BIRLAN Daniel GM3 - SCHATZ Yohann GC3

INSA de Strasbourg

#TrustedCode

-

Projet VBA

[DOSSIER DE PRÉSENTATION]

2018

A – Objectifs

Dans le cadre du module électif d'Excel avancé et de programmation VBA, nous avons choisi d'orienter notre projet autour d'un enjeu de plus en plus important de nos jours, qui est la sécurité informatique, en particulier de nos moyens de communication. Ainsi, notre choix de projet s'est porté sur la création d'une interface de communication sécurisée. En tant qu'étudiants ingénieurs respectivement en Génie Mécanique et en Génie Civil, ce projet nous a permis de chercher au-delà de nos spécialités et d'aborder un sujet qui nous est que très peu connu afin d'approfondir et de mettre en pratique nos connaissances de programmation en Excel-VBA. Le choix de ce projet a eu également pour objectif de nous confronter à des problèmes puisque nous n'étions initialement que très peu adeptes du langage de programmation VBA, et c'est en trouvant des solutions à nos problèmes que nous sommes parvenus à mieux comprendre les modes de fonctionnement du langage VBA. D'autant plus qu'en tant que futurs ingénieurs, il sera de notre ressort de trouver des solutions à des problèmes complexes ; et chercher en dehors de nos compétences acquises, à l'école par exemple. Pour finir, il est fondamental qu'un ingénieur soit capable de créer ses propres outils lui permettant de trouver des solutions à certains problèmes ; et non uniquement d'utiliser divers logiciels et feuilles Excel prêts à l'emploi, puisque les problèmes auxquels l'ingénieur doit savoir répondre changent constamment avec l'avancée technologique.

Plus concrètement, l'objectif fixé de ce projet a été de créer une interface, relativement facile d'utilisation, permettant une communication entre plusieurs utilisateurs par un système d'écriture et de lecture de fichiers *« .txt ».* L'intérêt majeur du projet étant de proposer divers algorithmes de cryptographie permettant de garantir la sécurité de la communication. Pour répondre à ces objectifs, le projet s'est articulé suivant une multitude de macros et de fonctions écrites en VBA. Les liens entre ces diverses macros et fonctions se sont faits sur la feuille Excel sur l'Userform mais aussi au niveau temporel, dans le but d'obtenir une interface de discussion relativement « instantanée ».

B - Mode d'emploi

1 - Prérequis

Contenu du package :

- Fichier *TrustedCodeF.xlsl* [messagerie sécurisée programmée en VBA]
- Fichier *Test.txt* [stockage du dernier message crypté envoyé]
- Fichier *User1.txt* [stockage du nom de l'utilisateur sur le Canal 1]
- Fichier *User2.txt* [stockage du nom de l'utilisateur sur le Canal 2]

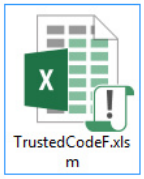
} **Dossier TrustedCode_files**

Installation :

- Installer l'application **Dropbox** sur les deux machines
- Partager via l'application Dropbox le dossier TrustedCode_files. Veiller à ce que les fichiers du ce dossier soient accessibles et modifiables depuis les deux machines.

