Procédure d'upgrade depuis toute version antérieure vers V3.1 Commandant Poulard

A. Le plus facile, l'interface web

Il y a plusieurs cas, différenciés entre autres par le fait que vous ayez ou non un historique de connexions issu de versions *antérieures à la V3.0* dans votre base de données. Donc concentrez-vous, décontractez les épaules et choisissez bien...

Cas A1. J'upgrade depuis V3.0 qui était ma première install de QoQ-CoT => je n'ai pas de connexions collectées par une version < V3.0 dans ma BD

1. Pour simplifier ici la lecture, on nommera la base mysql `QoQCoT` (*pas de tirets « - » autorisés dans les noms de bases MySQL...*) et l'utilisateur associé `QoQCoT_user`.

2. Détarer la précieuse archive V3.1 Commandant Poulard, par exemple dans `/tmp`

3. Copier votre `qoq-cot/config.php` actuel dans `/tmp/qoq-cot/src/`

4. Vos données ne risquent rien, bien sûr, lors de cet upgrade... mais sauvez-les quand même... Dumpez donc soigneusement votre base `QoQCoT`.

5. Modifiez votre base de données pour la rendre v3.1 compatible. Pour vous éviter de trop toucher à ces rouages internes recouverts de graisse forcément salissante, car nous tenons à votre confort, nous avons regroupé et automatisé toutes ces modifs dans le seul script `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh`.
Hmm, v2toV3 me dites-vous ? Mais je n'ai jamais eu de v2 !!!
Oui, c'est vrai... mais vous allez quand même devoir lancer le script pour certaines de ses actions qui corrigent quelques erreurs de jeunesse de la v3.0...

Mais quelles sont donc ces fameuses actions, me direz-vous, circonspect et méfiant comme doit l'être à raison un informaticien à l'heure du big data et de wikileaks ? Détaillons donc ça ensemble :

- a) NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0 – *Pour sécuriser l'insertion des données de connexion, QoQ-CoT depuis la version 3.0 Commandant Poulard préconise l'ajout d'un trigger dans la base de données, qui interdira toute possibilité (forcément malicieuse) de mettre à jour l'heure de fin d'une connexion avec une heure antérieure à celle déjà inscrite dans la base (pour masquer une présence sur une machine, par exemple). Pour ce faire, il faut donner le privilège `TRIGGER` à l'utilisateur `QoQCoT_user` sur la base `QoQCoT`, depuis la machine où tourne l'interface web. C'est ce que va faire ce script. Nous verrons plus bas comment créer effectivement le trigger.*
- b) INUTILE SI ON N'A JAMAIS EU DE V2.0, mais on va le faire quand même, pour avoir une belle structure de BD up-to-date – Modifier les propriétés du champs `idProcess` de la table « Connexions » de façon à ce qu'il puisse être NULL. Ceci est nécessaire en mode hybride rsyslog/poussin-coq. En effet, en mode poussin-coq, l'info du numéro de process, qui servait en mode rsyslog à appairer les événements de connexion et déconnexion, n'a plus de sens, du coup elle n'est pas renseignés lors des requêtes d'insertion dans la base par le coq. Conséquance, surtout si votre serveur MySQL est en mode « strict » : il faut que `idProcess`

puisse être NULL. Ainsi, les 2 modes pourront cohabiter.
- c) Supprimer la table inutile « Aliases ». Nous avions créé historiquement cette table en prévision, dans une version future, de la gestion le problème épineux du renommage des machines et, plus généralement, de l'évolution dans le temps des salles machines dont on mesure le taux d'utilisation. Mais notre réflexion a évolué et nous avons changé d'algorithme : cette table ne sera finalement pas utile. See you en V4 😉
- d) INUTILE SI ON N'A JAMAIS EU DE V2.0 – *La « raccourcissation » : transformation des noms longs (FQDN) en noms courts (FQDN tronqué avant le premier « . ») dans les tables « Connexions » et « MachinesToSalles ». C'est crucial en cas d'upgrade v2 → v3 car en mode v2, ce sont les noms longs qui sont insérés dans la base tandis qu'à partir du mode poussin-coq, ce sont les noms courts.*

Maintenant que nous savons POURQUOI il faut exécuter ce script, voyons COMMENT :

5.1 Exécution du script `ModifieDBV2toV3.sh`

5.1.1 copier `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh` sur un serveur Linux disposant du client MySQL ET depuis luquel l'utilisateur `root` de MySQL ET l'utilisateur `QoQCoT_user` sont autorisés à se connecter (table `user` de la base `mysql`). Par exemple le serveur MySQL, ou le serveur qui tourne l'interface web QoQ-CoT…

5.1.2 copier dans le même répertoire votre fichier `config.php` (si vous avez suivi les indications, il est dans `/tmp/qoq-cot/src/`) : le script utilise ce fichier pour récupérer toutes les données utiles à la création de la base.

