Hewlett Packard Enterprise

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Jörg Hildebrandt
HPE TS Consulting

11.05.2016

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Über mich

Jörg Hildebrandt

privat

- 48 Jahre, geschieden, 3 Kinder
- Wohnhaft in Weimar, Thüringen
- Hobbies: Fahrrad fahren, Gleitschirmfliegen

beruflich

- Mehr als 18 Jahre Datenbank-Erfahrung, spezialisiert auf das Produkt-Portfolio von Oracle
- Entwickler, DBA, SAP Consultant, u.v.a.m.
- aktuell: Solution Architekt im Consulting bei Hewlett Packard Enterprise

in der DOAG aktiv seit 2001 als

- Mitglied im Vorstand, Aufbau vom Publishing-Bereich, Gründung DOAG-Dienstleistungen GmbH
- Leiter der Special Interest Group Oracle + SAP (bis heute)
- Regionaler Repräsentant für Thüringen (bis heute)

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Aufgabenstellung

In einer sehr großen Kundenumgebung wurde nach neuen Möglichkeiten gesucht, ***Datenbank-Daten*** (Inhalte) und ***Datenbank-Strukturen*** (Definitionen) zu separieren.

Die Strukturen werden hierbei als Teil eines Applikations-Release gesehen, sollen also innerhalb einer Version nicht verändert werden.

Im ***Normalbetrieb*** der Applikation soll nur mit den Daten selber gearbeitet werden (können).

Im ***Wartungsmodus*** der Applikation dürfen sowohl Strukturen als auch Daten verändert werden.

Die angestrebte Separierung soll dabei für die Applikation ***transparent*** sein.

Die Nutzung der Datenbank soll unter der o.a. Prämisse ***sicher*** möglich sein.

Aufgabenstellung

Kollission der Interessen von Entwicklung und Betrieb

Software wird oft *auf dem Server unter dem Schreibtisch* entwickelt

- Einfachste Anforderungen an Security
- Neuaufsetzen der Umgebung jederzeit einfach möglich

Unternehmens-Datenbanken sind oft Sammel-Instanzen mit einer Vielzahl parallel betriebener Schemata

- Umgebungen müssen gerade im Hinblick auf Security und Ressourcen-Konsum klar abgrenzbar sein
- Nutzung teilweiser komplexer Prozesse für Betrieb und Monitoring
- Neuaufsetzen ganzer Datenbanken im Regelfall nicht möglich

Bildquelle: Timo Klostermeier / pixelio.de

Aufgabenstellung

Kollission der Interessen von Entwicklung und Betrieb

Änderungen von DB-Strukturen müssen auf einfachste Weise möglich sein

- Im Entwicklungszyklus einer Software werden ständig sowohl Änderungen an den Daten, wie auch den Objekt-Strukturen vorgenommen
- Oftmals wird als DBA entwickelt, oft werden Rechte-Konzepte erst kurz vor Fertigstellung des Produktes zum Einsatz gebracht bzw. vernachlässigt
- Auditing wird nicht benötigt

DB-Strukturen dürfen zur Laufzeit eines Software-Releases nicht verändert werden

- Die einen DB-Service nutzende Applikation soll nur mit den ***Daten*** einer Datenbank arbeiten.
- Struktur-Änderungen werden meistens ausschließlich im Rahmen von Deployments vorgenommen.
- Es sind oft feingranulare Rechte-Konzepte im Einsatz.
- DB-Aktivitäten werden oft protokolliert und auditiert, teilweise gibt es rechtliche Notwendigkeiten hierzu

Aufgabenstellung

Kollission der Interessen von Entwicklung und Betrieb

Die Software soll zu vielen Datenbanken kompatibel sein

- Entwicklungskosten sollen möglichst gering sein
- Auf spezielle Konzepte oder DB-Features eines Herstellers wird im Regelfall verzichtet.
- Auf spezielle Hersteller ausgerichtetes Know How ist oft nicht in der Tiefe vorhanden.

Im Unternehmen gibt es eine klare Datenbank-Strategie

- Die Kosten für die Lizenzierung der Umgebung sollen minimiert werden.
- Alle Betriebsverfahren und Prozesse sind auf einen oder wenige Datenbank-Hersteller ausgerichtet.
- Das für Betrieb und Support benötigte Know How ist auf die strategischen Datenbanken fokussiert.

