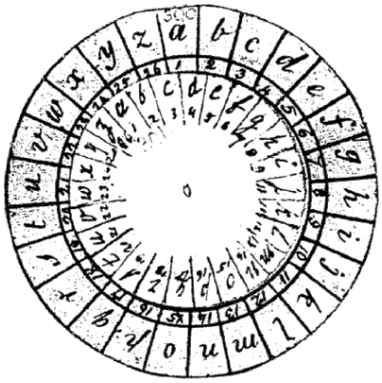
[*A* 4 L]

Microsoft WCF

Java EE

Programmation en C# avancé

PROJET SECUR-WS



CESI exia
ÉCOLE SUPÉRIEURE D'INFORMATIQUE

Barbolini / Sbriglio – eXia Aix en Provence

Crypter [verbe transitif] : Coder un message afin de le protéger et de le rendre incompréhensible à ceux qui ne disposent pas du code.

L'internaute : http://www.linternaute.com/dictionnaire/fr/definition/crypter/

Sommaire

1. Contexte 4
2. Présentation de la mission 4
3. Spécifications fonctionnelles 4
4. Infrastructure technique 5
5. Spécifications détaillés des activités 5
6. Spécifications techniques de la plateforme .Net 6
7. Spécifications techniques de la plateforme Java EE 8
8. Livrables 10
9. Planification générale 10
10. Critères d'évaluation 10

1. Contexte

Une Agence de cyber-sécurité souhaite s'approprier des informations sensibles. Elle désire donc identifier un dangereux Black Hat afin de stopper les agissements de son organisation criminelle.

Des documents récemment dérobés contiennent le nom d'un informateur qui connait potentiellement l'identité du chef de l'organisation criminelle que vous devez démasquer (le Black Hat). Ces documents sont actuellement en possession de l'agence. Le problème est que ces informations sont chiffrées. Des sources fiables ont indiqué à l'agence, que le chiffrement n'est pas asymétrique.

L'agence vous confie la mission '*gen*erator'.

2. Présentation de la mission

–

Vous êtes une entité informatique (de type White Hat) capable de répondre au besoin de l'agence, à savoir, casser les documents chiffrés. Cette première mission, si elle est réussie, engendrera un contrat financièrement fructueux pour votre structure. Une quantité importante d'informations chiffrées vous sera alors confiée dans le but de répondre au même objectif initial, mais en quantité plus importante. Il est nécessaire que votre solution soit une plateforme distribuée, car à terme, d'autres applications viendront bénéficier des services de cette dernière.

Sur ce coup d'essai, vous devez :

- Rentrer en contact avec l'agence (gen.project.exia@gmail.com). Ils vous remettront les documents chiffrés qui contiennent l'email de l'informateur.
- Déchiffrer ces premiers documents pour trouver l'email de l'informateur.
- Rentrer en contact avec l'informateur. Débrouillez-vous pour qu'il vous communique le nom du chef de l'organisme.
- communiquer à l'agence l'identité du chef.

3. Spécifications fonctionnelles

Vous devez concevoir et développer une plateforme distribuée capable :

- SPF1 - d'authentifier l'utilisateur.
- SPF2 – de déchiffrer plusieurs documents de type '.txt' en simultané. Fournir un document déchiffré de type '.txt' par document chiffré source. Fournir un rapport de type '.pdf' présentant le taux de confiance d'un document déchiffré par document chiffré source.

4. Infrastructure technique

Votre équipe se compose d'experts .NET (C# /WCF) et d'experts Java EE.

L'application .NET se charge de déchiffrer les fichiers codés en appliquant sur chaque fichier une série de clés. La plateforme java EE a pour objectif de déterminer si la clé appliquée sur le fichier est correcte (celle qui permet de déchiffrer le fichier). La plateforme .Net envoie donc à la plateforme Java EE un message comportant le texte décodé, le nom de fichier associé et la clé utilisée. La plateforme Java doit s'assurer que l'information est déchiffrée correctement (qu'il s'agit d'un texte en français). Si c'est le cas, elle doit rechercher une adresse email. Si l'email est trouvé, la plateforme Java EE doit fournir à la plateforme .NET les informations suivantes : adresse email, nom du fichier contenant le mail et la clé de déchiffrement.

