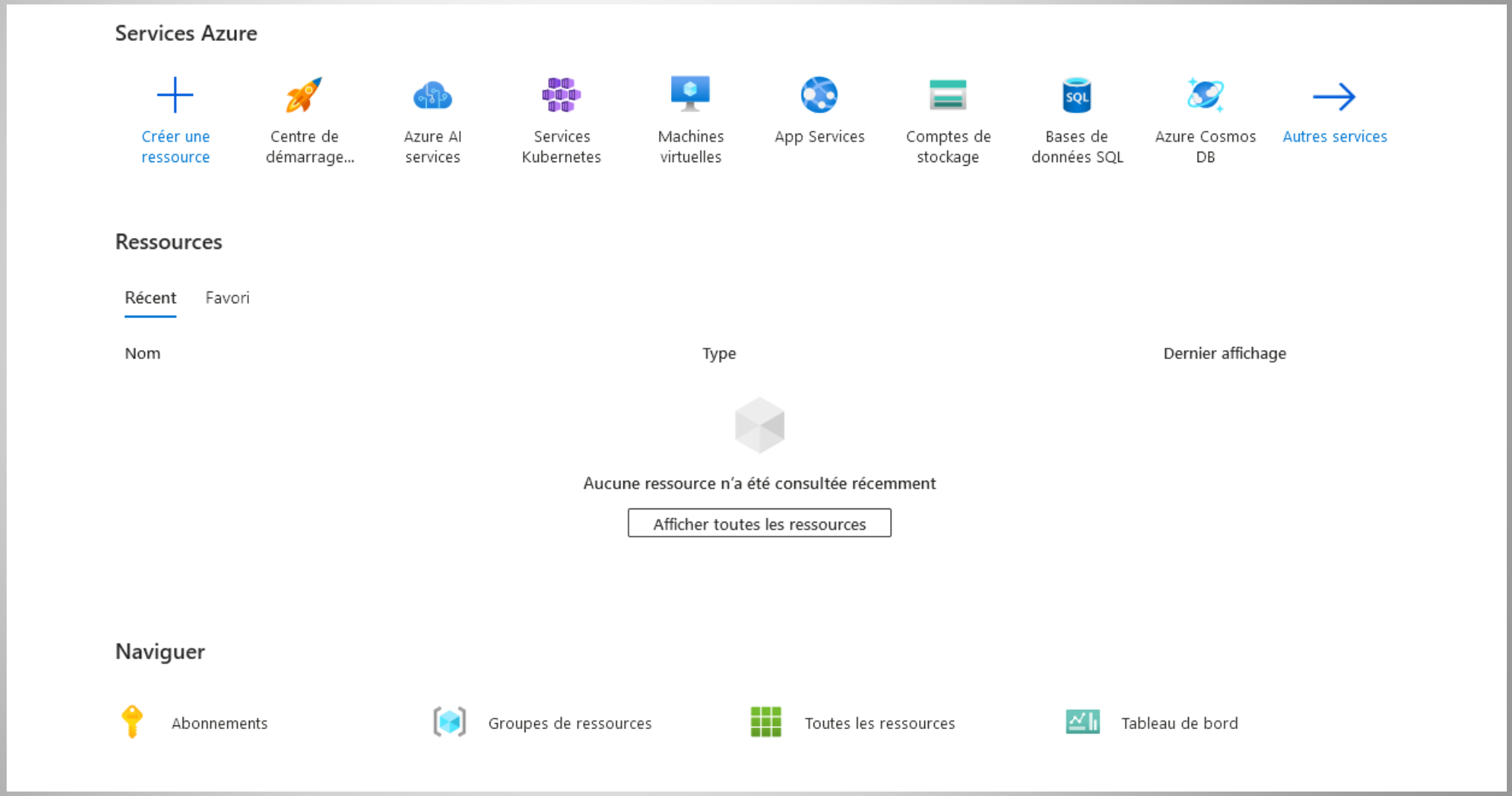
Affiche dans mon exemple : *code-server*
docker-influxdb-grafana
docker-localai

TD Azure

- Objectif :
  - Prise en main en 2 étapes
    - Instancier un Container Serverless (container instance)
    - Instancier un Container App Serverless
- Pour utiliser le container (port 80/443) – il faut faire du NAT ou avoir un reverse proxy (traefik) ➔ d'où le container App

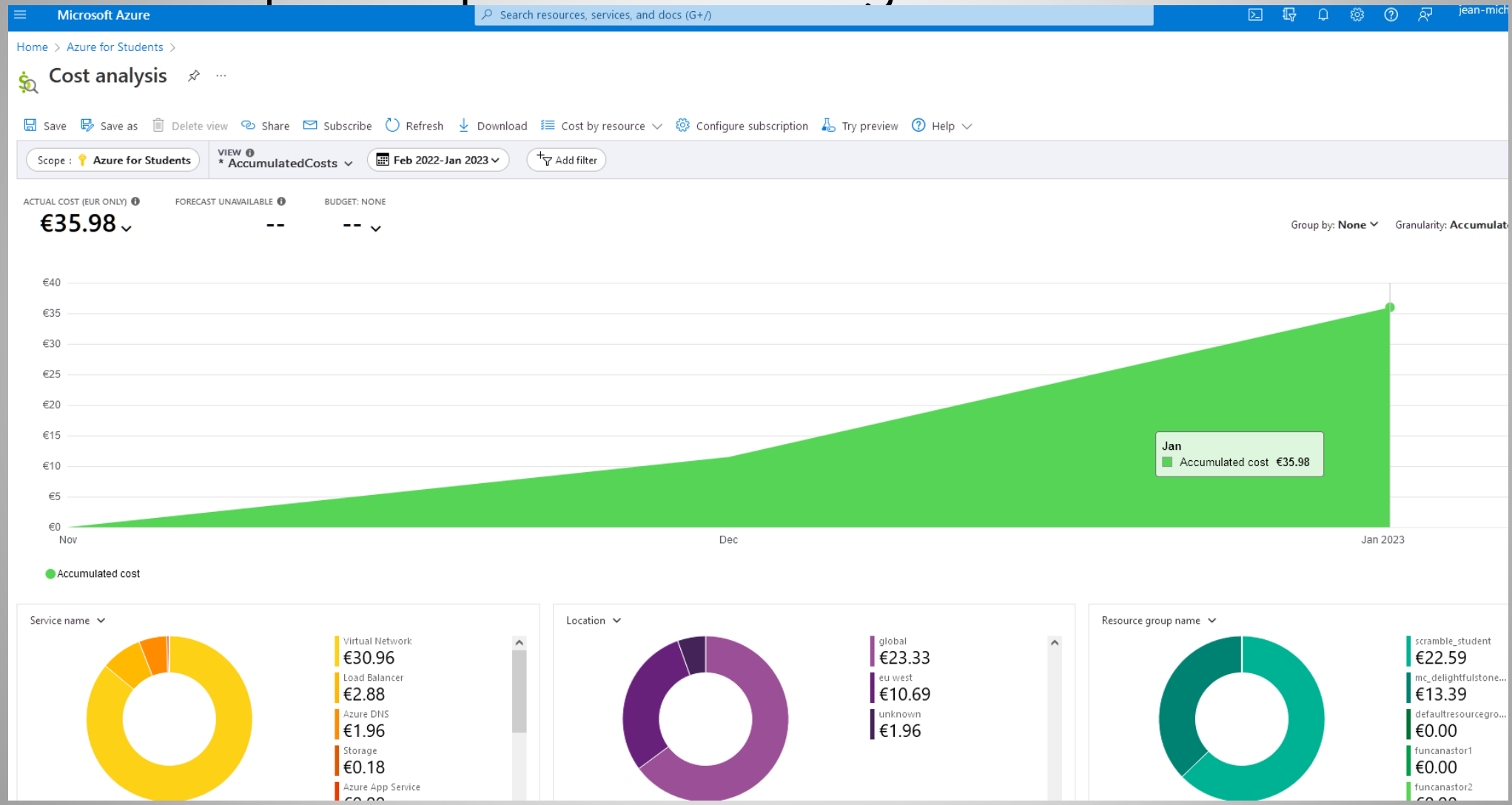


- Basculer en anglais



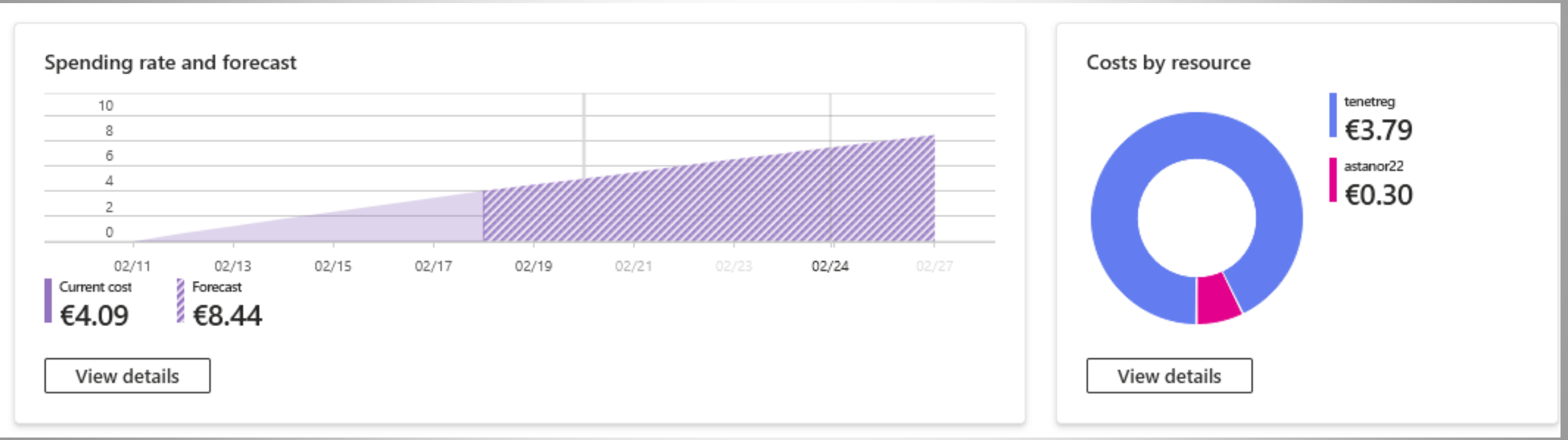
TD Azure

- Le plus important : le budget



Le budget

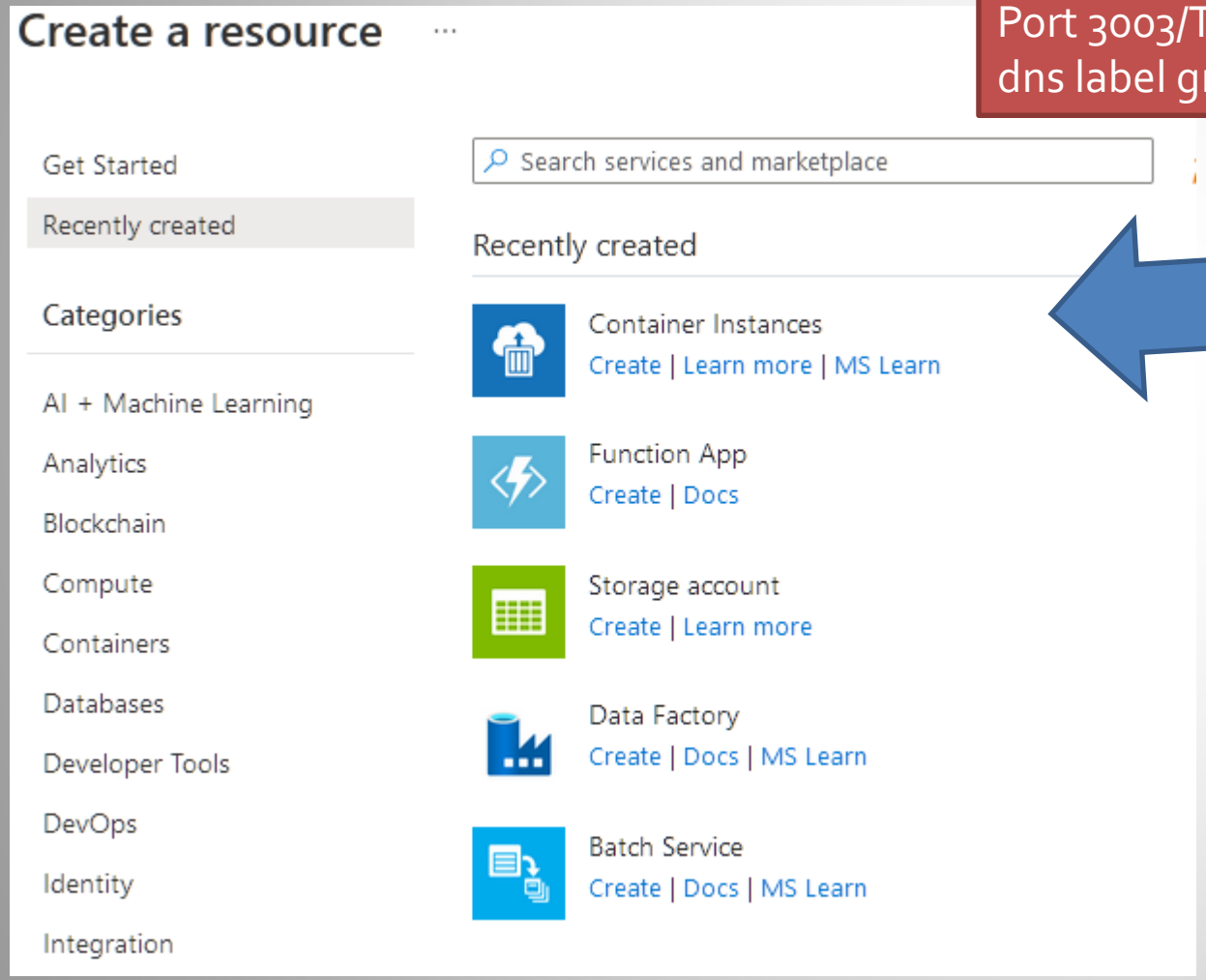
- Au bout d'une semaine pour la registry : presque 4€ !

