Keys Operating System (KeysOS)

Numéro de révision du document : 006/20

Date de publication : 28/12/2020

Gestionnaire du projet : Mr. Aneto Minanga GUYLLIT

Signatures d'approbation

Approuvé par : Mr. **Aneto Minanga GUYLLIT**

Préparé par : Mr. **Aneto Minanga GUYLLIT**

Examiné par : Mr. Mr. **Aneto Minanga GUYLLIT**

Table des matières

Suivi de modification des documents

Numéro de révision	Date de Publication	Auteur(s)	Brève description des modifications
006/20	**28/12/2020**	Mr. Aneto Minanga GUYLLIT	Structure complète du système
--//--	**--//--**	**--//--**	Modification complète du système

1. Aperçu du système

1.1 But, portée et objectifs

- La circulation des informations sur internet devienne les plus en plus inquiète, les nouvelles générations d'applications natives ou distantes avec une myriade des données circulent en permanence et sans pour autant s'assurer de la sécurité des informations à la portée des utilisateurs. Les consommateurs (Internautes) ses retirent petit à petit de la toile suite à ce désagrément par rapport aux informations personnelles stocké sans garanties, les mots de passe erronés avec des questions que les gens ne se souviennent plus pour la restauration de leurs comptes, les accès piraté, même si l'arrivé du **Token** (Jeton) et d'autres systèmes des sécurisations des informations sur la base des données, les internautes continuent toujours d'inquiétude sans pour autant avoir un changement aux informations qu'ils sauvegardent sur la toile.

- L'arrivé de ce système des traitements des clés (Keys OS) a pour but de sécurisé tous type d'information sur internet ou hors ligne sans connexion internet grâce à une clé chiffré (Crypté) qui contienne les informations relative aux utilisateurs ou même pour l'internet des objets (Internet Of Things). Tant que son système de sécurisation utilise le format des caractères textes et non des bytes codes, la clé peut être intégré dans tous les systèmes de gestion de la base des données sans inquiétude comme des simples caractères textes.

- Le système de traitement des clés chiffrées n'a pas été dédié spécifiquement pour les entreprises. Même les particuliers peuvent l'utiliser pour la sécurisation des leurs informations personnels. L'objectif est de faire en sorte que, toutes les

informations qui circulent sur la toile (Internet) depuis un serveur distant ou en locale depuis une application native, soient sécurisé de bout en bout pour que toutes les informations relatives à la clé chiffrée (Crypté) ne soient pas divulgué par d'autres personnes hors de la portée.

1.2 Hypothèses, contraintes et risques

- En effet, l'intégration de ce système de chiffrement des informations depuis une application web ou locale ne requise pas une expérience du système **Windows**, **Linux** ou **Mac OS**. Même les débutants des plates-formes cités peuvent l'intégré facilement dans leurs applications natives ou distantes avec des petites lignes des commandes. Le système est capable de sécuriser tout type d'information **Texte**, **Image**, **Vidéo**, **PDF** et autres depuis un serveur, disque dur local ou les périphériques amovibles (**USB**, **CD**, **DVD** et **SD CARD**). Grace à cette portabilité des informations que KeysOS offerte à tous ceux qui veulent sécuriser leurs informations, le système est censé de généré des clés portable pour les Token (Jetons), clé de licence et autres types des clés non cités.

- C'est ne pas obligatoire pour que les informations soient sécurisé, il faudra que la clé soit caché à la portée des autres inconnus. C'est ce qui fait que lorsque ces inconnus vont récupérer votre clé, après son déchiffrement, ils vont avoir toutes les informations relatives à votre compte. L'importance est d'avoir une clé (Token) portable, capable d'interagir d'un serveur à un autre et en toute sécurité même si la clé est partagée à la portée de tous.

- Les risques relatif au système KeysOS, est que si le téléphone portable de l'utilisateur de la clé a été volé avec un numero de téléphone mobile utilisé à la génération de la clé chiffrée, ou un compte mail qui n'utilise pas cette clé de chiffrement depuis son système et que son mot de passe a était dérobé par une autre personne, la clé chiffrée par le système KeysOS peut être déchiffrée facilement à la portée d'une personne indésirable parce que le système utilise une méthode d'envoi des informations relatives à la clé vers un

serveur dédié de la société à l'adresse qui lui a été spécifié suivi d'un numero de téléphone de l'utilisateur ou son adresse électronique (E-mail).