5.1.3 Désactivez le cron de `peuplade.sh`

5.1.4 lancer `./ModifieDBV2toV3.sh` dans ce répertoire. À noter :

- Vous serez interactivement sollicité pour toutes les actions exécutées par le script. À chaque fois, vous aurez le choix d'effectuer ou non l'action.
- Le script peut être lancé un nombre quelconque de fois, les actions pouvant être refaites indéfiniment sans dommage aucun pour les données.

Les actions, dans l'ordre d'apparition à l'écran :

5.1.4.1 : NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0, donc dire NON au script sauf si vous aviez sauté cette étape à l'époque – *Ajout du droit `TRIGGER` : le script vous demandera d'indiquer sur quelle machine tourne l'interface web QoQ-CoT afin de pouvoir paramétrer dans MySQL le droit `TRIGGER` sur la base `QoQCoT` depuis ladite machine pour l'utilisateur `QoQCoT_user`. Une fois l'action terminée, `QoQCoT_user` pourra créer le trigger, ça va se passer en 5.2.*

5.1.4.2 : Modification des propriétés du champs `iDProcess` de la table « Connexions » : bien qu'inutile dans votre cas, faites-le, vous aurez une structure de base up-to-date, c'est toujours mieux.

5.1.4.3 : Suppression de la table inutile « Aliases »

5.1.4.4 : INUTILE POUR VOUS, dites fermement NON au script. – *La « raccourcissation » : transformation des noms longs (FQDN) en noms courts (FQDN tronqué avant le premier « . ») dans les tables « Connexions » et « MachinesToSalles »*

5.1.5 C'est terminé. Vous pouvez éventuellement détruire les copies de `ModifieDBV2toV3.sh` et `config.php`, ils ont fini leur travail, furtif, mais crucial.

5.2 NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0 – Création du trigger : cette opération va se faire au travers du script `/tmp/qoq-cot/src/setup.php`. Jetez un oeil sur la doc d'installation si vous avez besoin de vous rafraîchir la mémoire par exemple au sujet du format du fichier .csv ou de l'expression régulière à laquelle doivent répondre vos noms de salles dans JeDDLaJ.

5.2.1 Cas csv

5.2.1.1 Retrouver le fichier csv décrivant vos salles, ou mieux (depuis V3.0 Commandant Poulard), regénérez-le dynamiquement à partir des données en base, en lançant `/tmp/qoq-cot/src/export_salles_to_csv.php > /chemin/vers/mon/fichier.csv`. Profitez-en pour le modifier/compléter si besoin pour refléter l'évolution de votre structure, très dynamique.

5.2.1.2 Lancer `/tmp/qoq-cot/src/setup.php /chemin/vers/mon/fichier.csv`

5.2.1.3 Ceci va regénérer les salles dans la base en fonction des données du fichier .csv (pas de changement si le fichier .csv n'a pas changé...) ET créer le trigger

5.2.2 Cas JeDDlaJ

5.2.2.1 Lancer `/tmp/qoq-cot/src/setup.php jeddlaj`

5.2.2.2 Ceci va regénerer les salles dans la base en fonction des groupes déclarés dans JeDDLaJ (pas de changement si les groupes n'ont pas changé...) ET créer le trigger

Et voilà, c'est déjà fini, le trigger est créé tout comme il faut, vos données sont désormais protégées et vous avez gardé les mains propres et les ongles nets. Merci qui ???

6. Copiez votre `qoq-cot/.htaccess` actuel vers `/tmp/qoq-cot/src/.htaccess`

7. Renommer votre répertoire `qoq-cot` actuel en `qoq-cot.old`

8. déplacez `/tmp/qoq-cot/src` dans votre emplacement de prod (c.-à-d. le répertoire qui contient `qoq-cot.old`) et renommez-le en `qoq-cot`.