2 - Lancement & paramétrage de TrustedCode

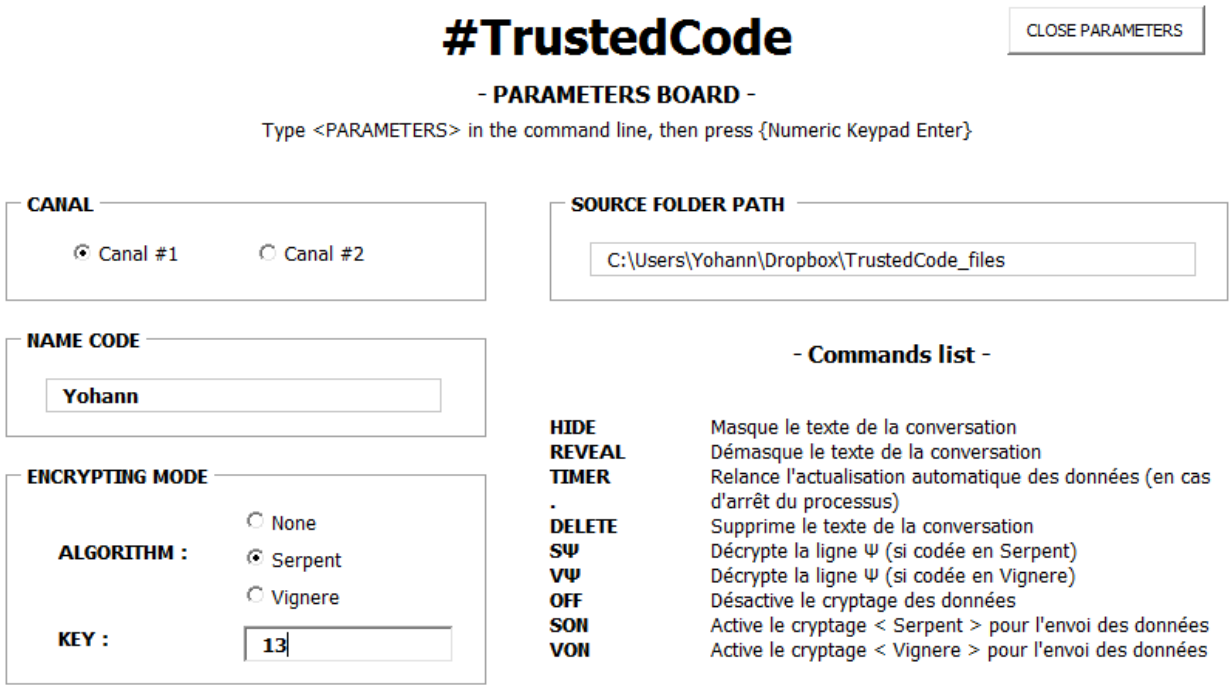
- Ouverture du fichier *TrustedCodeF.xlsl*



- Le tableau des paramètres s'affiche automatiquement, permettant les entrées suivantes :

Ordre à respecter :

