Authentication and Authorization in Oracle 12.2 based Systems

Dr. Günter Unbescheid
Database Consult GmbH, Jachenau
Database Consult GmbH

- Founded in 1996
- Specialized in ORACLE-based Systems
- Focus Areas
  - Security, Identity Management
  - Tuning, Installation, Configuration, Systemanalysis
  - Support, Troubleshooting, DBA-Tasks
  - Database Design, Datamodelling und –design
  - Custom made Workshops
  - www.database-consult.de
Agenda

- Introduction
- Authentication
  - Password based
  - OS based
  - Kerberos, SSL/TLS
  - Enterprise User
- Authorization
  - Privileges and Roles
  - Types of Grants
- Identities in Multitier Environments
  - Client Identifier
  - Real Application Security
Introduction

- Database Security and integrity are still key aspects of an organisation’s security posture
  - despite “autonomous” databases
- Authentication
  - process or action of verifying the identity of a user or process
- Authorization
  - process of giving someone permission to do or have something
- DB Security is based on both
Database User Types

- **Local User**
  - The user is only present in a specific PDB of a CDB
  - Same username can be present in multiple PDBs, but they are unrelated due to distinct namespaces, passwords, profiles etc.
  - Must not use Common User Prefix

- **Common User**
  - Two Subtypes since 12.2: Application Common User (ACU) and CDB Common User (CCU)
  - ACU present in Application Root Container and its dependents
  - CCU present in all containers (root and all PDBs and App. PDBs)
  - Security Context is applied via Name-Prefix (next slide)

- **Non-CDB User**
  - “local” user in Non-CDB
Common Users

- **CDB Common User**
  - connected to a common user with the `CREATE USER` privilege for creation
  - the current container must be the root container.
  - username prefixed with "C##" or "c##" and contain only ASCII or EBCDIC characters (init Parameter `common_user_prefix`)
  - username must be unique across all containers
  - either `CONTAINER=ALL` clause, or omit it (default setting when in root)

- **Application Common User**
  - created in application root container
  - name must not start with common prefix
  - prefix is empty string by default
  - After creation application has to be synched:
    ```sql
    alter pluggable database application <name> sync
    ```
Local Users

- must be connected to a user with the **CREATE USER** privilege for creation
- username must **not** be prefixed with "C##" /"c##"
- The username must be unique within the PDB or CDB$ROOT
- either specify the **CONTAINER=CURRENT** clause, or omit it, as this is the default setting when the current container is a PDB
Profiles and Password Verification

- Local an common profiles (with common prefix `c##` in Root)
- New profile resource `INACTIVE_ACCOUNT_TIME` (days)
- Script `catpvf.sql` provides improved password functions
  - `ora_complexity_check`, `ora_string_distance` – helper
  - `oral2c_verify_function` – Rules: C8, L1, D1, S1, PW contains username, username reversed, servername, oracle, distance > 2
  - `verify_function_11G` – Rules: C8, L1, D1, equals username (+digits), username reversed, servername, dictionary check etc.
  - `verify_function` – old version (pre 11)
  - `oral2c_strong_verify_function` – Rules: C9, UL2, LL2, D2, S2, distance >=4
  - `oral2c_stig_verify_function` – Rules: C15, UL1, LL1, D1, S1, distance >=8 used in STIG User Profile

- Maximum PW length in any case is 30, beware `by values` clause
Database User Password Storage

- stored as one-way hash in data dictionary, can not be decrypted
- starting with version 6 use DES algorithm, no case sensitivity
  - username and password concatenated, refer to rainbow tables
- Version >= 11gR1 use SHA1 (enable case sensitivity)
  - PW concatenated with SALT (random data), renewed when PW changes
  - stored in $user$\texttt{spare4}
- Version 12.1.0.1 – choices DES, SHA1 and HTTP (used for digest authentication of XDB, weak!)
- Version >= 12.1.0.2 – added PBKDF2 followed by SHA512 (SHA2)
  - SHA2: better collision resistance, longer outputs due to longer key length
  - Password-Based Key Derivation Function 2 – protect against brute force by producing a derived key
  - disable HTTP by default (HTTP DIGEST clause of \texttt{user} command)
Oracle 12c Password Hash