9. That's all, folks : votre poulette a retrouvé ses 20 ans et est toujours toute à vous...

Cas A2. J'upgrade depuis V3.0 qui était un upgrade d'une V2.0 ou inférieure = j'ai des connexions collectées par une version < V3.0 dans ma BD

1. Pour simplifier ici la lecture, on nommera la base mysql `QoQCoT` (*pas de tirets « - » autorisés dans les noms de bases MySQL...*) et l'utilisateur associé `QoQCoT_user`.

2. Détarer la précieuse archive V3.1 Commandant Poulard, par exemple dans `/tmp`

3. Copier votre `qoq-cot/config.php` actuel dans `/tmp/qoq-cot/src/`

4. Commentez dans votre crontab, si vous l'avez mis en place, le lancement du script peuplade.php. Vérifiez ensuite qu'une exécution de peuplade.php n'est pas en cours. Si c'est le cas, attendez qu'elle

se termine avant de poursuivre.

4,5 Vos données ne risquent rien, bien sûr, lors de cet upgrade… mais sauvez-les quand même… Dumpez donc soigneusement votre base `QoQCoT`.

5. Modifiez votre base de données pour la rendre v3.1 compatible. Pour vous éviter de trop toucher à ces rouages internes recouverts de graisse forcément salissante, car nous tenons à votre confort, nous avons regroupé et automatisé toutes ces modifs dans le seul script `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh`. Mais que fait donc ce script, me direz-vous, circonspect et méfiant comme doit l'être à raison un informaticien à l'heure du big data et de wikileaks ? Détaillons donc ça ensemble : :

- a) NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0 – *Pour sécuriser l'insertion des données de connexion, QoQ-CoT depuis la version 3.0 Commandant Poulard préconise l'ajout d'un trigger dans la base de données, qui interdira toute possibilité (forcément malicieuse) de mettre à jour l'heure de fin d'une connexion avec une heure antérieure à celle déjà inscrite dans la base (pour masquer une présence sur une machine, par exemple). Pour ce faire, il faut donner le privilège `TRIGGER` à l'utilisateur `QoQCoT_user` sur la base `QoQCoT`, depuis la machine où tourne l'interface web. C'est ce que va faire ce script. Nous verrons plus bas comment créer effectivement le trigger.*
- b) Modifier les propriétés du champs idProcess de la table « Connexions » de façon à ce qu'il puisse être NULL. Ceci est nécessaire en mode hybride rsyslog/poussin-coq. En effet, en mode poussin-coq, l'info du numéro de process, qui servait en mode rsyslog à appairer les événements de connexion et déconnexion, n'a plus de sens, du coup elle n'est pas renseignés lors des requêtes d'insertion dans la base par le coq. Du coup, et surtout si votre serveur MySQL est en mode « strict », il faut que idProcess puisse être NULL. Ainsi, les 2 modes pourront cohabiter.
- c) Supprimer la table inutile « Aliases ». Nous avions créé historiquement cette table en prévision, dans une version future, de la gestion le problème épineux du renommage des machines et, plus généralement, de l'évolution dans le temps des salles machines dont on mesure le taux d'utilisation. Mais notre réflexion a évolué et nous avons changé d'algorithme : cette table ne sera finalement pas utile. See you en V4 
- d) **LE PLUS GROS CHANGEMENT** : modification des données des tables « Connexions » et « MachinesToSalles » pour transformer les noms longs (`toto.mon.domaine`) en noms courts (`toto`). C'est crucial en cas d'upgrade v2 → v3. Pourquoi ? Parce qu'en mode v2, ce sont les noms longs qui sont insérés dans la base (via `peuplade.sh`) et qu'à partir du mode poussin-coq, ce sont les noms courts. Pas de problèmes, aucune donnée de connexion n'est perdue mais, dans votre base de données issue du temps v2, qui contient tout votre précieux historique de connexions, les salles sont définies comme contenant des machines à noms longs. Or toutes les connexions poussins remontant en nom court, elles n'apparaîtront pas dans les graphes car elles seront vues comme correspondant à d'autres machines… Bref, soit vous modifiez les noms des machines dans les salles par les noms courts et vous ne verrez que les connexions issues des poussins, soit vous laissez les salles telles quelles et vous ne verrez que les connexions issues du fonctionnement V2. La modif. que nous vous proposons ici va permettre de disposer dans les graphes de tout l'historique des connexions AINSI que des nouvelles tout en permettant de fonctionner en mode hybride V2 et V3 en n'ayant que des noms courts en base (oui, pour cela, bravo vous avezc suivi, il faut aussi modifier `peuplade.sh`, nous en proposons une nouvelle version, on reviendra sur ce point plus bas…)