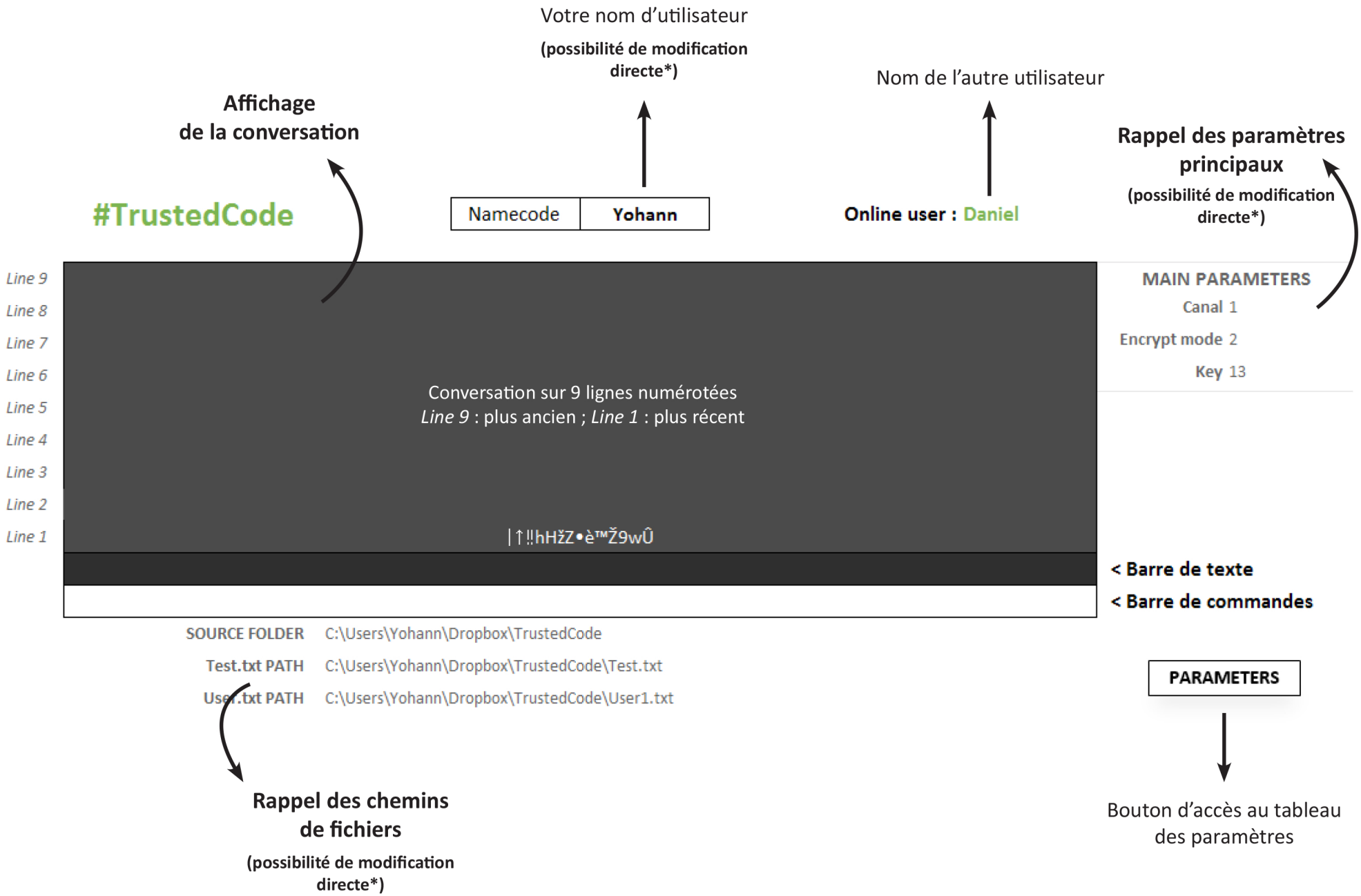
1. **[CANAL]** le choix du canal (les deux utilisateurs doivent choisir un canal différent)
2. **[SOURCE FOLDER PATH]** la saisie du chemin du dossier TrustedCode_files (dans le dossier Dropbox)
3. **[NAME CODE]** la saisie du nom d'utilisateur
4. **[ENCRYPTING MODE]** **ALGORITHM :** le choix du mode de cryptage pour les messages envoyés
   KEY : la saisie de la clé de cryptage



Une fois le paramétrage terminé, cliquer sur le bouton [CLOSE PARAMETERS].

- On retrouve l'interface suivante :

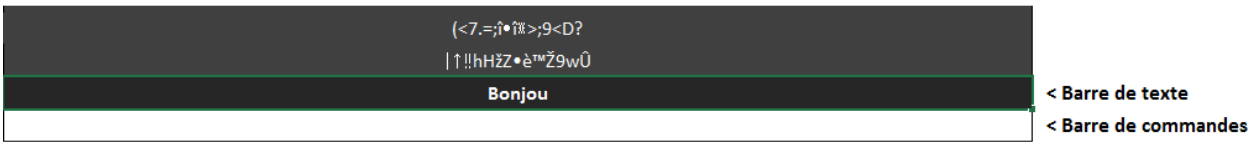
Interface utilisateur



Possibilité de modification directe : l'information peut être éditée directement dans l'interface*

