Essbase : chiffrement des paramètres de connexion des scripts MaxL

par Sébastien Roux (sroux.developpez.com)

Date de publication : 18 avril 2009

Dernière mise à jour :

Cet article décrit les options de chiffrement proposées par la ligne de commande MaxL à partir de la version 9.3.1 d'Essbase. Ces options permettent de masquer les paramètres de connexion aussi bien dans le script que dans son compte rendu d'exécution en apportant ainsi une bien meilleure sécurité.

I - Introduction

Cet article aborde les différentes options de chiffrement apportées par MaxL, il vient enrichir l'article **MaxL et ESSCMD, les langages d'administration d'Essbase** réalisé par mon confrère **Antoun**.
Un des reproche qui peut être fait à MaxL concerne son manque de sécurité. Il arrive souvent que les paramètres de connexion soient stockés en dur dans le script ; ceux-ci comprennent le compte utilisateur, s'agissant souvent d'un compte administrateur, ainsi que le mot de passe associé. Par ailleurs le compte rendu d'exécution, en interactif, dans le shell, ou redirigé vers un fichier (cf commande **spool**), restitue toutes ces informations en clair.

Exemple de compte rendu d'un script MaxL

```
 Essbase MaxL Shell - Release 11.1.1 (ESB11.1.1.2.0B110)
 Copyright (c) 2000, 2008, Oracle and/or its affiliates.
 All rights reserved.

MAXL> login 'admin' 'password' on 'sr-dev';

 OK/INFO - 1051034 - Logging in user [admin].
 OK/INFO - 1051035 - Last login on Sunday, April 19, 2009 3:19:26 PM.
 ...
 OK/INFO - 1007067 - Total Restructure Elapsed Time : [1.062] seconds.
 OK/INFO - 1013273 - Database Demo.Basic altered.

MAXL> alter application 'Demo' unload database 'Basic';

 OK/INFO - 1056013 - Application Demo altered.

MAXL> logout;

      User admin is logged out


  MaxL Shell completed
```

A partir de la version 9.3.1 d'Essbase, MaxL propose des fonctionnalités de chiffrement des paramètres de connexion, nom d'utilisateur et mot de passe. Le principe reste relativement simple, deux clés sont générées, la première servant au chiffrement du script MaxL existant, la seconde permettant son exécution sécurisée.

Tout au long de cet article nous aborderons la démarche de chiffrement/déchiffrement qui se résume aux étapes suivantes :

1 génération des clés,
2 production des versions chiffrées (.mxls) des scripts existants (.mxl),
3 exécution des scripts chiffrés à l'aide de la clé de déchiffrage.

II - Génération des clés

II-A - Rappel du principe de chiffrement par clé publique et clé privée

Je vous invite à consulter l'article de **ram-0000** intitulé : **Introduction à la cryptographie**. Le **chapitre 8** aborde les algorithmes asymétriques employés par MaxL. Nous y apprenons que l'algorithme asymétrique impose l'utilisation de deux clés, une pour le chiffrement et l'autre pour le déchiffrement. On parle alors de clé publique et de clé privée.

II-B - Cas pratique appliqué à MaxL

L'argument gk (*generate keys*) permet de générer deux clés, une publique et une privée, destinées, dans l'ordre, au chiffrement ou au déchiffrement des paramètres de connexion (nom d'utilisateur et mot de passe) du script MaxL.

Syntaxe de génération des clés publique et privée
```
essmsh -gk
```

Compte rendu de la génération des clés
```
Essbase MaxL Shell - Release 11.1.1 (ESB11.1.1.2.0B110)
Copyright (c) 2000, 2008, Oracle and/or its affiliates.
All rights reserved.


Public Key for Encryption: 31457,2452005869
Private Key for Decryption: 1886885153,2452005869

 MaxL Shell completed
```

Dans cet exemple, la chaîne numérique *31457,2452005869* désigne la clé publique (*Public Key for Encryption*). La chaîne 1886885153,2452005869 désigne la clé privée (*Private Key for Encryption*).

III - Chiffrement d'un script MaxL

III-A - Support d'exercice

Nous travaillerons à partir de l'exemple suivant :

Exemple de script Maxl
```
/*
 * Script MaxL (essmsh)
 * Nom : test_encrypt.mxl
 * Description : retructuration dense de la database
 */

/* Connexion */
login 'admin' 'password' on 'sr-dev';

/* Démarrage base */
alter application 'Demo' load database 'Basic';
/* Restructuration */
alter database 'Demo'.'Basic' force restructure;
/* Arrêt database */
alter application 'Demo' unload database 'Basic';

/* Deconnexion */
logout;
```

Ce script complet permet d'effectuer une restructuration dense de la base *Basic* de l'application *Demo* après son démarrage. La base est ensuite arrêtée.
La partie qui nous intéresse ici concerne la chaîne de connexion *login*. En effet, dans MaxL, toute exécution de commandes visant le serveur, les applications ou les bases requiert au préalable une identification grâce à la commande *login*. Cette commande prend en arguments le nom de l'utilisateur, le mot de passe ainsi que le serveur concerné.