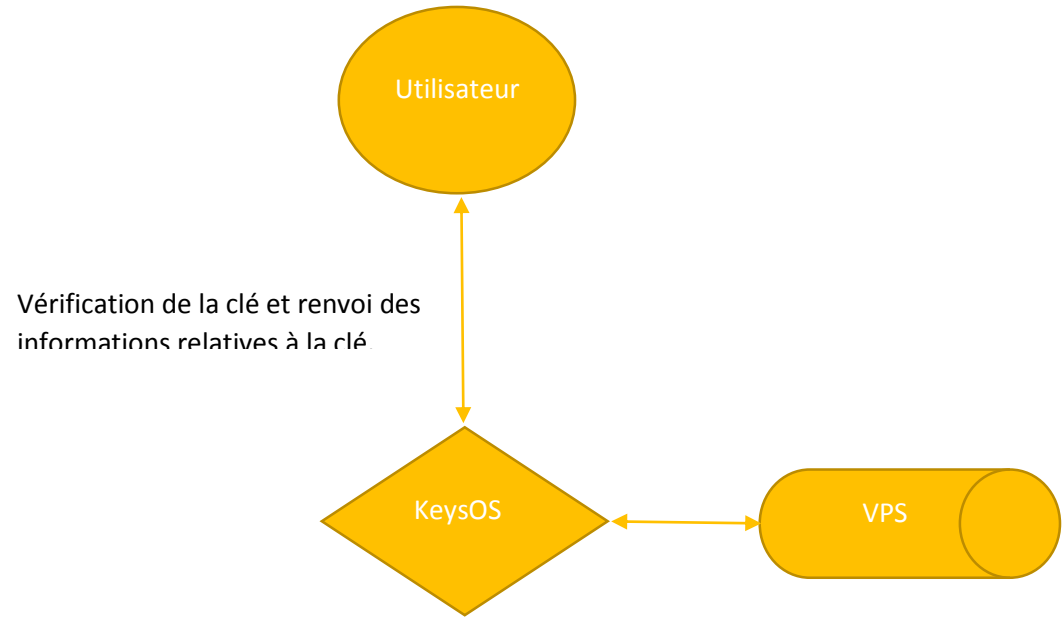
NB : A bien gardé le numero de téléphone ou les accès de votre adresse mail hors de la portée d'un inconnu pour que vos informations depuis la clé chiffrée ne soient pas déchiffrées sans votre confirmation.

- **Comment utiliser cette clé de chiffrement des informations avec mon compte mail, ou des réseaux sociaux** ?

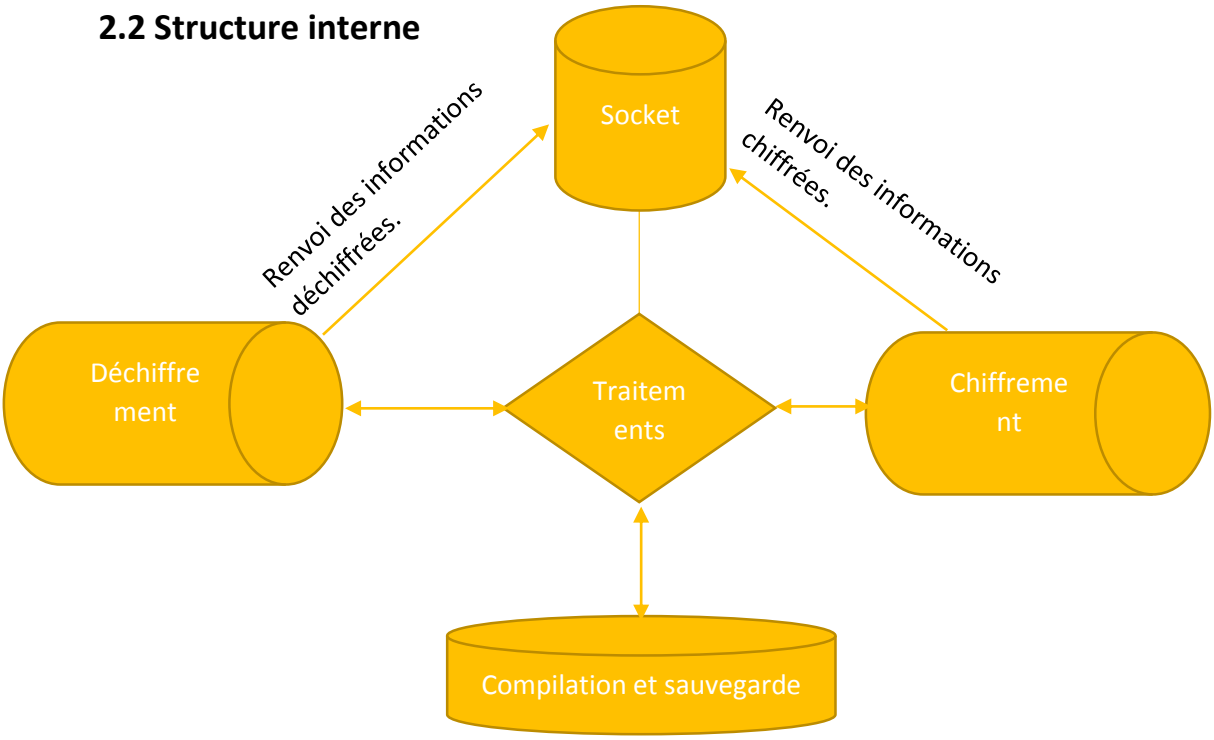
Les utilisateurs n'ont pas des problèmes pour l'utilisation de ce système de cryptage des informations, seuls les administrateurs de leurs plates-formes peuvent intégrer ce système de cryptage à leurs systèmes pour que les utilisateurs arrivent à l'utiliser aux chiffrements de leurs informations personnelles.

