Automatisierte Oracle Passwort-Verwaltung mit KeePass

Martin Bracher
Trivadis AG
Zürich

Schlüsselworte
KeePass, Oracle, Passwort, Security

Einleitung

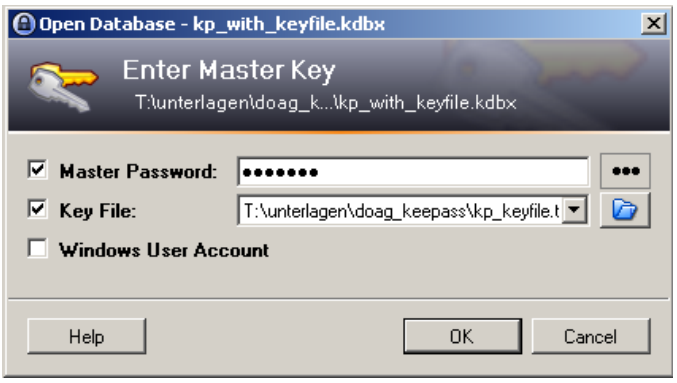
KeePass ist in der Windows (und Linux) Desktop Welt ein weit verbreiteter grafischer Passwort-Manager, welcher die Passwörter in einer verschlüsselten Datei (*.kdbx) ablegt. Was weniger bekannt ist, diese KeePass Dateien können auch auf Server-Seite mit entsprechenden Perl-Modulen bearbeitet werden. Dies können wir somit auch gut für die Verwaltung von Oracle Passwörtern auf dem Server nutzen.

Wenn wir eine Datenbank automatisiert per Script erstellen, können wir ein Random-Passwort setzen und dieses gleich im KeePass File eintragen lassen, so dass der DBA dieses auch verfügbar hat. Es ist nicht mehr notwendig, dieses in einem Klartext-File auf dem Server herumliegen zu lassen.

Bei Entwicklungsdatenbanken, wo verschiedene Entwickler arbeiten, kann es schon mal passieren, dass einer das Passwort ändert, ohne den anderen das mitzuteilen. Ich zeige nun eine Möglichkeit, wie man via Passwort-Verifizierungsfunktion (welche von Oracle bei der Passwort-Änderung aufgerufen wird) das neue Passwort gleich in die KeePass Datei schreiben kann. Somit kennen einerseits alle Entwickler wieder das neue Passwort, andererseits kann man nach dem neu Klonen der Entwicklungsdatenbank aus der Produktion die Passwörter wieder auf den bisherigen Entwicklungsstand zurücksetzen.

Das Passwort der Passwort-Verwaltung...

KeePass speichert die Passwörter in einer Datei mit der Endung .kdbx ab. Die Datei selbst wird üblicherweise durch ein Master Password gesichert. Alternativ kann man auch ein Key-File angeben. Dies ist eine Datei, aus welcher KeePass ein Passwort errechnet. Gut geeignet sind einmalige Dateien, die niemand anderes besitzt, beispielsweise Fotos, oder eine Datei, die mit Zufallswerten gefüllt ist.



Dies eröffnet die Möglichkeit, die Keepass-Datei selbst auf einem zentralen Share abzulegen, wo viele, auch nicht Berechtigte Zugriff haben. Das Key-File verteilt man dann nur lokal auf die Systeme oder an User, die Zugriff brauchen.

Es lässt sich auch Master-Passwort und Key-File kombinieren. Auf einem Server, wo die Verwaltung automatisiert erfolgen soll, muss in irgend einer Form das Passwort und das Key-File im Zugriff liegen. Es besteht natürlich ein gewisses Restrisiko, dass sich jemand darauf Zugang verschafft. Deshalb sind die Dateien nur für die berechtigten User (dba-Gruppe) lesbar zu machen. Als weitere Hürde kann man das Passwort noch verschleiern, damit es nicht direkt im Klartext sichtbar ist:

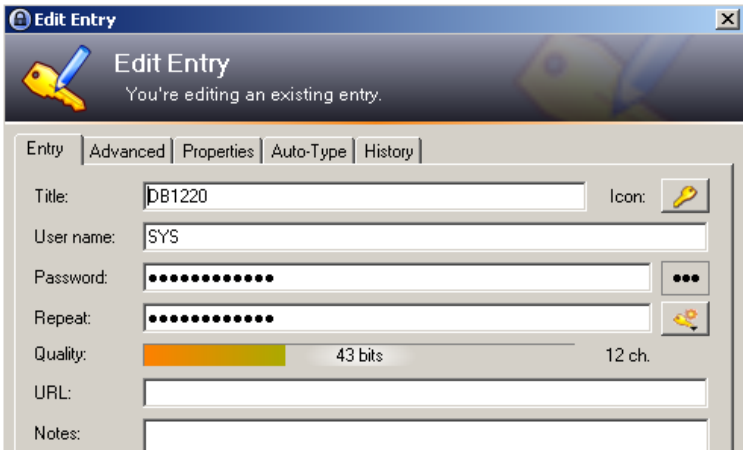
```
{TWOFISH}Q3VzdG9tU2FsdF9fRXPv4fFW2VBUVkRFdWdlblNhbBRlZF9f3Tle9ULZxJErmmnYn2I2GhTTHQP+s3rk
```

Bild mit nfs-Share

Struktur der KeePass Einträge

Jeder Eintrag muss einer Gruppe zugeordnet werden. Wenn die KeePass Datei auch für andere User wie Oracle verwendet wird, bietet es sich an, die Oracle-User unter der Gruppe „ORACLE“ zu speichern. Es kann auch eine Unterteilung nach Verwendung erfolgen, so dass man Gruppen die Gruppen DEV, TEST, INT, PROD erstellt.