- 12C password hash protects against password-based security threats by including support for mixed case passwords:
  - cryptographic hash function used is based on a de-optimized algorithm involving Password-Based Key Derivation Function 2 (PBKDF2) and the SHA-512 cryptographic hash functions
  - 12C password version adds a salt to the password when it is hashed
- Algorithm makes dictionary and brute force attacks more expensive
- By default two versions of the password hash: 11G and 12C
  - The 10G password version, which was generated in Oracle Database 10g, is not case sensitive and is DES-based
  - 11G and 12C password versions are case sensitive.
- Version of PW hash used for authentication depends on
  - `SQLNET.ALLOWED_LOGON_VERSION_CLIENT` and` SQLNET.ALLOWED_LOGON_VERSION_SERVER`
Passwords and Case Sensitivity

- init Parameter `SEC_CASE_SENSITIVE_LOGON` deprecated in 12.1.0, implicit/default value **TRUE**
  - you should not change it to **FALSE**, otherwise
  - startup warning:
    `ORA-32004: obsolete or deprecated parameter(s) specified for RDBMS instance`

- SQLNET Parameter `ALLOWED_LOGON_VERSION_SERVER` changed
  - Default value 12.1 – 11 – Default value 12.2 – 12
  - New Default may lead to `ORA-1017` when running with old pw versions

```
select username, password_versions from DBA_USERS where ...
```

<table>
<thead>
<tr>
<th>SQLNET.ALLOWED_LOGON_VERSION_SERVER</th>
<th>10G, 11G, and 12C</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLNET.ALLOWED_LOGON_VERSION_SERVER=8</td>
<td></td>
</tr>
<tr>
<td>SQLNET.ALLOWED_LOGON_VERSION_SERVER=12</td>
<td></td>
</tr>
<tr>
<td>SQLNET.ALLOWED_LOGON_VERSION_SERVER=12a (&gt;=12.1.0.2)</td>
<td>12C</td>
</tr>
</tbody>
</table>
Passwords and Database Links

9i Client (Unpatched) (U)

Oracle Net Services protocol negotiation fails

10g Client (E)

Oracle Net Services protocol negotiation succeeds

11.2.0.3 Client (C)

Oracle Net Services protocol negotiation succeeds

Fixed Database Link (Host)

Database Link (Target)

12g

9i

H acts as client to T by using SQLNET.ALLOWED_LOGON_VERSION_CLIENT = 8

H also acts as server for C, E, and U by using SQLNET.ALLOWED_LOGON_VERSION_SERVER = 12

T uses SQLNET.ALLOWED_LOGON_VERSION=8

T

©Database Consult GmbH - Jachenua
Database Internal Authentication

- **PROs**
  - Default Solution for Database Users
  - Improved Password (Hash) Functions in current release
  - Extensive Tool Support

- **CONs**
  - No central administration
  - Difficult to keep in Sync with many sites, databases and users
  - Big administrative overhead
Administrative Database Users

- Need to authenticate and authorize even when DB is not online
- Authentication via OS groups
  - recommended for local connects
  - in CDB only effective for Root Container
- via password files – in addition to internal storage (no display in PDB
  - init Parameter `remote_login_passwordfile`
  - in 12.2 extended functionality (next slides)
  - also good for remote authentication
- via external services, e.g. OUD/OID and enterprise users
  - Privileges `SYSDBA` and `SYSOPER`
  - no need for password files
  - init Parameter `LDAP_DIRECTORY_SYSAUTH`
Password File 12.2