2. Organisation du système

2.1 Structure externes



2.2 Structure interne



2.3 Rôles et responsabilités

- **CREATE :** Cette balise (TAG) est utilisée à la création d'une clé contenant les informations personnelles ou une clé de licence pour une société privée. Elle est utilisée avec d'autres balises qui font en sorte qu'elle soit une balise du début et de la fin pour la requête complète. Autrement appelé, le corps de la commande.

- **KEYS :** La balise spécifique pour la clé chiffrée. Sera introduit pour la génération, la sécurisation et la vérification de la clé. Quant à la sécurisation de la clé, elle utilise la méthode booléenne avec deux variables **TRUE** et **FALSE**. Le **TRUE** pour demander au système KeysOS de sécuriser la clé et **FALSE** pour désactiver la sécurisation de la clé. Si vous l'avez désactivé durant la génération de votre clé, cette clé fonctionnera partout depuis un ordinateur de n'importe quel utilisateur pour la lecture des informations chiffrée qu'elle contienne. En outre, si elle est sécurisé avec la méthode **TRUE**, la clé fonctionnera uniquement depuis votre serveur virtuel ou votre propre ordinateur sur lequel la clé a été générée et si d'autres personne veulent le lire ou recevoir des informations sur la clé, le système KeysOS rejettera la demande.

NB : Par défaut cette balise est désactivée par la méthode **FALSE** pour la génération de la clé qui fonctionnera dans n'importe quel ordinateur sans vérification de sa société expéditrice.

- **SOCIETY** : L'entête qui contiendra toutes les informations relatives à la société sur laquelle veut générer la clé personnelle d'un utilisateur ou de sa propre clé de licence. Elle marche ensemble avec les balises **NAME** et **DOMAINNAME**.

- **NAME** : Définit le nom de la société entrain de générer la clé.

- **DOMAINNAME** : Le nom de domaine de la société pour un site internet (www.votresite.com).

- **SET** : Demande une clé unique depuis un autre département grâce à un fichier des scripts KeysOS enfin de générer une clé chiffrée pour le partage des informations entre deux établissements différents (B2B). Elle se combine avec la balise **FROM**.

- **FROM** : Spécifie une adresse sur laquelle le système va demander l'autorisation de se communiquer entre les établissements partenaires (Business to Business) en toute sécurité.

- **SEND** : Balise de renvoi de la clé unique du département associé une fois qu'elle a été demandé. Elle fonctionne quasiment avec la fonction **TO**.

- **TO** : Une adresse sur laquelle le système KeysOS vérifiera si et seulement si, elle est autorisé de partagé les informations avec le département ciblé (Notamment une adresse **IP**).

- **VERIFY** : L'entête qui contiendra toutes les informations relatives à la clé pour une demande de confirmation avant la lecture des contenus chiffré dans la clé. Elle se combine avec les commandes **ACTION**, **URLCONFIRM**, **DESTINATION** et **CONFIRM**.