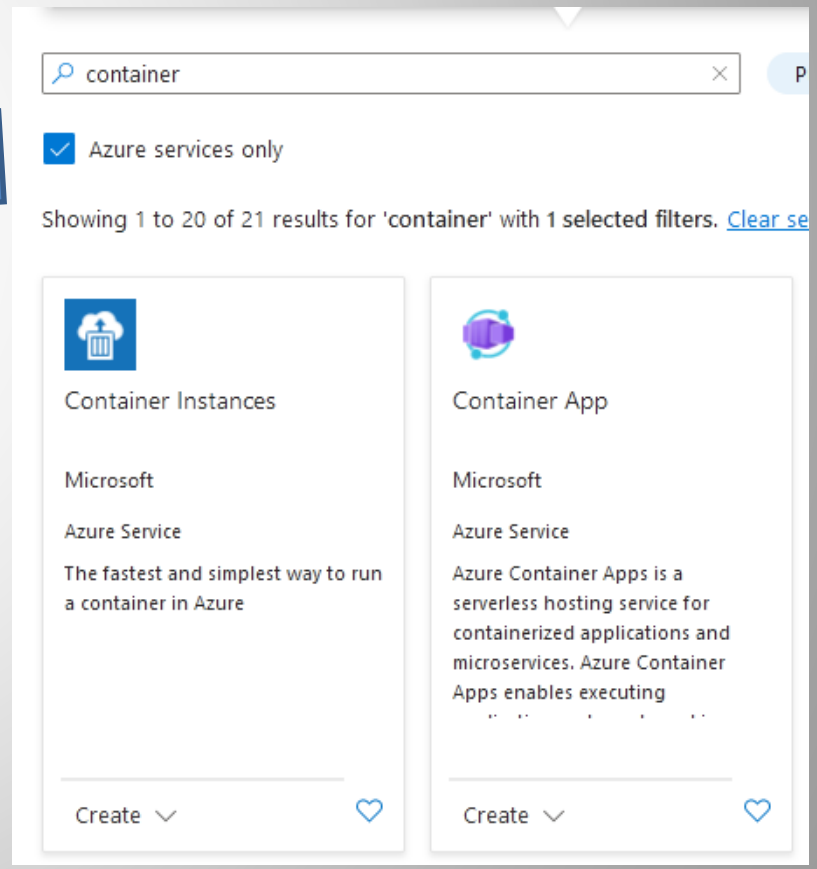


Création d'un container

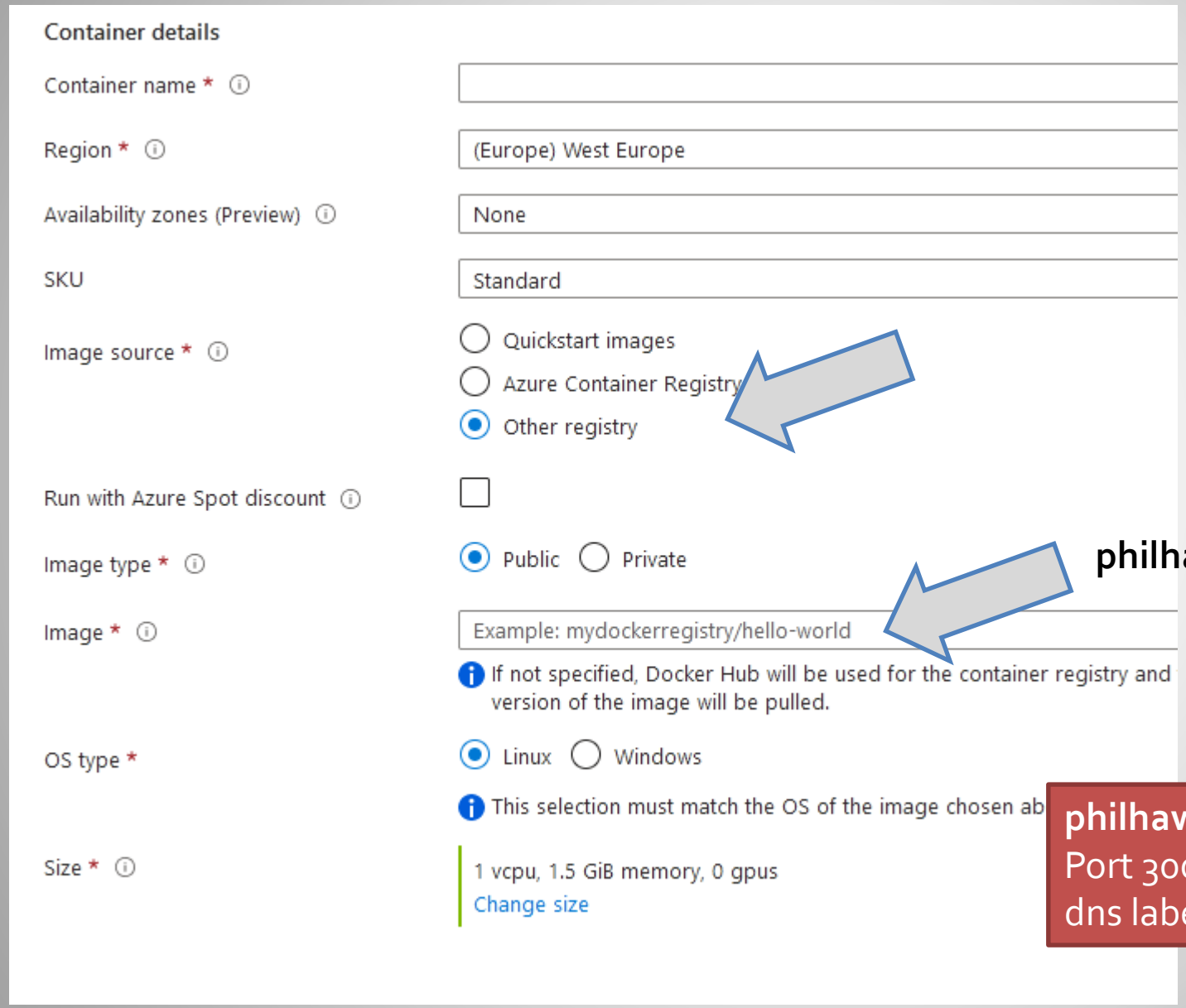
- Créer une instance « container » sur grafana

philhawthorne/docker-influxdb-grafana
Port 3003/TCP ➔ pb avec accès « eduspot »
dns label grafana





Détails du conteneur



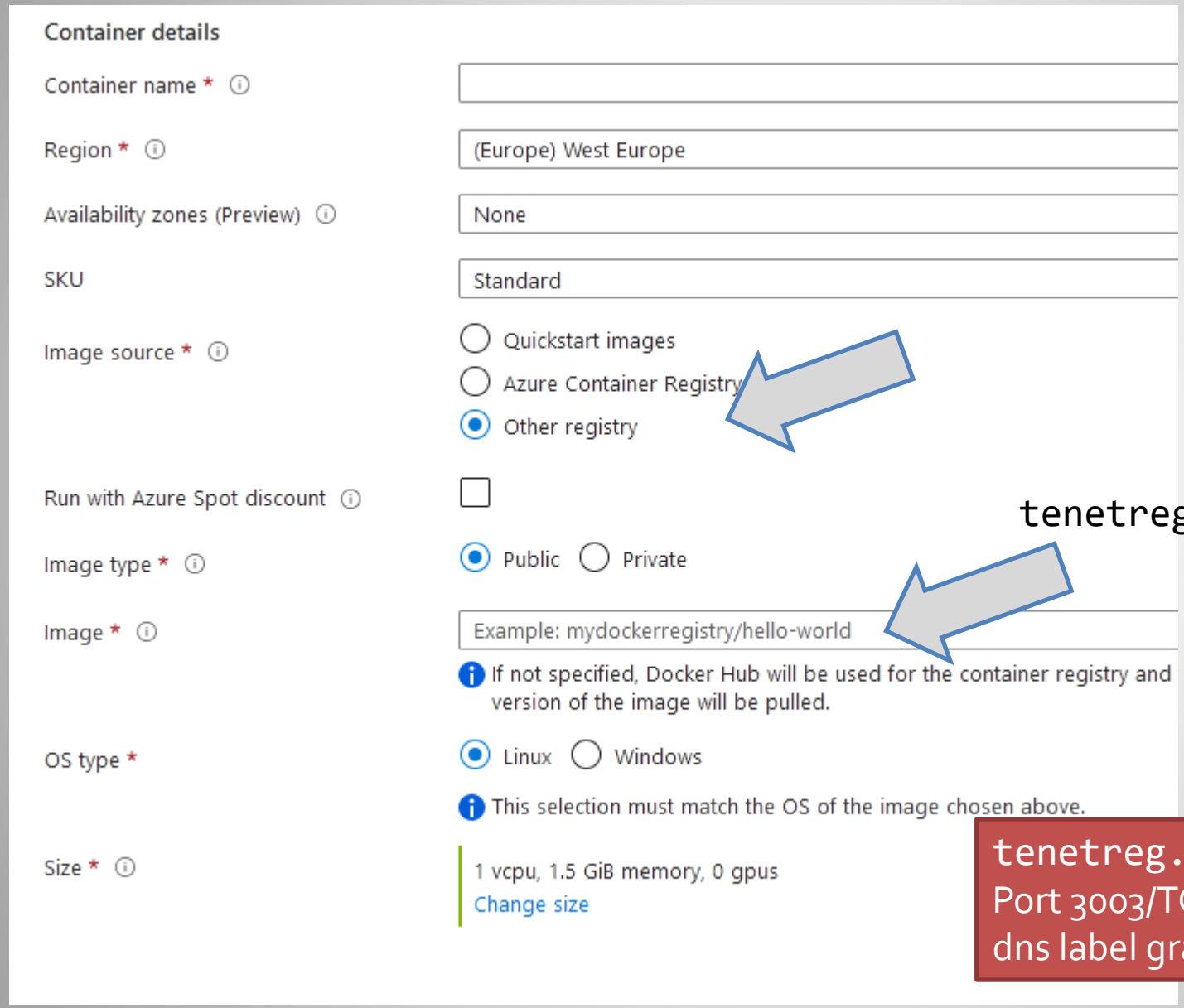
Je l'appelle ….anastor22

philhawthorne/docker-influxdb-grafana

Ne fonctionne pas car limite sur dockerhub

philhawthorne/docker-influxdb-grafana
Port 3003/TCP ➔ pb avec accès « eduspot »
dns label grafana

Détails du conteneur



Je l'appelle ….anastor22

`tenetreg.azurecr.io/docker-influxdb-grafana`

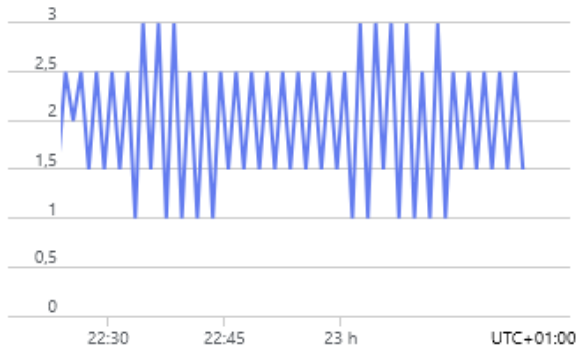
`tenetreg.azurecr.io/docker-influxdb-grafana`
Port 3003/TCP ➔ pb avec accès « eduspot »
dns label grafana