Les paramètres de connexion, nom d'utilisateur et mot de passe, sont en clair dans le script, c'est à dire qu'il ne sont pas cryptés et donc accessibles et "réutilisables" à souhait par tout utilisateur, mal intentionné ou pas, accédant au code source. Il est cependant possible de passer les chaînes de connexion au script MaxL pour plus de sécurité, cependant le compte-rendu les restituera en clair comme évoqué plus haut.

III-B - Chiffrement du script à partir de la clé publique

Une fois la clé publique générée, le script MaxL peut-être chiffré.

Chiffrement d'un script MaxL (test_encrypt.mxl)
```
essmsh -E nom_du_script clé_publique

essmsh -E test_encrypt.mxl 31457,2452005869
```

L'argument E (*encrypt*) chiffre le script passé en paramètre ainsi que les scripts imbriqués (scripts appelés à partir du script principal).
L'argument Em (*manual encrypt*) ne chiffre en revanche que le script passé en paramètre en laissant le choix de chiffrer manuellement les scripts imbriqués.
Lors du chiffrement, l'extension du script est renommée en *mxls* (*secured*).

Script MaxL chiffré (test_encrypt.mxls)
```
/*
 * Script MaxL (essmsh.exe)
 * Nom script: test_encrypt.mxl
 * Description : retructuration dense de la database
 */

/* Connexion */
login $key 98163042910494032910488356101 1 $key 533312566168686488815523440980877053878 1 on 'sr-dev';

/* Démarrage base */
alter application 'Demo' load database 'Basic';
/* Restructuration */
alter database 'Demo'.'Basic' force restructure;
/* Arrêt database */
alter application 'Demo' unload database 'Basic';

/* Deconnexion */
logout;
```

Nous pouvons constater que seuls les arguments de la commande de connexion sont chiffrés.

Chaînes de connexion chiffrées
```
/* Connexion */
login $key 98163042910494032910488356101 1 $key 533312566168686488815523440980877053878 1 on 'sr-dev';
```

Pour information, l'absence de la commande *login* dans le script n'est pas bloquante pour le bon déroulement de l'opération de chiffrement.

IV - Chiffrement d'une chaîne de caractères

MaxL propose également une option pour chiffrer une chaîne de caractère au choix.

Chiffrement d'une chaîne de caractères
```
essmsh -ep ma_chaîne clé_publique

essmsh -ep admin 31457,2452005869
```

Compte-rendu d'exécution du chiffrement d'une chaîne de caractères
```
 Essbase MaxL Shell - Release 11.1.1 (ESB11.1.1.2.0B110)
 Copyright (c) 2000, 2008, Oracle and/or its affiliates.
 All rights reserved.

 Encrypted Data: 98163042910494032910488356101

  MaxL Shell completed
```

L'argument ep (*encrypt input data*) permet de chiffrer une chaîne de caractères. La chaîne cryptée peut-être récupérée à partir des commandes *cut*, *grep*, *awk* etc. accessibles à partir de la ligne de commande Windows (cf modules gnu), les shells Unix/Linux (shell, korn shell etc.) ou les langages de scripts (Perl, VBScript etc.)

V - Exécution d'un script MaxL chiffré

Le script chiffré est déchiffré puis exécuté grâce à la clé privée préalablement générée.

Déchiffrement et exécution d'un script MaxL chiffré (test_encrypt.mxls)

```
essmsh.exe -D nom_du_script clé_privée

essmsh -D test_encrypt.mxls 1886885153,2452005869
```

L'argument D (*decrypt*) permet de déchiffrer les paramètres chiffrés puis d'exécuter le script.

Compte-rendu d'exécution du script MaxL chiffré

```
MAXL> login $key 9816304291049403291048835610l1 $key 5333125661686864888l5523440
9808770538781 on 'sr-dev';

 OK/INFO - 1051034 - Logging in user [admin].
 OK/INFO - 1051035 - Last login on Saturday, April 18, 2009 11:19:27 AM.
 OK/INFO - 1241001 - Logged in to Essbase.

MAXL> alter application 'Demo' load database 'Basic';

 OK/INFO - 1056013 - Application Demo altered.

MAXL> alter database 'Demo'.'Basic' force restructure;

 OK/INFO - 1019017 - Reading Parameters For Database [Drxxxxxx].
 OK/INFO - 1019013 - Writing Outline For Database [Basic].
 ...
 OK/INFO - 1007067 - Total Restructure Elapsed Time : [0.25] seconds.
 OK/INFO - 1013273 - Database Demo.Basic altered.

MAXL> alter application 'Demo' unload database 'Basic';

 OK/INFO - 1056013 - Application Demo altered.

MAXL> logout;

      User admin is logged out
```

Comme nous pouvons le constater dans le compte-rendu d'exécution, les paramètres de connexion sont bel et bien chiffrés. Cela permet d'éviter de retoucher la sortie pour en supprimer les chaînes de connexion.

VI - Conclusion

Cet article était l'occasion de présenter en détail les différentes fonctionnalités de chiffrement proposées par la ligne de commande MaxL. Je remercie **Antoun** pour sa relecture et ses conseils toujours aussi avisés.