Bildquelle: Christian Beuschel / pixelio.de

Aufgabenstellung

Strikte Umsetzung von ITIL-Prozessen im Lebenszyklus von Unternehmens-Anwendungen

- Implementierungen und Deployments dürfen nur im Rahmen vereinbarter Change-Prozesse durchgeführt werden.
- Alle Änderungen und deren Zeitrahmen sind damit bekannt und dokumentiert.

Absicherung von Software-Release-Ständen in Datenbanken

- Die zu einem Software-Release gehörenden DB-Strukturen dürfen sich nur im Rahmen eines Release-Wechsels ändern.

Konsequente Trennung von administrativen und die Datenbank nutzenden Operationen

- Die einen DB Service nutzenden Sessions müssen derart eingeschränkt werden, dass sie nur mit den Daten operieren können.
- Strukturelle Änderungen von DB-Objekten sollen im Regelbetrieb ausgeschlossen werden.

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Herkömmliche Realisierung

Verwendung von Admin User und Runtime User

Jeder in der zentralen DB-Infrastruktur betriebenen Applikation sind mindestens 2 Datenbank-Schemata pro genutztem DB-Service zugeordnet

Admin User

- Eigentümer sämtlicher DB-Objekte, die von der Applikation genutzt werden
- Durchführung von strukturellen Änderungen an zur Applikation gehörenden DB-Objekten und ggf. Befüllung mit Daten im Wartungsmodus (DDL- und DML-Berechtigungen)

Runtime User

- Nutzer der im Admin-Schema angelegten DB-Objekte
- Lesen, Schreiben und Verändern von Nutzdaten der Applikation (DML-Berechtigungen), Ausführung von Funktionen, Prozeduren, Triggern und Sequenzen

Herkömmliche Realisierung

Verwendung von Admin User und Runtime User

Die Nutzung der DB-Objekte des Admin Users durch den Runtime User wird mittels eines einfachen, zweistufigen Verfahrens ermöglicht.

Freigabe von Objekt-Berechtigungen seitens des Admin Users

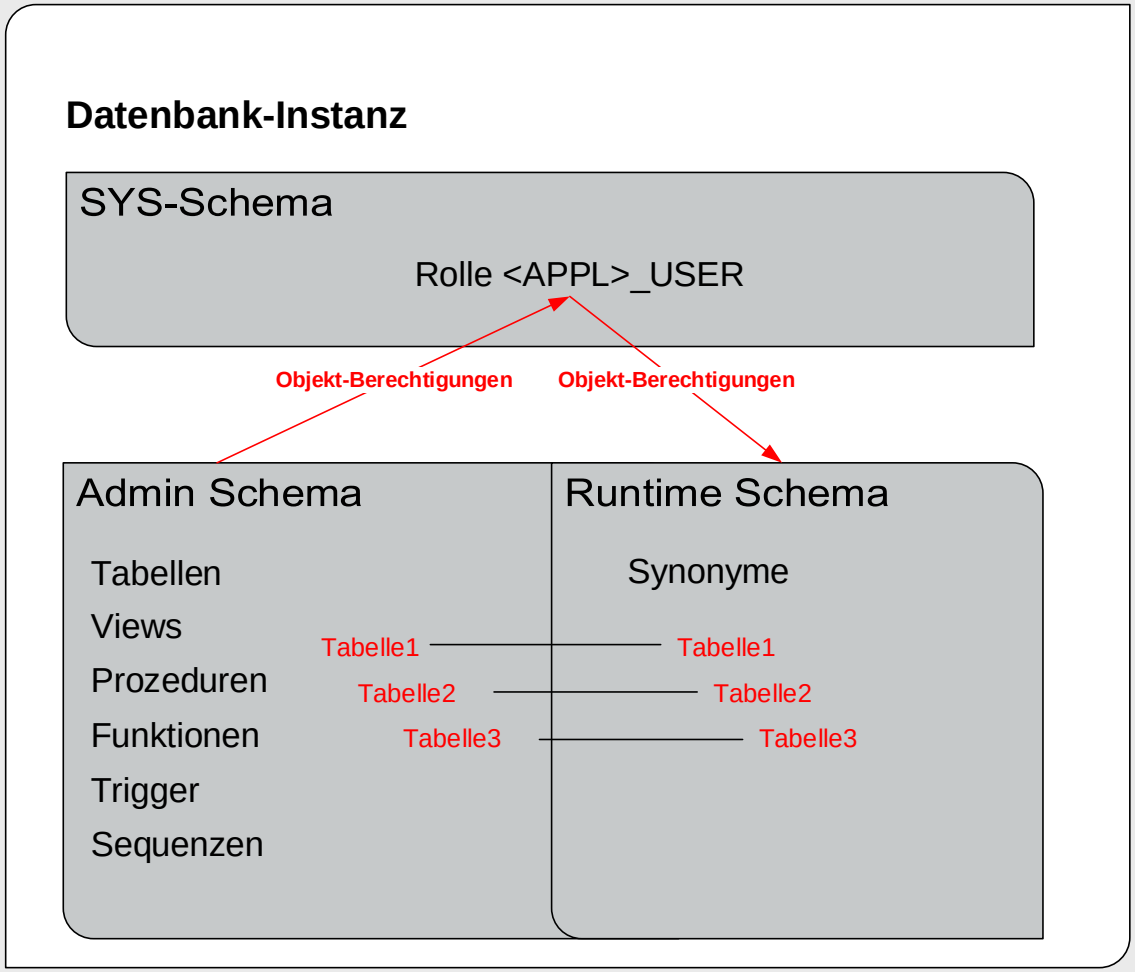
- Im Schema der Admin Users werden die für den Regelbetrieb vorgesehenen DML-Berechtigungen z.B. an eine Rolle <APPL>_USER weitergegeben.