En résumé, la plateforme .NET se charge du déchiffrement et la plateforme Java EE se charge de valider le déchiffrement et de retrouver l'email de l'informateur.

Votre plateforme distribuée doit remplir au mieux les exigences non fonctionnelles (scalabilité, disponibilité, modularité, sécurité…) liées à ce type de projet.

5. Spécifications détaillés des activités

SPF1	**Authentification utilisateur**
In : login, password, tokenApp	L'utilisateur doit fournir son login et son mot de passe. La platefome.Net se charge de l'authentification. Le client lourd doit fournir son token de sécurité spécifique à l'application. La plateforme ne donne accès à ces services qu'en cas de token valide. En cas de succès, la plateforme fournit un token qui permettra à l'utilisateur de bénéficier des services de la plateforme.
Out : bool, tokenUser	

SPF2	**Déchiffrer**
In : tokenUser, tokenApp, txt files	A l'aide du logiciel client, l'utilisateur indique quels fichiers il souhaite déchiffrer. Le logiciel client communique son token d'application et le token de l'utilisateur. Si la plateforme accepte la requête, elle reçoit les fichiers à déchiffrer. Elle tente de déchiffrer avec un premier code (brute force) les fichiers reçus. Une fois ce travail effectué elle envoie les fichiers à la plateforme Java EE. Elle continue à déchiffrer ces mêmes fichiers avec un autre code. Elle réitère cette opération jusqu'à ce que la plateforme Java EE lui indique qu'elle (la plateforme .Net) a réussi à trouver le bon code. Elle arrête alors de vouloir identifier le code et fournit au client le code, le fichier en clair et le rapport au format txt de taux de confiance du document. Pour savoir si un document a bel est bien été déchiffré par la plateforme .Net, pour chaque 'document' la plateforme Java EE constitue un échantillon de mots présents dans le document. Plus l'échantillon est
Out : bool, file, key, txt file, email	

	grand, plus le taux de confiance du document sera important, mais plus le temps de traitement sera long. Elle compare les échantillons avec un dictionnaire de mots présents en base. Si elle estime que suffisamment de mots présents dans le document sont en correspondance avec son dictionnaire de mot, alors elle conclura que le 'document' a été déchiffré. Elle recherche alors l'information secrète cachée dans le 'document'. Elle enverra alors à la plateforme .net, le triplet {information secrète/nom du document/ code de déchiffrement}. La plateforme .NET informera à son tour le client. Un email d'avertissement sera envoyé à l'utilisateur pour le prévenir de la réussite de l'opération. Cet email doit être envoyé par la plateforme .NET.

6. Spécifications techniques de la plateforme .Net

Ci-après, la liste des spécifications techniques à respecter fournissant ainsi les critères d'acceptations techniques :

- Stx 1 - Architecture : 3 tiers - SOA
- Stx 2 - Framework : Microsoft Framework 4.5
- Stx 3 - Langage : C# 4.5
- Stx 4 - Solution : 9 couches (modèle historique SOA)
- Stx 5 - Client : (Solution technique conseillée)
    - A déployer sur les postes clients.
    - Lourd (Winform/WPF) / asynchrone (Callback).
    - 3 couches MVC (1 composant physique).
    - Parallélisation des traitements issus du middleware (traitements lourds). Client non-bloqué lorsqu'il attend les réponses de la plateforme.
    - WPF possible, donc pas obligatoire.
- Stx 6 - Middleware :
    - A déployer sur un serveur.
    - Architecture de type service.
    - Point d'entrée unique.
    - Structure des messages : STG [bool statut_op – string info – object[] data – string operationname – string tokenApll – string tokenUser]
    - Une seule méthode exposée sur le point d'entrée unique de signature : STG m_service(STG msg)
    - Distribué : oui.
    - Journalisation de l'activité du CAM.
    - Les contrôleurs de workflow et les composants métiers doivent supporter des charges de travail importantes.
    - 5 couches.
    - 1 composant physique pour les couches 5 – 4 – 3.
    - 1 composant physique pour la couche 2.
    - 1 composant physique pour la couche 1.