- Password Files for ASM and DB can be created/stored in ASM Diskgroup
- Admin Passwords are case sensitive (no \texttt{IGNORECASE} in \texttt{orapwd} anymore)
- Password profile parameters and pw complexity functions can be used for admin user authentication too
  - \texttt{FAILED_LOGIN_ATTEMPTS}, \texttt{INACTIVE_ACCOUNT_TIME (days)}, \texttt{PASSWORD_LOCK_TIME}, \texttt{PASSWORD_LIFE_TIME}, \texttt{PASSWORD_GRACE_TIME}
- Setting the \texttt{ORAPWD} utility \texttt{FORMAT} parameter to \texttt{12.2} (default)
  - enables the password profile parameters for administrative users
Password File 12.2

- **FORMAT** to **12.2** enforces the following rules for PW
  - contains no fewer than 8 characters and includes at least one numeric and one alphabetic character.
  - is not the same as the user name or the user name reversed.
  - is not the same as the database name.
  - does not contain the word oracle (such as oracle123).
  - differs from the previous password by at least 8 characters.
  - contains at least 1 special character, not exceed 30 characters.
  - does not contain the double-quotiation character (").
  - However, it can be surrounded by double-quotiation marks

- The password file can also contain **external users**, e.g. Kerberos:

  ```
  orapwd FILE='+DATA/orcl/orapworcl' DBUNIQUEUSERNAME='orcl', FORMAT=12.2
  sys=external('KerberosUserSYS@example.com')
  syskm=external('KerberosUserSYSKM@example.com')
  ```
Proxy User

- connect to a schema (proxy client) via different user (proxy user)
  - proxy client has to be database user (DBU)
  - proxy user can be DBU or Enterprise User
  - authentication by password, Kerberos or SSL/TLS, also ext. password st.
- available since 9iR2 – **connect through** clause of **alter user**
  - since 12.1: **proxy only connect** possible
- Proxy user inherits all or part of proxy client roles
- Useful for (administrative) tasks under „foreign“ schema without PW transfer
OS Authentication – non administrative

- Possible without database password via Prefix *(os_authent_prefix)*
- Useful when
  - personal OS users exist at target site *and*
  - personal local database connects are necessary
- Also available in CDB context for common users
  - sync of os_authent_prefix with common_user_prefix
- Better choices for PW-less connects: Kerberos or SSL/TLS
  - local as well as remote connects
SSL/TLS and certificates

- SSL/TLS – protocol for securing network connections
  - use of private/public keys (certificates) and symmetric keys
- Delivers PW less authentication through certificates and network encryption
- Components for configuration:
  - Certificates on client and server site – *existing PKI* is of advantage
  - Storage of certs in wallet or HSM
  - Configure Listener (*listener.ora*) and Network profile (*sqlnet.ora*)
  - External Users (*identified externally*) in database carrying distinguished name of client certificate
- Can be combined with Enterprise Users in OUD/OID
Kerberos

- K. is a computer network authentication protocol
- Used for authentication only
  - no encryption of session data
- It makes use of Tickets to establish identities
  - Ticket Granting Ticket (TGT) at connect
  - Service Tickets at request (PW less)

Components
- Service User in KDC
- Keyfile and krb5.conf at DB target server
- External User in DB with User Principal Name of Service Ticket

Can be combined with EUS
Enterprise User (EUS)

- User stored in (central) directory Service (OUD/OID)
  - Models: direct storage, chaining or proxy
- Can handle authentication and authorization
- EUS authentication via password, Kerberos or SSL/certificate
- Components:
  - Directory Service with EUS object classes
  - Metadata: DB registration, user- and role mappings
  - Wallet with DB authentication
  - DB global user (*identified globally*)
EUS Mappings

- Private Schema/Exclusive Schema
  - 1:1 EUS to Global database User via distinguished name of DB user

- Shared Schema
  - Entry Level – 1:1 via Enterprise User
  - Subtree Level – all EUS of subtree to shared global schema of DB

- Proxy Permission – EUS to DB proxy client
  - DB: `grant connect through enterprise users` clause

- EUS and Container Database
  - PDBs register like Non-CDB
  - wallets in standard directory
  - Mappings for local users
Client Identifier (CI)

