Wie sicher sind Database Links?

Dani Schnider
Trivadis AG
Zürich/Glattbrugg, Schweiz

Schlüsselworte:

Data Warehouse, ETL, Database Links, Security, Datenschutz

Einleitung

In vielen Data Warehouses werden die Quelldaten über Database Links in die Staging Area geladen. Dabei wird häufig die Frage nach der Sicherheit dieser Zugriffsmöglichkeit gestellt. Ist es nicht gefährlich, wenn ein ETL-Prozess via Database Link direkt auf das Quellsystem zugreifen kann? Wie kann sichergestellt werden, dass keine unberechtigten Zugriffe auf sensitive Daten erfolgen? Basierend auf konkreten Problemstellungen aus verschiedenen Kundenprojekten werden verschiedene Konfigurationsmöglichkeiten von Database Links mit ihren Vor- und Nachteilen bezüglich Datensicherheit aufgezeigt.

Data Warehouse mit Database Link auf Quellsystem

Eine häufige anzutreffende Konfiguration in Data Warehouses sieht folgendermaßen aus: Auf der DWH-Datenbank wird ein Database Link angelegt, der auf das Applikationsschema der Quellsystems zeigt. Im nachfolgenden Beispiel besteht das „Quellsystem“ aus dem Schema SCOTT auf der Datenbank SOURCE_SYSTEM. Damit alle relevanten User auf der DWH-Datenbank den Database Link verwenden können, wird folgender Public Database Link angelegt:

```
CREATE PUBLIC DATABASE LINK scott_dbl
CONNECT TO scott IDENTIFIED BY tiger
USING 'SOURCE_SYSTEM'
```

Diese Konfiguration funktioniert einwandfrei und erlaubt es den ETL-Prozessen, über den Database Links SCOTT_DBL auf die Quelltabellen zuzugreifen und die Daten in die Staging Area des Data Warehouses zu laden. Alles andere als einwandfrei ist die hier aufgezeigte Konfiguration in Bezug auf Datenschutz. Sie weist nämlich verschiedene Sicherheitslücken auf, die nachfolgend aufgezeigt werden. Um diese Sicherheitslücken zu beheben, werden verschiedene Maßnahmen vorgestellt. Schließlich wird eine sichere Konfiguration einer DWH-Umgebung gezeigt, in welcher die erwähnten Sicherheitslücken geschlossen werden und somit ein sicherer Zugriff auf die Daten des Quellsystems gewährleistet werden kann.

Sicherheitslücke 1: Database Link auf Schema Owner

Die erste Sicherheitslücke kann schon fast als „Todsünde“ im Zusammenhang mit Sicherheit bezeichnet werden. Der Database Link verweist direkt auf den Schema Owner SCOTT, also das Applikationsschema, unter dem die Tabellen gespeichert sind. Dies hat zur Folge, dass jeder Benutzer, der über den Database Link auf das Quellsystem zugreift, die Privilegien des Schema Owners besitzt! Das bedeutet, dass nicht nur die für das Data Warehouse relevanten Daten gelesen werden können,

sondern unter Umständen auch sensible Daten, die gar nicht ins DWH geladen werden sollen und dürfen. Doch es kommt noch schlimmer: Die Tabelleninhalte können über den Database Link auch verändert werden! Das tönt verlockend und ist auch ganz einfach und praktisch – zumindest für Betrüger. Wenn also ein schlecht bezahlter DWH-Entwickler plötzlich mit dem Porsche zur Arbeit fährt und sich eine teure Villa kauft, könnte der Verdacht aufkommen, dass er via Database Link ein paar „Korrekturen" in der Gehaltstabelle des HR-Systems vorgenommen hat.

Immerhin ist es nicht möglich, DDL-Befehle über einen Database Link auszuführen. Obwohl dies aus Sicht mancher DWH-Entwickler recht praktisch wäre. Wer hat sich nicht schon gewünscht, ein paar „Vereinfachungen" im Datenmodell eines Quellsystems vorzunehmen, unnötige Tabellen zu löschen oder zusätzliche Indizes anzulegen. Natürlich ist dies nicht erlaubt und glücklicherweise über einen Database Link auch nicht möglich. Wir werden aber später sehen, dass es möglich ist, über ein paar Umwege das Passwort eines Database Links zu ermitteln. Und wenn es sich dabei um das Passwort des Applikationsschemas handelt, sind den Missbrauchsmöglichkeiten kaum noch Grenzen gesetzt.