Maintenant que nous savons POURQUOI il faut exécuter ce script, voyons COMMENT :

5.1 Exécution du script `ModifieDBV2toV3.sh`

5.1.1 copier `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh` sur un serveur Linux disposant du client MySQL ET depuis luquel l'utilisateur `root` de MySQL ET l'utilisateur `QoQCoT_user` sont autorisés à se connecter (table `user` de la base `mysql`). Par exemple le serveur MySQL, ou le serveur qui tourne l'interface web QoQ-CoT...

5.1.2 copier dans le même répertoire votre fichier `config.php` (si vous avez suivi les indications, il est dans `/tmp/qoq-cot/src/`) : le script utilise ce fichier pour récupérer toutes les données utiles à la création de la base.

5.1.3 Désactivez le cron de `peuplade.sh`

5.1.4 lancer `./ModifieDBV2toV3.sh` dans ce répertoire. À noter :

- Vous serez interactivement sollicité pour toutes les actions exécutées par le script. À chaque fois, vous aurez le choix d'effectuer ou non l'action.
- Le script peut être lancé un nombre quelconque de fois, les actions pouvant être refaites indéfiniment sans dommage aucun pour les données.

Les actions, dans l'ordre d'apparition à l'écran :

5.1.4.1 : NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0, donc dire NON au script sauf si vous aviez sauté cette étape à l'époque – *Ajout du droit `TRIGGER` : le script vous demandera d'indiquer sur quelle machine tourne l'interface web QoQ-CoT afin de pouvoir paramétrer dans MySQL le droit `TRIGGER` sur la base `QoQCoT` depuis ladite machine pour l'utilisateur `QoQCoT_user`. Une fois l'action terminée, QoQCoT_user pourra créer le trigger, ça va se passer en 5.2.*

5.1.4.2 : Modification des propriétés du champs `iDProcess` de la table « Connexions ».

5.1.4.3 : Suppression de la table inutile « Aliases »

5.1.4.4 : **Le principal, la « raccourcissation »** : transformation des noms longs (FQDN) en noms courts (FQDN tronqué avant le premier « . ») dans les tables « Connexions » et « MachinesToSalles »

5.1.5 C'est terminé. Vous pouvez éventuellement détruire les copies de `ModifieDBV2toV3.sh` et `config.php`, ils ont fini leur travail, furtif, mais crucial.

5.2 NORMALEMENT DÉJÀ FAIT LORS DE L'INSTALL de la V3.0 – Création du trigger : cette opération va se faire au travers du script `/tmp/qoq-cot/src/setup.php`. Jetez un oeil sur la doc d'installation si vous avez besoin de vous rafraîchir la mémoire par exemple au sujet du format du fichier .csv ou de l'expression régulière à laquelle doivent répondre vos noms de salles dans JeDDLaJ.

5.2.1 Cas csv

5.2.1.1 Retrouver le fichier csv décrivant vos salles, ou mieux (depuis V3.0 Commandant Poulard), regénérez-le dynamiquement à partir des données en base, en lançant `/tmp/qoq-cot/src/export_salles_to_csv.php > /chemin/vers/mon/fichier.csv`. Profitez-en pour le modifier/compléter si besoin pour refléter l'évolution de votre structure, très dynamique.

5.2.1.2 Lancer `/tmp/qoq-cot/src/setup.php /chemin/vers/mon/fichier.csv`

5.2.1.3 Ceci va regénérer les salles dans la base en fonction des données du fichier .csv (pas de changement si le fichier .csv n'a pas changé...) ET créer le trigger

5.2.2 Cas JeDDlaJ

5.2.2.1 Lancer `/tmp/qoq-cot/src/setup.php jeddlaj`

5.2.2.2 Ceci va regénerer les salles dans la base en fonction des groupes déclarés dans JeDDLaJ (pas de changement si les groupes n'ont pas changé...) ET créer le trigger

Et voilà, c'est déjà fini, le trigger est créé tout comme il faut, vos données sont désormais protégées et vous avez gardé les mains propres et les ongles nets. Merci qui ???