- Application Users in Middle Tier are usually no database users
- MT connect through connection pool via pool DB user
  - Enduser Identity is lost on DB site
- CI mark sessions with arbitrary string, e.g. enduser identity
  - use DBMS_SESSION or OCI Calls
  - present in `v$session, aud$` etc.
- global application context
  - global variables used to set/evaluate client identifies
Authorization by Privileges

- Administrative privileges – **SYSDBA, SYSBACKUP** etc.
  - bundle specific administrative tasks
  - in **format=12.2** (default) also **password_profile** present
  - can not be granted to roles

- System privileges (SP)
  - right to perform actions, or actions on behalf of schema objects
  - e.g. **create table, select any table** etc.

- Object privileges (OP)
  - permission to perform an action on a particular schema object

- SP or OP are granted users or roles
  - commonly granted (**container=all**) – for each and every container
  - locally granted (**container=current**) (default) only locally in PDB
User Roles

- User roles group privileges and other roles and can be granted themselves to users and other roles
  - share namespace with users

- Types of roles
  - *common role* – created in CDB$ROOT and present in all PDBs, pref. `c##`
  - *local role* – created and valid for specific PDB only

- Activation of roles
  - default roles of user via *grant*
  - non-default roles of user via *grant* and *set role* (password possible)
  - *secure application roles* – activated via invoker’s right PL/SQL, no grant
  - *global roles (identified globally)* via OUD/OID based on enterprise roles
  - code based access control (CBAC) – grant to PL/SQL program
EUS-Mappings

EUS-Umgebung

EDom
ERo
EUs

regDB
EGr

GLRo

Role(s)
Priv(s)

GLUs

shared schema or explicit global schema

Enaty Level or Subtree Level

©Database Consult GmbH - Jachenau
Database Vault

- Define access privileges against security realms
  - override **any** and owner privileges
  - option for Enterprise Edition
- prevent DBAs from viewing and modifying table data
- even prevent schema owner from viewing and modifying table data
Label Security

- Framework for Row level access control based on labels, i.e. sensitivity levels
  - ~ mandatory access control (MAC=)
- Labels are attached to rows and users
- Users see rows according to own clearance level (and below)
  - grant is not sufficient
  - enforcement via policies
- Build on top virtual private database (VPD)

![Diagram showing clearance levels and flight table]

Source: http://mohamednabeel.blogspot.de
Real Application Security

- RAS – Real Application Security
  - introduced under 12c R1
  - building upon/enhancing concepts like VPD, secure application context, lightweight sessions in OCI, application roles, proxy authentication etc.

- Centralizing security models and authorization logic
  - deploying security policies inside Oracle databases instead of implementation in (distributed) application code
  - avoid inconsistencies by multiple entry access points
  - improve maintenance

- Enforce end-to-end security for 3-tier and 2-tier applications
  - independant of entry points, usable for:
    - direct logins, APEX, Java/Middletier application servers
  - enforce least-privilege and preserve client identities
Real Application Security

- Fills the gap by utilizing basic techniques like VPD etc. and by adding an enhanced security framework
- Implemented in 12c Enterprise Edition – no options required
  - 32 dictionary tables under SYS named $XS$
  - Views: 30 DBA_XS% and 13 USER_XS%
  - 4 predefined database roles named XS_
  - PL/SQL Packages: 18 XS_% and 17 DBMS_XS_
  - predefined application user and application roles
  - Java Packages oracle.security.xs and oracle.security.xs.ee.session
  - Apex application rasadm and demo scripts for testing
- Adds new concepts
  - application user, application role, application sessions, ACL, security class, security policy
Summary

- Lots of choices for authentication
  - internal and external/central solutions
  - varying degrees of implementation and maintenance complexity
- Lots of authorization models
  - internal and external/central solutions
- Ease of configuration against ease of maintainability
- Careful Design towards central solutions in big environments
Thank you for listening
www.database-consult.de