

WHITE PAPER

Why Quantum Computing and Harvest Now, Decrypt Later Attacks Cannot Compromise Data Stored in AsterionDB

An architectural analysis of Oracle Tablespaces, Oracle Securefiles, and the AsterionDB API-first data enclave

Prepared by: Asterion Database Technology Group, LLC

Document Status: For Review and Distribution

EXECUTIVE SUMMARY

The cybersecurity community is increasingly alarmed by two related threats: the eventual arrival of cryptographically relevant quantum computers (CRQCs) capable of breaking today's public-key encryption, and the Harvest Now, Decrypt Later (HNDL) strategy in which nation-state adversaries are collecting encrypted data today to decrypt once quantum capability matures.

These are legitimate concerns — but they apply unevenly depending on how data is stored. For unstructured data stored in conventional file systems or object storage, the threat is real and urgent. For unstructured data stored inside an AsterionDB/Oracle database system, the threat is structurally neutralized at the architecture level, independent of encryption algorithm selection.

Central Finding

An adversary who successfully uses a quantum computer to decrypt Oracle tablespace files gains nothing of intelligence value. They cannot read the data without running the Oracle database engine — which immediately reinstates the full AsterionDB security enclave. The quantum computer returns the attacker to exactly the problem they started with: they need a database administrator password.

This white paper explains the technical basis for this conclusion, describes the four successive barriers an attacker must defeat, and identifies the specific operational conditions required to maintain this protection.

The Nature of Harvest Now, Decrypt Later

Modern encryption algorithms — particularly RSA, Elliptic Curve Cryptography (ECC), and Diffie-Hellman key exchange — derive their security from mathematical problems that classical computers cannot solve efficiently. A sufficiently powerful quantum computer, using Shor's Algorithm, could solve these problems and break the encryption.

Because quantum computers capable of doing this do not yet exist, an adversary's only current option is to collect encrypted data now and store it until decryption becomes feasible. This is the HNDL strategy. Organizations with data that will retain intelligence value for 10-20 years — the timeframe most often cited for CRQC development — face a concrete, present-day risk.

Why Unstructured Data in File Systems Is Particularly Vulnerable

The HNDL attack is especially effective against unstructured data stored in conventional file systems or object storage because the attack chain from interception to exploitation is short:

- A network-layer capture or storage exfiltration event yields encrypted files.
- The adversary stores those files until quantum decryption is available.
- Upon decryption, each file is immediately readable by a standard application: a PDF opens in Acrobat, an image renders in a viewer, a DICOM file opens in medical imaging software.
- Intelligence exploitation follows directly.

This vulnerability applies to text files, images, PDF documents, spreadsheets, DICOM files, AI model weights, and virtually any other data format stored outside a database. The key characteristic: once decrypted, the data is directly usable without additional infrastructure.

SECTION 2 — HOW ORACLE DATABASES STORE UNSTRUCTURED DATA

Oracle Tablespaces: Data Without File Boundaries

Oracle databases store all of their data — structured and unstructured — within tablespaces. A tablespace is a logical storage unit comprised of one or more physical files on the operating system's file system. Oracle manages the internal layout of these files entirely; no data element has an independently accessible file representation.

Concrete Scale Example

A 2TB production database may consist of sixteen 125GB tablespace files. Within those sixteen files, over two million images may be stored — but not as two million image files. They exist as database objects interleaved across Oracle's internal data pages. There is no file manifest, no directory, no recoverable image boundary without the database engine.

Oracle Securefiles: Eliminating the File System as a Vulnerability

Before Oracle Securefiles, unstructured data (BLOBs) had to be stored as BFILEs — which actually placed the binary content in a file on the operating system's file system, with only a pointer stored in the database. This created exactly the exposure that HNDL exploits: a discrete, locatable, directly-decryptable file.

Oracle Securefiles eliminates this exposure. Unstructured data is stored entirely within tablespace files, under the complete control of the Oracle storage engine. The data has no independent file-system existence. It cannot be located, isolated, or extracted without the database engine's participation.

The System Change Number (SCN) Consistency Requirement

Oracle maintains internal consistency across tablespace files using System Change Numbers (SCNs) — monotonically increasing transaction counters. For a database to open and operate, all tablespace files must be consistent with one another at the same SCN. If an attacker captures tablespace files at different moments, the files will not be in sync, and Oracle will refuse to open them.

On a live production system, capturing all tablespace files in a single atomic, synchronized snapshot without shutting down the database is not feasible through an external, covert channel.