- **ACTION** : Commande booléenne **TRUE** ou **FALSE** pour une demande de confirmation depuis le propriétaire de la clé en cours d'utilisation. Si et seulement si l'action est activée avec la méthode **TRUE**, le système synchronisera la requête avec le serveur privé de la clé pour une demande de confirmation avant sa lecture. Une connexion internet est requise pour la lecture des clés avec des URL de confirmation.

- **URLCONFIRM :** Permet de définir une adresse URL (https://www.votresite.com/mapage.php) sur laquelle le système KeysOS en verra un code de confirmation à l'adresse spécifique pour son utilisation avant la lecture de la clé.

NB : Le système KeysOS n'a pas un serveur privé d'envoi des mails ou des SMS aux destinateurs pour la confirmation de lecture de la clé. Vous devez spécifier une adresse URL privé que vous utilisé personnellement pour l'envoi de Mail et SMS.

- **DESTINATION** : Commande de destination d'un code de confirmation. Elle peut être un numero de téléphone mobile ou une adresse électronique (E-mail).

- **CONFIRM** : Elle demande la confirmation au système avant d'envoyer ou de ne pas envoyer le code de confirmation avec les propriétés **AUTO**, **MANUEL** et **USER**.

- **AUTO** : Si vous avez définit la confirmation par une propriété AUTO, le système affichera automatiquement toutes les informations relatives à la clé chiffrée (Cryptée) sans demande de la confirmation à l'utilisateur ou l'administrateur de la clé.

- **MANUEL** : En spécifiant la propriété MANUEL, le dit système KeysOS va procéder à la demande de confirmation avant que votre clé soit utilisée et activera depuis la console une session pour vous permettre de saisir le code de confirmation avant la lecture de la clé chiffrée.

- **USER** : Et lorsque la propriété est classé sur USER, le système l'utilisera pour envoyer un message de confirmation à l'utilisateur de la clé. Cette propriété ressemble à seule de la propriété MANUEL mais ici l'insertion d'un code de confirmation est depuis une adresse spécifique que le système KeysOS va spécifier. Du genre http://localhost:5364/?votrecodeici/

- **SECURITY** : Définit l'entête de la sécurité de la clé que vous voulez chiffrer.

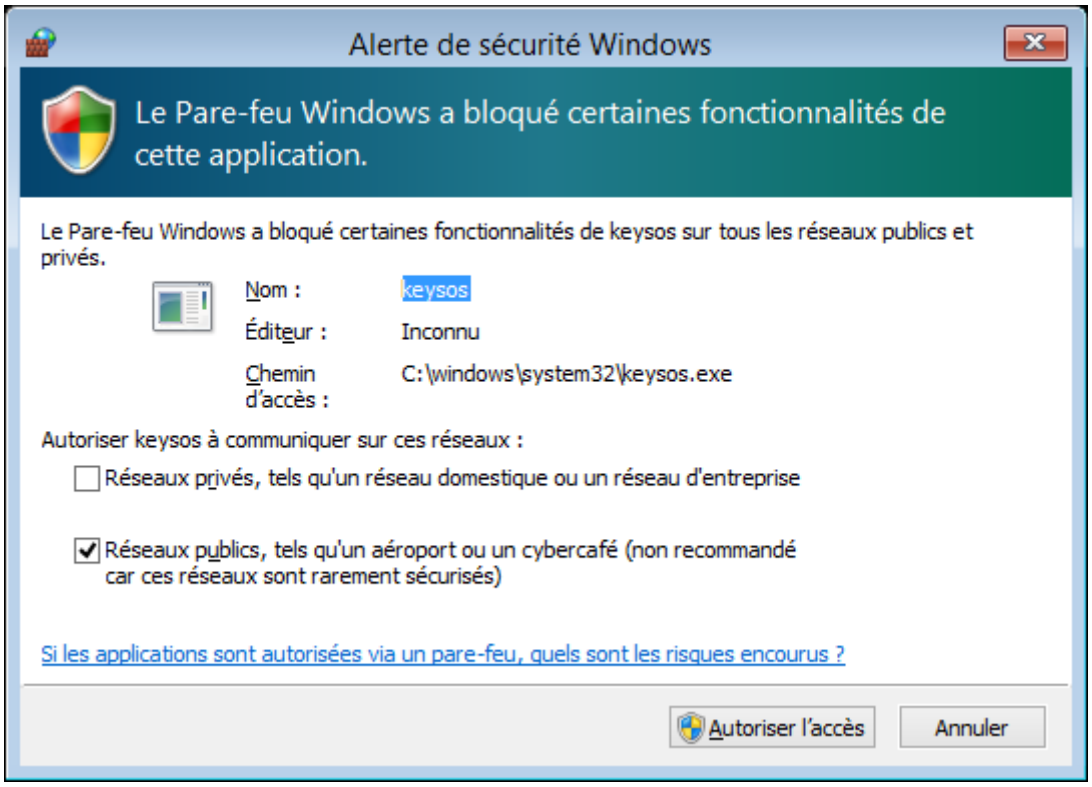
- **LEVEL** : Niveau de sécurité de la clé que vous voulez chiffrer. Par défaut le niveau de sécurité est sur 1. Ça dépend de votre processeur et de vos RAMs. Vous pouvez changer le niveau de sécurité par 3 ou 4 et ça dépend de vous et de votre processeur.

