PF-979
La dernière trackbonification

Les rules et les dashboards “upstream”

Introduction

(Nolwenn)

- Autrefois :
  - les fichiers de surcharge (values.yaml) servant au déploiement des rules/dashboards upstream étaient nombreux et répartis sur plusieurs répertoires, avec jusqu'à 3 niveaux de surcharge ;
  - les rules upstream étaient déployées avec un script shell qui lance une commande Helm ;
  - les dashboards upstream étaient déployés avec Ansible/Concourse.
- État actuel :
  - les rules/dashboards upstream sont déployés avec une unique chart Helm ;
  - la chart est déployée avec Trackbone.


deploy.sh
imgflip.com

Le nouveau repo git

(Yves)

caascad/applications/caascad-prometheus-rules-dashboards-upstream

La nouvelle chart helm

(Yves)

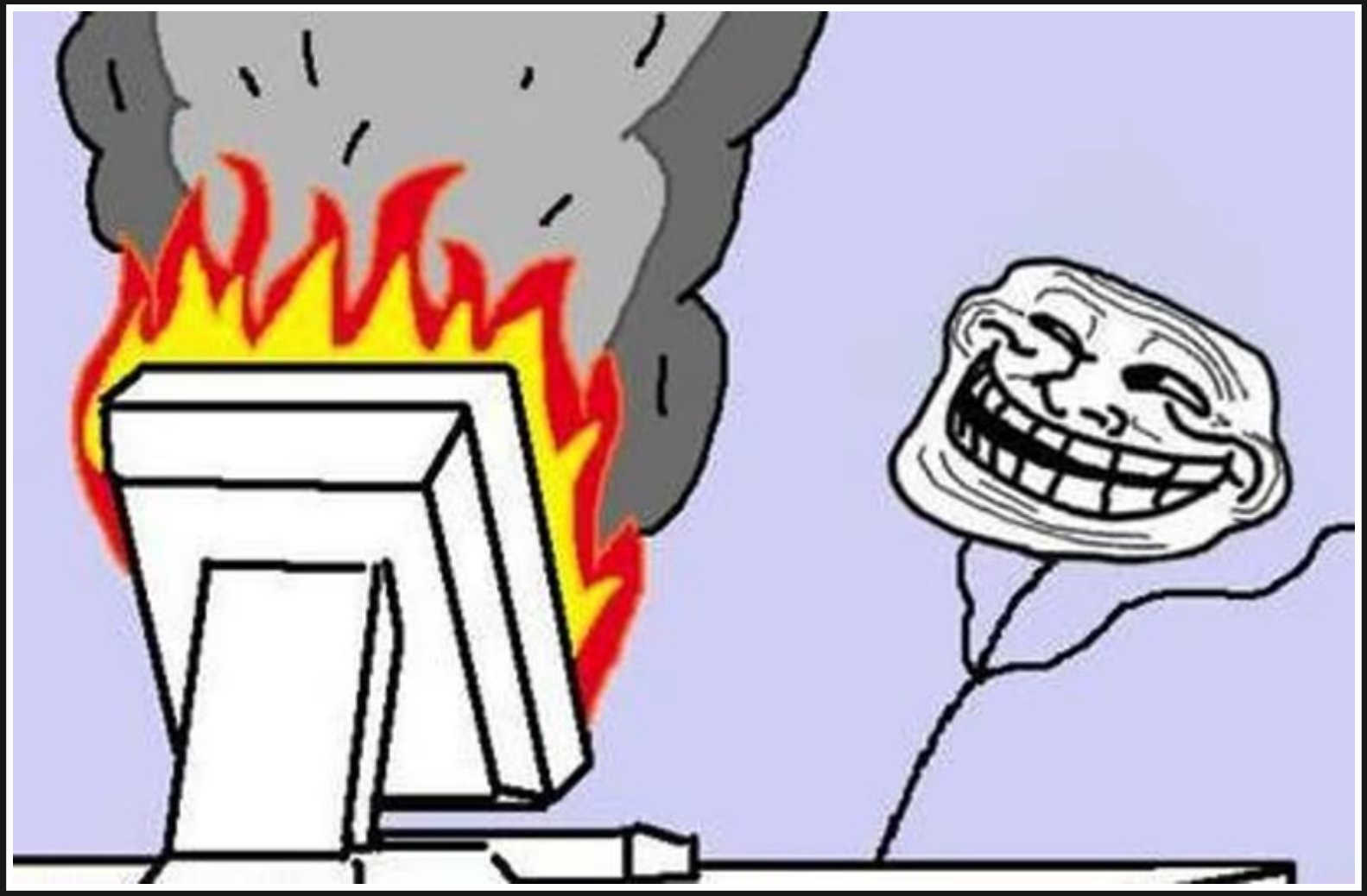
- Répertoire helm/.
- Deux répertoires :
  - `rules/` pour les rules,
  - `dashboards/` pour les dashboards.
- Un template pour les dashboards custom.
- Un répertoire `dashboards/` à la racine pour les dashboards custom.
- les dashboards custom doivent être supprimés à terme.