Die Datenbankbezeichnung wird in das „Title“ Feld eingefügt. Als Bezeichnung wird üblicherweise der Tnsnames-Alias, db_name, db_unique_name oder der Service-Name verwendet.



Kompilierung der Perl-Module

Bei Verwendung des Perls der Linux-Distribution gibt es möglicherweise bereits bestehende Pakete, die nur noch installiert werden müssen, beispielsweise bei RedHat/Oracle Enterprise Linux aus dem EPEL-Repository
https://centos.pkgs.org/7/epel-x86_64/perl-File-KeePass-2.03-10.el7.noarch.rpm.html

Wenn man ein anderes Perl, z.B. das Perl von Oracle verwenden will, muss man in der Regel folgende zusätzliche Module installieren, welche bei cpan.org heruntergeladen werden können:

- Crypt-Rijndael-1.13
- Digest-SHA-5.95

- XML-Parser-2.44
- File-KeePass-2.03

Die Installation der Module erfolgt immer nach dem gleichen Schema

```
export PERL=/<path_to_the_desired_perl>/bin/perl
tar zxvf Crypt-Rijndael-1.13.tar.gz
cd Crypt-Rijndael-1.13
$PERL Makefile.PL
make
make test
make install
```

Wer die BasEnv Umgebung unserer Trivadis Toolbox verwendet, hat in den gelieferten Perl-Versionen für die meisten Plattformen die Module bereits integriert.

Verwendung der Perl-Module

Erstellen eines neuen KeePass Files

```
  use File::KeePass;
  my $kp;
  $kp=File::KeePass->new;
  $kp->add_group({ title => "DOAG" });
 $kp->save_db("doag.kdbx", "manager" );
```

Falls man Passwort plus Key-File will, lautet die Syntax wie Folgt:

```
$kp->save_db("doag.kdbx", [ "manager", "/path/key-file.jpg" ] ) ;
```

User SYS der Datenbank DB1220 mit Passwort „manager“ hinzufügen:

```
  use File::KeePass;
  my $kp;
  $kp=File::KeePass->new;
  $kp->load_db("doag.kdbx", "manager" );
  $kp->unlock;
  my $group = $kp->find_group({title => "DOAG" });
  my $gid = $group->{'id'};
  my $entry= $kp->add_entry({ group => "$gid", title => "DB1220",
                              username => "SYS", password => "manager" });
  $kp->save_db("doag.kdbx", "manager" );
```

Das Passwort von SYS@DB1220 aus der KeePass Datei auslesen

```
  use File::KeePass;
  my $kp;
  $kp=File::KeePass->new;
  $kp->load_db("doag.kdbx", "manager" );
  $kp->unlock;
  my $group = $kp->find_group({title => "DOAG" });
  my $gid = $group->{'id'};
  my $entry=$kp->find_entry({ group_id => "$gid", title => "DB1220",
                              username => "SYS" });
  if ( $entry ) { print "Passwd of SYS@DB1220 is: $entry->{password} \n" };
```

Wer sich die Arbeit einer eigenen Implementierung sparen will, der findet in unserer Software zum Setzen von Oracle-Umgebungen, dem BasEnv, entsprechende Tools. Beispielsweise das setpw.pl und getpw.pl Script. Für eigene Scripts steht eine Library mit entsprechenden Funktionen zur Verfügung.

Automatische Aktualisierung von KeePass via Datenbank

Eine Passwort-Verwaltung ist nur solange von Nutzen, wie die Passwörter aktuell sind. Aber wie bringt man die Benutzer dazu, die Einträge in KeePass zu aktualisieren?

Am Besten gar nicht... ☺
Man lässt dies von der Datenbank selbst machen.

Jedem Benutzeraccount in Oracle ist ein Profil zugeordnet. Bei diesem Profil kann man gewisse Einschränkungen und Prüfungen setzen, wie etwa die Gültigkeitsdauer eines Passworts und die Anzahl fehlerhafter Logins. Um bestimmte Passwortregeln zu erzwingen, kann man eine Passwort-Verifizierungsfunktion angeben.

```
alter profile default limit password_verify_function my_verify_function;
```

Dies ist ein PL/SQL Programm, welches wir frei gestalten können; in der Regel prüfen wir das eingegebene Passwort auf die Einhaltung der Regeln. Hier setzen wir aber auch mit der Schnittstelle zum KeePass an. In der Präsentation werde ich zeigen, wie man mit Hilfe des dbms_scheduler auf OS-Seite ein Programm aufruft, das die Passwörter im KeePass aktualisiert.

```
dbms_scheduler.create_job(job_name=>jobname, job_type=>'executable',
job_action=>'/pfad/pw2keepass', number_of_arguments=>4, enabled=>FALSE);
…
dbms_scheduler.set_job_argument_value(jobname,3,password);
dbms_scheduler.enable(jobname);
```

Eine Anwendungmöglicheit dieser aktuellen KeePass Datei ist das Duplizieren. Oft werden Produktionsdatenbanken zu Test-/Entwicklungsdatenbanken geklont. Auf dem Duplikat will man meist nicht die Passwörter der Produktion haben, sondern die Passwörter des überschriebenen Klons. Man kann nun ein kleines Programm erstellen, das einen Loop über alle User und dblinks macht, das bisherige Passwort aus KeePass ausliest und dieses wieder in der Datenbank neu setzt.

Viel Erfolg beim Einsatz von Trivadis-Know-how wünscht Ihnen

Kontaktadresse:
Martin Bracher
Trivadis AG
Sägereistrasse 29
CH-8152 Glattbrugg

Telefon:	+41 58 459 56 56
Fax:	+41 58 459 56 66
E-Mail	martin. bracher@trivadis.com
Internet:	www.trivadis.com