- Stx 7 - Data : (Solution technique conseillée)
  - SQL Server STD Ed ou ENT Ed.
  - ~~Mode d'authentification mixte.~~
  - ~~Mise en place de l'agent SQL Server dans un processus Windows distinct.~~
  - ~~Mise en place du moteur SQL Server dans un processus Windows distinct.~~
  - ~~Sauvegarde automatique journalière (sauvegarde complète).~~
- Stx 8 – Communication
  - Liaison de type netTcp entre le client lourd et la plateforme.
  - Communication sécurisée avec un chiffrement à 128b.

Classification des composants de la solution .NET (« couches »)

- CAD, le composant d'accès aux données est une primitive de communication et il est le seul composant disposant de capacités à se connecter et fournir un service d'interaction avec le système de gestion de base de données relationnelles.

- EM, les entités de mapping encapsulent les instructions de manipulations des données.

- CM, les composants métiers proposent la mécanique nécessaire à l'acquittement du ou des services.

- CW, les contrôleurs de workflow gèrent et coordonnent l'activité transactionnelle des services.

- CAM, le composant d'accès métier est une primitive de communication, il ressemble quelque peu au Composant d'Accès aux Données. En effet, il fournit un niveau d'abstraction aux utilisateurs de la couche n-1, offrant la possibilité d'un brassage de workflow ultérieur.

- SERVM, les services métiers exposent une série de comptoirs sous forme de thématiques, propres aux ressources que renferme la plate-forme. Ils constituent le point d'entrée de fonctionnalités proposées par la plate-forme.

- SERVC, le serveur de composant rendra accessible les services aux applications. Plusieurs choix de format sont possibles à ce stade et notamment celui des Web Services dans le but de rendre interopérables les plates-formes, les services et les applications.

- CUC, le composant utilisateur de communication est une primitive de communication. Il fournit une propriété d'abstraction à la couche n-1, permettant de centraliser les éléments et donc de communiquer avec la plate-forme.

- CUT, le composant utilisateur de travail est propre aux contraintes de l'application et non pas de la plate-forme, dans un contexte de workflow.

- SERVU, les services utilisateurs s'exécutent en tâches de fond sur les différents hôtes.

- CUP, les composants utilisateurs publiques, tout comme les composants utilisateurs, proposent de s'acquitter d'une ou plusieurs tâches, à la différence près qu'ils coopèrent avec plusieurs applications.

- CU, les composants utilisateurs agissent en tant que pupitres de commande de l'application. Ils proposent l'interfaçage nécessaire à tout dialogue avec l'utilisateur.

7. Spécifications techniques de la plateforme Java EE

Java EE 7 et Payara 4.1.X sont fortement recommandés pour développer et exécuter l'application.

La réception des messages générés par la plateforme .NET (contenu des documents décodés par .NET) et la logique de traitement des messages pour repérer l'email du contact doivent s'exécuter idéalement dans deux domaines différents. (Deux CM différentes)

Ce sera un plus si l'application Java EE est répartie sur des machines distinctes. Les machines distinctes peuvent être des machines virtuelles.

L'intégration du tiers 'réception des messages' et du tiers 'traitement des messages' se fait au moyen de l'API JMS. Un middleware orienté message s'interpose donc entre la logique d'acquisition et la logique de traitement.

Par logique de traitement du message, nous entendons opérations sur le texte décodé par l'application .NET

Vous avez le choix d'exposer la plateforme Java EE à la plateforme client .NET via services web SOAP ou REST

Le traitement (qui peut être long) des messages est asynchrone. Les messages de la destination sont pris en charge par un composant de type Message-driven Bean. Les messages reçus sont envoyés dans une destination JMS.

Le choix du type de message au sens JMS est à votre discrétion : TextMessage, ByteMessage…

Rappelons que tout message JMS est composé d'une enveloppe contenant un header, une section optionnelle de propriétés personnalisables et d'un corps de message qui contient l'information à transférer. La section de propriétés permet de transporter / transférer des informations complémentaires qui peuvent être entre autre de type entier ou chaîne de caractère. Ces propriétés définies par le développeur sont des pairs clé/valeur. Pour plus d'informations : http://docs.oracle.com/javaee/7/api/javax/jms/Message.html

Pour des raisons de scalabilité la plateforme Java EE doit être essentiellement (voire totalement) « stateless ».

