Rocket.Chat

Situation initiale

La prod actuelle se trouve sur `escandalo`, une machine mise à disposition par Yulpa, et uniquement accessible en IPv4 (`185.217.152.58`). Cette machine n'est d'ailleurs plus présente dans l'infra maintenue de Yulpa ; pour preuve le reverse DNS utilisé par ces derniers : `58.152.217.185.not.updated.as199712.fr`.

Il n'est pas non plus possible de mettre à jour sa base (son OS), car il s'agit d'un conteneur LXC. Tenter de le mettre à jour casse l'ABI de virtualisation utilisée sur l'hôte et le conteneur ne redémarre plus.

Cette machine dispose de la version de Rocket.Chat 3.0.11 publiée le 2020-04-03 (src.). Elle a été installée selon une installation standard, sans Docker, mais via un rôle Ansible. (src.)

Cette machine est malheureusement infectée par un crypto miner qui accapare souvent toutes les ressources de la machine si bien qu'il faille la redémarrer régulièrement. Ce crypto miner se serait installé via le fait que la base de données MongoDB ait été rendue accessible publiquement (port 27017). Bien que ce port soit maintenant bloqué, le mal est fait et le crypto miner persistant. Il n'est d'ailleurs pas détectable, car il s'est greffé à pas mal de libs systèmes (dont les `coreutils`).

En outre, il est fort à parier que le serveur soit membre d'un botnet, car, régulièrement, une charge utile s'ouvre sur le port 2000 (en TCP). Il s'agit vraisemblablement du port utilisé par un serveur distant de type « Command and control » (src.). Nous avons détecté ce comportement via l'outil de Monitoring fourni par Shodan. (src.)

En outre, le certificat de cette machine retourne un domaine pour `chat-temp.lghs.be` qui semble indiquer que le chat était précédemment accessible par cette adresse également. Voici le rapport d'état qu'on reçoit par mail pour ce souci (toujours via Shodan.io) :

```
185.217.152.58


// Trigger: ssl_expired
// Port: 443 / tcp
// Hostname(s): 58.152.217.185.not.updated.as199712.fr, chat-temp.lghs.be
```

```
// Timestamp: 2022-12-17T00:11:09.909515
// Alert ID: lghs-chat1 (IJCF1NPFL1DV31Q0)
```

Plan de migration

Bien que l'objectif final soit de permettre une migration à Mattermost, l'état du serveur actuel est tel qu'une migration directe à Mattermost dans des conditions instables (à cause du crypto miner) n'est pas possible sans prendre le risque d'une corruption de données.

De même, il n'existait pas avant ce projet de migration de script permettant un import de données au sein de Mattermost. Ce dernier, lorsque la fonctionnalité de « compliance report » est active ne permet pas d'insérer des données en conservant la date de publication. (Notons que la fonctionnalité de confirmité - « compliance report » - est activée par défaut et ne peut, à notre connaissance, pas être désactivée facilement sur les versions actuelles de Mattermost). Insérer des données dans le passé est donc impossible. Pour qu'elles soit conservées, il faut que les données historiques soient migrées directement dès le début avant même que de nouvelles données soient insérées.

La migration vers Mattermost doit donc se faire en 2 étapes :

Phase 1 : Upgrade à la dernière version de Rocket.Chat ce qui permet :

1. D'avoir un serveur stable qui ne soit plus vulnérable et à partir duquel continuer la migration
2. D'activer la version Entreprise de Rocket.Chat gratuitement pour 30 jours afin de bénéficier de la levée des limites en matière de notifications push. En effet, la version Community de Rocket.Chat est désormais bridée à 1000 notifications push par mois. Il est nécessaire de passer à la version Entreprise pour lever cette limitation. (src.)

Phase 2 :

1. Migration des données (canaux, chats, threads, fichiers joints et émojis)
2. Installation d'une nouvelle configuration oAuth sur Keycloak
3. Migration des bots vers Mattermost.

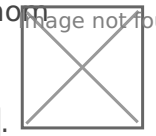
Déploiement d'une nouvelle machine

Bien qu'on dispose d'une machine offerte par Hivane Network (src.) pour notre nouvelle machine de prod, notre phase 1 va nécessiter une machine temporaire. Pour ce faire, nous allons déployer une machine sur Scaleway.

Nous utilisons le compte personnel de William Gathoye.

Pour que les autres membres du LgHS puissent y accéder, voici les étapes de création que nous avons suivies.

1. Connectez-vous à un compte existant sur `https://console.scaleway.com`.
2. Dans la barre du haut, dans `Organization Dashboard`, cliquez sur le lien `Create Project`.