N'oubliez pas que Docker Hub a récemment introduit une limite de débit d'extraction sur les images Docker. Lorsque vous spécifiez une image à partir du registre Docker Hub, cela peut affecter votre instance de conteneur. En savoir plus

Bases Vue JSON

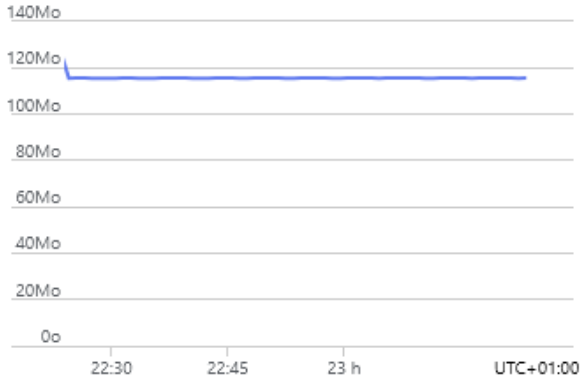
Groupe de res... (déplacer)	: test	Référence SKU	: Standard
Statut	: En cours d'exécution	Type de système d'exploi...	: Linux
Emplacement	: West Europe	Adresse IP (Public)	: 9.163.206.90
Abonnement (déplacer)	: Azure for Students	FQDN	: grafana.g5fcc7cjazbvaabt.westeurope.azurecontainer.io
ID d'abonnement	: fcc775b0-9367-47f6-b9ca-6c87491fe5b1	Nombre de conteneurs	: 1
Étiquettes (modifier)	: Ajouter des étiquettes		

Processeur



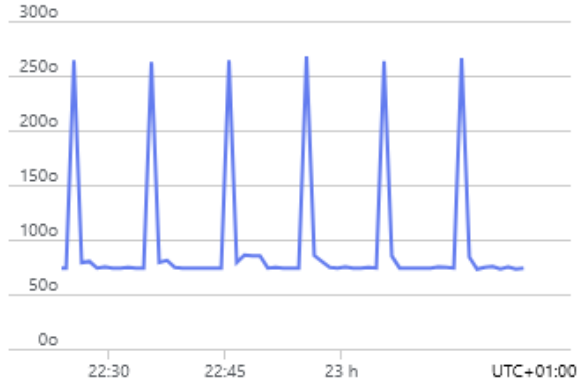
CPU Usage (Moy), testenet | 1,98

Mémoire



Memory Usage (Moy), testenet | 115,52Mo

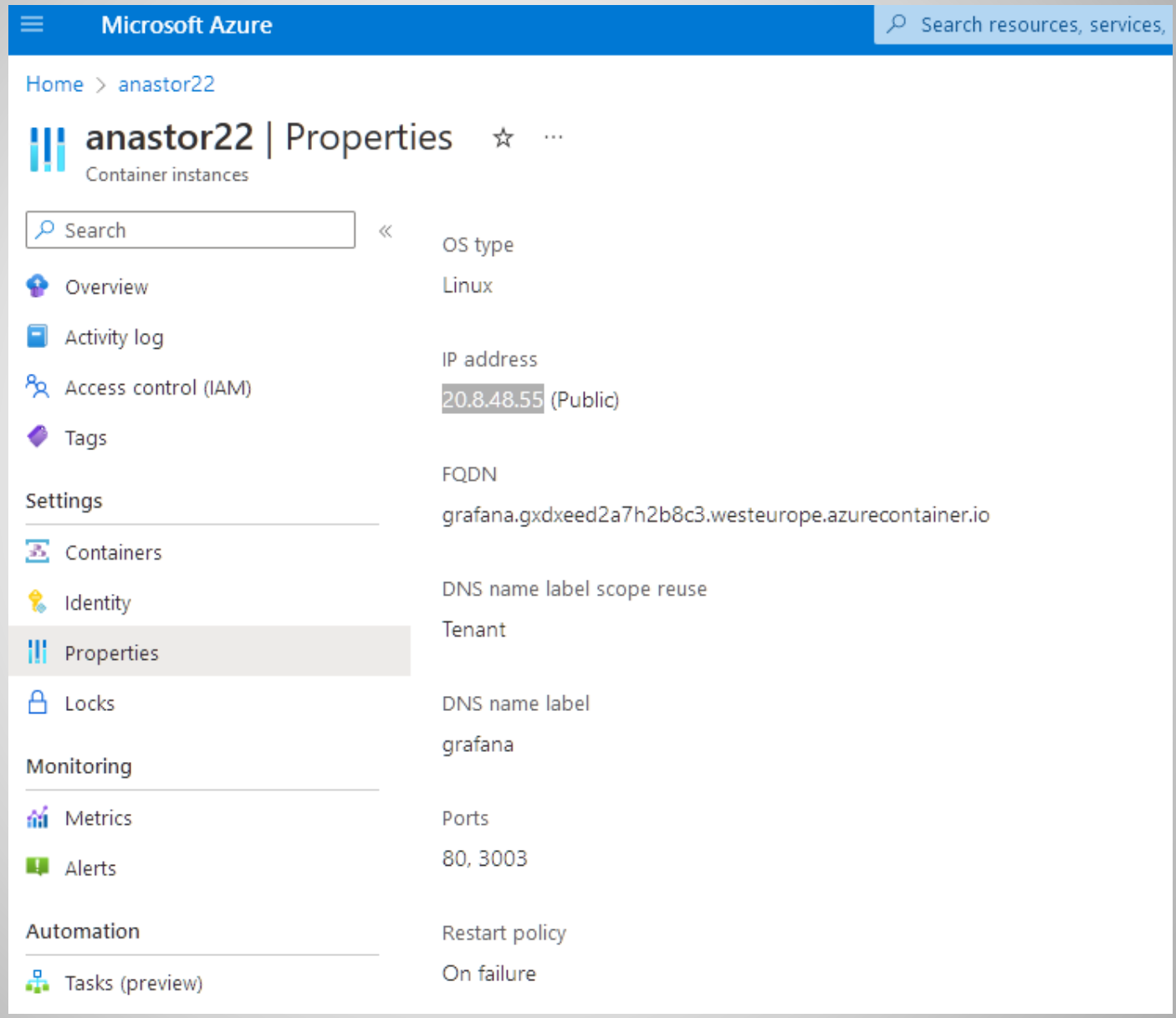
Octets réseau reçus



Network Bytes Received Per Second (Moy), test... | 94,84o

Octets réseau transmis

Information du container



Déploiement personnalisé

Déployer à partir d'un modèle personnalisé

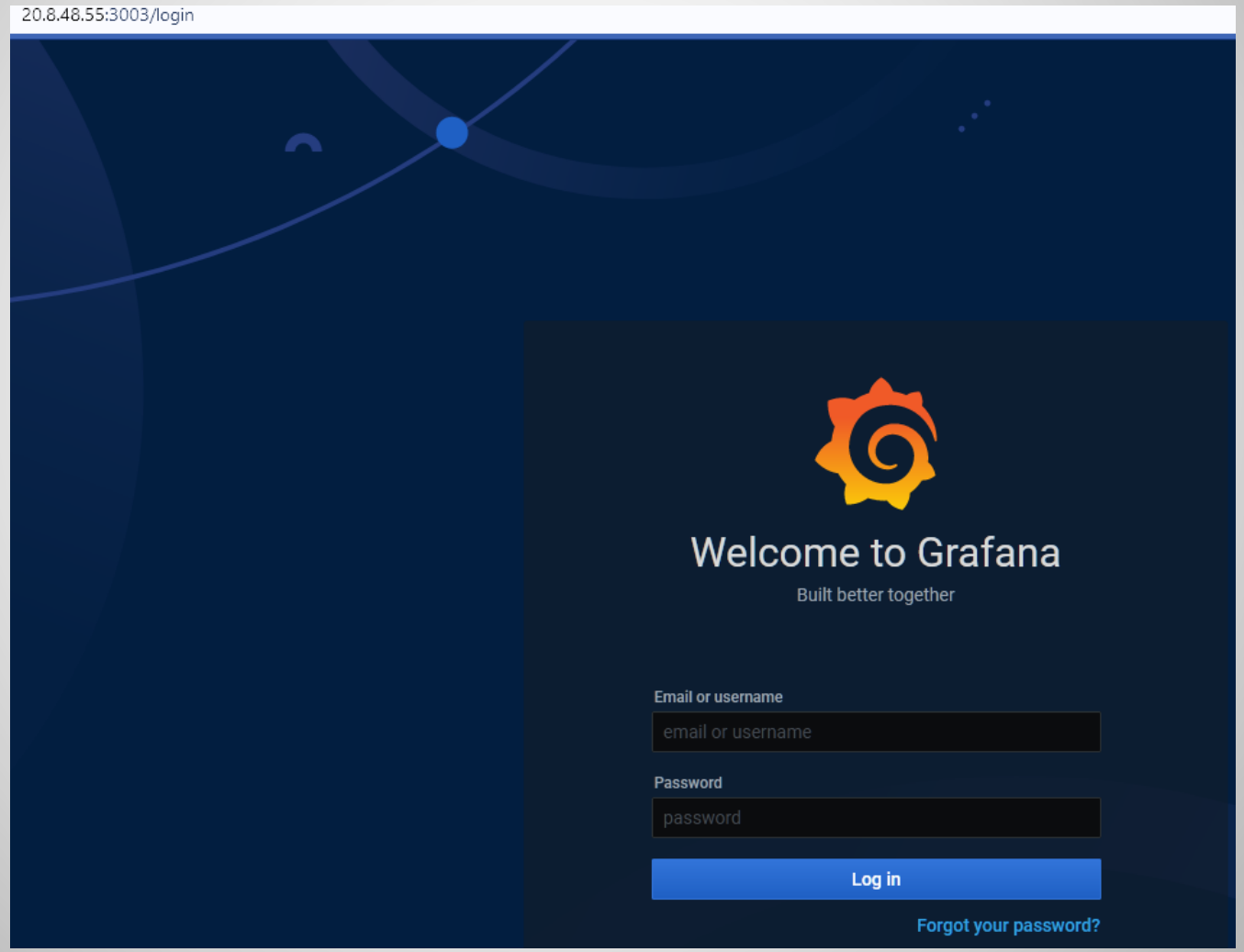
New! Deployment Stacks let you manage the lifecycle of your deployments. Try it now →

Availability Zones *	[]
Location *	westeurope
Container Name *	astanorweb
Image Type *	Public
Image Name *	philhawthorne/docker-influxdb-grafana
Os Type *	Linux
Number Cpu Cores *	1
Memory *	1.5
Restart Policy *	OnFailure
Sku *	Standard
Ip Address Type *	Public
Ports *	[{"port":"80","protocol":"TCP"}][{"port":"3003","protocol":"TCP"}]
Workspace Region *	westeurope
Workspace Sub Id *	dad45ca4-4fad-484d-8731-444104547c8a
Workspace Resource Group Name *	DefaultResourceGroup-WEU
Workspace Name *	DefaultWorkspace-dad45ca4-4fad-484d-8731-444104547c8a-WEU
Oms Sku	standalone

80 et 3003, attention à la construction du tableau

Vérification de l'accès

- On doit spécifier le port 3003 (pb sur certains réseaux qui n'autorisent que les ports 443 et 80)



Container App – étape 2

- https://learn.microsoft.com/fr-fr/azure/container-apps/
- Il s'agit d'un environnement pour le Container

➔Enrichissement :

➔Ingress (proxy réseau) ➔ pour avoir le port 443 aulieu de 3003

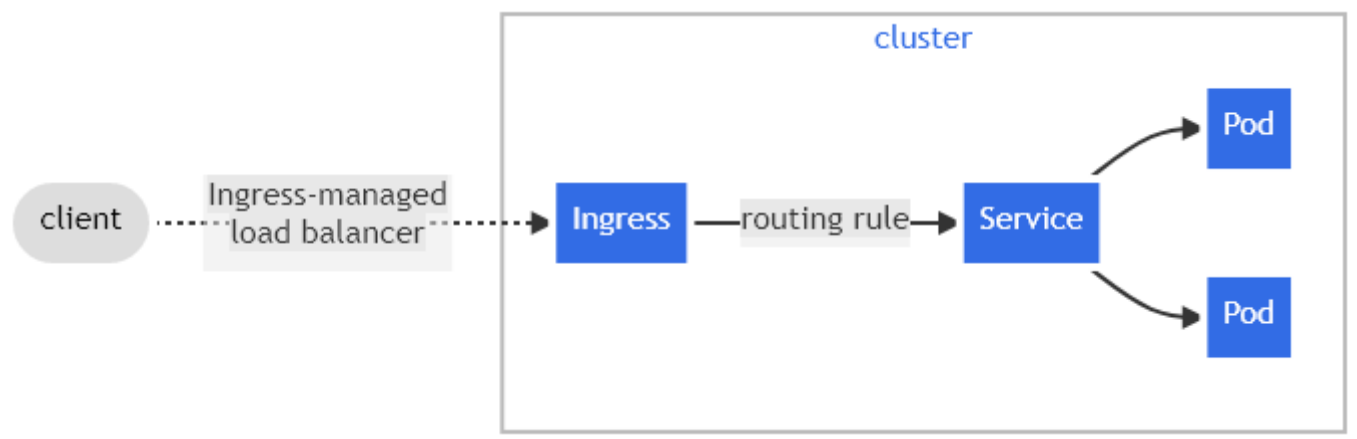
➔Monitor : possibilité d'avoir une console

➔Scale : plusieurs instances concurrentes

➔Nom de domaine dédié (Custom domain)

Load balancing / Sharding

- Ingress : réalise le mapping port 443/3003



- Load balancing = distribution de la charge, avec un outil de scale-up. L'algorithme « standard » Round Robin
- Sharding = distribution des données (de shard), pour maximiser l'effet de localité (1 service = 1 localité). L'algorithme « standard » fonction de hash sur 1 ou plusieurs colonnes.