6. Copiez votre `qoq-cot/.htaccess` actuel vers `/tmp/qoq-cot/src/.htaccess`

7. Renommer votre répertoire `qoq-cot` actuel en `qoq-cot.old`

8. déplacez `/tmp/qoq-cot/src` dans votre emplacement de prod (c.-à-d. le répertoire qui contient `qoq-cot.old`) et renommez-le en `qoq-cot`.

9. Remettez en service le lancement du script peuplade.php dans votre crontab. ATTENTION : le peuplade.php lancé doit ABSOLUMENT être celui du nouveau répertoire qoq-cot, dans sa splendide livrée 3.1 commandant Poulard. En effet, il a été modifié pour remonter des noms courts afin d'être compatible avec les remontées des poussions. Notons qu'une fois que vous serez intégralement passé en mode poussin/coq, le script peuplade.php n'aura plus d'objet et vous pourrez le supprimer de la crontab.

10. That's all, folks : votre poulette a retrouvé ses 20 ans et est toujours toute à vous...

Cas A3. J'upgrade depuis V2.0 Bernadette

1. Pour simplifier ici la lecture, on nommera la base mysql `QoQCoT` (*pas de tirets « - » autorisés dans les noms de bases MySQL...*) et l'utilisateur associé `QoQCoT_user`.

2. Détarer la précieuse archive V3.1 Commandant Poulard, par exemple dans `/tmp`

3. Copier votre `qoq-cot/config.php` actuel dans `/tmp/qoq-cot/src/`

4. Commentez dans votre crontab, si vous l'avez mis en place, le lancement du script peuplade.php. Vérifiez ensuite qu'une exécution de peuplade.php n'est pas en cours. Si c'est le cas, attendez qu'elle se termine avant de poursuivre.

4,5 Vos données ne risquent rien, bien sûr, lors de cet upgrade... mais sauvez-les quand même... Dumpez donc soigneusement votre base `QoQCoT`.

5. Modifiez votre base de données pour la rendre v3.1 compatible. Pour vous éviter de trop toucher à ces rouages internes recouverts de graisse forcément salissante, car nous tenons à votre confort, nous avons regroupé et automatisé toutes ces modifs dans le seul script `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh`. Mais que fait donc ce script, me direz-vous, circonspect et méfiant comme doit l'être à raison un informaticien à

l'heure du big data et de wikileaks ? Détaillons donc ça ensemble : :

- a) Pour sécuriser l'insertion des données de connexion, QoQ-CoT depuis la version 3.0 Commandant Poulard préconise l'ajout d'un trigger dans la base de données, qui interdira toute possibilité (forcément malicieuse) de mettre à jour l'heure de fin d'une connexion avec une heure antérieure à celle déjà inscrite dans la base (pour masquer une présence sur une machine, par exemple). Pour ce faire, il faut donner le privilège `TRIGGER` à l'utilisateur `QoQCoT_user` sur la base `QoQCoT`, depuis la machine où tourne l'interface web. C'est ce que va faire ce script. Nous verrons plus bas comment créer effectivement le trigger.
- b) Modifier les propriétés du champs idProcess de la table « Connexions » de façon à ce qu'il puisse être NULL. Ceci est nécessaire en mode hybride rsyslog/poussin-coq. En effet, en mode poussin-coq, l'info du numéro de process, qui servait en mode rsyslog à appairer les événements de connexion et déconnexion, n'a plus de sens, du coup elle n'est pas renseignés lors des requêtes d'insertion dans la base par le coq. Du coup, et surtout si votre serveur MySQL est en mode « strict », il faut que idProcess puisse être NULL. Ainsi, les 2 modes pourront cohabiter.
- c) Supprimer la table inutile « Aliases ». Nous avions créé historiquement cette table en prévision, dans une version future, de la gestion le problème épineux du renommage des machines et, plus généralement, de l'évolution dans le temps des salles machines dont on mesure le taux d'utilisation. Mais notre réflexion a évolué et nous avons changé d'algorithme : cette table ne sera finalement pas utile. See you en V4 😉
- d) **LE PLUS GROS CHANGEMENT** : modification des données des tables « Connexions » et « MachinesToSalles » pour transformer les noms longs (`toto.mon.domaine`) en noms courts (`toto`). C'est crucial en cas d'upgrade v2 → v3. Pourquoi ? Parce qu'en mode v2, ce sont les noms longs qui sont insérés dans la base (via `peuplade.sh`) et qu'à partir du mode poussin-coq, ce sont les noms courts. Pas de problèmes, aucune donnée de connexion n'est perdue mais, dans votre base de données issue du temps v2, qui contient tout votre précieux historique de connexions, les salles sont définies comme contenant des machines à noms longs. Or toutes les connexions poussins remontant en nom court, elles n'apparaîtront pas dans les graphes car elles seront vues comme correspondant à d'autres machines... Bref, soit vous modifiez les noms des machines dans les salles par les noms courts et vous ne verrez que les connexions issues des poussins, soit vous laissez les salles telles quelles et vous ne verrez que les connexions issues du fonctionnement V2. La modif. que nous vous proposons ici va permettre de disposer dans les graphes de tout l'historique des connexions AINSI que des nouvelles tout en permettant de fonctionner en mode hybride V2 et V3 en n'ayant que des noms courts en base (oui, pour cela, bravo vous avezc suivi, il faut aussi modifier `peuplade.sh`, nous en proposons une nouvelle version, on reviendra sur ce point plus bas...)