Mise à jour des rules+dashboards upstream

(Nolwenn)

Méthode :

1. lancer ici les scripts :
    - `generate_prometheus_rules.sh`,
    - `generate_dashboards.sh` ;
2. voir les différences avec `git diff` (les étudier…) ;
3. présenter les différences si elles sont significatives (Monitoring, Supervision) ;
4. `git commit && git push` puis MR puis tag ;
5. modifier la configuration envs-ng puis lancer Trackbone.



Le script de hack : rules (1/3) ➜ le wrapper

(Yves)

- Le wrapper avant et après.
- Retour au manifest codé en dur dans le script :
  - on perd des rules Grafana ;
  - et on les retrouve ici.
- Resynchro partielle avec upstream du script `sync_prometheus_rules.py` :
  - ajout des annotations ;
  - reprise des `condition_map`, `alert_condition_map` et `replacement_map` ;
  - suppression d'une modif nécessaire seulement aux rules Grafana ;
  - Remplacement de nombreux guillemets (`"` -> `'`) ;
  - suppression de code mort.
- Déplacement de certaines modifications dans `prometheusrules-patches/`.
- Création d'un système de patchs.

Le script de hack : rules (2/3)
➜ le système de patchs

(Yves)

- Pour chaque patch, le code retour est vérifié.
- Deux répertoires de patchs :
  - hack-patches/ : pour patcher le script de hack.
    - ⚠️ Il faudra éviter d'en ajouter chaque fois que c'est possible,
  - prometheusrules-patches/ : pour modifier les rules après leur génération.

Le script de hack : rules (3/3) ➜ méthodes pour patcher

(Yves)

- Avec **diff/patch** comme dans 10-alertmanager-add-message.sh pour le multi-ligne.
- Avec **sed** comme dans 10-change-k8s-naming-schema.sh pour le mono-ligne.
- Avec des scripts plus complexes (c’est faisable mais il n’y en a pas pour le moment).
- Chaque patch prend un argument : le répertoire contenant les patchs.

Le script de hack : dashboards (1/3) ➜ le wrapper

(Nolwenn)

- Le wrapper avant et après.
- Les patchs ne sont plus explicitement spécifiés : tous les scripts dans `dashboards-patches/` sont executés dans un ordre bien précis.

Le script de hack : dashboards (2/3) ➜ le système de patchs

(Nolwenn)

- Pour chaque patch, le code retour est vérifié.
- Liste des patchs :
  - dashboards-patches/* : pour modifier les dashboards après le lancement du script de hack `sync_dashboards.py`,
  - dashboard_json_to_configmap.sh : patch exécuté en dernier permettant de convertir les dashboards JSON en YAML.

Le script de hack : dashboards (3/3) ➜ les modifications

(Nolwenn)

- On préfixe les titres des dashboards avec `OBS /`.
- On préfixe les uids des dashboards avec `obs-`.
- On transforme `CHANGEME_DEFAULT_DATASOURCE` en variable Helm.
- On change la définition de la variable de dashboard `cc_prom_source` : utilisation d'une métrique au lieu d'une recording rule.
- Le dashboard Remote Write est ajouté à notre liste des dashboards upstream pour qu'on puisse le déployer là où nous en avons besoin.


Hacker
Zone

Conclusion sur les scripts de hack

(Yves)

- On utilise maintenant les mêmes mécanismes pour les rules et dashboards upstream.
- Les 2 scripts sont au même endroit et se lancent de la même façon.

values.yaml et envs-ng

(Nolwenn)

Avant :

- `helm-prometheus-rules/values.yaml`
- `values/default/infra/helm-prometheusrules-default.yaml`
- `values/default/infra-consumption/helm-prometheusrules-default.yaml`
- `values/default/cloud-app/helm-prometheusrules-default.yaml`
- `values/default/cloud-caascad/helm-prometheusrules-default.yaml`
- `values/default/cloud-client/helm-prometheusrules-default.yaml`

Après :