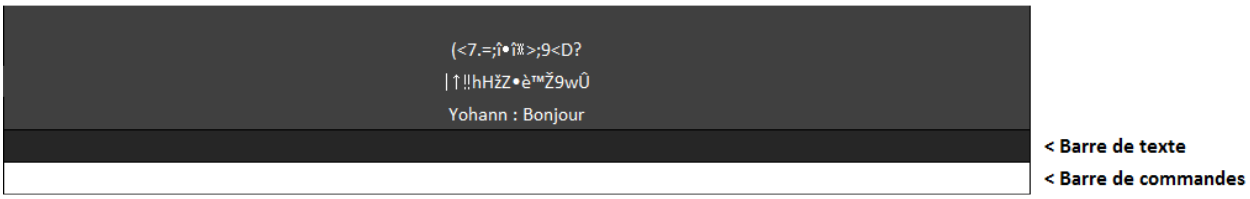
3 - Envoi d'un message

- Écriture du message dans la barre de texte



- Envoi du message en pressant la touche ENTRÉE

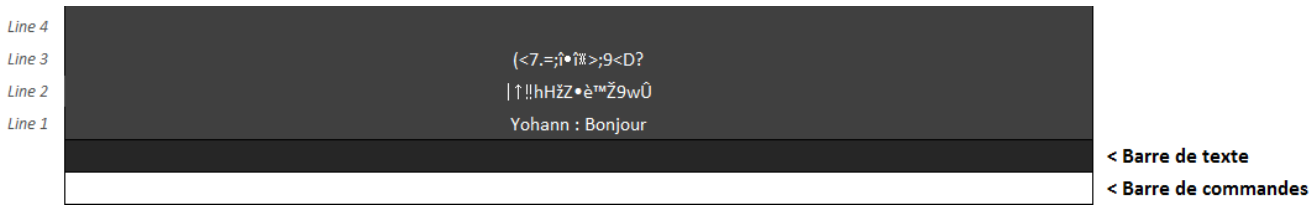
La conversation est automatiquement mise à jour :



4 - Réception et décryptage d'un message

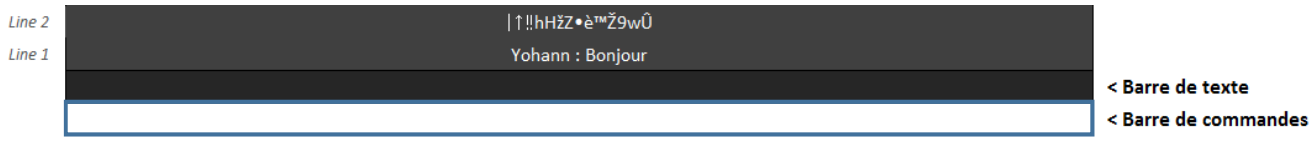
- Le rafraîchissement de la conversation s'effectue automatiquement grâce à la boucle Timer. En cas d'interruption du timer, la combinaison ALT + HAUT permet un rafraîchissement manuel.

Dans cet exemple, la conversation est la suivante :



- Le message en ligne 1 (line 1) n'a pas été crypté
- Le message en ligne 2 (line 2) a été crypté en langage Serpent
- Le message en ligne 3 (line 3) a été crypté en langage Vignere

Il n'y a pas besoin de manipulation pour décrypter la ligne 1, celle-ci apparaît déjà en Français.
Pour décrypter la ligne 2, il est nécessaire d'utiliser la **barre de commandes** :



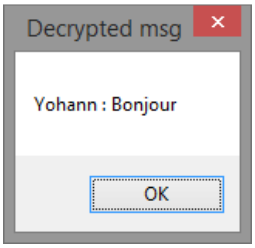
La **barre de commande** fonctionne comme celle d'*Autocad* : une action peut être exécutée en tapant le **code** correspondant dans la barre et en pressant **CTRL** + **HAUT**

Une liste complète des commandes est donnée dans la table des paramètres :

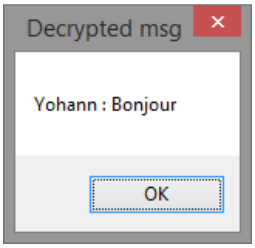
- Commands list -

HIDE	Masque le texte de la conversation
REVEAL	Démasque le texte de la conversation
TIMER	Relance l'actualisation automatique des données (en cas d'arrêt du processus)
DELETE	Supprime le texte de la conversation
SΨ	Décrypte la ligne Ψ (si codée en Serpent)
VΨ	Décrypte la ligne Ψ (si codée en Vignere)
OFF	Désactive le cryptage des données
SON	Active le cryptage < Serpent > pour l'envoi des données
VON	Active le cryptage < Vignere > pour l'envoi des données

Pour décrypter la **ligne 2** cryptée en **Serpent**, on saisit dans la barre de commandes : **S2**
(S pour Serpent, 2 pour le numéro de la ligne)
En pressant **CTRL** + **HAUT** , une boîte de dialogue apparaît présentant le message décodé.