Dans un environnement orienté EJB, les API de threading Java SE ne doivent pas être utilisées par les développeurs. La gestion des threads est sous contrôle du container Java EE. Si vous vouliez explicitement manipuler des threads, il faudrait utiliser l'API Java EE de gestion de la concurrence (https://docs.oracle.com/javaee/7/tutorial/concurrency-utilities.htm). Cela n'est pas une exigence car ça n'a pas été vu en prosits. Il est important de rappeler que l'architecture EJB est adaptée aux environnements extrêmement concurrents.

La plateforme valide le déchiffrement en comparant un échantillon de mots /motifs prélevés dans le 'document' décodé avec une bases de mots (français). La base est (éventuellement) de type MySQL. Internet propose un grand nombre de dictionnaires de mots. Vous pouvez choisir le dictionnaire que vous voulez.
Cependant vous pouvez utiliser le fichier texte *liste_francais.txt* fourni en ressource qui contient 22740 mots à raison d'un mot par ligne.
Ce fichier texte provient du site http://www.freelang.com/dictionnaire/dic-francais.php

Voici un exemple de procédure pour alimenter une table simple avec le fichier texte.
On considère la base (le schéma) MySQL **dico** contenant la table **mots(id,mot)** où id est la colonne clé primaire auto-incrémentée et mot une colonne de type varchar(255) :

```
CREATE TABLE `dico`.`mots` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `mot` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`));
```

On suppose aussi que MySQL utilise un encodage Latin1 et que l'option *secure-file-priv* MySQL n'est pas activée. Cette option est accessible depuis MySQL Workbench > INSTANCE > Option File > onglet Security. Il faut redémarrer le serveur pour que les modifications soient prises en compte.
Voici le script pour remplir la table :

```
LOAD DATA INFILE 'liste_francais/liste_francais.txt' INTO TABLE dico.mots CHARACTER SET latin1 lines TERMINATED BY '\r' (mot);
```

Le dossier liste_francais étant dans le répertoire data de MySQL.

Vous pouvez décider d'utiliser ou non JPA pour manipuler les données de la base.
Pour information, voici comment injecter d'une source de données JDBC pour directement « attaquer » une base via l'API JDBC sans passer par JPA :

```
@javax.annotation.Resource(lookup= « jdbc/myDS ») private javax.sql.DataSource ds ;
```

Où jdbc/myDS est le nom JNDI d'une source de données définie dans Payara.

La plateforme Java doit fournir une interface web permettant au minimum :

- d'insérer un nouveau mot dans la base
- de lister des mots contenant un motif donné (par exemple lister les mots contenant « Aban »).

L'exigence suivante est secondaire : L'application Java EE doit pouvoir stocker en base le nom du fichier décodé, les premiers mots en clairs contenus dans ce fichier, la clé utilisée pour le décodage.
L'interface web devra permettre d'accéder à ces informations.

Enfin, si le mail du terroriste a été trouvé dans le ' document', la plateforme Java EE invoquera un service WCF pour communiquer les informations à la plateforme .NET.

8. Livrables

Le projet est à réaliser en groupe de 3 ou 4 exars.

- Power point de présentation de soutenance.
- Rapport projet entièrement sur GitHub documenté en Markdown :
  - Introduction.
  - Modélisation UML commentée de l'application distribuée
  - Analyse technique.
  - Analyse des écarts.
  - Bilan.
- L'URL du GitHub devra être envoyée à rbello@cesi.fr dès le 22/06/17
  - L'ensemble du code source et des livrables devra être « mis en configuration »
  - L'utilisation de Docker est tout à fait possible
  - Pensez à renommer le projet
- La plateforme distribuée 'GEN' basée sur les technologies .NET WCF et java EE
- Le client 'GEN'.
- Les fichiers déchiffrés et leurs rapports de probabilité.
- Le nom de l'informateur. → PAS SUR GITHUB
- Le fichier secret. → PAS SUR GITHUB

9. Planification générale

Mercredi 21 juin 2017	Lancement du projet
Lundi 26 juin après-midi	Prise de contact avec l'agence
~~Jeudi 29 juin 2017~~	~~Remise du rapport pdf au Jury~~
vendredi 30 juin 2017	Soutenance de projet

Ne confondez pas vitesse et précipitation. Ne vous lancez pas à la va-vite dans la construction d'un petit logiciel de décryptage qui ne sera pas (ou très difficilement) intégrable à l'architecture finale.

10. Critères d'évaluation

Voir fichier Excel.