Deshalb ist es unbedingt zu vermeiden, einen Database Link zu erstellen, der direkt auf einen Schema Owner oder einen hochprivilegierten User zeigt. Stattdessen wird empfohlen, auf dem Quellsystem einen spezifischen User zu erstellen, über den via Database Link zugegriffen werden kann. Dieser User hat nur die notwendigsten Berechtigungen, die für die Extraktion der relevanten Daten notwendig sind. Zusätzliche Systemprivilegien sowie Zugriffsrechte auf Tabellen, die nicht zwingend benötigt werden, sind zu vermeiden.

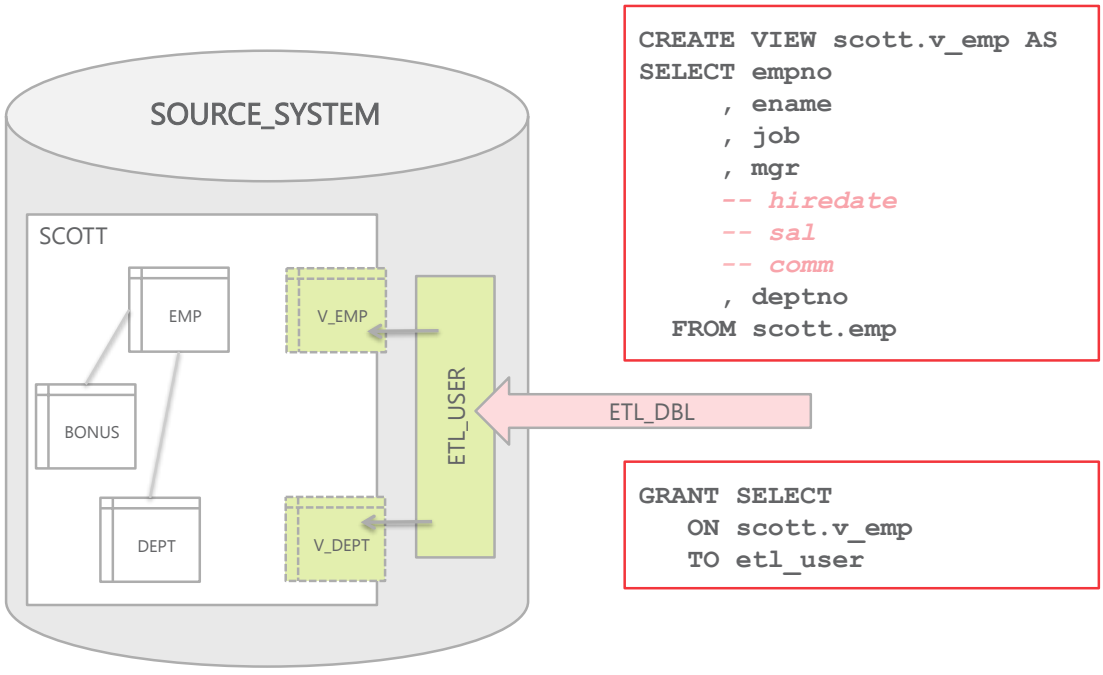

Abb. 1: Zugriff auf Quellsystem über spezifischen User und View Layer

Im DWH-Umfeld eignet sich dazu ein spezieller User auf dem Quellsystem – nennen wir ihn für unser Beispiel ETL_USER. Dieser User hat SELECT-Privilegien auf diejenigen Daten, die ins Data Warehouse geladen werden sollen. Weitere Berechtigungen, insbesondere INSERT-, UPDATE- und DELETE-Privilegien sind nicht erlaubt. Zusätzliche Sicherheit kann implementiert werden, indem der User nicht direkt auf die Quelltabellen zugreift, sondern über einen View Layer, in welchem nur die erforderlichen Attribute selektiert werden können. Abbildung 1 zeigt anhand der Tabelle EMP im

Schema SCOTT, wie sensitive Daten (Attribute HIREDATE, SAL und COMM) ausgeblendet werden können. Das geht natürlich nur, wenn diese Informationen nicht ins Data Warehouse geladen werden sollen. Für DWH-relevante Informationen benötigt der User ETL_USER die entsprechenden Leseberechtigungen – aber alle anderen User nicht. Um dieses Thema geht es bei der nächsten Sicherheitslücke.

Sicherheitslücke 2: Public Database Link auf fixen User

Laut Oracle-Dokumentation[1] ist ein Private Database Link „more secure than a public or global link". Was ist damit gemeint?