The environment is a secure boundary around one or more container apps that can communicate with each other and share a virtual network, logging, and Dapr configuration. Container Apps Pricing

Show environments in all regions ☐

Region * France Central

Container Apps Environment * (new) managedEnvironment-SubPerso2-a20d (SubPerso2)

Create new

philhawthorne/docker-influxdb-grafana
Ingress HTTP Port 3003 anywhere
Scale = 1 // sinon pas d'accès

Create a resource

Get Started

Recently created

Categories

AI + Machine Learning

Analytics

Blockchain

Compute

Containers

Databases

Developer Tools

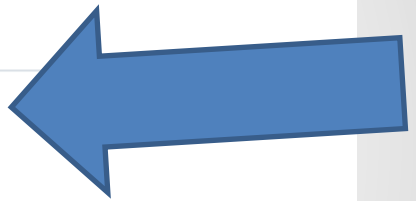
DevOps

Identity

Integration

Search services and marketplace

Recently created



Container App
Create | Docs

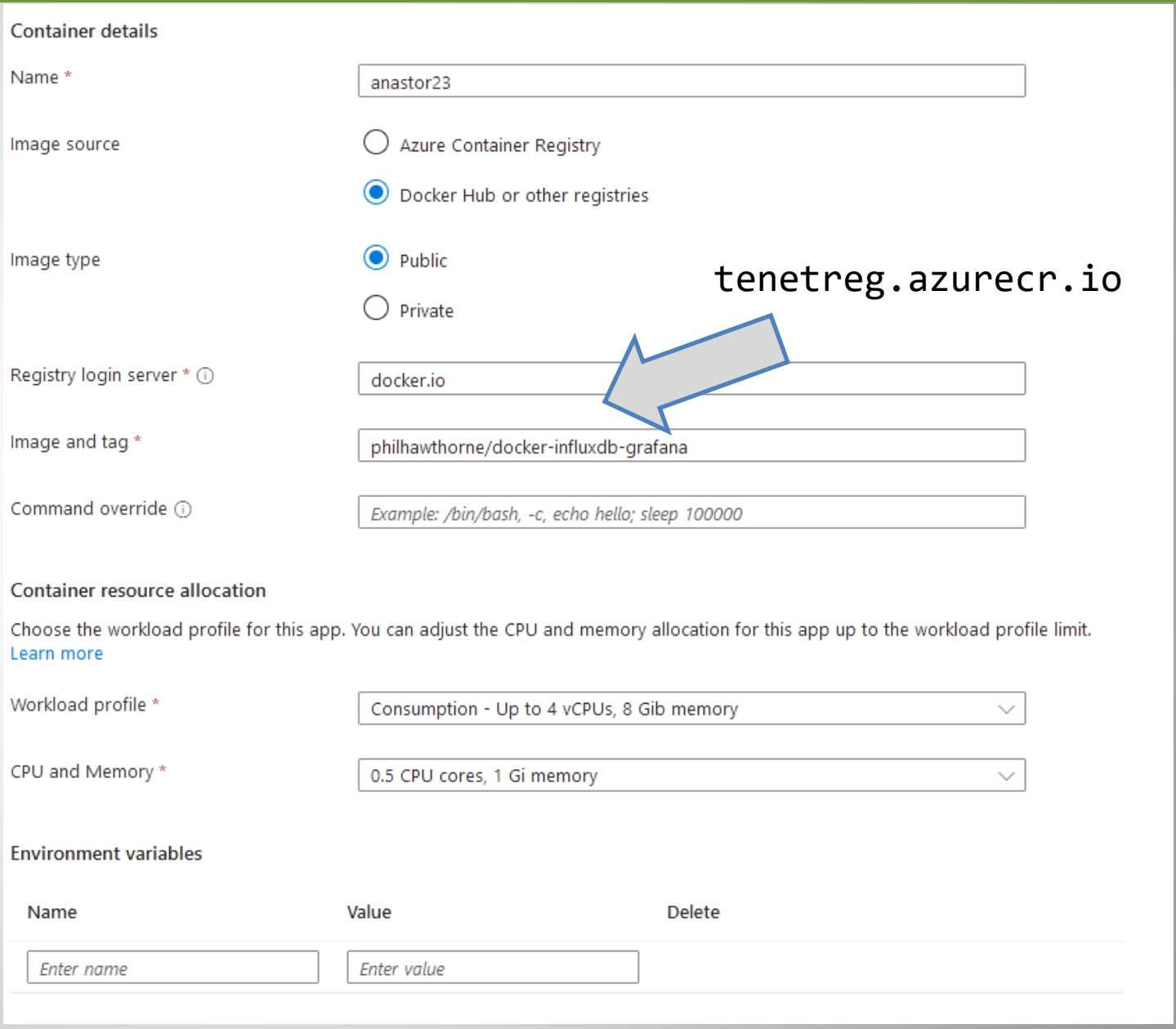
Container Instances
Create | Learn more | MS Learn

Function App
Create | Docs

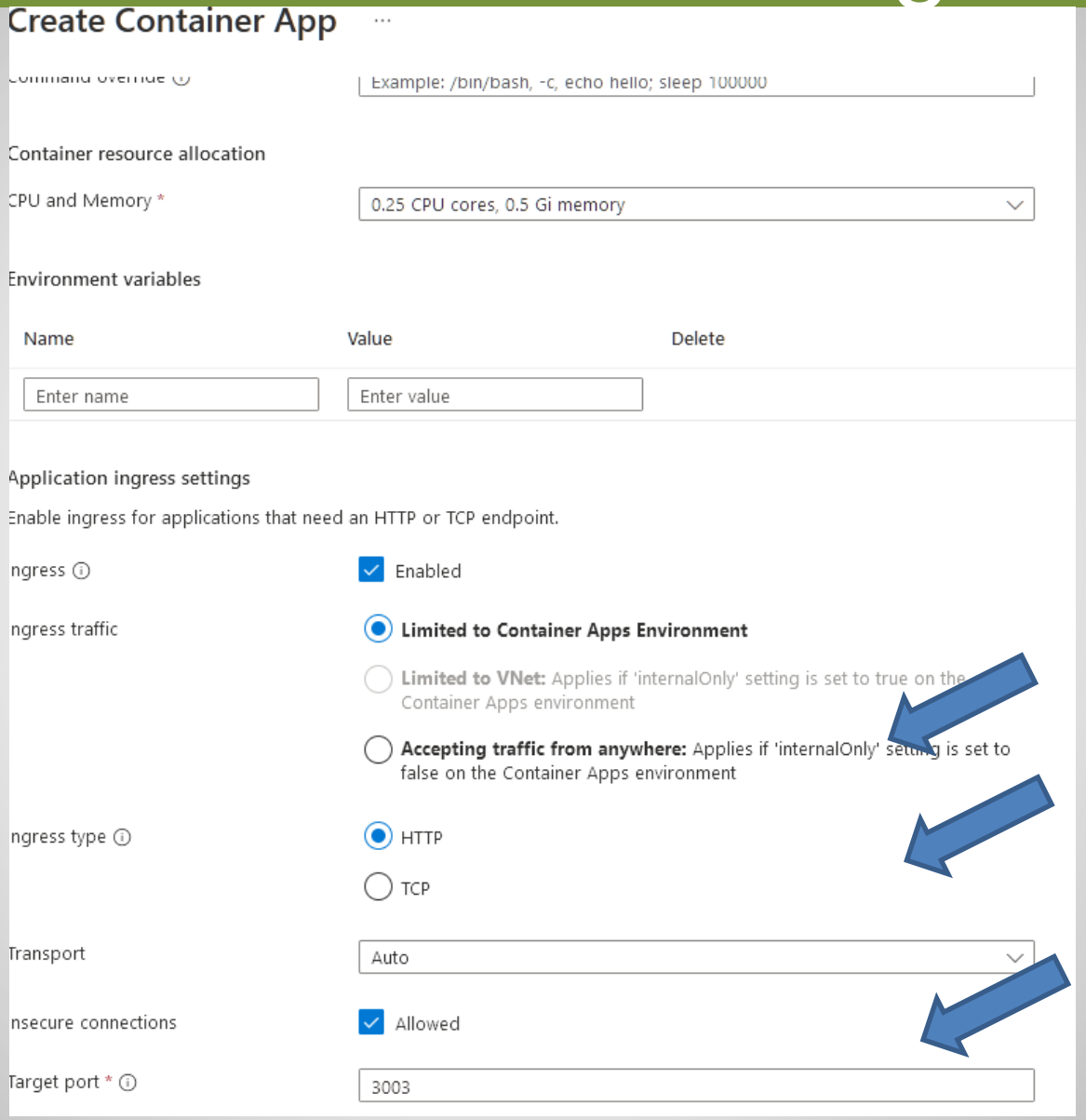
Storage account
Create | Learn more

Data Factory
Create | Docs | MS Learn

Utilisation de la registry azure tenetreg.azurecr.io



Modification Ingress



Basics | Container | Bindings | Ingress | Tags | **Review + create**

Project details

Subscription	Subscription Perso2
Resource group	SubPerso2
Name	grafana2

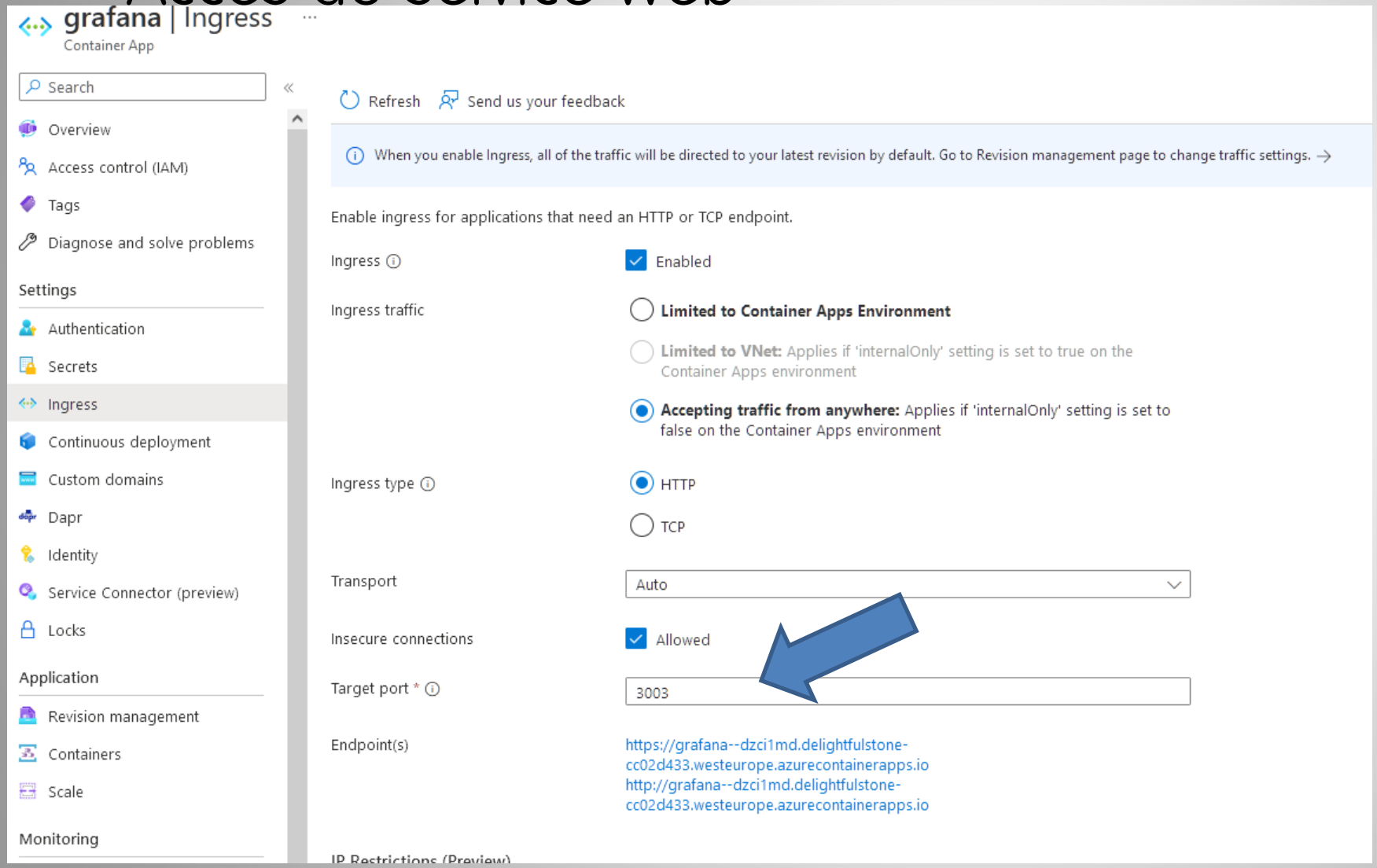
Container Apps Environment (New)