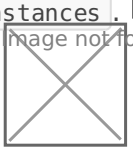

3. Spécifiez le nom du projet (ici `lghs`) et cliquez sur le bouton `Create new project`.
4. Ajoutons maintenant les clés SSH des différents protagonistes du LgHS afin qu'ils puissent accéder à la machine que nous allons créer. Pour ce faire, cliquez sur votre nom d'organisation en haut à droite dans la barre du haut et cliquez sur `SSH Keys`.


5. Cliquez ensuite sur le bouton `Add a new SSH key`.
6. Créez ensuite une clé au format ed25519 avec la commande suivante (src.) :

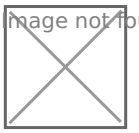
```
ssh-keygen -t ed25519
```

7. Dans le premier champ de la boite de dialogue suivante, collez la clé publique (fichier `.pub`) qui vient d'être générée.
8. Dans le second champ, collez un nom de clé pour savoir qui est qui, ici `william_gathoye_ssh_key_2021-10-23_lghs_ed25519` (avoir les prénoms + noms, la date de génération et le type de clé est une bonne idée, car ces informations ne seront plus affichées par la suite au sein de l'interface de Scaleway).


9. Cliquez enfin sur le bouton `Add an SSH key`

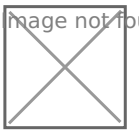

10. Rendez-vous maintenant dans la partie `Instances`.


11. Cliquez sur le bouton `Create an instance`.
12. Sélectionnez la zone de `Paris 1`



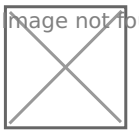
13. Sélectionnez le type de machine `Dev&Test`

14. Sélectionnez la gamme `DEV1-M`



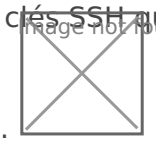
15. Choisissez `Debian Bullseye` (c'est-à-dire Debian 11)

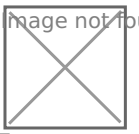
16. Nommez votre volume `lghs-chat`



17. Sélectionnez le volume qui vient d'être créé (`lghs-chat`)

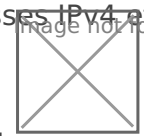
18. Nommez votre machine `lghs-chat`
19. Assurez-vous que les clés SSH que vous avez créées précédemment soient bien toutes visibles à cette étape.



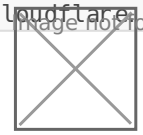


20. Confirmez la création de l'instance par `Create a new instance`

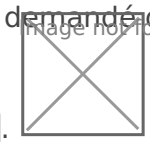
21. Attendez quelques secondes que la machine se crée.
22. Vous allez tomber sur cet écran avec un résumé de la configuration de la machine. Ce qui nous intéresse ici, ce sont les adresses IPv4 et IPv6. Cliquez sur les boutons ad-hoc pour les copier dans votre presse papier.



23. Assurez-vous d'avoir vos identifiants à portée de main ou d'avoir une délégation d'accès sur le compte Cloudflare du Liege Hackerspace. Connectez-vous à l'interface de Cloudflare via `https://dash.cloudflare.com` afin de configurer les entrées DNS relatives à cette nouvelle instance.



24. Tapez votre code d'identification multifacteur (cette option devrait être activée pour votre compte; dans le cas contraire, ça représenterait un risque de sécurité qu'il serait nécessaire de corriger).

25. Dans le cas où vous administrez plusieurs comptes Cloudflare à l'aide des mêmes identifiants, il se peut qu'il vous soit demandé de choisir le compte approprité. Dans pareil cas, ici choisissez `Liège HackerSpace`. 

26. Le LgHs dispose de plusieurs noms de domaines. Les sous-domaines placés sur `lghs.be` désignent les URLs que les utilisateurs finaux devront taper pour accéder au service, là où les domaines sur `lghs.space` concernent les domaines techniques. Créons ici un sous-domaine technique. Pour ce faire, choisissez `lghs.space`. 

27. Assurez-vous que le bon domaine ait été sélectionné (1); choisissez le menu `DNS` (2); le sous-menu `Records` (3) devrait alors se sélectionner automatiquement; cliquez sur le bouton `Add record` (4), sélectionnez le type `AAAA` (5), spécifiez un sous-domaine (ici `chat-migration`) (6); spécifiez l'IPv6 que vous avez copiée précédemment à partir du panneau de résumé chez Scaleway (7); désactivez Cloudflare en proxy de l'adresse IP (8); et cliquez sur le bouton `Save`. Réitérez l'opération pour l'adresse IPv4 précédemment copiée (même procédure, changez juste le type en `A`). 