Ein Private Database Link ist nur für den User sichtbar, unter dem er erstellt wurde. Im Gegensatz dazu ist ein Public Database Link für alle Benutzer der Datenbank sichtbar. Das gleiche gilt für einen Global Database Link, der über einen entsprechenden Eintrag im Oracle Names Server erstellt werden kann.

Public Database Links müssen nicht in jedem Fall ein Sicherheitsproblem sein. In unserer Beispielkonfiguration wurde jedoch ein Public Database Link auf einen fixen User (der erst noch der Schema Owner ist) erstellt. Dies erlaubt es jedem Benutzer, der Zugriff auf die DWH-Datenbank hat, über diesen Database Link auf die Tabellen von User SCOTT zuzugreifen. Es genügt also, irgendein Passwort auf der DWH-Datenbank zu kennen, um sich Zugriff auf das Quellsystem zu verschaffen. Im Falle eines Private Database Links ist ein solcher Zugriff nur möglich, wenn das Passwort des Users bekannt ist, unter welchem der Database Link erstellt wurde.

Wird also im Data Warehouse unter dem User DWH_STAGE (Schema Owner der Staging Area) folgender Database Link erstellt, so kann nur jemand über den Database Link zugreifen, der das Passwort des Users DWH_STAGE kennt.

```
CREATE DATABASE LINK etl_dbl
CONNECT TO etl_user IDENTIFIED BY etlpw
USING 'SOURCE_SYSTEM'
```

Anders sieht die Situation aus, wenn ein sogenannter „Connected User Link" erstellt wird. Für diese Art von Database Links wird nicht ein spezifischer User und ein Passwort hinterlegt, sondern der Zugriff auf das Quellsystem erfolgt mit dem gleichen Usernamen und Passwort wie auf der DWH-Datenbank. Selbst wenn eine solche Verbindung als Public Database Link angelegt wird, haben nur jene Benutzer Zugriff, die einen User (mit dem gleichen Passwort!) auf der referenzierten Datenbank haben.

```
CREATE DATABASE LINK etl_dbl
USING 'SOURCE_SYSTEM'
```

Eine mögliche Konfiguration besteht zum Beispiel darin, auf der DWH-Datenbank sowie auf jedem Quellsystem einen spezifischen Benutzer ETL_USER anzulegen und dann über Database Links (public oder private) via Connected User auf die Quellsysteme zuzugreifen. Voraussetzung dafür ist, dass das Passwort für ETL_USER auf allen Datenbanken identisch ist.

[1] Siehe Oracle Database Adminstrator's Guide 11g Release 2, Kapitel 30

Damit die Ladeprozesse in die Stage-Tabellen vom Benutzer ETL_USER ausgeführt werden können, muss dieser natürlich entsprechende Schreibrechte für die Staging Area besitzen, d.h. INSERT-Privilegien für alle Tabellen des Schemas DWH_STAGE. Da das Passwort für den Benutzer ETL_USER auf allen Datenbanken gleich sein muss, kann somit jemand, der entsprechenden Zugriff auf ein Quellsystem hat, über einen Connected User Link Daten in die Stage-Tabellen einfügen. Das ist in der Regel nicht weiter tragisch, aber sicher nicht wünschenswert.

Es gibt noch ein weiteres Problem, wenn das Laden der Staging Area nicht direkt in DWH_STAGE, sondern über einen weiteren User ETL_USER erfolgt. Vor dem Laden der Stage-Tabellen wird üblicherweise ein TRUNCATE TABLE auf die Stage-Tabellen ausgeführt. Dies ist nur für Tabellen im eigenen Schema möglich, oder über das Systemprivileg DROP ANY TABLE! Dass dies keine brauchbare Option ist, versteht sich von selbst. Stattdessen wird für eine solche Konfiguration empfohlen, den TRUNCATE-Befehl über eine kleine PL/SQL-Prozedur mittels dynamischem SQL auszuführen.[2]

Dass ein Private Database Link sicherer ist als ein Public Database Link, ist relativ einfach nachvollziehbar. Aber was ist nun die bessere Wahl zwischen Fixed User Link und Connected User Link? Ein Connected User Link hat – wie erwähnt – ein paar Nachteile und führt zu einer etwas höheren Komplexität, da die ETL-Prozesse nicht mehr direkt vom User DWH_STAGE ausgeführt werden können. Ein Fixed User Link ist einfacher in der Anwendung. Allerdings müssen die Zugriffsinformationen – insbesondere das Passwort – bei Erstellen des Database Links gespeichert werden. Und dies führt uns zur dritten Sicherheitslücke.