SECTION 3 — HOW ASTERIONDB ENHANCES THE ORACLE SECURITY POSTURE

API-First Architecture: One Access Pathway, Fully Audited

AsterionDB's fundamental design principle is that there are exactly two ways to access data in the database:

Pathway 1

Traverse audited, authenticated business logic within the database via the AsterionDB API layer. All access is logged, governed, and subject to role-based controls.

Pathway 2

Connect to the database as a DBA (Database Administrator) with direct credential access. This is a conventional privileged-identity security problem — not a cryptographic one.

The AsterionDB API layer — built on AsterionDB's DbTwig technology — ensures that no process can issue a raw database query or directly access stored content without first satisfying the authentication and authorization requirements of the business logic tier. This eliminates entire classes of injection, bypass, and privilege-escalation attacks that target exposed data layers.

Oracle RAC and ASM: Additional Structural Resilience

In deployments using Oracle Real Application Clusters (RAC) or Automatic Storage Management (ASM), the storage architecture becomes even more resistant to exfiltration. ASM manages storage as raw disk groups rather than conventional file system files. The ASM kernel module — running as a separate Oracle process — mediates all I/O. An attacker attempting to copy ASM disk group contents would need to defeat both the ASM metadata layer and the multi-node coordination of a RAC cluster, in addition to all the barriers described above.

SECTION 4 — THE FOUR BARRIERS: WHY QUANTUM DECRYPTION FAILS

The following diagram traces the attack path an adversary must navigate to extract intelligence from an AsterionDB/Oracle system using HNDL. Each barrier is independently formidable; together, they make the attack path non-viable.

Barrier 1

Tablespace File Acquisition

The attacker must obtain ALL tablespace files, each perfectly synchronized at the same Oracle System Change Number. While the database is running, these files are open and OS-locked; any external access attempt triggers monitoring alerts. Covert exfiltration of a live multi-file database is operationally implausible in a professionally managed environment.



If Barrier 1 is defeated: attacker has all synchronized tablespace files

Barrier 2

PQC Decryption (The HNDL Bet)

	Tablespace files may not be encrypted at all — in which case decryption is moot and the attacker has binary data in Oracle's internal page format. If Oracle TDE is active, decryption produces raw binary page data — not readable files. There are no file headers, no MIME types, no image boundaries. The decrypted output is Oracle's internal block format, not recoverable data.
↓	<i>If Barrier 2 is defeated: attacker has decrypted binary tablespace data</i>
Barrier 3	Database Engine Required Oracle's binary page format cannot be parsed into usable data without Oracle itself. BLOB objects in Securefiles have no file boundaries in the raw binary tablespace files — they are LOB segments interleaved across data blocks. Oracle-format forensic tools exist, but reliably reconstructing Securefiles LOBs from offline binary pages without a running instance requires Oracle internal expertise and is not a routine capability even for sophisticated nation-state actors.
↓	<i>If Barrier 3 is defeated: attacker has a running Oracle instance with the exfiltrated data</i>
Barrier 4	AsterionDB Access Control and Audit Layer Once Oracle is running, the attacker is fully inside the AsterionDB security enclave. All data access traverses the API-first business logic layer with authentication, role-based authorization, and audit logging. Bypassing this layer requires DBA credentials — a conventional privileged-identity security problem entirely unrelated to quantum computing.

The Logical Conclusion

Defeating all four barriers returns the attacker to exactly the access problem they faced at the start: they need a DBA password. The quantum computer — and the entire HNDL operation — accomplishes nothing. A rational adversary would abandon the tablespace exfiltration approach in favor of credential compromise, insider threat, or application-layer attacks. These are conventional security problems with established and understood defenses.

SECTION 5 — THREAT COMPARISON: FILE SYSTEM vs. ASTERIONDB/ORACLE

The following table illustrates how the HNDL threat applies differently to conventionally stored unstructured data versus data stored in an AsterionDB/Oracle system.