Maintenant que nous savons POURQUOI il faut exécuter ce script, voyons COMMENT :

5.1 Exécution du script `ModifieDBV2toV3.sh`

5.1.1 copier `/tmp/qoq-cot/ressources/UPGRADE_FROM_PREVIOUS_VERSIONS/ModifieDBV2toV3.sh` sur un serveur Linux disposant du client MySQL ET depuis luquel l'utilisateur `root` de MySQL ET l'utilisateur `QoQCoT_user` sont autorisés à se connecter (table `user` de la base `mysql`). Par exemple le serveur MySQL, ou le serveur qui tourne l'interface web QoQ-CoT...

5.1.2 copier dans le même répertoire votre fichier `config.php` (si vous avez suivi les indications, il est dans `/tmp/qoq-cot/src/`) : le script utilise ce fichier pour récupérer toutes les données utiles à la création de la base.

5.1.3 Désactivez le cron de `peuplade.sh`

5.1.4 lancer `./ModifieDBV2toV3.sh` dans ce répertoire. À noter :

- Vous serez interactivement sollicité pour toutes les actions exécutées par le script. À chaque fois, vous aurez le choix d'effectuer ou non l'action.
- Le script peut être lancé un nombre quelconque de fois, les actions pouvant être refaites indéfiniment sans dommage aucun pour les données.

Les actions, dans l'ordre d'apparition à l'écran :

5.1.4.1 : Ajout du droit `TRIGGER` : le script vous demandera d'indiquer sur quelle machine tourne l'interface web QoQ-CoT afin de pouvoir paramétrer dans MySQL le droit `TRIGGER` sur la base `QoQCoT` depuis ladite machine pour l'utilisateur `QoQCoT_user`. Une fois l'action terminée, QoQCoT_user pourra créer le trigger, ça va se passer en 5.2.

5.1.4.2 : Modification des propriétés du champs `iDProcess` de la table « Connexions ».

5.1.4.3 : Suppression de la table inutile « Aliases »

5.1.4.4 : **Le principal, la « raccourcissation »** : transformation des noms longs (FQDN) en noms courts (FQDN tronqué avant le premier « . ») dans les tables « Connexions » et « MachinesToSalles »

5.1.5 C'est terminé. Vous pouvez éventuellement détruire les copies de `ModifieDBV2toV3.sh` et `config.php`, ils ont fini leur travail, furtif, mais crucial.

5.2 Création du trigger : cette opération va se faire au travers du script `/tmp/qoq-cot/src/setup.php`. Jetez un oeil sur la doc d'installation si vous avez besoin de vous rafraîchir la mémoire par exemple au sujet du format du fichier .csv ou de l'expression régulière à laquelle doivent répondre vos noms de salles dans JeDDLaJ.