Sicherheitslücke 3: Passwort gespeichert im Data Dictionary

Beim Erstellen eines Fixed User Links muss der Username und das Passwort definiert werden, mit welchem auf die referenzierte Datenbank zugegriffen wird. Diese Informationen müssen im Data Dictionary gespeichert werden, um über den Database Link zugreifen zu können. Das bedeutet in unserem Fall, dass in der DWH-Datenbank Passwörter des Quellsystems gespeichert sind. Was bedeutet dies bezüglich Zugriffsicherheit?

Viele Entwickler und DBAs schrecken vor Database Links zurück, weil dadurch „Passwörter in der Datenbank gespeichert werden“. Zwar werden die Passwörter seit Oracle 10g Release 2 verschlüsselt abgespeichert, aber durch eine einfache SQL-Query lassen sie sich entschlüsseln: [3]

```
SELECT userid
     , utl_raw.cast_to_varchar2(dbms_crypto.decrypt
                 (substr(passwordx,19),4353,substr(passwordx,3,16)))
  FROM sys.link$
```

Diese Abfrage lässt sich zwar nur mit Leseberechtigung auf die Tabelle SYS.LINK$ ausführen. Außerdem müssen EXECUTE-Privilegien für das Package DBMS_CRYPTO vorhanden sein, was für die meisten Datenbankuser auch nicht der Fall ist. Wird sichergestellt, dass diese Berechtigungen für normale Datenbankbenutzer nicht vorhanden sind, kann das Sicherheitsproblem zumindest entschärft werden. Aber es existiert nach wie vor! Jeder Datenbankuser kann mit dem Package DBMS_METADATA die DDL-Befehle für seine Objekte (also auch für Database Links) extrahieren.

---

[2] Siehe Buch „Data Warehousing mit Oracle“, Seite 106 (Hanser Verlag, ISBN 978-3-446-42562-0)

[3] Siehe http://www.oracleforensics.com/wordpress/wp-content/uploads/2012/11/database_link_security.pdf

Mit der Rolle SELECT_CATALOG_ROLE ist dies sogar für fremde Objekte möglich. Im Falle von Database Links wird zwar „nur“ das verschlüsselte Passwort extrahiert. Dieses kann aber kopiert und mit der oben erwähnten SQL-Query auf einer beliebigen Datenbank entschlüsselt werden.

Glücklicherweise hat Oracle diese Sicherheitslücke mittlerweile geschlossen. Ab Oracle 11.2.0.3 wird ein anderer Verschlüsselungsalgorithmus verwendet, der mit der erwähnten SQL-Query nicht mehr zu knacken ist. Die verschlüsselt gespeicherten Passwörter sind also vorerst wieder sicher. Doch für wie lange? Irgendwann wird auch diese Verschlüsselung wieder geknackt werden.

Tipps für sichere Konfiguration von Database Links

Nach all diesen Hinweisen auf Sicherheitslücken stellt sich die Frage: Sollen und dürfen Database Links überhaupt eingesetzt werden, um Daten von Quellsystemen in ein Data Warehouse laden? Oder anders gefragt: Erlauben Database Links eine sichere Verbindung zwischen zwei Datenbanken?

Es ist möglich, sichere Lösungen mit Database Links zu implementieren. Allerdings gehört dazu mehr als nur das Anlegen eines Database Links. Wichtig ist vor allem die Erstellung eines spezifischen Users, der ausschließlich Leseberechtigungen auf die notwendigen Views im Quellsystem hat. Durch Verwendung von Private Database Links statt Public Database Links kann sichergestellt werden, dass der Zugriff nur vom vorgesehenen Schema aus erfolgen kann.