Erstellung von Synonymen im Kontext des Runtime Users

- Im Schema des Runtime Users werden Aliasnamen (Synonyme) auf die potenziell zu nutzenden DB-Objekte des Admin Users angelegt.
- Die Rolle <APPL>_USER wird dem Runtime User zugewiesen.

Herkömmliche Realisierung

Verwendung von Admin User und Runtime User



Herkömmliche Realisierung

Verwendung von Admin User und Runtime User

Vorteile

- Einfaches Konzept
- Umgebungen „sehen sowohl für den Admin User als auch den Runtime User genauso aus“

Bildquelle: Rainer Sturm / pixelio.de

Herkömmliche Realisierung

Verwendung von Admin User und Runtime User

Nachteile

- Verwendung von zwei Schemata pro Anwendungsfall und von Synonymen ist prinzipiell technisch nicht notwendig
- Aufwändige Administration besonders im Hinblick auf
  - Verwaltung von Passworten (z.B. verschiedene Anforderungen an Komplexität und Ablauf)
  - Verwendung von Synonymen
- Deployments müssen bezüglich des DB-Teils in zwei User-Umgebungen durchgeführt werden.
- Für spezielle Verwendungen sind oftmals ohnehin ergänzende Rollen-Konzepte im Einsatz, die die Gesamt-Komplexität erhöhen.

Bildquelle: Rainer Sturm / pixelio.de

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Alternative Realisierung

Ziele in Kundenprojekt

- Strukturelle Einfachheit aus Applikationssicht (Transparenz)
- Einfache Administrierbarkeit aus Betriebssicht
- Verbesserung der Sicherheit
- Einfachere Ausführbarkeit von automatischen Deloyments hinsichtlich ihres DB-Teils
- Möglichst wenig Anpassung an den Applikationen zur Nutzung des neuen Verfahrens
- Einfache Umstellung auf das neue Konzept
- Weitere Lizenzkosten sollen wenn möglich vermieden werden

Alternative Realisierung

Jeder Applikation die einen DB-Service nutzt, ist idealerweise nur noch 1 Datenbank-Schema pro DB-Service zugeordnet.

(Ehemaliger) Admin User

- Eigentümer und Nutzer sämtlicher DB-Objekte, die zur Applikation gehören
- Durchführung von strukturellen Änderungen an zur Applikation gehörenden DB-Objekten und ggf. Befüllung mit Daten im Wartungsmodus (DDL- und DML-Berechtigungen)
- Lesen, Schreiben und Verändern von Nutzdaten der Applikation im Normalbetrieb (DML-Berechtigungen), Ausführung von Funktionen, Prozeduren, Triggern und Sequenzen