Pour décrypter la **ligne 3** cryptée en **Vignere**, on saisit dans la barre de commandes : **V3**
(S pour Vignere, 3 pour le numéro de la ligne)
En pressant **CTRL** + **HAUT** , une boîte de dialogue apparaît présentant le message décodé.



C - Fonctionnement général

C1 - Synthèse du code VBA

Note : *pour expliquer plus en détail le code, celui-ci comporte de nombreuses annotations*

Dans cet exemple, nous serons dans le cas d'un utilisateur sur le **Canal 1** ; l'autre utilisateur est sur le **Canal 2**

Notation : psb → Private Sub ;

Déclenchement	Nom	Description
Ouverture du fichier	Workbook_Activate	- Affecte des actions à la touche [ENTRÉE] (lancer **psbTouche_envoi**) et aux combinaisons [CTRL+HAUT] (lancer **psbTouche_commande**) et [ALT+HAUT] (lancer **psbActualisation**) de façon permanente (code exécuté en mode continu) - Ouvre l'UserForm « Paramètres » - Lance l'application **psbCycle_timer**
	psbCycle_timer	- Lance l'application psbActualisation à t + 5 secondes
	psbActualisation	- Lance simultanément les applications **psbLecture** (lecture des données dans *Test.txt*), **psbName** (récupération et contrôle de votre nom d'utilisateur dans *User1.txt*) et **psbConnexion** (récupération et contrôle du nom de l'utilisateur 2 dans *User2.txt*) - Relance **psbCycle_timer** (bouclage du timer)
	psbLecture	***Lecture et contrôle du fichier Test.txt*** - Ouvre le fichier *Test.txt* et récupère le texte de celui-ci - Processus de contrôle : si le texte récupéré dans *Test.txt* ne correspond pas à la dernière ligne de la conversation, celle-ci doit être mise à jour → Démarre l'application **psbAffichage**
	psbAffichage	***Mettre à jour l'affichage de la conversation (afficher les nouveaux messages)*** - Décale les lignes de la conversation d'un cran vers le haut (ligne suivante) et insère le texte récupéré en tête de conversation (première ligne)
	psbName	***Mettre à jour le fichier User1.txt*** - Ouvre le fichier *User1.txt* et récupère votre nom d'utilisateur - Processus de contrôle : si le nom récupéré ne correspond pas à celui affiché dans « Namecode : » → remplace le nom dans le fichier *User1.txt*
	psbConnexion	***Mettre à jour le nom de l'autre utilisateur affiché dans l'interface*** - Ouvre le fichier *User2.txt* et récupère le nom de l'utilisateur 2 - Processus de contrôle : si le nom récupéré ne correspond pas à celui affiché dans « Online user : » → remplace le nom affiché sur l'interface
Touche [ENTRÉE]	psbTouche_envoi	- Processus de contrôle : si la cellule contenant le texte à envoyer est vide (message blanc) → affichage d'une boite de dialogue demandant de poursuivre ou non l'envoi - Processus de contrôle : si le chemin du fichier Test.txt n'est pas spécifié → affichage d'un message d'erreur - Lance l'application **psbEnvoyer**
	psbEnvoyer	- Cryptage du texte selon le mode défini dans l'interface paramètres (cryptage Serpent, Vignere ou cryptage inactif) - Lance l'application **psbAffichage** - Lance l'application **psbÉcriture**
	psbÉcriture	- Écrit le texte crypté (ou non) dans le fichier *Test.txt*
Combinaison touches [CTRL] + [HAUT]	psbTouche_Commande	- Lance l'application **psbContrôleur_commandes** - Efface le contenu de la barre de commandes
	psbContrôleur_commandes	- Processus de contrôle : en fonction du code saisi dans la barre des commandes → lance l'application correspondante (voir **partie B** pour une description détaillée des applications de commande)
Fermeture du fichier	Workbook_BeforeClose	- Démarre l'application **psbSuppression** - Sauvegarde le fichier
	psbSuppression	***Restauration du fichier à son état initial (efface les saisies)*** - Supprime toutes les informations saisies dans l'interface (paramètres, conversation) et modifie les fichiers *User.txt*