- un seul fichier values.yaml (et la configuration dans envs-ng) ;
- toutes les rules et tous les dashboards sont désactivés par défaut ;
- refacto des values : suppression de code mort ;
- deux principales parties :
  - values caascad
  - values upstream
- les infos de **caascad/monitoring** sont rationalisées dans la config envs-ng.

envs-ng et Trackbone

(Yves)

- En zone cloud, on déploie trois configurations pour chaque Prometheus (cloud-caascad, cloud-client, cloud-app).
- Toutes les rules/dashboards upstream sont dans un même namespace : `prometheusrules-dashboards-upstream`.
- La spécificité du filtrage d'alertes de la zone ocb-corp (namespaces des zones corp, prdcasa…) est gérée sous forme d'override.
- Nous en avons profité pour déployer le dashboard remote-write sur ocb-demo.

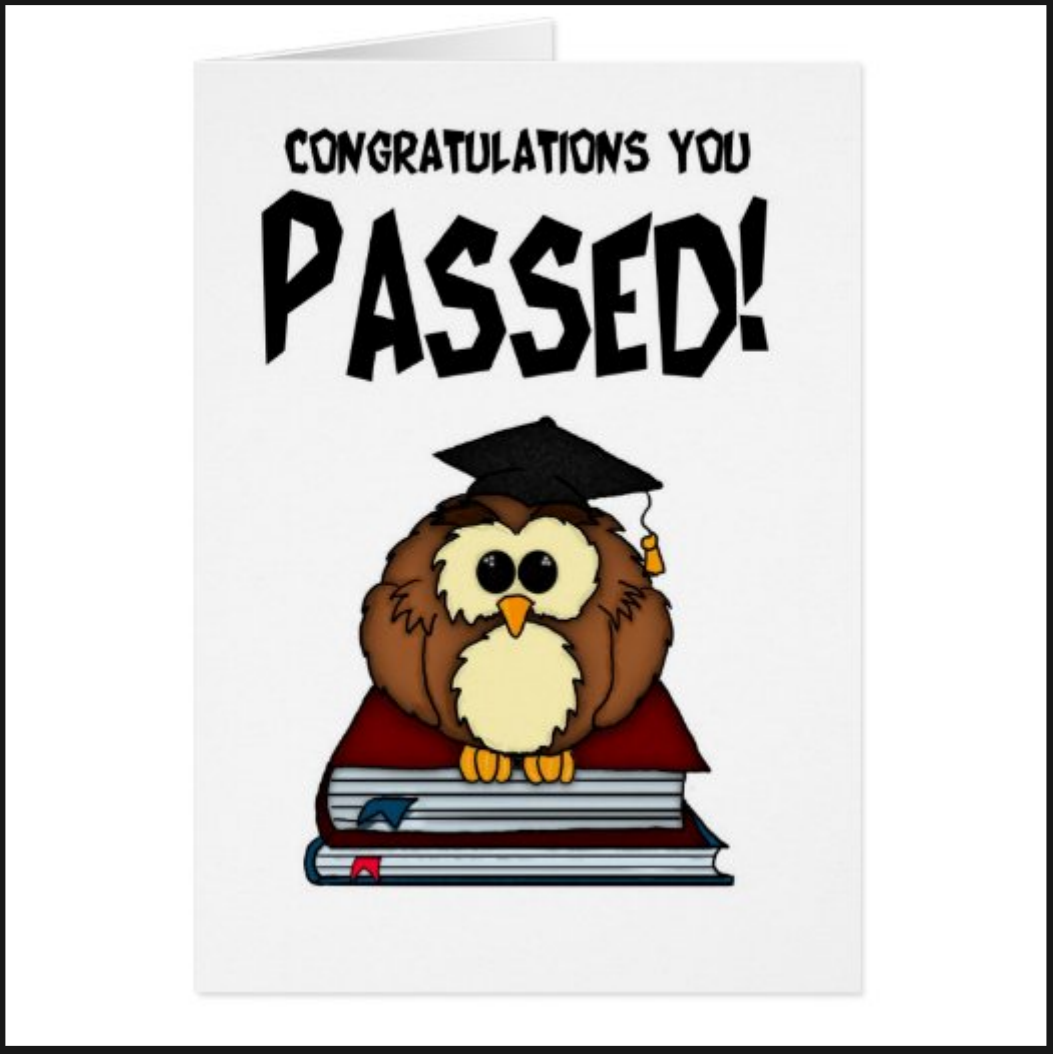
Tests fonctionnels

(Nolwenn)

Répertoire tests/

On vérifie :

- qu’aucune alerte n’utilise la fonction `absent()` ;
- qu’aucune alertingRule n’est déployée pour Prometheus cloud-app : avant la migration, des alertingRules étaient déployées ;
- la version de la chart helm (test usuel).