5.2.1 Cas csv

5.2.1.1 Retrouver le fichier csv décrivant vos salles, ou mieux (depuis V3.0 Commandant Poulard), regénérez-le dynamiquement à partir des données en base, en lançant `/tmp/qoq-cot/src/export_salles_to_csv.php > /chemin/vers/mon/fichier.csv`. Profitez-en pour le modifier/compléter si besoin pour refléter l'évolution de votre structure, très dynamique.

5.2.1.2 Lancer `/tmp/qoq-cot/src/setup.php /chemin/vers/mon/fichier.csv`

5.2.1.3 Ceci va regénérer les salles dans la base en fonction des données du fichier .csv (pas de changement si le fichier .csv n'a pas changé...) ET créer le trigger

5.2.2 Cas JeDDlaJ

5.2.2.1 Lancer `/tmp/qoq-cot/src/setup.php jeddlaj`

5.2.2.2 Ceci va regénerer les salles dans la base en fonction des groupes déclarés dans JeDDLaJ (pas de changement si les groupes n'ont pas changé...) ET créer le trigger

Et voilà, c'est déjà fini, le trigger est créé tout comme il faut, vos données sont désormais protégées et vous avez gardé les mains propres et les ongles nets. Merci qui ???

6. Copiez votre `qoq-cot/.htaccess` actuel vers `/tmp/qoq-cot/src/.htaccess`

7. Renommer votre répertoire `qoq-cot` actuel en `qoq-cot.old`

8. déplacez `/tmp/qoq-cot/src` dans votre emplacement de prod (c.-à-d. le répertoire qui contient `qoq-cot.old`) et renommez-le en `qoq-cot`.

9. Remettez en service le lancement du script peuplade.php dans votre crontab. ATTENTION : le peuplade.php lancé doit ABSOLUMENT être celui du nouveau répertoire qoq-cot, dans sa splendide livrée 3.1 commandant Poulard. En effet, il a été modifié pour remonter des noms courts afin d'être compatible avec les remontées des poussions. Notons qu'une fois que vous serez intégralement passé en mode poussin/coq, le script peuplade.php n'aura plus d'objet et vous pourrez le supprimer de la crontab.

10. That's all, folks : votre poulette a retrouvé ses 20 ans et est toujours toute à vous...

Cas A4. J'upgrade depuis une version antérieure à V2.0 Bernadette

Vous devez d'abord passer en 2.0 Bernadette...

B. Le plus long, les poussins et le coq

Cas B1. J'upgrade depuis V3.0 Commandant Poulard

Vous êtes déjà en mode poussin/coq, le plus dur est fait

Il y a cependant quelques nouveautés (bio) dans les poussins et le coq V3.1. :

Côté poussin

- il y a dans la conf. du poussin V3.1 un nouveau paramètre : « hostname ». Il permet de répondre à un cas auparavant non géré : il peut arriver qu'on souhaite que la machine remonte dans la base avec un nom différent de celui connu par le système (hostname sous linux, Nom de l'ordinateur sous Windows...).
  - Un exemple de cas où ça peut être utile : machine en double boot avec un nom système différent sous les 2 OS. Si on ne remplit pas ce paramètre, la machine va remonter dans la base sous 2 noms différents selon qu'elle tourne sous 1 OS ou l'autre, ce qui peut être ennuyeux pour les statistiques d'utilisation... On règle le problème en remplissant ce paramètre, dans le `poussin.yml` de l'OS #2, avec le nom système de la machine sous l'OS #1 (sous l'OS #1, on laisse le paramètre à vide. C'est-à-dire qu'on peut laisser le poussin v3.0 si on l'avait déjà installé).
  - Ce paramètre permet aussi de répondre au cas inverse : différencier des machines aux mêmes noms courts, dans des domaines différents : `toto.chicken.org` et `toto.fried.chicken.org`.
- **ATTENTION** : si vous renseignez le paramètre `hostname`, alors ce nom **doit**, comme celui de

toutes les autres machines, apparaître dans le fichier `.csv` donné en entré à `setup.php`, sinon les connexions issues de cette machine n'apparaîtront jamais dans les graphes...
Dans le cas du double boot vu ci-dessus, vous avez logiquement pensé à mettre le nom de machine dans le fichier `.csv`, puisqu'il s'agit du nom système fourni par l'un des deux OS.
Mais si l'on souhaite, pour d'autres et obscures raisons, changer le nom de machine remonté par un nom *qui n'est le nom système d'aucune des machines du parc surveillé*, alors il ne faut pas oublier de faire figurer ce nom dans le fichier `.csv`...