- **FIND** : L'entête de la recherche des données chiffrées. Fonctionnel avec la commande **DATA**.

- **DATA** : Permet d'inséré les informations personnels que vous voulez protéger depuis la clé à générer. On peut utiliser les caractères textes ou un lien d'un fichier locale.

- **RUN** : Une commande d'entête pour la demande d'exécution d'un socket.

- **SOCKET** : Commande avec deux propriétés booléenne **TRUE** et **FALSE** pour l'exécution d'un socket. Sous la plate-forme Windows desktop, si cette propriété est sur **TRUE**, vous verrez une boite de dialogue d'un pare-feu qui vous indique si vous voulez confirmer l'exécution d'un socket depuis le système KeysOS comme indiqué sur l'image ci-après :



NB : A sa confirmation, le message ne s'affichera plus et vous pouvez utiliser KeysOS pour la sécurisation de vos clés et leurs lectures en internes ou externes.

3. Plans du processus technique

3.1 Modèle du processus

- Le système des traitements des clés chiffrées (KeysOS) utilise trois modelés des processus de base pour sa mise en forme des informations des chiffrements ou des déchiffrements. L'une pour le traitement des clés qui ne requise pas sa confirmation depuis son expéditeur basé sur le traitement automatique du système, la seconde modèle de processus fonctionnelle avec l'aide d'un administrateur et la dernière

modèle de processus avec l'aide d'un utilisateur spécifique de la clé en cours d'utilisation.

A la génération d'une nouvelle clé, l'utilisateur peut spécifier quelle modèle de processus a utilisé depuis son propre système ou un autre système hors de sa portée. Si la clé crypté avec la première méthode de cryptage basé sur le traitement automatique, le système KeysOS affichera les informations relatives à la clé sans pour autant demandé une confirmation depuis l'administrateur ou l'utilisateur de la clé. A la génération de la clé crypté d'une modèle secondaire pour l'administrateur, la sélection et l'affichage des informations sécurisé depuis la clé requise une confirmation de l'administrateur du dit système par un code de confirmation uniquement pour la session ouverte. Pour la dernière modèle des processus, le système des traitements des clés est censé d'envoyer un code de confirmation à l'utilisateur pour s'assurer que la requête demandé a bien été effectué par l'utilisateur dédié pour que les informations personnelles ne soient pas divulgué par d'autres personnes inattendus.

3.2 Méthodes et techniques

- Le système des traitements des clés peut être utilisé non seulement depuis la console mais peut être aussi appelé depuis une application web et intégré dans une application de bureau grâce à une bibliothèque dynamique personnalisé disponible l'une pour l'extension **.dll** pour la plate-forme **Windows** et l'autre en **.so** pour les plates-formes **Linux**. Depuis l'invité des commandes, Shell ou autres, l'administrateur peut générer des différents types des clés avec des commandes spécifiques à KeysOS. Pour une application web, l'administrateur peut utiliser un fichier d'une extension .**KS** de KeysOS enfin de l'appelé depuis le même fichier pour avoir le résultat de la clé généré parce qu'il s'exécute comme un fichier des scripts.

- L'application web peut aussi utiliser les commandes Shell pour ces requêtes en faisant un appel du système KeysOS depuis une invité des commandes ou Shell. Pour les bibliothèques dynamiques offertes, les développeurs peuvent intégrer tous les systèmes de chiffrement dans leurs applications enfin d'interagir avec les clés généré sans passé par l'invité des commandes ou Shell mais avec des langages spécifiques. Une fois que le système KeysOS est

appelé et la clé sécurisé pour une utilisation permanente, à la vérification de la clé, le système des traitements des clés est censé de vérifier toutes les informations relatives à la clé avant d'afficher le résultat final. A l'intérieur de la clé généré, KeysOS a d'autre fonctionnalités que vous allez voir ci-dessous leurs rôles et fonctionnalités sur la section 5.