Region	northeurope
Container Apps Environment	managedEnvironment-SubPerso2-9ab6
Log Analytics workspace (New)	workspacesubperso28e43
Virtual network	Default
Zone redundancy	Disabled

Container

Name	grafana2
Image source	Public
Registry login server	docker.io
Image and tag	philhawthorne/docker-influxdb-grafana
Command	
Workload profile type	Consumption
Number of CPU cores	0.5
Memory size (Gi)	1
Ingress settings	Accepting traffic from anywhere
Ingress type	HTTP
Transport	Auto
Insecure connections	Allowed
Port	3003

- Accès au service web



Étapes de création d'une image

Exemple avec LocalAI (alternative à ollama)

//pas de majuscule dans le nom d'une image Docker

Sur sa machine

- docker build -t docker-localai:latest .
- docker tag docker.io/library/docker-localai:latest tenetreg.azurecr.io/docker-localai:latest
- az acr -login -n tenetreg
- docker push tenetreg.azurecr.io/docker-localai:latest
- ➔ensuite on est sur le web Azure

Informations de base | Conteneur | Entrée | Étiquettes | Vérifier + créer

Créez une application conteneurisée et exécutez-la sur une plateforme serverless, sans gérer l'infrastructure cloud.Guide de démarrage rapide

Détails du projet

Sélectionnez un abonnement pour gérer la création et les coûts des ressources, et un groupe de ressources pour organiser toutes vos ressources pour ce déploiement.

Abonnement * : Azure for Students

Groupe de ressources * : DefaultResourceGroup-WEU

Créer un groupe de ressources

Nom de l'application conteneur * : testlocalai

Source de déploiement *

- (●) Image conteneur
  Apportez votre propre registre de conteneurs ou générez un conteneur à partir d'un Dockerfile.
- ( ) Code source ou artefact
  Générez et déployez votre code sans utiliser de Dockerfile.

Environnement Container Apps

Un environnement est une limite sécurisée autour d'un groupe d'applications conteneur. Tarification de Container Apps

Afficher les environnements dans toutes les régions ⓘ [ ]

Région * : West Europe

Environnement Container Apps * : (nouveau) managedEnvironment-DefaultResource-ae72 (DefaultResourceG...

Source de l'image
- ○ Azure Container Registry
- ◉ Docker Hub ou d'autres registres

Type d'image
- ◉ Public
- ○ Privé

Serveur de connexion au registre * ⓘ : tenetreg.azurecr.io

Image et balise * : docker-localai:latest

Remplacement de commande ⓘ : *Exemple : /bin/bash*

Remplacement des arguments ⓘ : *Exemple : -c, while true; do echo hello; sleep 10; done*

Fonctionnalités spécifiques à la pile de développement

Lorsque vous sélectionnez une pile de développement spécifique, vous obtenez des fonctionnalités supplémentaires adaptées à cette pile, en optimisant Container Apps pour qu'elles s'exécutent pour vos paramètres uniques.

Pile de développement : Non spécifié

Allocation des ressources de conteneur

Choisissez le profil de charge de travail pour cette application. Vous pouvez ajuster l'allocation de processeur et de mémoire pour cette application jusqu'à la limite du profil de charge de travail. En savoir plus

Profil de charge de travail * : Consumption : jusqu'à 4 processeurs virtuels, 8 Gio de mémoire

Processeur et mémoire * : 4 cores de processeur, mémoire 8 Gio

Créer Application de conteneur ...

Informations de base | Conteneur | **Entrée** | Étiquettes | Vérifier + créer

Paramètres d'entrée d'application

Activez l'entrée pour les applications qui ont besoin d'un point de terminaison HTTP.

Entrée ⓘ — ☑ Activé

Trafic entrant
- ○ Limité à l'environnement Container Apps
  Sélectionnez cette option si vous voulez limiter le trafic vers cette application conteneur à partir de l'environnement Container App
- ◉ Acceptation du trafic depuis n'importe où
  Sélectionnez cette option si vous voulez autoriser le trafic vers cette application conteneur à partir de n'importe où

Type d'entrée ⓘ
- ◉ HTTP
- ○ TCP

Transport : Auto

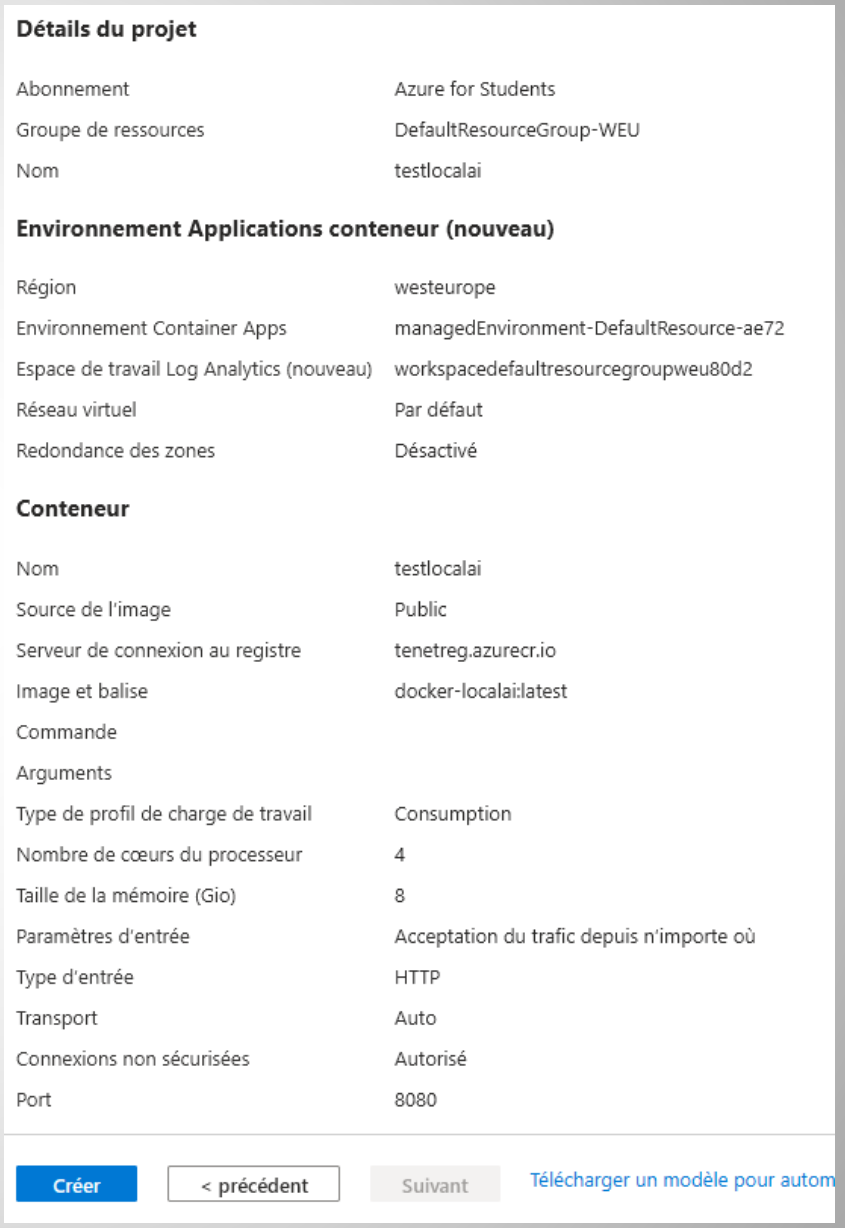
Connexions non sécurisées : ☑ Autorisé

Port cible ⓘ : 8080

Affinité de session ⓘ : ☐ Activé

▲ Ports TCP supplémentaires

- On vérifie



Détails du projet

Abonnement	Azure for Students
Groupe de ressources	DefaultResourceGroup-WEU
Nom	testlocalai

Environnement Applications conteneur (nouveau)

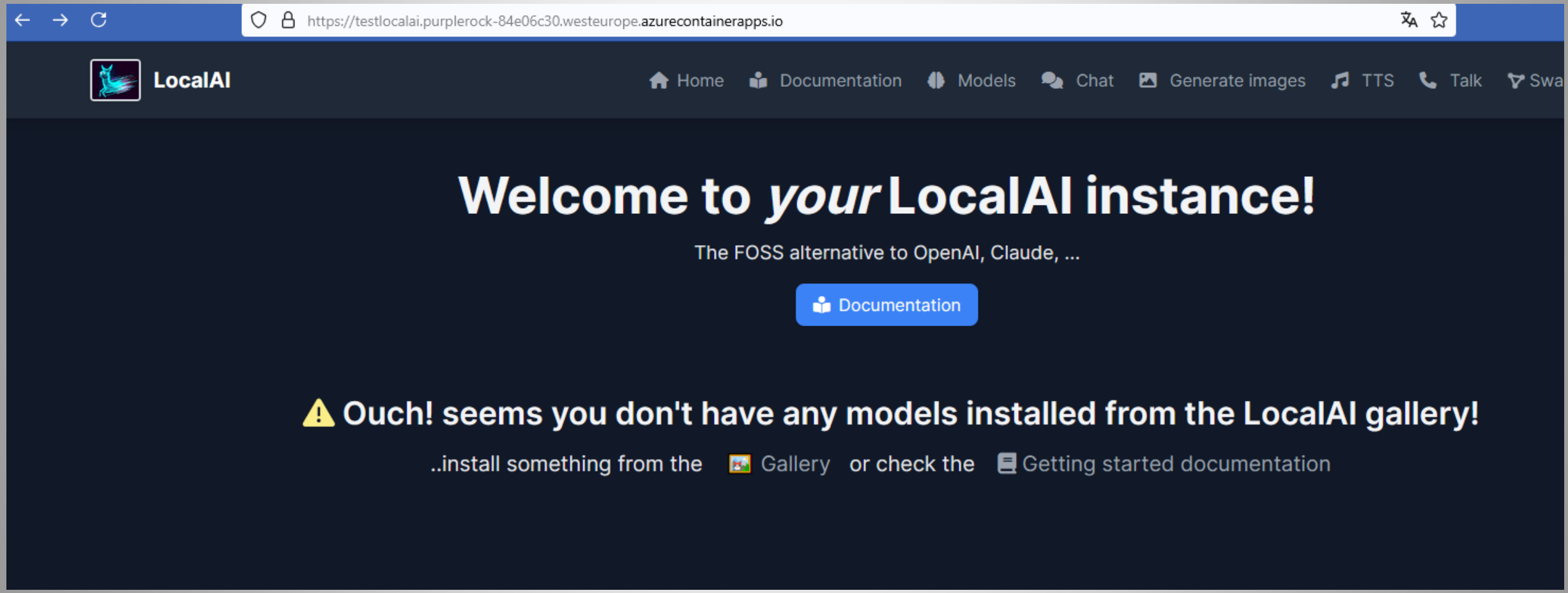
Région	westeurope
Environnement Container Apps	managedEnvironment-DefaultResource-ae72
Espace de travail Log Analytics (nouveau)	workspacedefaultresourcegroupweu80d2
Réseau virtuel	Par défaut
Redondance des zones	Désactivé

Conteneur

Nom	testlocalai
Source de l'image	Public
Serveur de connexion au registre	tenetreg.azurecr.io
Image et balise	docker-localai:latest
Commande	
Arguments	
Type de profil de charge de travail	Consumption
Nombre de cœurs du processeur	4
Taille de la mémoire (Gio)	8
Paramètres d'entrée	Acceptation du trafic depuis n'importe où
Type d'entrée	HTTP
Transport	Auto
Connexions non sécurisées	Autorisé
Port	8080

Créer | < précédent | Suivant | Télécharger un modèle pour autom

- Accès à la page après…10 minutes



Pourquoi appliquer un théorème?