CONGRATULATIONS YOU
PASSED!

Le script de migration (1/2)
➜ Description

(Yves)

- Le script est déposé en pièce jointe du ticket PF-979.
- Une copie se trouve ici.
- Il se lance pour chaque zone cloud indiquée en argument.
- Il effectue toutes les opérations de migration (sauf un backup, opération oubliée) ainsi que les vérifications à chaque étape.
- La migration a consisté à lancer le script et vérifier qu'il allait jusqu'au bout sans erreur.
- Un script de rollback a également été écrit. Il est plus rudimentaire et a principalement servi aux tests (déploiement, rollback, déploiement, rollback…)

Le script de migration (2/2)
➜ Son fonctionnement

(Yves)

- Certaines variables peuvent être surchargées par une variable d'environnement (comme `STEP` ou `KSWITCH`).
- Grâce à `STEP` on peut relancer la migration depuis une étape précise (utile en cas d'échec).
- Préparation :
  - le script se connecte à l'environnement avec `KSWITCH` (`kswitch` pouvant être changé en `rswitch login`) ;
  - il vérifie qu'il est connecté sur le bon cluster (avec `kubectl get ingress -A`) ;
  - il vérifie qu'il est dans le bon répertoire pour Trackbone (`contexts/caascad`).
- Chaque étape est accompagnée d'une vérification. Le script s'arrête en cas d'erreur.
- En cas de problème, de nombreuses traces sont conservées dans un répertoire (diffs, output Trackbone…).
- À la fin, une différence avant/après a lieu sur les rules et les dashboards.

La migration…

(Yves)

- La migration a été conforme à nos attentes.
- Quelques échecs sur STG suite à des particularités de zones (shared, ngotalpha…).
- Un seul échec sur PRD : **ocb-be1** à cause de dashboards ayant été déployés sur cloud-client alors que c’était inutile.
- Quelques difficultés à cause de Concourse, mais non bloquantes.

CAASINC-1121

(Yves)

- CAASINC-1121
- Lors de la dernière vérification, des rules supprimées existent encore…
- Cause : oubli de vérification des rules hors namespaces clients (il manquait `dashboard`).
- Impact : nul car oubli de suppression des rules `caascad-k8s-recordingrules` .
- Fait amusant : ces rules oubliées ne sont pas utilisées !

caascad/monitoring : les restes du passé

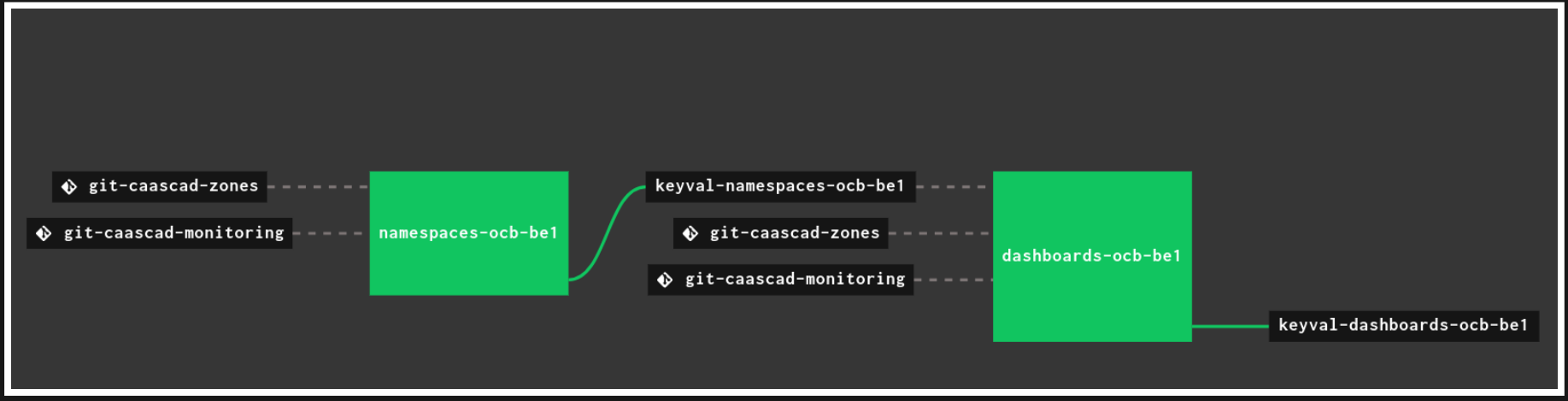
(Nolwenn)

- Le repo caascad/monitoring aujourd'hui.
- Il reste 4 ou 5 docs à migrer.
- Il reste 2 jeux de scripts (`get_rl` et `metricscli`).
- PF-1371 dernier ménage et archivage du repo.