4. **Prix**

Le système des traitements des clés tant qu'il donne à la portée des tous deux modes des générations des clés et deux modes pour les lectures des clés chiffrées (Cryptées), les générations des clés avec ou sans licence sont quasiment gratuites pour qu'il soit intégré dans tous les systèmes informatique Web ou Local. Vous avez l'opportunité d'utilisé les commandes **KeysOS** comme n'importe quel langage de programmation dans un fichier contenant les commandes (Script) KeysOS ou faire un appel de vos commandes depuis une interface des commandes (Console) sans frais de paiement.

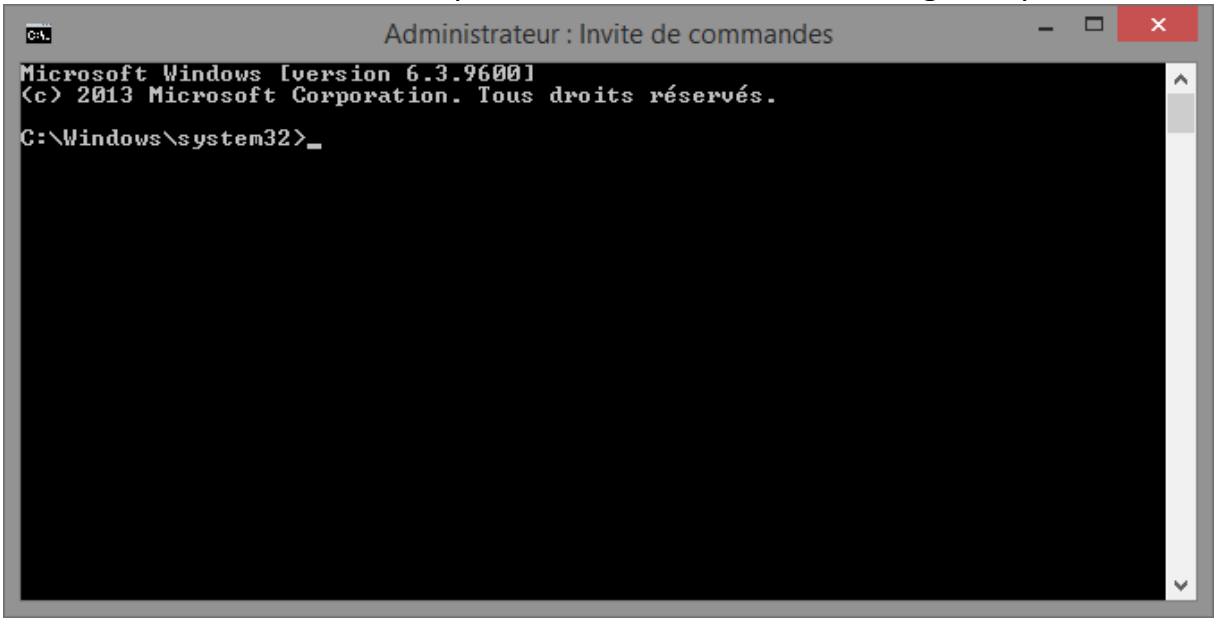
Quant à la lecture des clés chiffrées par le dit système, tant qu'il donne deux modes des lectures avec une clé de licence et sans clé de licence en mode **FREE**, la lecture d'une clé chiffrée en mode FREE et gratuite sans frais de paiement. En outre, pour la lecture d'une clé chiffrée avec une licence spécifique pour les entreprises, les paiements ses font par **20 sessions** pour **0.1$**. Tel est le prix unitaire pour les sessions. La société à qui utilise le système KeysOS pour les lectures des clés chiffrées peut valider sa clé de licence pour une durée d'un mois, six mois ou même plus ça dépend de la société.