Connexion à la machine

Créez une entrée dans votre fichier `~/.ssh/config` :

```
Host lghs-chat-prod
    User root
    Hostname lghs-chat-prod.lghs.space
    Port 22
    IdentityFile ~/.ssh/keys/william_gathoye_ssh_key_2021-10-23_lghs_ed25519
```

Attention, notez que si vous décidez de placer par la suite la machine derrière Cloudflare, Cloudflare ne pourra pas par défaut jouer le rôle de proxy SSH, il faudra alors remapper le domaine sur les adresses IP réelles et non celles de Cloudflare. Pour ce faire il faudra placer les adresses IP réelles dans votre fichier `hosts`. C'est la seule méthode valable, OpenSSH est alors assez malin pour choisir la bonne adresse IP selon la stack IP employée (il comprend le fichier `hosts` et ne prendra donc pas le premier venu). (src.)

Voici un exemple avec `vahine` :

```
/etc/hosts
```

```
[...]
2001:bc8:600:1b1e::1 lghs-chat-prod.lghs.space
163.172.177.119 lghs-chat-prod.lghs.space
[...]
```

```
~/.ssh/config
```

```
Host lghs-chat-prod
    User root
    Hostname lghs-chat-prod.lghs.space
    # Add an entry in /etc/hosts in order to force the resolution to the
    # following non Cloudflare proxy addresses
    # 2001:bc8:600:1b1e::1 lghs-chat-prod.lghs.space
    # 163.172.177.119 lghs-chat-prod.lghs.space
    Port 22
    IdentityFile ~/.ssh/keys/william_gathoye_ssh_key_2021-10-23_lghs_ed25519
```

Déploiement d'un Rocket.Chat 3.0.12

Bien que la version actuellement en production soit la 3.0.11, nous allons installer une 3.0.12, car la 3.0.11 ne dispose pas une image Docker disponible. (src.)

Connectez-vous à ladite machine et préparons l'environnement Docker.

Commencez par mettre à jour la machine et à installer les outils dont nous aurons besoin :

```
apt update && apt dist-upgrade -y && apt install -y tmux vim rsync
```

Lancez un tmux, très utile pour conserver le shell lors de la migration, et éviter que celle-ci soit perdue en cas de déconnexion.

```
tmux new -s wget
```

Installons le moteur Docker pour Debian 11 (src.) :

```
apt-get remove docker docker-engine docker.io containerd runc
apt-get update
apt-get install -y ca-certificates curl gnupg lsb-release
mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
apt-get update
apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Une fois installé, vérifiez que la version 2 de Compose a bien été installée (en effet nous utilisons cette version plutôt que la version 1 dépréciée) :

```
$ docker compose version
Docker Compose version v2.14.1
```

Installez le frontend NGINX (dépôt Debian) et certbot via snap (la méthode officielle pour avoir la dernière version) : src.

```
apt install -y nginx
apt install -y snapd
snap install core
snap refresh core
snap install --classic certbot
ln -s /snap/bin/certbot /usr/bin/certbot
snap set certbot trust-plugin-with-root=ok
snap install certbot-dns-cloudflare
```

Créez les répertoires dont on a besoin pour le déploiement Docker :

```
mkdir /srv/chat.lghs.be
cd /srv/chat.lghs.be/
```

Placez dans ce dossier le fichier Docker Compose (`docker-compose-prod-3.0.12.yml`) suivant (basé sur l'ancien Docker Compose officiel du tant de la 3.0.11 (src.)):

```
version: "3.9"
```

```
services:
  #  rocketchat:
  #    image:  rocketchat/rocket.chat:3.0.12
  #    command:  >
  #      bash -c
  #        "for i in `seq 1 30`; do
  #          node main.js &&
  #          s=$$? && break || s=$$?;
  #          echo \"Tried $$i times.  Waiting 5 secs...\";
  #          sleep 5;
  #        done; (exit $$s)"
  #    restart:  unless-stopped
  #    volumes:
  #      - "/srv/chat.lghs.be/data/www:/app/uploads/"
  #    environment:
  #      - PORT=3000
  #      - ROOT_URL=http://localhost:3000
  #      - MONGO_URL=mongodb://mongo:27017/rocketchat
  #      - MONGO_OPLOG_URL=mongodb://mongo:27017/local
  #    depends_on:
  #      - mongo
  #    ports:
  #      - 3000:3000

  mongo:
    image:  mongo:4.2.22
    restart:  unless-stopped
    volumes:
     # - ./data/db:/data/db
     - "/srv/chat.lghs.be/data/db:/data/db/"
     - "/srv/chat.lghs.be/backups:/backups/"
    # --smallfiles not supported with mongo 4.2
    # --storageEngine=mmapv1 deprecated in mongo 4.2
    #command:  mongod --smallfiles --oplogSize 128 --replSet rs0 --storageEngine=mmapv1
    #command:  mongod --oplogSize 128 --replSet rs0 --storageEngine=mmapv1
    command:  mongod --oplogSize 128 --replSet rs0