Alternative Realisierung

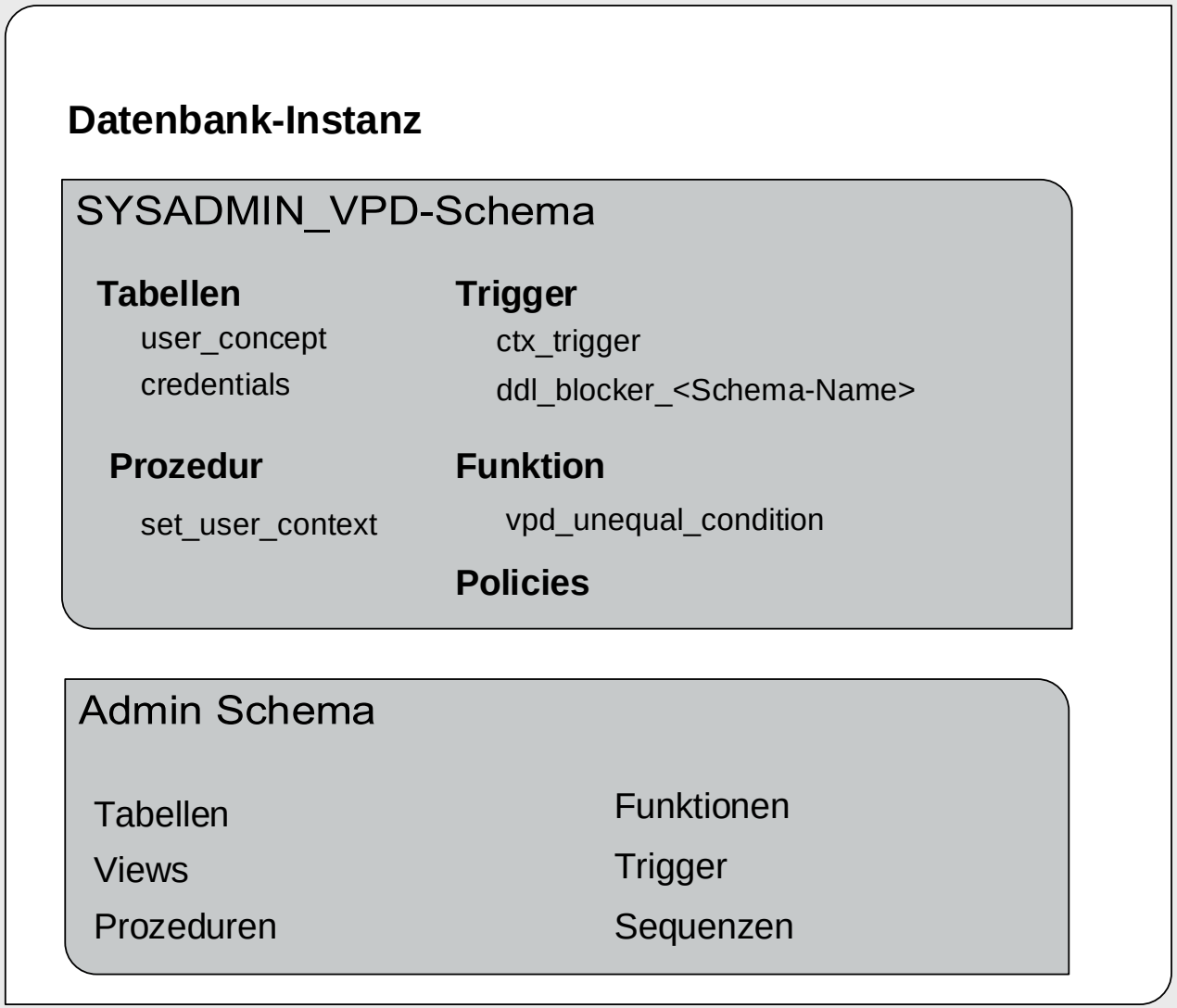
Die Nutzung der DB-Objekte wird mittels des folgenden Verfahrens ermöglicht.