5. **Manuel d'utilisation et de l'installation**

Ce manuel d'utilisation et de l'installation est spécifiquement pour les amoureux de la commande. Nous allons voir comment installer KeysOS sous **Windows 8.1** et **Ubuntu 18.4**. Pour d'autres versions les méthodes d'installation sont similaires. Avant de procéder, vous pouvez télécharger le master au format .zip à l'adresse ci-après :

https://github.com/AnetoEnterprise/KeysOS/archive/main.zip

- ✓ **Installation sous Windows 8.1** :
  Premièrement vous devez lancer la console en tant qu'un administrateur. Pour procéder, allez sur la barre de recherche de Windows et tapez la commande **CMD**. Une liste des applications apparaissent et faites un clic droit sur l'application de la console et sélectionner : **Exécuter comme administrateur**. La console va s'exécuter en tant qu'administrateur comme l'image ci-après :



Une fois que vous avez l'invité de commande (Console) a votre portée, dezipper le fichier téléchargé depuis :
https://github.com/AnetoEnterprise/KeysOS/KeysOS.zip/
Et tapez les commandes :
cd D:/Windows
cp ./keysos.exe C:/Windows/System32/keysos.exe
clear

- ✓ **Installation sous Ubuntu 18.4** :
  Même procédure d'installation depuis la plate-forme linux.
  Premièrement télécharger le fichier :
  https://github.com/AnetoEnterprise/KeysOS/KeysOS.tar.gz/
  Et procéder à l'installation en saisissant les commandes suivantes :
  tar xpf KeysOS.tar.gz
  cd /Linux
  cp ./keysos /usr/bin/
  reset

Exemple1 :
Maintenant, nous voulons tester notre système des traitements des clés chiffré avec un exemple simple d'une clé qui affiche le Nom, Post-nom et Prénom de l'utilisateur sans demandé la confirmation depuis l'administrateur du système ou l'utilisateur de la clé lui-même.

Notre utilisateur à comme informations : **Aneto, Minanga, Guyllit** tel est son Nom, Post-nom et prénom. Pour procéder, depuis la console nous allons taper les commandes ci-après :

```
CREATE KEYS{
SOCIETY{
NAME="MySociety",
DOMAINNAME="www.mysociety.com"
}
SECURITY{
LEVEL=1
}
FIND{
DATA="Aneto,Minanga,Guyllit"
}
};
```

Exemple2 :
Nous allons recommencer le même exemple mais cette fois si nous allons spécifier la date et l'heure d'expiration de la clé. Si la date du jour et l'heure est supérieur à la date et l'heure de la clé. L'affichage des informations relatives à la clé sera aboli.

```
CREATE KEYS{
SOCIETY{
NAME="MySociety",
DOMAINNAME="www.mysociety.com"
}
SECURITY{
LEVEL=1
}
EXPIRATION{
DATE=CURRENT_DATE,
```

```
TIME="07:50"
}
FIND{
DATA="Aneto,Minanga,Guyllit"
}
};
```

Exemple3 :
Génération d'une clé qui fonctionnera uniquement depuis la plate-forme sur laquelle elle a était généré. Si une autre personne veut l'utiliser dans un autre environnement, le système rejettera sa requête. En outre si l'environnement est correcte, le système envoi un code de confirmation à l'adresse spécifique de la commande URLCONFIRM pour la destination de la commande DESTINATION et la confirmation du système automatique de la commande CONFIRM.

```
CREATE KEYS{
SOCIETY{
NAME="MySociety",
DOMAINNAME="www.mysociety.com",
KEYS=TRUE
}
VERIFY{
ACTION=TRUE,
URLCONFIRM="https://www.yoursite.cd/getconfirm.php",
DESTINATION="yourmail@yoursite.com",
CONFIRM=AUTO
}
SECURITY{
LEVEL=1
}
FIND{
DATA="Aneto,Minanga,Guyllit"
}
RUN{
SOCKET=TRUE
}
};
```

Exemple4 :
Génération d'une clé qui permettra la communication entre deux départements pour le partage des informations relatives à leurs utilisateurs (Business to Business). Cette méthode est fonctionnelle entre deux fichiers des scripts **KeysOS**. Le premier fichier a pour but de demandé l'autorisation d'une clé unique au département secondaire enfin de générer la clé qui va transmettre les informations à partager et le deuxième fichier permettra de renvoyer la clé unique au premier département pour la génération de la clé de partage des informations. Pour se faire, créer un fichier des scripts du nom de **test1.ks** et mettez-y les commandes ci-après :

```
# !/usr/bin/keysos

CREATE KEYS{
SOCIETY{
NAME="MySociety",
DOMAINNAME="www.mysociety.com",
KEYS=TRUE
}
SET KEYS{
FROM="https://www.yoursite.cd/cgi-bin/script.ks"
}
VERIFY{
ACTION=TRUE,
URLCONFIRM="https://www.yoursite.cd/getconfirm.php",
DESTINATION="yourmail@yoursite.com",
CONFIRM=AUTO
}
SECURITY{
LEVEL=1
}
FIND{
DATA="Aneto,Minanga,Guyllit"
}
RUN{
SOCKET=TRUE
}
};
```

