back office eco système n’gondo

fonctions

- enregistrement
- activation
- authentification
- recharge

les modules

tous les modules ont un fichier de configuration .ini

il contient les adresses ip des serveurs et celles des bases de données

chaque serveur a un doublon qui sert de back-up

chaque serveur enregistre simultanément les commandes dans 2 bases de données *REPORT*

clients

WEB
N’GONDO
IOS
ANDROID
autres

serveurs

PPPX
proxy

AAR
authentification
activation

SR

enregistrement

LB
répartition de charges

TOPUP

rechargement

base de données

REPORT

enregistrement des requêtes

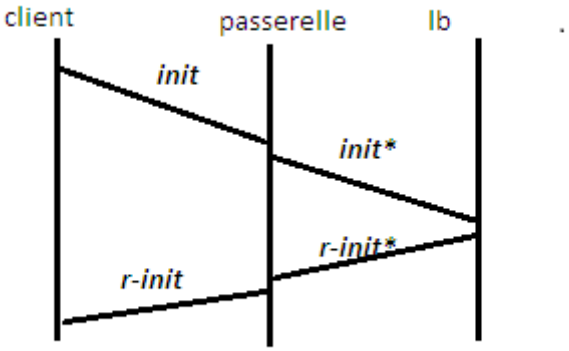
CRYPTO

enregistrement des box n’gondo

commande

INIT

workflow



format

```
<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>INIT</cmd>
        <time>$(TIME)</time>
</xml>
```

```
<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>INIT*</cmd>
	<time>$(TIME)</time>
	<bridge>$(BRIDGE)</bridge>
</xml>

<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>R-INIT*</cmd>
	<time>$(TIME)</time>
	<bridge>$(BRIDGE)</bridge>
	<AAR><first></first><second></second></AAR>
	<TOPUP><first></first><second></second></TOPUP>
	<LB><first></first><second></second></LB>
	<PPPX><first></first><second></second></PPPX>
	<REPORT><first></first><second></second></REPORT>
	<CRYPTO><first></first><second></second></CRYPTO>
</xml>

<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>R-INIT</cmd>
	<time>$(TIME)</time>
	<bridge>$(BRIDGE)</bridge>
	<AAR><first></first><second></second></AAR>
	<TOPUP><first></first><second></second></TOPUP>
	<LB><first></first><second></second></LB>
	<PPPX><first></first><second></second></PPPX>
	<REPORT><first></first><second></second></REPORT>
	<CRYPTO><first></first><second></second></CRYPTO>
</xml>
```

protocole de dialogue

au démarrage chaque module envoi la commande *INIT* à *PPPX*

la réponse obtenue permet de mettre à jour le fichier de configuration du module

PPPX forward la requête à *LB* ce dernier retourne les adresses ip des serveurs

et des bases de données.

Chaque serveur publie un rapport à chaque requête reçue.

LB consulte ces rapports pour répartir la charge.

Chaque module réitère la commande *INIT* périodiquement.

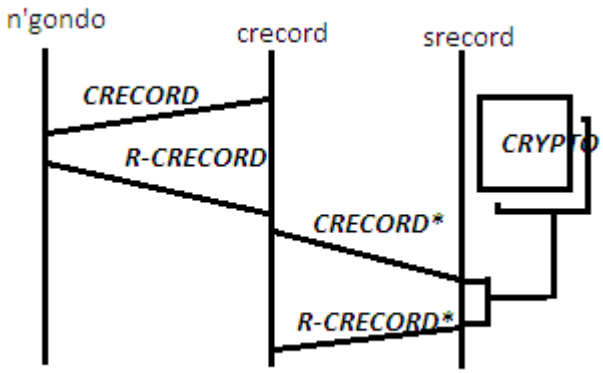
Chaque retour donne lieu à la mise à jour des paramètres de connexion

aux serveurs et à la base de données.