- Definition von logischen Rollen (nicht DB-Rollen) entsprechend der vorgesehenen Arten der Nutzung des DB-Services der Applikation <APPL>
  - o WRITEALL Einfügen, Ändern und Löschen bez. aller Datensätze von <APPL>
  - o MAINTENANCE allumfassende Rechte bezüglich der DB-Objekte von <APPL>
  - o CUSTOM adaptive Rechte, z.B. nur Leserechte auf DB-Objekte von <APPL>

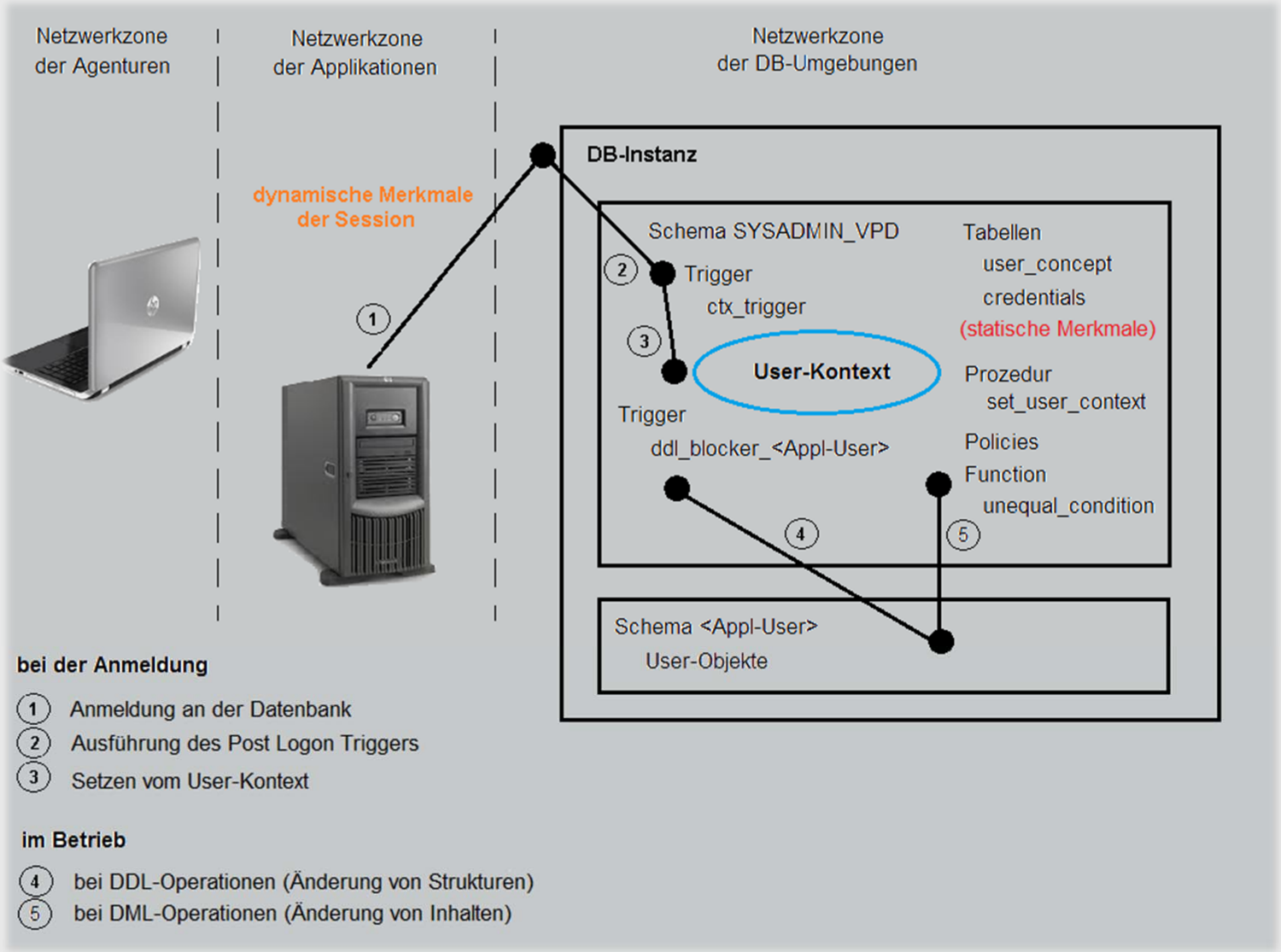
- Definition eindeutiger Metadaten der zugreifenden Clients (z.B. Bezeichnung des Clients, zugreifendes Programm, zugreifender OS-User, zugreifender Service-Name), sowie Zuordnung von logischen Rollen, Metadaten, Berechtigungsstrukturen und Zeitfenstern
  - o Ablage der genannten Inhalte in einem Schema mit administrativen Berechtigungen

- **Kontext-sensitive Zuweisung der Berechtigungsstrukturen beim Login**

Alternative Realisierung



Alternative Realisierung



Alternative Realisierung

Beim Anmeldevorgang ausgewertete Merkmale

- Name des Login-Users <APPL-User>
- Aufgerufener Service-Name
- Name des Clients
- IP-Adresse des Clients
- Aufrufendes Programm
- Betriebssystem-User am Client

Bildquelle: Thorben Wengert / pixelio.de

Nicht alle Merkmale müssen gesendet werden. Sie müssen aber so wie gesendet in der Datenbank vordefiniert werden, damit eine 100%ige Übereinstimmung der Merkmale vorliegt.