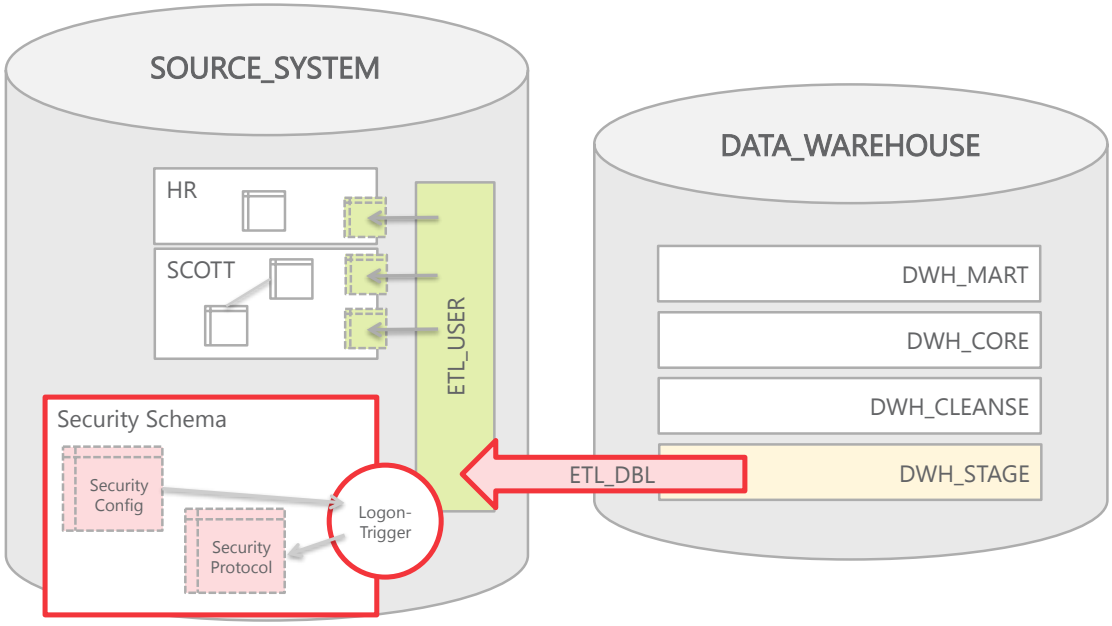

Abb. 2: Beispiel für sichere Konfiguration von Database Link

Um zu gewährleisten, dass auf den User ETL_USER nur über den dafür vorgesehenen Database Link zugegriffen werden kann, wird auf der Quelldatenbank ein Logon-Trigger implementiert, wie in Abbildung 2 skizziert. Der Trigger prüft, ob von der richtigen Datenbank aus über den korrekten Database Link zugegriffen wird. Direkte Connections als ETL_USER oder Verbindungen von einem „gefälschten“ Database Link von einer anderen Datenbank aus werden so verhindert. Außerdem wird empfohlen, alle erfolgreichen und zurückgewiesenen Connections in einer Log-Tabelle zu protokollieren, um Missbrauchsfälle bei Bedarf aufdecken zu können.

Das nachfolgende Beispiel zeigt das Grundprinzip einer solchen Logon-Überprüfung. Zusätzliche Checks auf IP-Adresse, OS-User oder weitere Kriterien können bei Bedarf eingebaut werden. In komplexeren Umgebungen mit mehreren Database Links, die jeweils auf einen spezifischen User zugreifen, kann die Überprüfung beispielsweise anhand einer Konfigurationstabelle durchgeführt werden.

```
CREATE OR REPLACE TRIGGER dbl_logon_trg
AFTER LOGON ON DATABASE
DECLARE
   v_username VARCHAR2(30)  := sys_context('USERENV','SESSION_USER');
   v_dbl_info VARCHAR2(200) := sys_context('USERENV','DBLINK_INFO');
BEGIN
   IF v_username = 'ETL_USER' THEN
      IF v_dbl_info IS NULL THEN
         write_log('failed: direct login', v_username, v_dbl_info);
         raise_application_error(-20101, 'Direct login not allowed.');
      ELSIF v_dbl_info NOT LIKE 'SOURCE_GLOBAL_NAME=DWH_PROD, DBLINK_NAME=ETL_DBL%' THEN
         write_log('failed: wrong dblink', v_username, v_dbl_info);
      raise_application_error(-20102, 'Login from wrong database link not allowed.');
      ELSE
         write_log('successful login', v_username, v_dbl_info);
      END IF;
   END IF;
END dbl_logon_trg;
```

Diese Beispielkonfiguration zeigt, dass es möglich ist, Datenbankumgebungen so zu konfigurieren, dass sichere Zugriffe über Database Links möglich sind. Das Erstellen des Database Links im Data Warehouse ist dabei der einfachste Teil. Entscheidend ist vor allem die Konfiguration auf Seite des Quellsystems. Durch spezifische User mit eingeschränkten Zugriffsrechten sowie einen Logon-Trigger, der unerlaubte Zugriffe verhindert, kann eine sichere Umgebung aufgebaut werden.

Kontaktadresse:

Dani Schnider
Trivadis AG
Europa-Strasse 5
CH-8152 Glattbrugg

Telefon: +41(0)44-808 70 20
Fax: +41(0)44-808 70 21
E-Mail dani.schnider@trivadis.com
Internet: www.trivadis.com