  # this container's job is just run the command to initialize the replica set.
  # it will run the command and remove himself (it will not stay running)
  mongo-init-replica:
```

```
    image:  mongo:4.2.22
    command:  >
      bash -c
        "for i in `seq 1 30`; do
          mongo mongo/rocketchat --eval \"
            rs.initiate({
              _id: 'rs0',
              members: [ { _id: 0, host: 'localhost:27017' } ]})\" &&
          s=$$? && break || s=$$?;
          echo \"Tried $$i times.  Waiting 5 secs...\";
          sleep 5;
        done; (exit $$s)"
    depends_on:
      - mongo
```

Le fait que le service Rocket.Chat soit commenté est tout à fait voulu. Ceci est nécessaire, car si le service Rocket.Chat est lancé alors qu'on réimporte les données, trop de mémoire et de puissance seront nécessaires, car Rocket.Chat passera continuellement son temps à réindexer les données menant inexorablement à un OOM.

Transfert des données

Connectez-vous sur `chat-migration` et assurez-vous que la connexion par mot de passe soit autorisée et que la machine dispose bien d'un mot de passe sur le compte root :

```
/etc/ssh/sshd_config
```

```
+++ sshd_config 2022-12-17 05:31:20.468748364 +0000
@@ -32,6 +32,7 @@

 #LoginGraceTime 2m
 #PermitRootLogin prohibit-password
+PermitRootLogin yes
 #StrictModes yes
 #MaxAuthTries 6
 #MaxSessions 10
@@ -56,6 +57,7 @@

 # To disable tunneled clear text passwords, change to no here!
```

```
 #PasswordAuthentication no
+PasswordAuthentication yes
 #PermitEmptyPasswords no

 # Change to yes to enable challenge-response passwords (beware issues with
```

```
# passwd
New password:
Retype new
password:

passwd: password updated successfully
# systemctl restart sshd
```

Se connecter sur `escandalo`. Se déplacer dans le home de l'utilisateur et exporter la base de données qui, par défaut, donnera un dossier nommé `dump` dans le répertoire de travail actuel :

```
$ cd /home/willget
$ mongodump
2022-12-17T06:22:29.149+0100    writing admin.system.version to dump/admin/system.version.bson
[...]
2022-12-17T06:22:46.957+0100    [####....................]
rocketchat.rocketchat_uploads.chunks  2366/11555  (20.5%)
```

Transférer le dossier sur `chat-migration` (< 1 min de transfert):

```
rsync -av --info=progress2 dump root@chat-migration.lghs.space:/srv/chat.lghs.be/dump-2022-12-17
```

```
sshfs lghs-chat-migration:/srv/chat.lghs.be/ ./chat-migration/
```

Se connecter sur `chat-migration`.

docker compose -f docker-compose-prod-3.0.12.yml up -d

et entrer dans le conteneur relatif à la base mongodb.

```
docker exec -it chatlghsbe-mongo-1 /bin/bash
root@79f9ab2ea43a:/# cd /backups/
root@79f9ab2ea43a:/# mongorestore --drop dump-2022-12-17/dump
[...]
```

```
2022-12-17T06:18:34.572+0000    [###############.........]
rocketchat.rocketchat_uploads.chunks  1.45GB/2.42GB  (60.0%)
2022-12-17T06:18:34.572+0000    [######################..]
rocketchat.rocketchat_message_read_receipt  40.0MB/43.4MB  (92.4%)
[...]
2022-12-17T06:19:06.402+0000    finished restoring rocketchat.rocketchat_uploads.chunks (490
documents, 0 failures)
2022-12-17T06:19:06.404+0000    513657 document(s) restored successfully. 0 document(s) failed
to restore.
```

L'option `--drop` a été nécessaire car nous rencontrions l'erreur suivante qui pouvait être causée à cause d'index invalides :

```
rocketchat rocketchat_uploads.chunks.bson: connection(localhost:27017[-5]) incomplete read of
message header: EOF
```

Fonctionne uniquement, si on commente le service Rocket.Chat du fichier Docker Compose pour éviter qu'il ne démarre.

La stack trace suivante est tout à fait normale. Il faut juste laisser mongo plus de temps à démarrer. (src.)

