**STOP VM**

Afin de permettre une exploitation efficace des ressources système d’un serveur de machines virtuelle, nous avons besoin d’automatiser l’arrêt de celles-ci car parfois les utilisateurs les laissent allumées ce qui ampute les ressources disponibles pour les machines actives et empêche le démarrage d’autres machines à la demande.

**Partie Client**

La partie client devra :

* Être déployé automatiquement par l’intermédiaire d’un setup d’installation sur la machine cible. (Dans toolsVM) // Ps : Penser à faire un tuto pour Inno Setup
* Fonctionner avec les toutes les versions de Windows, de XP à maintenant.
* Surveiller l’activité d’un ou plusieurs processus cible.
* Surveiller l’activité d’une connexion distante type RDP sur la machine cible.
* A partir des données collectées, évaluer le degré d’activité et en fonction des résultats et d’un seuil configuré puis déclencher une série d’actions (définies plus bas).
* Logger avec horodatage dans un fichier l’ensemble des évènements de l’application (à définir).
* Une interface (verrouillée par mot de passe) permettant la saisie de paramètres en local, temporisation par type d’action, adresse IP serveur de conf.
* Utiliser un fichier de configuration type app.conf ou ini (verrouillé par cryptage) afin de mémoriser les paramètres personnalisés.
* Se connecter à un serveur distant en attente de réception de message (définit plus bas).
* Déclencher une action au bout d’une temporisation configurable.
* Démarrer en tant que service.
* Sur annulation du message d’arrêt, relancer à nouveau le message via une temporisation configurée.

Les actions à déclencher séquentiellement seront :

* L’Affichage d’un message d’avertissement d’arrêt toujours au premier plan avec le temps restant avant le déclenchement de l’action suivante + un bouton d’annulation.
* La sauvegarde du travail ouvert par nos applications métier SCE, ACE + la fermeture de celles-ci.
* La mise en suspension de la machine virtuelle en enregistrant son état.
* L’arrêt normal de la machine.

Les messages à traiter seront :

* La mise à jour d’une ou plusieurs la temporisations cible.
* Mise à jour de silencieuse de l’application.
* L’activation du message d’avertissement d’arrêt.

Choix du langage :

Langage Objet, Visual Basic, C++.

Structure du programme :

Au moins une class par métier identifié, sans limite :

IHM, Réseau, Action, Gestionnaire principale, Fichier, etc...

Techniques de mesures :

Compteurs de performance.

Techniques de cryptage :

Simple par algorithme ou sous traitée en externe par un exe type scramble.