C2 - Réseau

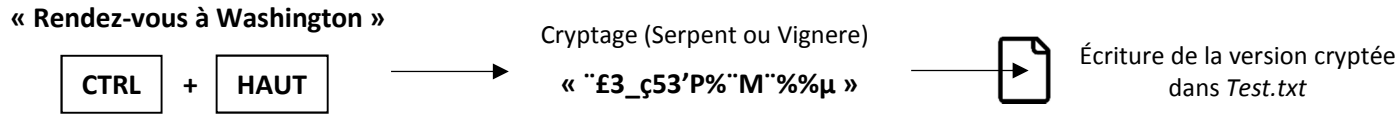
L'intérêt de Dropbox est de permettre une synchronisation en réseau des fichiers textes accessibles localement sur chaque machine.

Le principe de fonctionnement de TrustedCode est le suivant :

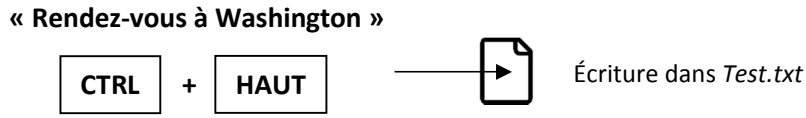
Cryptage et envoi des données -

- Le fichier *Test.txt* contient une ligne unique de texte correspondant au message envoyé. Lorsqu'un utilisateur écrit un message puis l'envoie avec [ENTRÉE], la saisie est écrite sur le fichier et **remplace** le contenu d'origine.

Si le **cryptage est activé** (Serpent ou Vignere), le texte est converti en texte crypté avant son écriture dans *Test.txt.*



Si le **cryptage est désactivé,** le texte est directement écrit dans *Test.txt* (sans traitement préalable).



Réception des données et décryptage -

Toutes les **5 secondes**, une actualisation est lancée par le logiciel : si le contenu de *Test.txt* n'est pas encore affiché sur l'interface de conversation, alors la conversation est mise à jour. *(psbLecture) (psbAffichage)*

Le contenu de Test.txt apparaît donc sur l'interface. Si celui-ci est crypté, il est possible d'afficher une traduction décryptée en utilisant la combinaison **S** + **CHIFFRE** ou **V** + **CHIFFRE** (*voir mode d'emploi*). Ces combinaisons permettent de lancer les algorithmes de décryptage donnant leur résultat via une **MsgBox**.

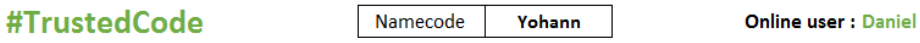
Affichage de l'utilisateur en ligne et principe des canaux -

L'utilisation de canaux permet de distinguer les deux utilisateurs :
- Si le **canal 1** a été choisi, l'utilisateur est l'**utilisateur 1** (ou *User 1*)
- Si le **canal 2** a été choisi, l'utilisateur est l'**utilisateur 2** (ou *User 2*)

- Lorsque l'utilisateur 1 saisit son nom (Namecode), celui-ci est écrit dans le fichier ***User1.txt***
- Lorsque l'utilisateur 2 saisit son nom, celui-ci est écrit dans le fichier ***User2.txt***

Si vous êtes **utilisateur 1** (vous avez sélectionné le canal 1), alors le nom de l'utilisateur 2 va être récupéré dans le fichier **User2.txt** et être affiché dans votre interface (**Online user : + NOM**). Si vous êtes **utilisateur 2,** la procédure est inverse.

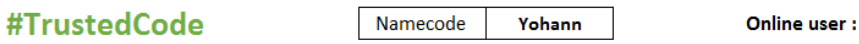
Ce processus est actualisé toutes les 5 secondes en cas de changement du nom d'utilisateur (via le timer).