Alternative Realisierung

Annahme in Kundenumgebung

Es muss sichergestellt sein, dass von den auf Datenbanken zugreifenden Servern nur autorisierte Applikationen mit „beglaubigten Merkmalen“ eingesetzt werden.

Vorsicht Spoofing!

Quelle Wikipedia: *Spoofing (englisch für Manipulation, Verschleierung oder Vortäuschung) nennt man in der Informationstechnik verschiedene Täuschungsmethoden in Computernetzwerken zur Verschleierung der eigenen Identität …*

Bildquelle: Bernd Kasper / pixelio.de

Alternative Realisierung

Bestimmte Merkmale lassen sich einer Session über das Oracle-Package dbms_application_info mitgeben, z.B.

dbms_application_info.set_client_info(client_info => ‘Main Procedure‘);

dbms_application_info.set_module(module => ‘Authentication‘,action_name => ‘Set Role‘);

Sie sind dann über die View v$session oder den hier zur Authentifizierung verwendeten sys_context so wie gesendet abrufbar.

Auch andere, vermeintlich eindeutige Merkmale können ge-faked sein

Beispiel Hostname

select sys_context('USERENV','HOST') from dual;

SQL> appserver4711

Der Hostname eines neu aufgesetzten Servers ist selbstverständlich frei bestimmbar …

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

1. Über mich
2. Aufgabenstellung
3. Herkömmliche Realisierung
4. Alternative Realisierung
5. **Implementierungs-Details**
6. Migrationspfade
7. Fazit

Implementierungs-Details

Trigger zum Verhindern von Strukturänderungen an DB-Objekten

```
create or replace trigger sysadmin_vpd.ddl_blocker_<Appl-User>
before create or alter or drop on <Appl-User>.SCHEMA
begin
  if sys_context("USER_CTX", "USER_ROLE") not in ("MAINTENANCE")
  then
     raise_application_error(-20001,"No permissions to change database objects");
  end if;
end;
/
```

Bildquelle: Tim Reckmann / pixelio.de

Implementierungs-Details

VPD-Policy zur Definition von Objekt-Rechten

```
dbms_rls_add_policy (
   object_schema => '<Appl-User>',
   object_name => '<Tabelle>',
   policy_name => '<Tabelle>_policy',
   function_schema => 'SYSADMIN_VPD',
   policy_function => 'UNEQUAL_CONDITION',
   statement_types => 'SELECT,INSERT,UPDATE',
   update_check => true);
```

Bildquelle: Tim Reckmann / pixelio.de

Implementierungs-Details

Funktion zur Bereitstellung einer „heimlichen“ Where Clause bei Operationen auf zu schützenden Objekten

```
create or replace function sysadmin_vpd.unequal_condition (schema_p in varchar2,
table_p in varchar2)return varchar2 is
begin
   if sys_context('USER_CTX', 'USER_ROLE') not in ('WRITEALL','MAINTENANCE')
   then return '1=0';
   else return NULL;
   end if;
end unequal_condition;
/
```

Bildquelle: TigerLime / pixelio.de

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

Migrationspfade

Randbedingungen

- Die Verwendung des alternativen Userkonzeptes ist nicht an die Nutzung einer bestimmten Datenbank-Betriebsumgebung gebunden.
- Das Userkonzept ist aufgrund der Einschränkungen von VPD nur für Umgebungen einsetzbar, die mit maximal 3 verschiedenen Betriebsmodi betrieben werden. Diese Betriebsmodi werden wie oben beschrieben, jeweils durch ihren User-Kontext WRITEALL, MAINTENANCE oder CUSTOM repräsentiert. Für andere Umgebungen kann das Userkonzept nur mit ergänzenden Regulierungen über Rechte und Rollen abgebildet werden.
- Die Adressierung von Tabellen muss ausschließlich ohne Angabe des Schema-Namens erfolgen.
  - Insbesondere dürfen Tabellen nicht mit <Runtime User>.<Tabelle> angesprochen werden.