```
[...]
$MONGO_OPLOG_URL must be set to the 'local' database of a Mongo replica se
[...]
```

https://docs.rocket.chat/quick-start/environment-configuration/configuring-ssl-reverse-proxy

```
[...]
Unsafe permissions on credentials configuration file: /etc/letsencrypt/cloudflare-api-
token.ini
[...]
```

chmod 600 /etc/letsencrypt/cloudflare-api-token.ini

Upgrade vers 3.18.2

Pour l'upgrade, il faut passer par chaque version majeure obligatoirement. (src.)

Passer à la dernière vesion de la branche 4.x ne fonctionne pas directement à cause d'étapes de schéma de migration qui ne sont plus présentes dans la version 4.

Cependant, passer directement de la 3.0.12 à la 3.18.2 ne fonctionne pas non plus sans souci, car la migration de schéma de DB ne passe pas correctement au schéma de base de données version 231. Il faut pour ce faire désactiver temporairement les modules oAuth/SAML. La migration 231 correspond à une requête mongo relative au plugin oAuth :

```
const query = {
    _id: { $in: [/^Accounts_OAuth_(Custom-)?([^-_]+)$/, 'Accounts_OAuth_GitHub_Enterprise'] },
    value: true,
};
```

(src.) L'erreur semble connue. (src.)

Considérons que RocketChat 3.0.12 est en cours d'exécution. Stoppons d'abord le conteneur Docker de Rocket.Chat tout en laissant Mongo tourner :

```
docker stop chatlghsbe-rocketchat-1
```

Désactivons temporairement le plugin de Rochet.Chat relative à l'auth oAuth :

```
root@lghs-chat-test:/srv/chat.lghs.be# docker exec -it chatlghsbe-mongo-1 /bin/bash
root@e80c9dfa0fb1:/# mongo
rs0:PRIMARY> use rocketchat
rs0:PRIMARY> var col = db.getCollection('rocketchat_settings')
rs0:PRIMARY> col.findOne({_id: { $in: [/^Accounts_OAuth_(Custom-)?([^-_]+)$/,
'Accounts_OAuth_GitHub_Enterprise'] }, value: true})
{
        "_id" : "Accounts_OAuth_Custom-Authlghsbe",
        "_updatedAt" : ISODate("2022-02-15T15:45:36.484Z"),
        "autocomplete" : true,
        "blocked" : false,
        "createdAt" : ISODate("2022-02-15T15:43:57.545Z"),
        "group" : "OAuth",
        "hidden" : false,
        "i18nDescription" : "Accounts_OAuth_Custom-Authlghsbe_Description",
        "i18nLabel" : "Accounts_OAuth_Custom_Enable",
        "packageValue" : false,
        "persistent" : true,
        "secret" : false,
```

```
        "section" : "Custom OAuth: Authlghsbe",
        "sorter" : 75,
        "ts" : ISODate("2022-02-15T15:43:57.546Z"),
        "type" : "boolean",
        "value" : true,
        "valueSource" : "packageValue"
}
```

La désactivation du plugin passe par cette commande :

```
rs0:PRIMARY> col.update({"_id" : "Accounts_OAuth_Custom-Authlghsbe"}, {$set: { "value" : false }})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Il est ensuite nécessaire de tuer les index de Rocket.Chat, sinon les migration de schéma échoueront. Pour ce faire, toujours à partir du shell Mondo DB :

```
db.rocketchat_nps_vote.dropIndexes()
db.users.dropIndexes()
db.rocketchat_room.dropIndexes()
db.rocketchat_message.dropIndexes()
db.rocketchat_integration_history.dropIndexes()
db.rocketchat_apps_logs.dropIndexes()
```

(src.)

Quittons le shell Mongo et le conteneur mongo et stoppons le reste de l'environnement Docker :

```
docker compose -f docker-compose-prod-3.0.12.yml down
```

Copions le fichier Docker Compose et changeons la version de Rocket.Chat :

```
cd /srv/chat.lghs.be
cp docker-compose-prod-3.0.12.yml docker-compose-prod-3.18.2.yml
```

```
--- docker-compose-prod-3.0.12.yml      2022-12-23 04:21:11.315632271 +0000
+++ docker-compose-prod-3.18.2.yml      2022-12-23 04:06:39.707945575 +0000
@@ -2,7 +2,7 @@

 services:
   rocketchat:
```

```
-    image: rocketchat/rocket.chat:3.0.12
+    image: rocketchat/rocket.chat:3.18.2
     command: >
       bash -c
         "for i in `seq 1 30`; do
```

```
docker compose -f docker-compose-prod-3.18.2.yml up -d
```