Attack Scenario	File System / Object Storage	AsterionDB / Oracle
Attacker intercepts encrypted data in transit	Encrypted file captured; stored for later decryption	Encrypted network traffic yields no locatable data object
Attacker exfiltrates storage	Individual encrypted files copied silently	Must capture all tablespace files in a synchronized snapshot while OS-locked
Quantum decryption succeeds	Data immediately readable by standard application	Decrypted binary is Oracle internal page format — unreadable without a running Oracle database
Attacker runs Oracle on exfiltrated data	N/A — data already accessible	Oracle reinstates AsterionDB API enclave; DBA credentials required in order to bypass
HNDL attack effectiveness	HIGH — short chain from ciphertext to intelligence	NEGLIGIBLE — quantum investment returns attacker to credential problem

SECTION 6 — CONDITIONS REQUIRED TO MAINTAIN THIS PROTECTION

AsterionDB's structural HNDL immunity is not unconditional. Organizations must ensure the following operational posture is maintained for the protections described in this paper to apply fully.

Condition 1: Oracle TDE Should Be Active

While the architecture is resistant to HNDL even without encryption — because the binary page format is not directly exploitable — enabling Oracle Transparent Data Encryption (TDE) eliminates the theoretical risk from advanced forensic tools capable of parsing Oracle block format offline. Organizations handling classified or long-sensitivity-life data should treat TDE activation as a baseline requirement, not an optional enhancement. The encryption algorithm selected for TDE should comply with current CNSA 2.0 standards.

Condition 2: Backups and DR Assets Must Be Managed With Equal Rigor

Oracle Recovery Manager (RMAN) backup sets, Data Pump export files (expdp), and cold standby copies represent offline artifacts that may be encrypted with different keys or at different levels than the live database. These artifacts are the most likely HNDL target, as they are typically stored on more accessible media than a live production database. Backup encryption must use FIPS 140-2 or 140-3 validated algorithms, and backup tapes or cloud storage must be protected with access controls equal to or exceeding those on the live database.

Condition 3: The AsterionDB API Layer Must Be Hardened

Adversaries who accept that the tablespace exfiltration path is infeasible will pivot to application-layer attacks against the AsterionDB API surface: SQL injection through DbTwig, authentication bypass, JWT manipulation, and database role privilege escalation. The API-first architecture concentrates and audits access, which is an advantage — but it also concentrates the attack surface. Standard application security practices (input validation, parameterized queries, JWT signature verification, least-privilege role assignment, and continuous audit log review) must be maintained.

Condition 4: DBA Credential Security Is the Primary Residual Risk

As established in this paper, the rational adversary path through an AsterionDB/Oracle environment leads ultimately to DBA credential compromise. Organizations must treat DBA credentials with the same rigor applied to the data itself: hardware security modules (HSMs) for credential storage, privileged access workstations (PAWs), session recording, and anomaly detection on all DBA-level database activity.

SECTION 7 — CONCLUSION

AsterionDB's architectural approach to unstructured data storage — placing data inside Oracle Securefiles within Oracle Tablespaces, accessed exclusively through an API-first, audited business logic layer — creates a structural defense against quantum computing and HNDL attacks that is independent of encryption algorithm selection.

The argument is not that the system is invulnerable. It is that the quantum/HNDL threat vector — the one creating significant concern across the security community — is structurally non-viable against this architecture. Decrypted tablespace files cannot be read without Oracle. Oracle reinstates the AsterionDB access enclave. The quantum investment yields nothing.

For organizations evaluating their post-quantum data security posture, the question should not be limited to "what encryption algorithm are we using?" It should also include "where is our unstructured data stored, and what does an attacker actually get if they decrypt our storage?" For AsterionDB customers, the answer to that second question is: they get a very large collection of Oracle binary pages they cannot read.

Key Takeaways

- Unstructured data in AsterionDB/Oracle has no file-system existence — there is nothing to harvest.
- Oracle tablespace files cannot be covertly exfiltrated from a running, professionally managed production system.
- Even if exfiltrated and decrypted by a quantum computer, tablespace binary data is not directly readable — Oracle must run to extract it.
- Running Oracle reinstates the full AsterionDB API-first security enclave, requiring DBA credentials to access data to bypass.
- The rational adversary path bypasses quantum entirely and targets credential compromise — a problem with well-understood conventional defenses.
- Full protection requires Oracle TDE activation, rigorous backup security, API layer hardening, and DBA credential controls.

About AsterionDB

AsterionDB is an Oracle-native data management platform built on a No File System (NoFS) architecture and API-first design philosophy. AsterionDB's five issued patents underpin a platform that has been briefed to DISA and CISA and holds credentials through Oracle's IL5/IL6-compatible infrastructure. AsterionDB is designed for organizations requiring a zero-trust data layer, post-quantum data architecture resilience, and CMMC/NERC CIP-015-1 compliance readiness.