Les pipelines Concourse/Ansible

(Yves)

- Ils n’existent plus ! Dernières captures d’écran dans CAASCHR-1777.
- Nous déployons tout avec Trackbone, enfin !
- La dette technologique avec Cue-0.2.2 n’est plus.
- La dette technologique avec l’image `caascad/ansible` qui posait problème à certains n’est plus.



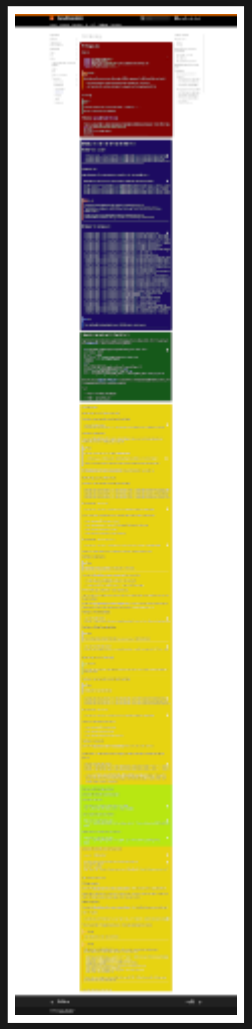
La documentation

(Nolwenn)

- Déploiement d’une nouvelle zone cloud simplifié. Il n’y a plus que quatre parties :
    1. préparatifs,
    2. déploiement avec Trackbone,
    3. déploiement des tests fonctionnels ,
    4. vérifications (automatiques et manuelles).
- Procédure pour upgrader les prometheusrules/dashboards upstream modifiée.
- Autres modifications mineures.

La documentation de déploiement des zones

(Nolwenn)



Les PF : état des lieux (1/2)

(Yves)

- Les PF restants pour les rules+dashboards upstream (et 4 custom n'ayant rien à faire là) :
  - PF-106 : migration des derniers dashboards custom :
    - 2 sous-tickets de discussion et d'architecture,
    - 4 sous-tickets, un par dashboard custom à sortir du repo ;
  - PF-1370 : Resynchronisation du script sync_prometheus_rules.py avec "upstream" ;
  - PF-977 vs PF-980 :
    - PF-977 : Refacto script hack pour générer les rules+dashboards upstream (avec gestion d'un commitID),
    - PF-980 : CI : récupération récurrente des rules+dashboards upstream ;
  - PF-970 : [Prometheus]Update evaluation interval pour les alerting/recording rules ;
  - PF-953 : [Prometheus] Update RL CPU/memory.
- Les PF fermés :
  - PF-110 : Trackbonifier le dashboard karma-alerts.json (il manquait la "review" depuis le 14/01/2022) ;
  - PF-420 : Cleanup images dockers (il manquait la "review" depuis le 30/03/2022) ;
  - PF-442 : Supprimer dashboard kubernetes-alertmanager-overview de grafana client (Traité le 21/09/2022 en ayant oublié l'existence du ticket) ;
  - PF-113 : Trackbonifier les dashboards storage-overview-* (Annulé : les dashboards ont été supprimés).

Les PF : état des lieux (2/2)

(Yves)

- Les nouveaux PF issus de PF-979 :
  - PF-1382 : monitoring de Kube-State-Metrics ;
  - PF-1371 : archivage du repo caascad/monitoring ;
  - PF-1370 : Resynchronisation du script sync_prometheus_rules.py avec “upstream”.
  - PF-1393 : Alerter lorsqu’un alertmanager tombe

Conclusion

(Nolwenn)

- Le repo caascad/monitoring est (quasi) vide. Encore un petit pas pour l'archiver…
- Tout le monitoring est enfin déployé uniquement avec Trackbone.
- Deux dettes technologiques supprimées.
- Diminution significative des problèmes de connaissances spécifiques et peu partagées sur les rules+dashboards upstream.
- Notre nouvel enfer de déploiement des zones : les vérifications manuelles.

Remerciements

- Nolwenn : Yves
- Yves : Nolwenn et Andelaï (incident PPDTAC en parallèle)
- Vous : pour votre attention 😀