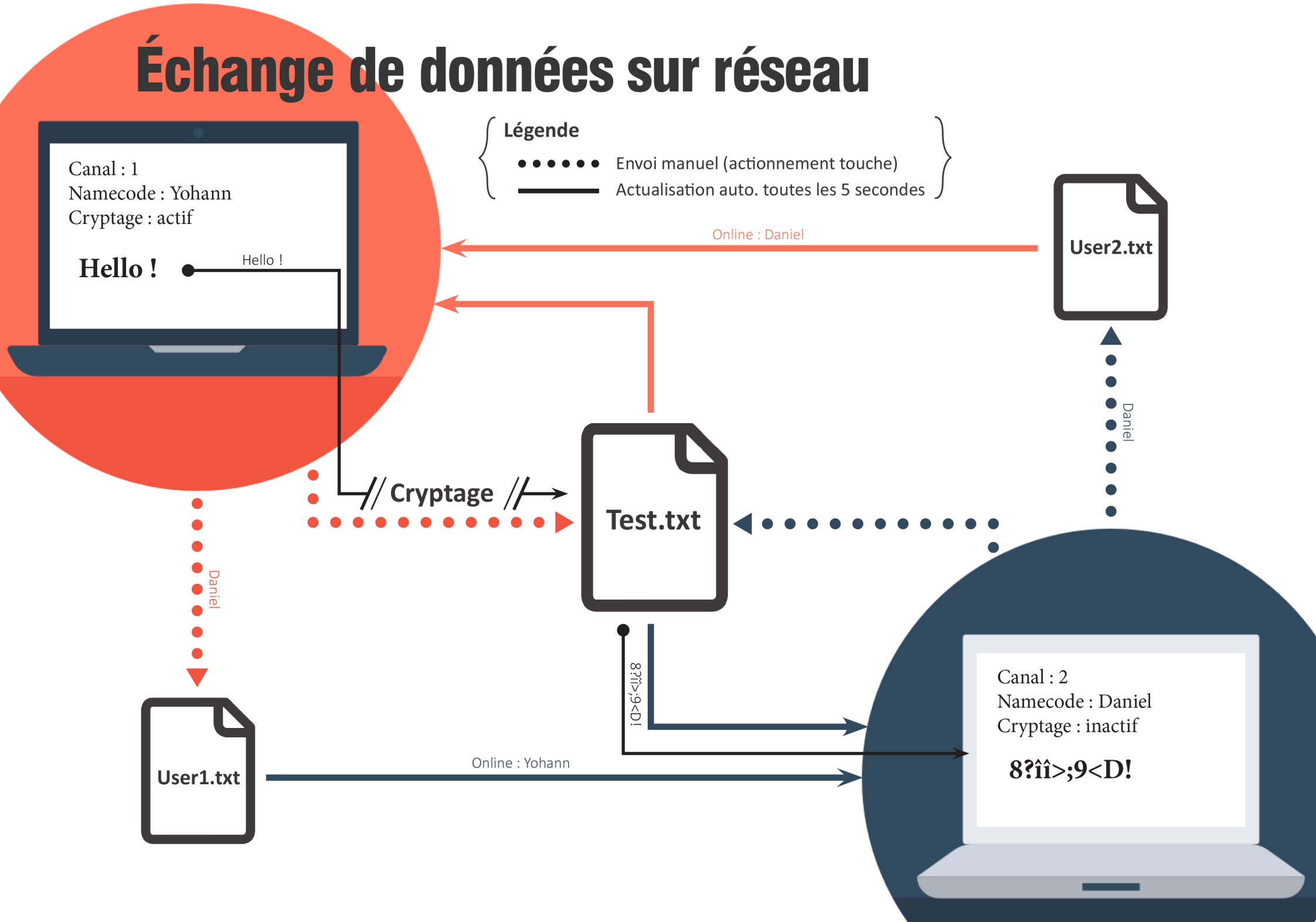
Lorsque l'**utilisateur 2** ferme *TrustedCode*, l'application **psbSuppression** est lancée et remplace le contenu du fichier *User2.txt* par la mention « DISCONNECTED »
L'actualisation de la lecture du nom d'utilisateur impose une boucle si :

Si le contenu du fichier User2.txt est « DISCONNECTED », alors rien n'est affiché (blanc)
Sinon
Le contenu est un nom et celui-ci et affiché

Donc, lorsque la mention DISCONNECTED va être récupérée par l'**utilisateur 1**, le nom sera effacé dans l'interface indiquant que l'utilisateur 2 a quitté la conversation.