Nous venons de codé notre premier fichier des scripts capable de demandé une clé unique avant la génération de la clé chiffrée. Maintenant nous allons voir comment créer un fichier **script.ks** qui est définit dans la balise FROM enfin de renvoyer la clé unique demandé. Créer un autre fichier **script.ks** et mettez-y les scripts ci-après :

```
# !/usr/bin/keysos

SEND KEYS{
TO="127.0.0.1"
};
```

NB : L'adresse IP 127.0.0.1 définit est seule que le système va vérifier si elle est autorisé a demandé la clé unique pour le partage des informations relatives aux utilisateurs. A bien modifier l'adresse à une adresse IP sur laquelle vous voulez autoriser à recevoir les informations concernant votre entreprise.

Une fois que tous les deux fichiers seront prêts, lancez le premier fichier **test.ks** pour demander la clé unique et générer une clé chiffrée (Cryptée).

Exemple5:
Faire un appel d'une bibliothèque dynamique depuis une application console en utilisant le langage **C++** sous **Windows 8.1**. Premièrement créer un fichier **keysos.h** et ajoutez-y les informations ci-après :

```
#ifndef KEYSOSLIB_H
#define KEYSOSLIB_H

#include <iostream>

using namespace std;

class KeysOSLib{
public:virtual string GetKeysForAdmin(string port, string code, string
licencelink, string keys)=0;
public:virtual string SelectKeys(string request, string keys)=0;
public:virtual string CreateSelectKeys(string request)=0;
};
#endif
```

Puis créer un autre fichier **test.cpp** et ajoutez-y les informations ci-après :

```
#include <iostream>
#include <string>

#include <json/json.h>
#include <json/reader.h>
#include <json/writer.h>
#include <json/value.h>
#include <unistd.h>
#include "keysoslib.h"



using namespace std;



void *handle;
KeysOSLib* ps;
typedef KeysOSLib* create_t();
typedef void destroy_t(KeysOSLib*);
create_t* openKeysOS;
destroy_t* closeKeysOS;



Json::Value root;
Json::CharReaderBuilder builder;
Json::CharReader * reader = builder.newCharReader();
bool parsingSuccessful=false;
string errorJSON="";
string valueFindXML="";
```

```
int main(int argc, char** argv)  {
handle=dlopen("./keysos.dll", RTLD_LAZY);
if(!handle){printf("%s", dlerror());}
openKeysOS=(create_t*)dlsym(handle, "create");
closeKeysOS=(destroy_t*)dlsym(handle, "destroy");
if(!openKeysOS){cout << "%s" << dlerror();}
if(!closeKeysOS){cout << "%s" << dlerror();}
ps=openKeysOS();
string response=ps->CreateSelectKeys("Your request here");
cout << response << "\n";
closeKeysOS(ps);

//Début obtention des données XML du resultat
parsingSuccessful = reader->parse(response.c_str(), response.c_str() +
response.size(), &root, &errorJSON);
if ( !parsingSuccessful ){
cout << errorJSON << "\n";
}else{
valueFindXML=root["DATA"].asString();
cout << valueFindXML << "\n";
}
//Fin obtention des données XML du resultat
return EXIT_SUCCESS;
}
```

A la fin, nous pouvons compiler notre application avec la commande suivante :

```
cd E:/votredossier
g++ DemoForWindows.cpp -o DemoForWindows.exe -I"C:/jsoncpp" -ljsoncpp -
ldl
./DemoForWindows.exe
```

Pour plus d'infos sur les commandes, voir le fichier **README** que vous avez téléchargé dans le dossier **Docs** ou allez sur :

https://github.com/AnetoEnterprise/KeysOS/README.txt

Liens YouTube :

https://www.youtube.com/channel/UCgpAwIaEGUAsf7ZAHp71zFg
https://youtu.be/aF5xeDx9zz0

Merci de votre disposition pour la lecture complète de ce système.

Pour plus d'infos :
E-mail: infos@keysos.org
Site web: http://www.keysos.org/
Whatsapp: 243 81 90 33 888