Une fois que `mongo-init-replica` est quitté, 20 secondes après, l'état de démarrage de Rocket.Chat peut être suivi avec :

```
$ docker logs -f chatlghsbe-rocketchat-1
[...]
➜ System ➜ startup
➜ +----------------------------------------------+
➜ |                SERVER RUNNING                |
➜ +----------------------------------------------
+


➜ |                                              |
➜ |  Rocket.Chat Version: 3.18.2                 |
➜ |       NodeJS Version: 12.22.1 - x64          |
➜ |      MongoDB Version: 4.2.22                 |
➜ |       MongoDB Engine: wiredTiger             |
➜ |             Platform: linux                  |
➜ |         Process Port: 3000                   |
➜ |             Site URL: https://chat.lghs.be   |
➜ |     ReplicaSet OpLog: Enabled                |
➜ |           Commit Hash: 03394ccaa5            |
➜ |        Commit Branch: HEAD
|


➜ |                                              |
➜ +----------------------------------------------+
```

Une fois que le serveur a redémarré, on applique la même recette que précédemment. On coupe Rocket uniquement en laissant Mongo tourner et on réactive le plugin oAuth.

```
root@lghs-chat-test:/srv/chat.lghs.be# docker stop chatlghsbe-rocketchat-1
root@lghs-chat-test:/srv/chat.lghs.be# docker exec -it chatlghsbe-mongo-1 /bin/bash
```

```
root@e80c9dfa0fb1:/# mongo
rs0:PRIMARY> use rocketchat
rs0:PRIMARY> var col = db.getCollection('rocketchat_settings')
rs0:PRIMARY> col.update({"_id" : "Accounts_OAuth_Custom-Authlghsbe"}, {$set: { "value" : true
}})
```

On quitte le reste de la stack Docker et on relance le tout :

```
root@lghs-chat-test:/srv/chat.lghs.be# docker compose -f docker-compose-prod-3.18.2.yml down
root@lghs-chat-test:/srv/chat.lghs.be# docker compose -f docker-compose-prod-3.18.2.yml up -d
```

Réactivation des notifications push

cf. screenshots

API d'enregistrement manuel (`api/v1/cloud.manualRegister`) semble être non dispo en 3.18.2, car l'appel du noeud retourne une 404. (src.)

Nouvelle passerelle enregistrée correctment et nouvelles CGU activées, mais dans les logs, on voit que le serveur a été précédemment enregistré sur le site cloud. Il faut retrouver l'accès à ce compte. De même lorsqu'on tente de se déconnecter, et qu'on reclique pour établir une nouvelle collection, Rocket réutilise les anciens identifiants sans nous laisser la possibilité d'en saisir des nouveaux (ou de nous indiquer quel était l'ancien qui a été employé). (src.)

En cliquant sur le bouton de test pour générer une notification sur les périphériques mobiles, on obtenait un message d'erreur indiquant qu'il n'y avait pas de jeton pour l'utilisateur en cours. Pour pallier ce problème, nous avons juste dû ous déconnecter de l'app mobile et nous reconnecter. Appuyer sur le bouton de test a alors fonctionné avec un toast indiquant que l'action s'était bien passée, mais nous ne recevions quand même pas la notification sur le mobile.

En regardant dans les logs, on a vu la raison :

```
root@chat-migration:/srv/chat.lghs.be# docker logs -f chatlghsbe-rocketchat-1
[...]
Push → info gateway rejected push notification. not retrying.
{

  statusCode:
422,
```

```
  content: '{"code":131,"error":"the amount of push notifications allowed for the workspace
was used","requestId":"290ba554-7c27-4286-8559-
633b2a29fd90","status":422}',
  headers:
{

    'access-control-allow-headers': 'Content-Type, Authorization, Content-Length, X-Requested-
With',
    'access-control-allow-methods': 'GET, PUT, POST, DELETE,
OPTIONS',

    'access-control-allow-origin':
'*',

    'access-control-expose-headers': 'Content-Type, Authorization, Cache-Control, Expires,
Pragma, X-powered-
by',
    'cache-control': 'private, no-cache, no-store, must-
revalidate',

    'content-length':
'154',

    'content-type': 'application/json; charset=utf-
8',

    date: 'Fri, 23 Dec 2022 05:26:47
GMT',

    expires: '-
1',

    pragma: 'no-
cache',

    vary: 'Accept-
Encoding',

    'x-powered-by': 'Rocket Fuel and
Rocketeers',
```

```
    connection:
'close'


},

  data:
{

    code:
131,

    error: 'the amount of push notifications allowed for the workspace was
used',

    requestId: '290ba554-7c27-4286-8559-
633b2a29fd90',

    status:
422


}

}
```

Une des solutions serait d'aller supprimer l'enregistrement en base de données selon les collections dont les noms commenceraient par `Cloud_*`. (src.)