commande

RECORD

workflow



format

```
<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>CRECORD</cmd>
        <time>$(TIME)</time>
</xml>

<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>R-CRECORD</cmd>
        <time>$(TIME)</time>
        <macaddr>$(MACADDR)</macaddr>
</xml>

<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>CRECORD*</cmd>
        <time>$(TIME)</time>
        <bridge>$(BRIDGE)</bridge>
        <macaddr>$(MACADDR)</macaddr>
</xml>
```

```
<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>R-CRECORD*</cmd>
        <time>$(TIME)</time>
        <bridge>$(BRIDGE)</bridge>
        <result>$(RESULT)</result>
</xml>
```

protocole de dialogue

la clé *4G* doit préalablement être relié au *PC* via une interface usb

cette interface est convertie en interface Ethernet ayant une adresse *IP* fixe.

À l'aide d'un outil on actionne un bouton qui génère une commande

cette commande est envoyé à la clé *4G*, la clé va retourner des informations

qui permettent de l'identifier de façon unique.

Cette information est ensuite envoyé au serveur *SRECORD* qui l'enregistre dans la base *CRYPTO*.

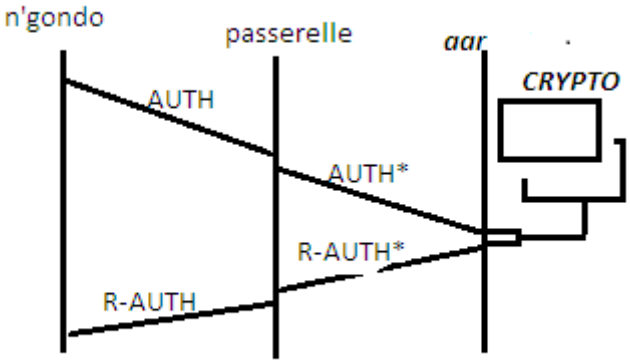
Avant l'enregistrement le serveur *SRECORD* génére une clé privé et crypte les *INFOS* avant de les stocker.

La base *CRYPTO* et le serveur *SRECORD* font partie du même *VPN*.

commande

AUTH

workflow



format

```
<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>AUTH</cmd>
	<time>$(TIME)</time>
	<crypto>$(CRYPTO)</crypto>
</xml>

<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>R-AUTH</cmd>
	<time>$(TIME)</time>
	<result>$(RESULT)</result>
</xml>

<xml>
	<app>$(NAME)</app>
	<version>$(VERSION)</version>
	<cmd>AUTH*</cmd>
	<time>$(TIME)</time>
	<bridge>$(BRIDGE)</bridge>
</xml>
```

```
<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>R-AUTH*</cmd>
        <time>$(TIME)</time>
        <bridge>$(BRIDGE)</bridge>
        <result>$(RESULT)</result>
</xml>
```

protocole de dialogue

La box N’gondo émet une demande d’authentification à chaque démarrage.

Si l’authentification est validée l’utilisateur peut surfer sinon c’est impossible.

commande

TOPUP

workflow

format

```
<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>TOPUP</cmd>
        <time>$(TIME)</time>
        <code>$(CODE)</code>
</xml>

<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>R-TOPUP</cmd>
        <time>$(TIME)</time>
        <result>$(RESULT)</result>
</xml>

<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>TOPUP*</cmd>
        <time>$(TIME)</time>
        <bridge>$(BRIDGE)</bridge>
        <code>$(CODE)</code>
```

```
</xml>

<xml>
        <app>$(NAME)</app>
        <version>$(VERSION)</version>
        <cmd>R-TOPUP*</cmd>
        <time>$(TIME)</time>
        <bridge>$(BRIDGE)</bridge>
        <code2>$(CODE)</code2>
</xml>
```

protocole de dialogue

La recharge de la box N'gondo est initié par l'utilisateur ou un administrateur.

Lorsqu'on actionne un bouton l'outil génère une commande TOPUP qui est émis vers le serveur TOPUP.

La commande TOPUP prend en paramétre un code secret obtenu lors de l'achat d'une recharge en ligne ou en boutique.

Le serveur TOPUP extrait le code de rechargement de la puce en fonction du code d'achat.

Le code de rechargement est transmis à l'outil qui l'exploite pour créditer la puce.