**Transition from DBMS\_OBFUSCATION\_TOOLKIT to DBMS\_CRYPTO in Oracle 23c**

**Introduction**

Oracle has deprecated DBMS\_OBFUSCATION\_TOOLKIT in favor of DBMS\_CRYPTO due to enhanced security, better cryptographic practices, and compatibility with modern systems. Oracle 23c enforces this transition by restricting or removing certain deprecated features.

**Key Differences**

| **Feature** | **DBMS\_OBFUSCATION\_TOOLKIT** | **DBMS\_CRYPTO** |
| --- | --- | --- |
| Availability | Deprecated in Oracle 23c | Fully supported in Oracle 23c |
| Security Standards | Outdated | Aligns with modern cryptographic standards |
| Hashing Algorithm | MD5 (legacy, insecure for sensitive data) | MD5, SHA1, SHA256, etc. |
| Performance | Limited optimization | Better performance with newer algorithms |
| Recommended Use | No longer recommended | Standard for all cryptographic operations |

**Why Use DBMS\_CRYPTO?**

DBMS\_CRYPTO offers several advantages over its predecessor:

1. **Security**: Supports modern algorithms like SHA256 and AES, which are more secure than legacy MD5.
2. **Performance**: Optimized for modern hardware and database operations.
3. **Future-Proof**: Ensures compatibility with future Oracle versions.

**Example: Transitioning Password Hashing to DBMS\_CRYPTO**

**Updated Function**

sql

Copy

CREATE OR REPLACE FUNCTION HASH\_PASSWORD

(p\_user\_name IN VARCHAR2,

p\_password IN VARCHAR2)

RETURN VARCHAR2

IS

l\_password RAW(255);

l\_salt VARCHAR2(255) := '1996USFGOJN2T3HW89EFGOBN23R5SDFGAKLGIGII';

BEGIN

l\_password := DBMS\_CRYPTO.HASH(

UTL\_I18N.STRING\_TO\_RAW(

p\_password ||

SUBSTR(l\_salt, 4, 14) ||

p\_user\_name ||

SUBSTR(l\_salt, 5, 10),

'AL32UTF8'),

DBMS\_CRYPTO.HASH\_MD5);

RETURN RAWTOHEX(l\_password);

END HASH\_PASSWORD;

/

**Granting Necessary Privileges**

sql

Copy

GRANT EXECUTE ON DBMS\_CRYPTO TO FBL;

**Why This Code Only Works in Oracle 23c**

1. **Modern Standards Enforcement**: Oracle 23c has removed or restricted access to DBMS\_OBFUSCATION\_TOOLKIT to enforce modern cryptographic standards.
2. **Functionality Shift**: The MD5 hashing capability previously provided by DBMS\_OBFUSCATION\_TOOLKIT is no longer available in 23c.
3. **Improved Security Practices**: The database prioritizes SHA256 or higher, aligning with security best practices.

**Legacy Code: Why It Fails in Oracle 23c**

The following legacy code will no longer work in Oracle 23c:

sql

Copy

CREATE OR REPLACE FUNCTION HASH\_PASSWORD

(p\_user\_name IN VARCHAR2,

p\_password IN VARCHAR2)

RETURN VARCHAR2

IS

l\_password VARCHAR2(255);

l\_salt VARCHAR2(255) := '1996USFGOJN2T3HW89EFGOBN23R5SDFGAKLGIGII';

BEGIN

l\_password := UTL\_RAW.CAST\_TO\_RAW(

DBMS\_OBFUSCATION\_TOOLKIT.MD5(

input\_string => p\_password ||

SUBSTR(l\_salt, 4, 14) ||

p\_user\_name ||

SUBSTR(l\_salt, 5, 10)));

RETURN l\_password;

END HASH\_PASSWORD;

/

**Why It Fails**

1. **Deprecation of MD5**: DBMS\_OBFUSCATION\_TOOLKIT.MD5 is deprecated and removed in Oracle 23c.
2. **Legacy Restrictions**: Oracle 23c restricts the use of older, insecure packages to ensure databases comply with modern security standards.

**Steps to Transition**

1. **Replace Deprecated Functions**: Rewrite functions to use DBMS\_CRYPTO instead of DBMS\_OBFUSCATION\_TOOLKIT.
2. **Grant Required Privileges**: Ensure the user executing the new code has privileges:

sql

Copy

GRANT EXECUTE ON DBMS\_CRYPTO TO your\_user;

1. **Test and Validate**: Verify the functionality of updated code in the Oracle 23c environment.
2. **Update Documentation**: Document the changes and ensure team members are aware of the migration to modern standards.

**Conclusion**

Migrating to DBMS\_CRYPTO ensures your application is secure, compliant, and future-proof. While legacy code using DBMS\_OBFUSCATION\_TOOLKIT may work in older versions of Oracle, its deprecation in Oracle 23c mandates transitioning to DBMS\_CRYPTO.