Un théorème : une proposition logique qui peut être démontrée

Utilité du théorème

- Avoir des propriétés communes (homogénéité)
- Avoir une approche efficace (factoriser l'effort)
- Identifier les exceptions (règles des 80/20)
- ➔avoir un gain de productivité

De la conceptualisation

Traiter un problème d'ingénieur peut se faire sans une conceptualisation préalable

- Éléments finis :1940… → bases rigoureuses:1973
- Génétique des espères : XIXème siècle → bases rigoureuses: mi XXème siècle

Avec des concepts ➔ symboles fonctionnels

Notre sujet : qu'est ce qu'un système distribué?

Pour finir

- Reste du cours
- ➔ vision pour un calculateur exaflopique

Superordinateur	Performance HPL (Rmax)	Nombre total de cœurs (CPU + GPU)	Architecture principale	Remarques
El Capitan (LLNL, USA)	~1.74 – 1.81 exaFLOPS	~11 millions (11,039,616)	AMD EPYC 4th Gen + AMD Instinct MI300A	Machine n°1 mondiale actuellement
Frontier (ORNL, USA)	~1.35 exaFLOPS	~8.7 – 9 millions	AMD EPYC 3rd Gen + AMD Instinct MI250X	Premier exaflopique historique (2022)
Aurora (ANL, USA)	~1.01 exaFLOPS	~9.3 millions	Intel Xeon Max + Intel Data Center GPU Max	Très fort en AI, beaucoup de GPU tiles
JUPITER (Allemagne)	~1 exaFLOPS (soutenu)	~10–12 millions estimés	NVIDIA Grace-Hopper ou mix	Premier exaflopique européen

Comment distribuer l'information?

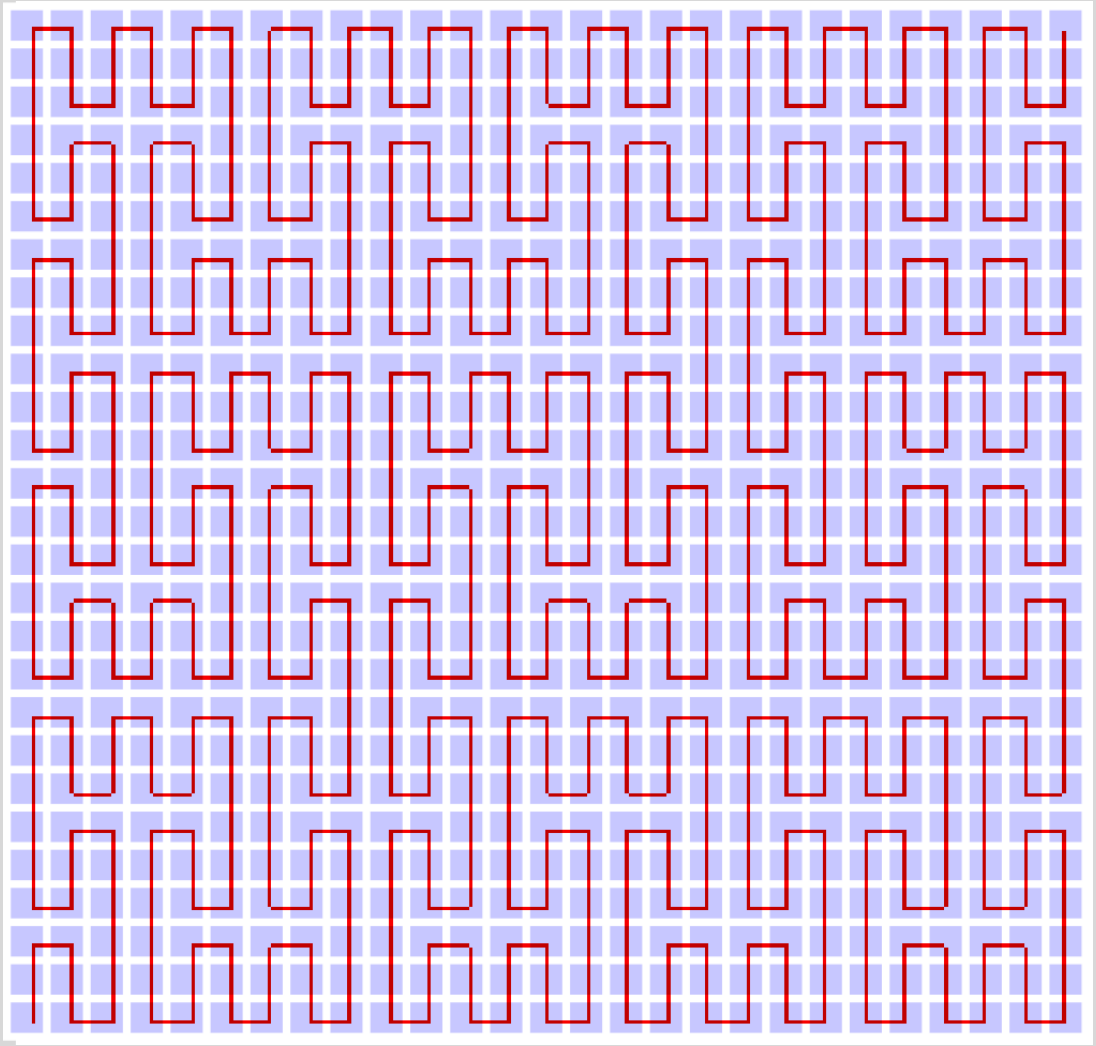
Soit un ensemble de 27*27 unités (891 unités de calcul.)

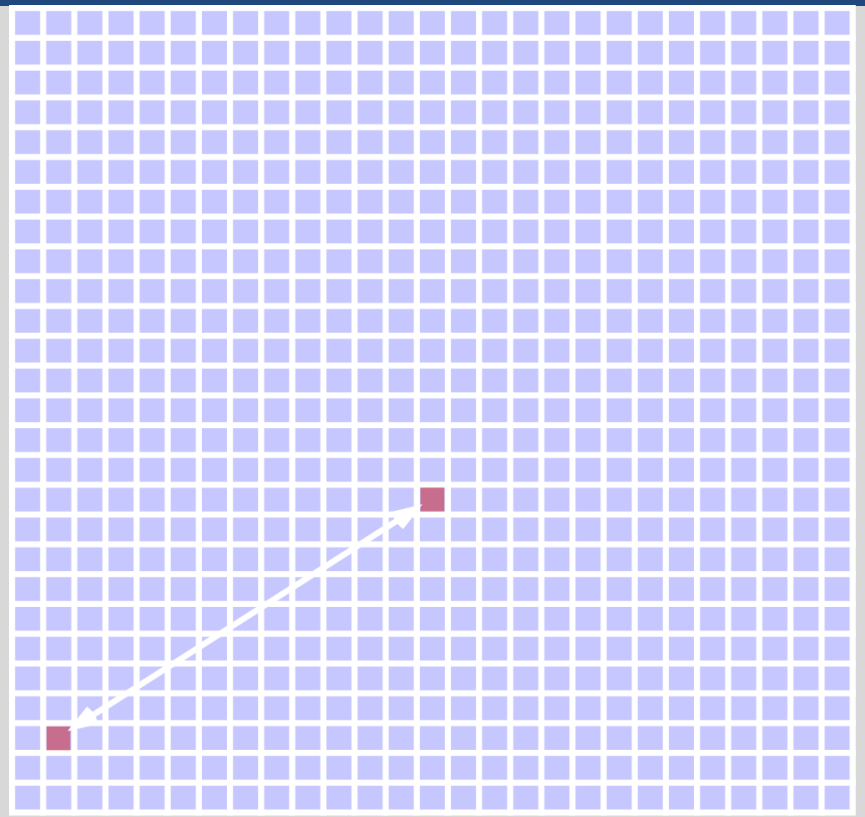
On peut imaginer faire une interconnexion.

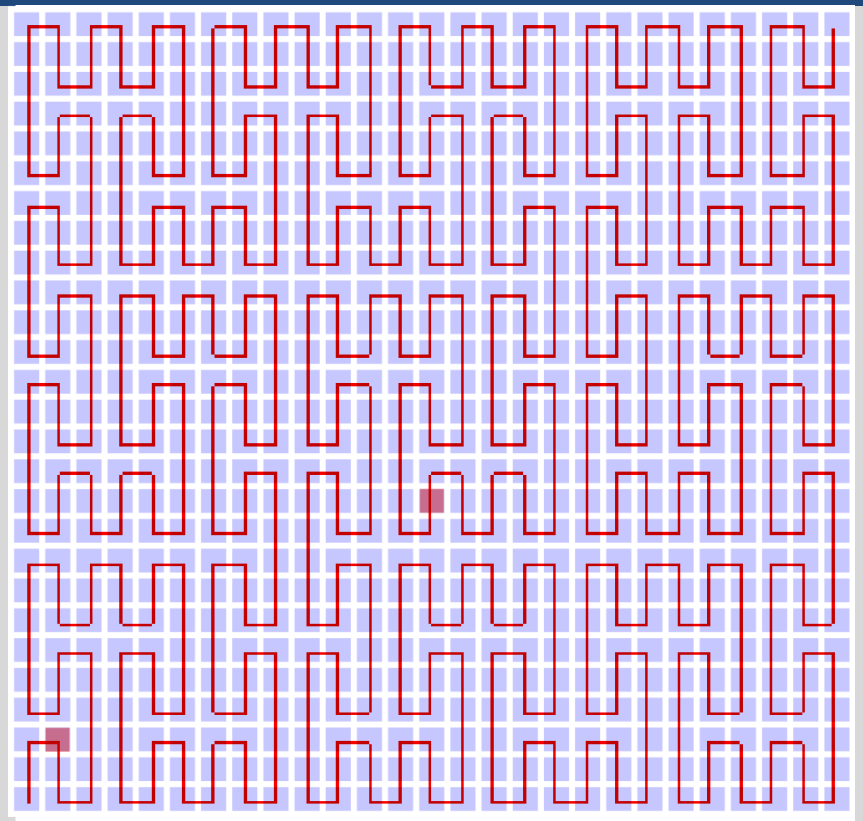
Quelle serait le chemin le plus économique?

➔courbe de Péano

C’est une courbe fractale, qui est générée par un automate







La distance entre 2 nœuds est algorithmique

Chaque nœud est autonome mais est garant du chemin

2 concepts à explorer d'un point de vue algorithmique :

- Système distribué
- Système autoconvergent

Qu'est ce qu'un message ?

Plusieurs hypothèses sous-jacentes sur le message diffusé

- le message est la preuve de vie
- il existe un délai (time-out) pour traiter le message
- le message ne connait pas la topologie du graphe
- le message peut être perdu
- le processus qui reçoit le message peut crasher

Le système distribué

➔ notion de temps universel (horloge)

➔ il existe une précédence des événements (causalité)

Système avec propagation synchrone

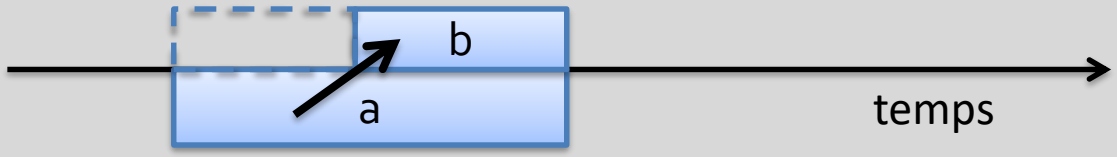
Équitable (fairness)

Auditable (proving)

Mode de sureté (safety)

Etat opérationnel, vivacité (liveness)

$$(\mathrm{a} \prec b \vee a = b) \equiv \mathrm{a} \preceq \mathrm{b}$$



$\preceq$ *poset : partially ordered set*

Les algorithmes à front d'onde conviennent à la résolution de nombreux problèmes de calculs distribués :

- diffusion (par exemple, notifications sur le début ou fin du calcul),
- synchronisation globale entre processus,
- déclencher l'exécution d'un événement dans chaque processus,
- calculer une fonction de données réparties entre tous les processus du système,
- élection du leader, détection de terminaison, calcul de snapshot, etc…

Un théorème outil pour l'ingénieur

Conceptualisation d'un algorithme à front d'onde

(1) **Terminaison**. Chaque calcul élémentaire est finit :

$\forall C: |C| < \infty$

(2) **Décision.** Chaque calcul élémentaire contient au moins un événement décisionnel :

$\forall C: \exists \mathrm{e} \in \mathrm{C}: e\ est\ un\ événement\ décisionnel$

(3) **Dépendance.** Dans chaque calcul élémentaire, chaque évènement décisionnel est associé à un évènement précédent qui est associé à un processus précédent.

