Tipps & Tricks: Juni 2004

Bereich:	DBA, Security	Erstellung:	06/2004
Versionsinfo:	10.2, 11.1, 11.2	Letzte Überarbeitung:	06/2009 MP

Verschlüsselung mit dbms_crypto

Bisher gab es in Oracle bereits ein Package zum Verschlüsseln von Daten oder Applikations-Passwörtern mit Namen dbms_obfusication_toolkit.

Leider hatte dieses Package einige Nachteile; so musste z. B. der zu verschlüsselnde String ein Vielfaches von 8 Byte betragen. Da dies in der Praxis selten vorkommt, kam nur eine Erweiterung des Strings mit Füll-Bytes in Betracht.

Oracle hat nun in der Version 10g ein neues Package herausgebracht, das diese Limitierung nicht mehr aufweist: dbms_crypto.

1. Hinweise

- Folgende Verschlüsselungsroutinen können verwendet werden:
  - dbms_crypto encrypt_des
  - dbms_crypto encrypt_3des_2key
  - dbms_crypto encrypt_3des
  - dbms_crypto.des_cbc_pkcs5
  - dbms_crypto.des3_cbc_pkcs5
  - dbms_crypto.encrypt_aes128
  - dbms_crypto.encrypt_aes192
  - dbms_crypto.encrypt_aes256
  - dbms_crypto.encrypt_rc4
- Die Schlüssellänge muss beim Algorithmus dbms_crypto.des_cbc_pkcs5 mindestens 8 Byte, beim dbms_crypto.des3_cbc_pkcs5 mindestens 24 Byte betragen.
- Weitere Verschlüsselungsmethoden können im folgenden Handbuch nachgelesen werden:
  PL/SQL Packages and Types Reference 11g Release 1 (11.1)
  Part Number B28419-01

2. Beispiel-Skript als anonymer Block

```
SET FEEDBACK OFF TERMOUT OFF VERIFY OFF ECHO OFF SERVEROUTPUT ON
DECLARE
   p_text VARCHAR2(4000) := 'Hallo Welt';
   p_text_raw RAW(128) := utl_raw.cast_to_raw(p_text);
   p_key RAW(128) := utl_raw.cast_to_raw('Key12345');
   p_crypto_typ NUMBER := dbms_crypto.des_cbc_pkcs5;
   p_encrypted_raw RAW(2048);
   p_decrypted_raw RAW(2048);
BEGIN
   DBMS_OUTPUT.put_line('Originaltext : ' || p_text);
```

```
   p_encrypted_raw := dbms_crypto.encrypt(
          src=> p_text_raw,
          typ => p_crypto_typ,
          key => p_key);
   DBMS_OUTPUT.put_line('Verschlüsselt: ' || rawtohex
                           (utl_raw.cast_to_raw(p_encrypted_raw)));
   p_decrypted_raw := dbms_crypto.decrypt(
          src=> p_encrypted_raw,
          typ => p_crypto_typ,
             key => p_key);
   DBMS_OUTPUT.put_line('Entschlüsselt: ' ||
                            utl_raw.cast_to_varchar2(p_decrypted_raw));
END;
/
```

Ausgabe:

```
Originaltext : Hallo Welt
Verschlüsselt: 333343454332374536423042323233 38
               343137363131383232313932383242 39
Entschlüsselt: Hallo Welt
```

3. Beispiel-Skript als Funktion:

```
CREATE OR REPLACE FUNCTION scott.crypt (
       text IN VARCHAR2,
       key IN VARCHAR2 DEFAULT 'Muniqsoft_Key',
       cryptmode IN VARCHAR2 DEFAULT 'E')
RETURN VARCHAR2
IS
       p_text_raw RAW(32760) := utl_raw.cast_to_raw(text);
       p_key_raw RAW(2048) := utl_raw.cast_to_raw(key);
BEGIN
     IF substr(upper(cryptmode),1,1)='E' THEN
          RETURN (utl_raw.cast_to_varchar2(dbms_crypto.encrypt(
             src => utl_raw.cast_to_raw(text),
             typ => dbms_crypto.des_cbc_pkcs5,
             key => p_key_raw)));
      ELSE
             RETURN (utl_raw.cast_to_varchar2(dbms_crypto.decrypt(
             src => p_text_raw,
             typ => dbms_crypto.des_cbc_pkcs5,
             key => p_key_raw)));
       END IF;
END;
/
show errors

COL crypt_string new_value  crypt_string
SELECT scott.crypt('Hallo Welt','Key12345','E') as crypt_string
```

```
FROM dual;
SELECT scott.crypt('&&crypt_string','Key12345','D')
FROM dual;
```

oder

```
SELECT scott.crypt(scott.crypt('Hallo Welt','Key12345','E'),'Key12345','D') as
crypt_string
FROM dual;
```

4. Einwegverschlüsselung mittels Hash Funktion

In vielen Distributionen ist es inzwischen üblich, die Dateien zum Download mit einer Prüffunktion zu versehen. Damit kann eine nachträgliche Manipulation leicht festgestellt werden.

```
SELECT dbms_crypto.hash(
              utl_raw.cast_to_raw('Mein Prüfsummencode'),2)
              FROM dual;
```

Ausgabe:

```
5C596ACEAFDE81488A8D024CF54D7799
```

Weitere Informationen zu diesem und anderen Themen erhalten Sie in unserem Packages oder Security-Kurs.