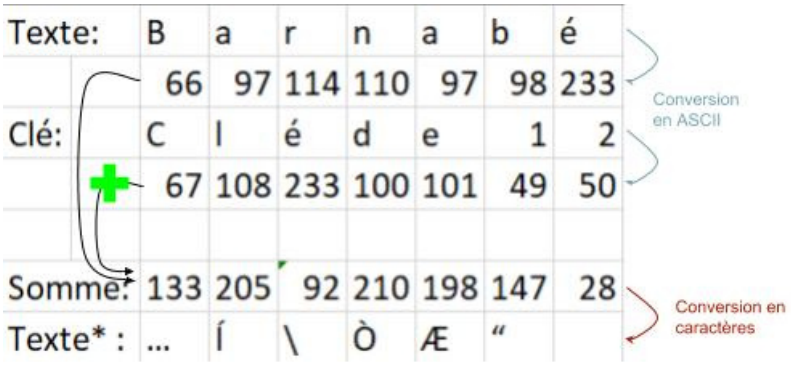
Échange de données sur réseau



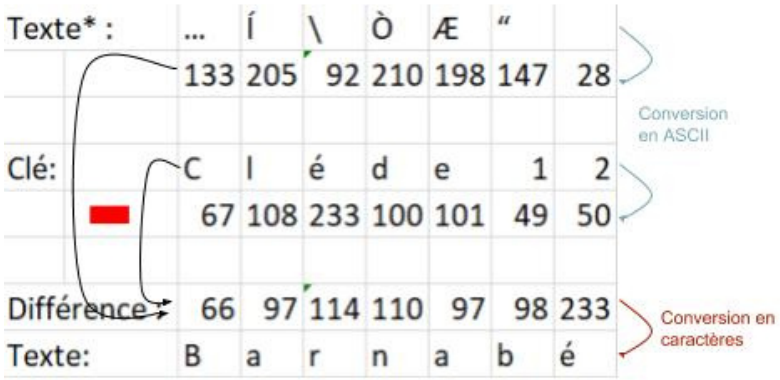
C3 - Algorithmes de cryptage

Nous avons implémenté deux algorithmes de cryptage pour ce projet : d'une part, un algorithme basé sur le chiffrement de *Vignere* qui fonctionne relativement facilement par substitution ; et d'autre part, un algorithme sophistiqué de chiffrement « par bloc » de 256 bits appelé Serpent. Le dernier étant très élaboré, nous l'avons récupéré au format Visual Basic depuis une base de données sur Internet. Chacun des deux algorithmes est utilisé uniquement par l'intermédiaire de fonctions « Encrypt » et « Decrypt » qui renvoient une série de caractères (String) à partir de données de texte (chiffré ou non selon le besoin de chiffrer ou de déchiffrer) et d'une clé de chiffrement. Nous expliciterons par la suite le fonctionnement de l'algorithme *Vignere.*

L'algorithme *Vignere* est basé sur le principe de substitution, c'est-à-dire que l'on vient additionner le numéro du premier caractère de la clé au premier caractère du texte, et le second caractère de la clé au second caractère du texte ; et ainsi de suite.



Dans l'exemple ci-dessus, nous avons chiffré le texte « Barnabé » avec la clé « Cléde12 » en suivant l'algorithme *Vignere.* Or, lors de la sommation de deux caractère, il est possible que la somme la soit pas entre 0 et 255 (qui est la plage des symboles ASCII), dans ce cas on soustrait simplement 255 au résultat (dans l'exemple, 114 « r » + 233 « é » -255 donne bien 92).



Réciproquement, pour déchiffrer un texte en suivant l'algorithme *Vignere*, il suffit de soustraire la clé au texte chiffré. A nouveau, il se peut que le résultat soit négatif, auquel cas on ajoute simplement 255 au résultat.

Ainsi, nous sommes bien parvenus à implémenter deux fonctions sous VBA permettant le cryptage et le décryptage de chaînes de caractères en fonction d'une clé.

D – Sources et références

https://vba.developpez.com/
http://www.freevbcode.com/ShowCode.asp?ID=3779