$\forall C: \forall \mathrm{e} \in \mathrm{C} : (e\ est\ un\ événement\ décisionnel \implies \forall \mathrm{q} \in \mathbb{P}: \mathrm{processus}, \exists \mathrm{f} \in \mathrm{Cq}: \mathrm{f} \preceq \mathrm{e})$

Quelques notions

Chan = fifo, bidirectionnel

Relile rrédécesseur/successeur entre les noeuds

receive ⟨tok,a⟩ from q

Réceptionne du canal **tok**, la valeur a de la part du nœud q

send ⟨tok⟩

Envoie un message {}sur **tok**

decide

Action

Taille du réseau : N – nombre de nœuds

Temps de garde du jeton, temps de garde d'une opération élémentaire

Collection d'algorithme de front d'onde

Topologie	Jeton/Autorité	Complexité
Anneau	Centralisée	N : nombre de nœud
	Décentralisée	O(N log N)
Arbre	Centralisée	2(N-1)
	Décentralisée	O(N)

Le jeton « est le symbole d'autorité »

Propagation sur un anneau

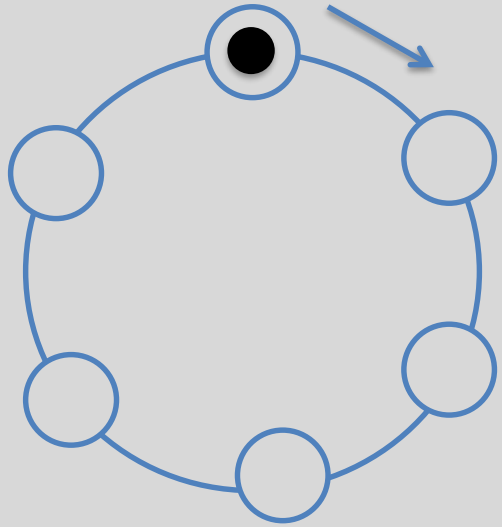
Anneau (graphe orienté, avec 1 jeton)

Pour l'initiateur :

send ⟨tok⟩ to $Next_p$
receive ⟨tok⟩
decide

Pour les autres :

receive ⟨tok⟩
send ⟨tok⟩ to $Next_p$

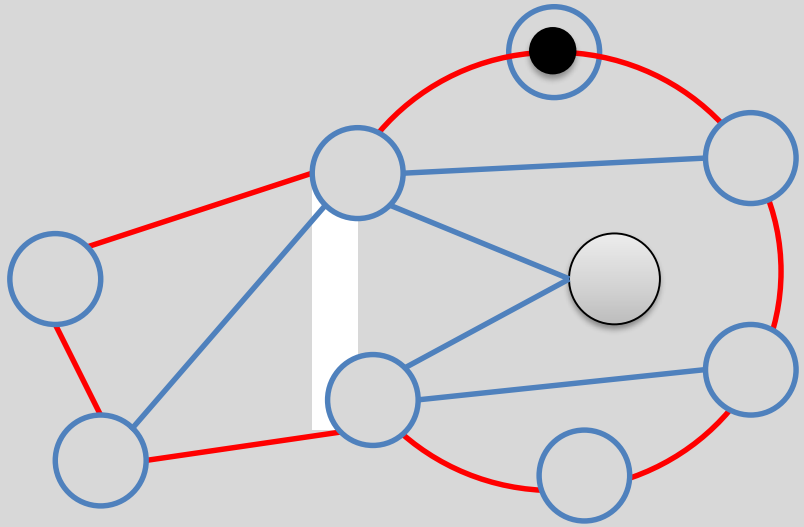


➔ Lorsque le jeton à fait le tour du graphe qu'une décision est prise

La topologie du système de communication est quelconque. L'anneau virtuel est un réseau superposé au réseau physique (overlay network) et utilisant les protocoles du réseau physique

L'anneau virtuel se définit dès qu'un nœud décrit un successeur et/ou un prédécesseur.

Chaque nœud surveille son prédécesseur/successeur et peut/doit provoquer une reconfiguration en cas de disfonctionnement.



←Ici un nœud dormant

Elections

Parmi un ensemble de processus, en choisir un et un seul (et le faire connaître à tous) – le même algorithme pour tous les processus.

- sécurité : un seul processus élu
- vivacité : un processus doit être élu en temps fini

L'élection peut être déclenchée par un processus quelconque, éventuellement par plusieurs processus

L'identité de l'élu est **déterministe** : par ex. choisir le processus qui a le plus petit numéro

Difficulté : des processus peuvent tomber en panne pendant l'élection

Utilité de l'élection

- Régénération d'un jeton perdu : un jeton et un seul doit être recréé
- Dans les algorithmes de type "maître-esclave" : élire un nouveau maître
- En cas de défaillance du maître courant. Exemples (cf plus loin) :
- validation de transactions, recherche du «reste à faire »
- Élection d'un maitre pour diriger un data-worker

Sur un anneau ➔ bouclage topologique

Sur un arbre ➔ segmentation en sous-arbre

Algorithme de la brute (bully)

Le réseau de communication est fiable et synchrone (timeout définit)

Chaque processus connaît l'identité des autres

- Les demandes d'élection sont diffusées par inondation.
- Choix déterministe : un processus répond à ceux de numéro inférieur au sien.

➔Un processus qui ne reçoit aucune réponse après un timeout constate qu'il est élu!

Constat :

- Algorithme coûteux en nombre de messages : $O(N^2)$
- Résiste bien aux pannes des processus (mais pb choix de timout)

Anneau sans régénération - élection

Election sur un anneau

var $List_p$ **:** ***int init*** {p} **;** //initialisation d'une liste, p identité du nœud
$state_p$ *: (cand, sleep, leader, lost)* ***init cand ;*** //statut de p
if p is initiator then //nous n'avons pas reçu de message de q
    $state_p$ *=cand*
    send ***⟨tok,p⟩*** *to* $Next_p$
    receive ⟨tok,q⟩
    while q ≠ p do //tant qu'on ne récupère pas sa propre identité
        $List_p$ **=** $List_p$ ∪ {q}
        send ⟨tok,q⟩ to $Next_p$
        receive ⟨tok,q⟩
    if p ==min($List_p$**)** ***then*** $state_p$ *=leader* // c'est l'id la plus petite qui gagne
    else $state_p$ *=lost*
else repeat
    receive ⟨tok,q⟩ //propagation
    send ⟨tok,q⟩ to $Next_p$
    if $state_p$ *==sleep* ***then*** $state_p$ *=lost*
    until false //le leader va envoyer un message pour arrêter ce repeat

Algorithme de LeLann, 1977

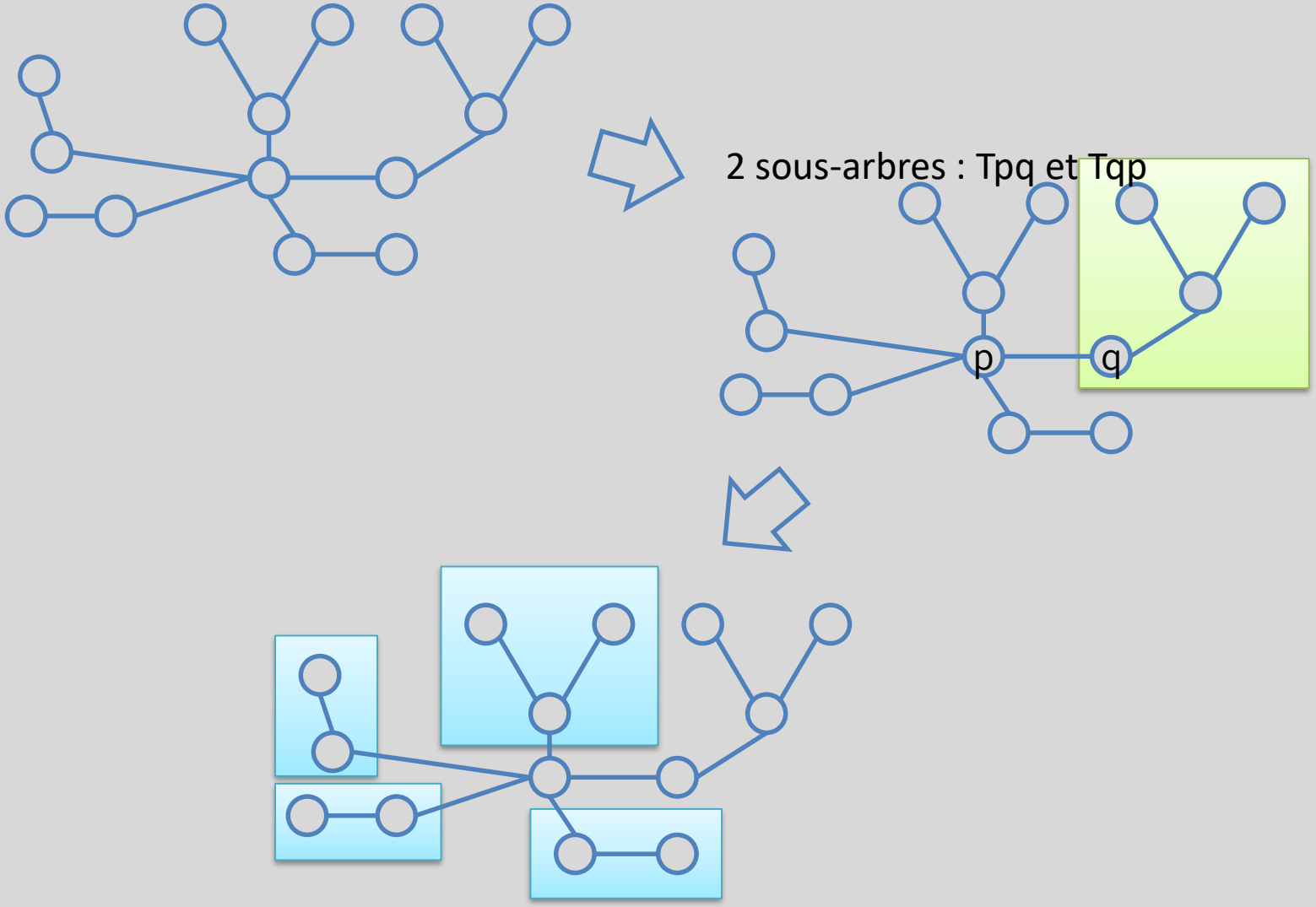
Anneau avec régénération - élection

Propagation Anneau avec régénération du jeton

```
var ci_p : int init p ; //identité de p – qui peut changer
    acn_p : int init udef ; //identité du nœud
    win_p : int init udef ; //identité du winner – régénération du jeton
    state_p : (active, passive, leader, lost) init active ; //statut de p
if p is initiator then state_p=active
else state_p=passive
while win_p == udef do
        if state_p==active then
                send ⟨one, ci_p⟩
                receive ⟨ one, q ⟩
                acn_p =q //enregistre l'identité du dernier processus actif
                if acn_p == ci_p then // si l'identité retournée a survécu au round
                        //on annonce à tous les nœuds que p est le leader
                        send ⟨smal, acn_p⟩
                        win_p = acn_p
                        receive ⟨ smal, q ⟩
                        else // acn_p est l'identité d'un voisin
                        send ⟨two, acn_p⟩
                receive ⟨ two, q ⟩ // on peut recevoir une autre identité
                if acn_p < ci_p and acn_p <q then //leader est le plus petit id reçu
                        ci_p = acn_p
                else state_p=passive
        else // passive – on relaie
                receive ⟨ one, q ⟩
                send ⟨one, q ⟩
                receive ⟨ m ⟩ // relaie qui peut être soit two soit smal
                send ⟨m ⟩
                if m is ⟨smal, q ⟩ then win_p =q
        if p==win_p then state_p=leader
        else state_p=lost
end
```

Peterson/Dolev-Klawe-Rodeh Algorithm, 1982

Algorithme de front d'onde - Arbre



Propagation dans un arbre

Arbre (graphe orienté)

//Neigh ➔ Voisin q de p

var ***$rec_p[q]$ for each $q \in Neigh_p$: boolean init false ; //initialisation de la variable***