Bildquelle: Kurt Michel / pixelio.de

Migrationspfade

Randbedingungen

- Vor der Umstellung von produktiven Applikationen zur Nutzung des alternativen Userkonzeptes müssen applikationsseitig geeignete Tests durchgeführt und von Key Usern bestätigt werden.
- Die Password-Policies müssen auf die andere Art der Nutzung des Admin Users hin angepasst werden, da der User-Name sich für dieselbe Verwendung ändern soll.
- Die für die Nutzung des User-Konzeptes notwendigen Objekte und Policies werden in einem separaten Schema namens SYSADMIN_VPD abgelegt, um sie nicht mit systemeigenen Daten zu vermischen. Das Passwort wird nach den Security-Vorschriften des Kunden behandelt und verwaltet.

Bildquelle: Kurt Michel / pixelio.de

Migrationspfade

Durchführung

- Anlegen des Users SYSADMIN_VPD und der Umgebung für Virtual Private Database, sowie weiterer zentraler Strukturen pro DB-Instanz, insofern nicht bereits geschehen
- Automatische Ermittlung der Berechtigungsstrukturen des oder der pro Applikation eingesetzten Runtime Users und Abbildung der Berechtigungen in Policies
- Erstellung des Triggers zum Blockieren von DDL-Operationen
- Ermittlung der Merkmale aller zugreifenden Clients und Ablage der Informationen in der credentials-Tabelle der DB-Instanz
- Erstellung eines DB-Profils gemäß den neuen Anforderungen des Admin Users und den Security-Anforderungen des Kunden

Bildquelle: Kurt Michel / pixelio.de

Migrationspfade

Durchführung

- Stoppen der die DB-Services nutzenden Applikationen
- Sperren des Runtime Users für die weitere Nutzung und Beenden potentiell verbliebener Sessions
- Tauschen des Anmelde-Users an der Applikation vom Runtime User auf den Admin User und seinem Passwort
- Änderung des User-Profils des Admin Users in der Datenbank auf das bereitgestellte neue Profil
- Änderung des Eintrags in der Tabelle user_concept für den Admin User auf ‚neu‘
- Neustart der Applikationen

Bildquelle: Kurt Michel / pixelio.de

Einsatz innovativer User-Konzepte ohne teure DB-Optionen

1. Über mich
2. Aufgabenstellung
3. Herkömmliche Realisierung
4. Alternative Realisierung
5. Implementierungs-Details
6. Migrationspfade
7. Fazit

Fazit

Einsatz von Virtual Private Database (VPD)

- VPD ist ein stabiles, kostenfreies Feature einer jeden Oracle-Datenbank bei Nutzung der Enterprise Edition und seit Release 8i verfügbar.
- Eine Alternative wäre die Verwendung von Database Vault. Hier ist es möglich, sog. Command Rules zu hinterlegen. Diese können sich auch auf die Verwendung bestimmter Kommandos beziehen. Bei Verletzung einer Rule wird wiederum eine Fehlermeldung generiert. **Database Vault erfordert zusätzliche Lizenzen.**

Fazit

Nachteil von Virtual Private Database

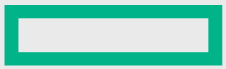
- Für die hier beschriebene Verwendung fehlt dem Package dbms_rls eine Gruppen-Funktionalität, mit der sich ein komplettes Set von Policies bündeln lässt. So wäre es z.B. denkbar, dass ein User nur Lese-Berechtigungen auf 3 Tabellen in einem Schema erhalten soll, ein anderer wiederum Vollzugriff auf alle Tabellen.
- Die Funktion create_policy_group stellt die o.g. Funktionalität leider nicht bereit, auch wenn der Name das vermuten lässt. Hiermit können lediglich Policies von mehreren Objekten zusammengefasst werden.

Fazit

Einsatz von Datenbank-Rollen

- Der Einsatz von Rollen zum Zwecke der Kontext-basierten Zuweisung oder dem Entzug von Berechtigungen ist nach dem Berechtigungskonzept von Oracle nicht möglich.

Der Schema-User der Datenbank-Objekte anlegt, hat automatisch und unumkehrbar allumfassende Berechtigungen auf diese.



Hewlett Packard
Enterprise

Vielen Dank!

joerg.hildebrandt@hpe.com