Cependant, nous n'avons rien trouvé de tel en tentant de les lister.

```
rs0: PRIMARY> db.getCollectionNames()
```

src.)

```
db.rocketchat_settings.update({_id: /Cloud_.*/}, {$set: {value: ""}}, {multi: true})
```

Upgrade vers 4.8.6

Pour le passage de la 3.18.2 à 4.8.6, exécuter les commandes précédentes jusqu'il n'y ait plus de souci de migration de schéma de base de données.

Upgrade vers 5.4.1

On a obtenu la stack trace suivante :

```
[...]
ervers: Map(1)
{
                                                                    [41/1821]
      'localhost:27017' => ServerDescription {
        _hostAddress: HostAddress { isIPv6: false, host: 'localhost', port: 27017
},

        address: 'localhost:27017',
        type:
'Unknown',

        hosts:
[],

        passives:
[],

        arbiters:
[],

        tags:
{},

        minWireVersion: 0,
        maxWireVersion:
0,
```

```
        roundTripTime: -1,
        lastUpdateTime:
34892529,

        lastWriteDate:
0,

        error: MongoNetworkError: connect ECONNREFUSED
127.0.0.1:27017

            at connectionFailureError
(/app/bundle/programs/server/npm/node_modules/meteor/npm-
mongo/node_modules/mongodb/lib/cmap/connect.js:381:20)
            at Socket.<anonymous> (/app/bundle/programs/server/npm/node_modules/meteor/npm-
mongo/node_modules/mongodb/lib/cmap/connect.js:301:22)
            at Object.onceWrapper
(events.js:520:26)

            at Socket.emit
(events.js:400:28)

            at emitErrorNT
(internal/streams/destroy.js:106:8)

            at emitErrorCloseNT
(internal/streams/destroy.js:74:3)

            at processTicksAndRejections (internal/process/task_queues.js:82:21)

}

    },
[...]
```

```
config = rs.config()
config.members[0].host = 'chatlghsbe-mongo-1:27017'
rs.reconfig(config)
```

(src.)

La précédente version de la branche 4.x (la 4.8.6) indique que MongoDB version 4.2 est pris en charge. (src.) Nous n'avons donc pas à mettre à jour la version de Mongo à présent.

Crash lors de l'upgrade direct de la v. 3.0.12 à 4.8.6.

Tentative de migratiob sur 4.0.0

(src.)

```
cp docker-compose-prod-3.0.12.yml docker-compose-prod-4.0.0.yml
```

Your database migration failed: |
| Start date cannot be later than expire date

même erreur en 4.0

https://github.com/RocketChat/Rocket.Chat/releases/tag/4.8.7

Il a ensuite fallu délocker les migrations pour qu'elles puissent s'exécuter à nouveau :

```
rs0:PRIMARY> use rocketchat
rs0:PRIMARY> db.migrations.update({"_id": "control"}, {$set:{locked: false}})
```

(src.)

```
{"line":"120","file":"migrations.js","message":"Migrations: Migrating from version 230 ->
232","time":{"$date":1671721103672},"level":"info"}
{"line":"120","file":"migrations.js","message":"Migrations: Running up() on version
231","time":{"$date":1671721103675},"level":"info"}
{"line":"120","file":"migrations.js","message":"Migrations: Running up() on version
232","time":{"$date":1671721103685},"level":"info"}
{"line":"120","file":"migrations.js","message":"Migrations: Finished
migrating.","time":{"$date":1671721103742},"level":"info"}
```

Versions prises en charge

https://docs.rocket.chat/getting-support/enterprise-support

Old workspace: https://cloud.rocket.chat/workspaces/63a56340b3c77e000185eaa2

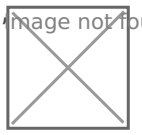
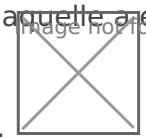
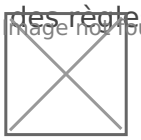
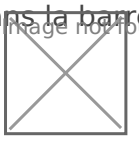
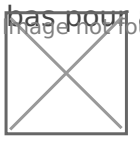
Firewalling

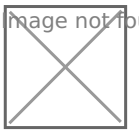
Au niveau de la machine

Au niveau de Scaleway

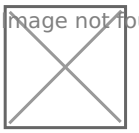
Le souci des dernières versions de Docker est qu'il modifie selon son bon vouloir un firewall qu'on aurait installé comme `firewalld`. Même si on ne l'utilise pas, Docker créera des CHAIN spéciales avec iptables/nftables, qui, en fonction des politiques de routing de la machine, pourrait entrainer certains ports internes des conteneurs à être exposés.