- **ATTENTION 2** : il est possible de renseigner le paramètre `hostname` avec un nom long, ce qui est désormais un des 2 seuls cas permettant d'introduire un nom long dans la base QoQ-CoT[1)]. C'est tout à fait possible, mais plutôt déconseillé dans un monde de noms courts... Mais si vous avez vos raisons, pourquoi pas 😉

- Si vous avez besoin de cette fonctionnalité sur certaines de vos machines, installez-y (comme indiqué dans le `README.TXT` du répertoire `ressources/poussin-coq/poussin`) le poussin en V3.1 ainsi que le `poussin.yml` associé (le même qu'avant, plus le paramètre `hostname` dûment rempli).
- Sinon, vous pouvez garder votre poussin 3.0, encore splendide et toujours bien côté à l'argus.

Côté coq

- le coq est désormais livré en version compilée coq32 et coq64, pour éviter d'avoir à installer les modules PERL. Mais le code est le même qu'en v3.0, vous pouvez donc sans souci laisser tourner le coq.pl de la v3.0.

Cas B2. J'upgrade depuis V2.0 Bernadette

Le principal, mais énorme, apport de la version 3.0 commandant Poulard réside dans l'abandon du rsyslog au profit du mode poussin/coq : les poussins (installés sur chaque machine du parc surveillé) remontent au coq (installé sur un serveur quelconque, nous préconisons le serveur MySQL) les données de connexions, et c'est le coq, il est le seul autorisé à le faire, qui va écrire ces données de connexions directement dans la table Connexions de la base MySQL.

Le truc, c'est que vous n'allez pas installer d'un coup les poussins sur toutes les machines, mais plutôt migrer petit à petit. Il y aura donc, pendant un temps, un mode hybride où les données de connexion seront remontées simultanément par deux procédés différents :

- pour les anciens clients, par les anciens mécanismes utilisant rsyslog puis écrites dans la table `Connexions` par le script (normalement croné) `peuplade.php` ;
- pour les nouveaux clients, par les poussins, qui communiqueront avec le coq, qui écrira directement dans la table `Connexions`.

Ceci ne pose pas de problème (si ce n'est que les données seront plus fiable en mode poussin/coq) et peut perdurer tout le temps nécessaire. Mais gardez à l'esprit que le mode poussin/coq est plus performant (moins de données sur le réseau, gain en termes de sécurité, etc.) et surtout, vous garantit des données plus fiables en éliminant tous les problèmes de pertes de fins de connexions rencontrés dans l'ancien fonctionnement.

Nous proposons la procédure suivante pour réaliser à votre rythme la migration de tout votre parc

surveillé vers le mode poussin/coq :

1. Installer le coq sur un serveur (de préférence le serveur MySQL) et de le configurer comme indiqué dans le `README.TXT` du répertoire `ressources/poussin-coq/coq`
2. Migrer petit à petit en mode poussin les machines du parc surveillé. Pour cela, il faudra, pour chaque machine :
    1. arrêter ou désinstaller l'ancien client rsyslog sur la machine ;
    2. installer et configurer le poussin adapté à l'OS de la machine comme indiqué dans le `README.TXT` du répertoire `ressources/poussin-coq/poussin`.
3. Quand toutes les machines auront été migrées, on pourra :
    1. supprimer du cron le lancement de `peuplade.php` ;
    2. arrêter le serveur rsyslog, voire le désinstalller s'il ne servait que pour QoQ-CoT ;
    3. supprimer la table `SystemEvents` de la DB rsyslog.

Cas B3. J'upgrade depuis une version antérieure à V2.0 Bernadette

Vous devez d'abord passer en 2.0 Bernadette...

---

1)
L'autre cas : une machine Linux – en mode poussin/coq ET dont le paramètre `hostname` de `poussin.yml` n'est pas renseigné – est (mal) configurée avec un nom long dans `/etc/hostname` (qui doit être normalement renseigné avec un nom court). En effet, le module Perl utilisé dans le poussin Linux remonte comme *nom court* le contenu du fichier `/etc/hostname`.