// $rec_p[q]$ est vraie si p a reçu un message de q

while count(q where $rec_p[q]$ is false) > 1 do //nous n'avons pas reçu de message de q

receive ⟨tok⟩ from q ; // bloquant

$rec_p[q]$=true ;

//>1 ➔ maintenant dans le tableau *$rec_p[q]$, il y a 1 q_0 avec $rec_q[q_0]$=false*

send ⟨tok⟩ to q_0 with $rec_p[q_0]$ is false

receive ⟨tok⟩ from q_0

$rec_p[q_0]$ =true

//l'état du nœud p est qu'il est maintenant actif/vivant à la racine de son sous-arbre

decide //peut devenir leader

//informe les autres processus de la decision

forall q ∈ ***$Neigh_p$: $q \neq q_0$ do send ⟨tok⟩ to q***

Arbre avec régénération décentralisée

Arbre (graphe orienté – connait le père)

var rec_p **:** ***int init 0 ; //compte le nombre de message***
$father_p$ *: int* ***init udef ;***
//Pour l'initiateur
forall q ∈ $Neigh_p$ ***do*** **send** ⟨tok⟩ to q;
while $rec_p < count(Neigh_p)$ ***do***
receive ⟨tok⟩
$rec_p = rec_p + 1$
decide
end
//Pour les non-initiateurs
receive ⟨tok⟩ from neighbor q
$father_p = q$
$rec_p = rec_p + 1$
forall q ∈ $Neigh_p$ ***: q ≠ father_p do*** **send** ⟨tok⟩ to q
while $rec_p < count(Neigh_p)$ ***do***
receive ⟨tok⟩
$rec_p = rec_p + 1$
send ⟨tok⟩ to $father_p$
end

Aussi appelé algorithme Echo

Propriétés génériques

PIF – Propagation of Information with Feedback pour synchronisation d'état suite à une diffusion

- Propagation avec rétroaction est un algorithme de front d'onde
- Tous les algorithmes à front d'onde peuvent faire du PIF

SYN – synchronisation globale (barrière)

- Chaque algorithme SYN est un algorithme de front d'onde
- Tous les algorithmes à front d'onde peuvent faire du SYN

Infinum – recherche d'un minimum d'un sous-ensemble

- Chaque algorithme Infinum est un algorithme de front d'onde
- Tous les algorithmes à front d'onde peuvent faire de l'évaluation Infinum

Opérateurs de réduction - Infimum

Sur un ensemble de nœud X, on peut mettre une relation d'ordre. Si on considère un sous-ensemble P (ie Processus Actifs), il existe toujours une relation d'ordre (et donc un infimum).

Théorème Infimum : soit l'opérateur ⊛ sur P

- ⊛ est commutatif : a ⊛ b = b ⊛ a
- ⊛ est associatif : (a ⊛ b) ⊛ c = a ⊛ (b ⊛ c)
- ⊛ est indempotant : a ⊛ a = a

Par identification on trouve que les opérateurs :

and, or, min, max, pgdc, ppmc, union, intersection d'un ensemble de valeurs partagées dans P sont compatibles avec ⊛ ➔ MPI_reduce

Le calcul de ⊛ peut se faire en 1 seul front d'onde.

la somme n'est pas généralisable en front d'onde

➔ces opérateurs sont disponibles dans MPI_reduce

MPI et le concept d'algorithme d'onde

La communication MPI se divise en 3 catégories principales

Synchronisation des états

- Barrière

Transfert de données (Map)

- Diffusion 1 vers plusieurs, Diffusion tous vers tous
- Collecte de plusieurs vers un tableau
- Diffusion depuis un tableau vers plusieurs

Calcul de réduction (Reduce)

- Somme, Min, Max
- Collecte des états

MPI permet une liaison complète de tous les noeuds

Taxonomie des fautes

Bon fonctionnement de la transmission synchrone

- Pas d'invention de message → il existe une causalité
- Pas de perte → le canal est fiable
- Pas de duplication → pas d'écho de transmission

Les fautes courantes sont associées à :

- Un processus dit Byzantin
- Une connexion défectueuse
- Une perte de transmission intermittente (bagotter) – bagotage de l'interface
- Un brouillage
- Un crash de processus

Gestion des fautes

On conceptualise le système distribué S

Il est exécuté sur une plateforme P (Cloud, hybride, on premises)

Chaque exécution de S dans P:

Il existe une configuration qui cible un état stable.

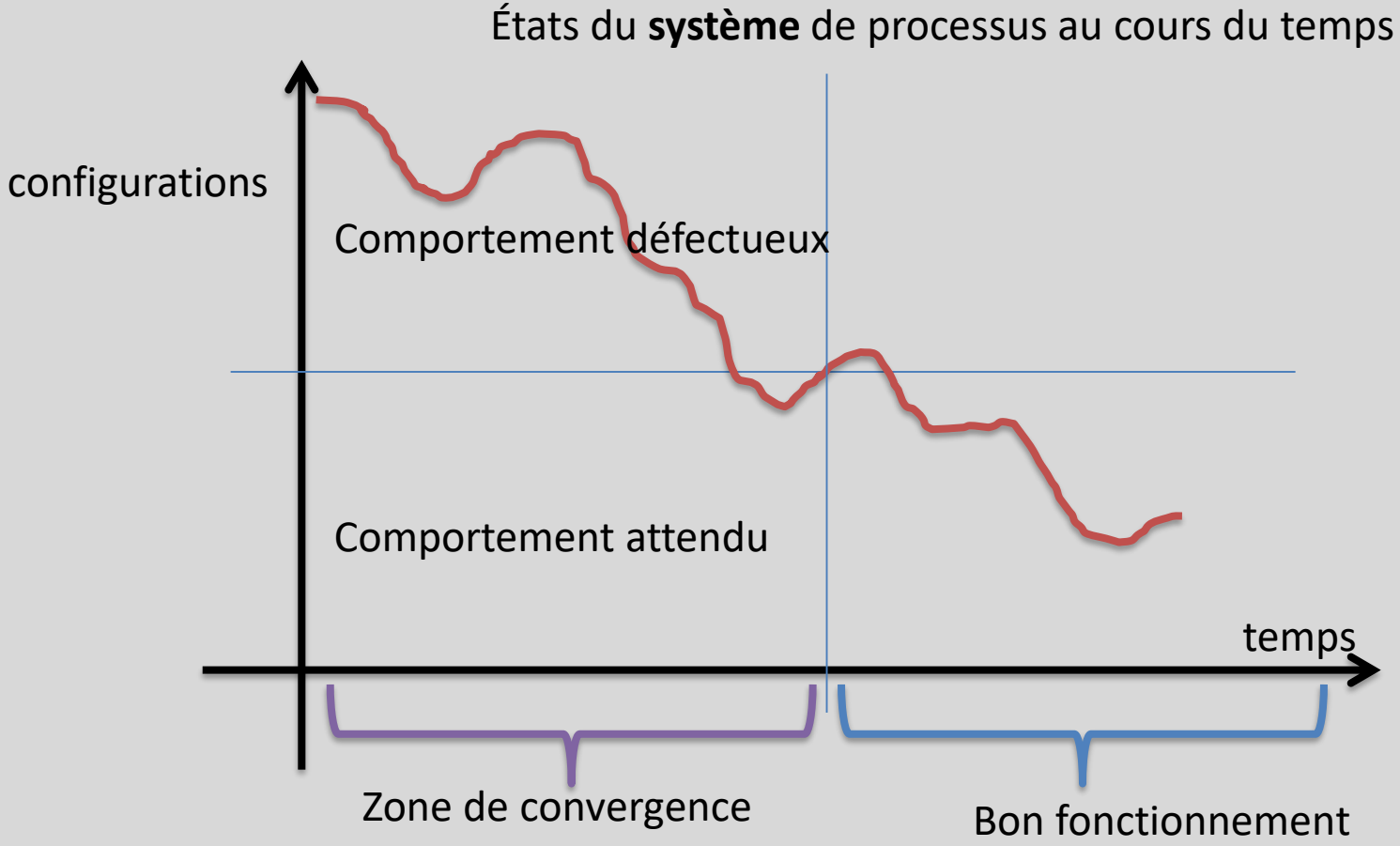
Chaque exécution à partir de la configuration, quelque soit l'état de démarrage aboutit à un état stable.

➔la configuration n'est pas vide

➔il peut exister des états défaillants atteignables à partir de la configuration

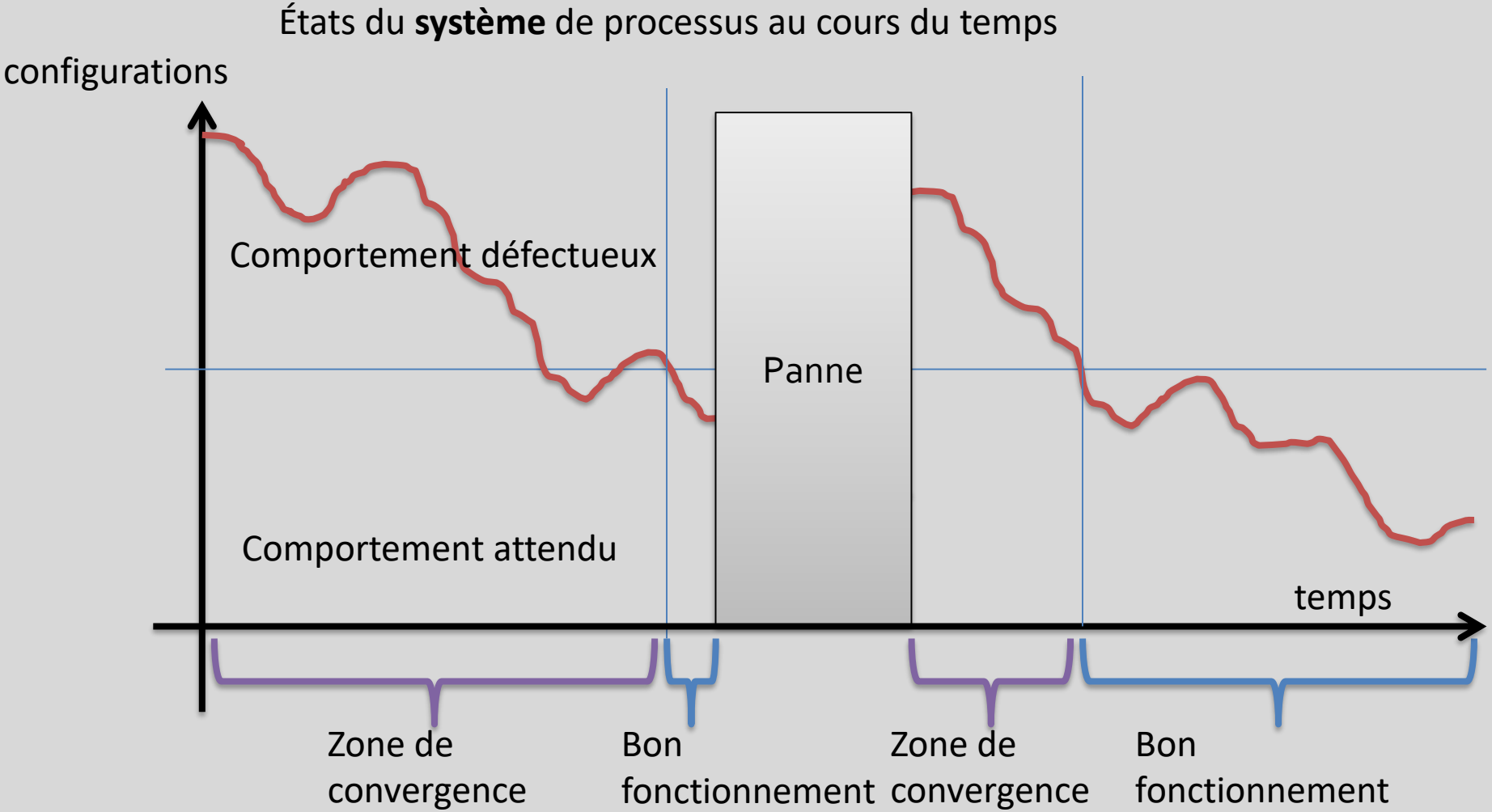
Peut-on concevoir des systèmes auto-stables?

Description du problème



Auto-stabilité

L'auto-stabilité est un mode de récupération



On peut implémenter l'autoréparation avec un algorithme à front d'onde.

➔2 approches pour l'autoréparation

- Une récupération au plus vite (avec zone de convergence)
- Une récupération consistante (sans zone de convergence)

➔d'un point de vue « implémentation Génie Log. » complexe

Calcul distribué

Data-worker

➔gestion centralisée, résilience nœud esclave

➔pas de résilience si le nœud maitre disparait

➔comment décrire la convergence d'état?