De façon a éviter ce cas de figure et de façon à éviter de reproduire la situation de compromission avec l'ancienne machine, il est fortement recommandé d'appliquer des règles de firewalling sur la machine au niveau réseau chez Scaleway.

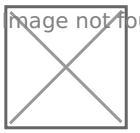
1. Retournez sur les instances via le lien `Instances` de la barre latérale de gauche, sélectionnez `Security Groups` et cliquez sur le bouton `Create a security group`. 
2. Attribuez un nom à votre groupe de sécurité. Ici, nous avons choisi de le nommer arbitrairement `hardened-conf`. 
3. Sélectionnez `Paris 1` comme zone de disponibilité. Attentio, cette zone doit être la même que celle dans laquelle a été placée l'instance de machine que nous avons créée précédemment. 
4. Défillez vers le bas pour atteindre le bas de la page, en passant la définition des règles (on les définira après) et cliquez sur le bouton `Create a new security group`. 
5. Retournez sur l'instance de machine, toujours via le lien `Instances` dans la barre latérale de gauche, onglet `Instances` et en cliquant sur l'instance `lghs-chat` : 
6. Faites défiler la page vers le bas pour atteindre les groupes de sécurité et cliquez sur l'icône en forme de crayon : 
7. Dans le menu déroulant, changez le groupe de `Default security group` à `hardened-conf` (le groupe qu'on a défini précédemment) :



8. Cliquez que le bouton `Save` :

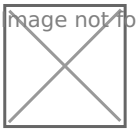


9. Constatez que le groupe de sécurité a changé pour notre machine :
10. Retournez dans les groupes de sécurité en passant par le lien `Instances` de la barre latérale de gauche et en cliquant sur l'onglet `Security Groups` :



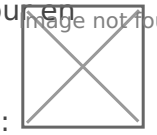
11. Sélectionnez le groupe de sécurité que vous venez de créer (`hardened-conf`) :

12. Allez dans l'onglet `Rules` :
13. Changez la politique par défaut pour le trafic entrant en cliquant sur le petit crayon :

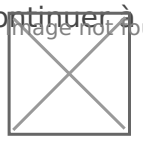


14. Passez le paramètre `Inbound default policy` sur `Drop` :

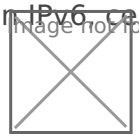
15. Confirmez le changement en cliquant sur le bouton avec la flèche verte :
16. Passons maintenant aux règles à proprement parler, ce qui nous intéresse. Pour en définir, cliquez sur l'icône en forme de crayon dans la sections `Rules` (règles) :

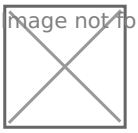


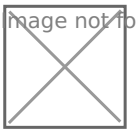
17. Cliquez sur le bouton `Add inbound rule` :
18. Définissez la première règle comme suit ; pour continuer à en ajouter, cliquez de nouveau sur le bouton `Add inbound rule` et ainsi de suite :

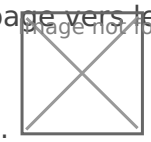


19. Le but est d'obtenir la liste suivante. On autorise juste le traffic web (ports 80/443 à la fois en TCP et UDP - UDP pour tout ce qui est HTTP/2+/QUICK), le SSH (port 22) et l'ICMP. Notez que l'interface web de Scaleway ne prend en charge que l'ICMP et non l'ICMPv6 ce qui empêchera la machine de répondre aux requêtes ICMP en IPv6, cependant ça fonctionnerait en CLI via la commande `scw` (à tester) (src.) :



20. Il faut autoriser les ports SMTP à soumettre des emails sinon Rocket.Chat ne sera pas autorisé à envoyer d'emails, ce qui posera problème (redéfinition de mot de passe et notifications par email notamment). Pour ce faire cochez la case `Enable SMTP Ports` :


21. Cocher la case a pour effet de vider la liste des règles prédéfinies de trafic sortant :


22. Faites défiler la page vers le haut et cliquez sur le bouton de crayon vert pour confirmer les changements. 

Révision #13
Créé 21 juin 2020 10:34:44
Mis à jour 30 décembre 2022 11:18:02 par wget