

CORE BANKING SYSTEMS SECURITY.

Every mobile app, ATM, and card swipe ultimately reads from and writes to a core banking system that, at much of the industry, still runs on decades-old COBOL and mainframe technology. **Modern APIs increasingly wrap that core, but wrapping it isn't replacing it — the legacy platform's risk profile doesn't disappear.**

- SYSTEM OF RECORD
- LEGACY MAINFRAME CORE
- PARALLEL-RUN TESTING
- VENDOR CONCENTRATION

01 THE MECHANISM



The System of Record for Every Dollar

The core banking system is the authoritative ledger of account balances, deposits, and transactions; **every other banking application ultimately reads from and writes to it.**



Decades-Old Technology, Still in Production

Much of the industry's core banking logic still runs on COBOL and mainframe systems built decades ago — **live infrastructure processing real transactions today**, not a museum piece.



Wrapped, Not Replaced

Because migrating the core is one of the highest-risk events a bank can undertake, most modernization builds **modern APIs around the legacy core rather than replacing the ledger itself.**

02 THE FLOW

The Governed Core Environment

CONTAINED

1

Direct, privileged access to the core banking system is tightly restricted, logged in detail, and requires independent approval — **no single administrator can alter a ledger entry unilaterally.**

2

Every change to core logic runs through a rigorous parallel-run process, **comparing new and old output side by side** before cutover.

3

The API layer wrapping the legacy core is treated as its own attack surface, with **dedicated security testing separate from the mainframe itself.**

The Unmanaged Legacy Core

EXPOSED

1

A handful of administrators hold broad, loosely audited access to the core banking system, **inherited from years of accumulated privilege** with no formal review.

2

A vulnerability in the modern API layer **provides a path to core-level functions** the mainframe itself was never directly exposed to.

3

A rushed change to core logic skips proper parallel-run validation, and a **subtle calculation error propagates** through every downstream system before anyone notices.

03 CORE PILLARS & COVERAGE

PILLAR	WHAT IT COVERS
Privileged Access Control	Tightly restricted, independently approved, and comprehensively logged access to direct core banking functions and the underlying ledger.
Change Management & Parallel Testing	Rigorous, slow-moving change control that runs new logic alongside the old and compares outputs before any cutover.
API/Wrapper Layer Security	Treating the modern interfaces built around the legacy core as their own attack surface, with dedicated security testing independent of the core itself.
Vendor & Third-Party Dependency Management	Managing the security posture of the small number of major vendors most banks license their core banking platform from.

04 DETECTION APPROACHES COMPARED

APPROACH	SUFFICIENT ALONE?	WHAT IT MISSES
Securing the API Layer While Ignoring the Core	No	A modern, well-tested API wrapper does nothing to address weak access controls or unmanaged privilege on the legacy system behind it.
Deploying Core Logic Changes Without Parallel-Run Validation	No	Errors in core banking logic are often invisible until real money is affected, which is exactly why pre-cutover comparison testing exists.
Broad, Unreviewed Administrator Access to the Core	No	Direct core-level access can alter ledger entries in ways that bypass every application-layer control built on top of it.
Treating the Core Banking Vendor's Security as Someone Else's Problem	No	The bank's own security posture is only as strong as the vendor's patching cadence and development practices for a system this central.
Restricted Core Access + Parallel-Run Change Control + Secured API Layer + Vendor Oversight	Yes	Addresses the legacy system's actual risk profile instead of assuming a modern interface makes the underlying core safe by association.

CORE BANKING SYSTEMS SECURITY.

RESTRICT · VALIDATE · MONITOR

05

PROGRAM OPERATIONS

The core banking system is the one piece of infrastructure where an error isn't just a bug — it's a discrepancy in who owns how much money, with direct regulatory and legal exposure.

Most banks don't build their own core banking platform; they license it from one of a handful of major vendors, making vendor security posture a direct dependency, not a peripheral concern.

A well-secured API layer can still become the new largest attack surface, since it's what's actually reachable from the internet even when the mainframe itself never was.

Parallel-run testing exists because core banking errors are often silent until they've already affected real account balances.

06

CONTROL MAPPING

FFIEC AIO BOOKLET

Architecture, Infrastructure, and Operations — supervisory expectations for how depository institutions manage core technology infrastructure risk.

FFIEC DAM BOOKLET

Development, Acquisition, and Maintenance — examiner guidance covering changes to and vendor sourcing of core banking systems.

NIST SP 800-53

Security and privacy controls catalog commonly referenced for the access control and change management expectations applied to core systems.

OCC THIRD-PARTY GUIDANCE

Risk management expectations for third-party relationships, directly relevant to the small number of vendors most core banking platforms depend on.

07

WHAT THIS DOES NOT SOLVE

Full Modernization Risk

Wrapping a legacy core in modern APIs reduces some exposure, but **the underlying platform's age and technical debt remain**, along with a shrinking pool of specialists.

Vendor Incident Response Beyond the Bank's Control

A security incident inside the core banking vendor's own environment is **largely outside the bank's direct visibility**, despite the stakes.

The Cost of a Full Migration

Recognizing the core's risk doesn't make a full replacement affordable or safe on a normal timeline; the **tradeoff between legacy risk and migration risk doesn't disappear**.

Business Logic Nobody Fully Documented

Decades of accumulated special-case rules embedded in core logic **can't be fully secured against what isn't understood**, only carefully tested around.

08

MYTHS VS. REALITY

MYTH

"Core banking systems on old technology are basically retired."

REALITY

COBOL and mainframe-based core banking platforms are **live production infrastructure processing real transactions** at a large share of major institutions today.

MYTH

"Wrapping a legacy core in modern APIs means the core is modernized."

REALITY

The ledger and its underlying logic remain the original legacy platform; **the API layer changes what's exposed, not what's running underneath**.

MYTH

"A well-tested API layer is the hardest part to secure."

REALITY

The API layer is often the new largest attack surface precisely because it's internet-reachable, while the **legacy core behind it may have weaker native access controls**.

MYTH

"Core banking vendor security is the vendor's problem, not the bank's."

REALITY

The bank's own security posture is **directly dependent on that vendor's patching cadence** for the single most consequential system it operates.

MYTH

"If a core banking change passes basic testing, it's safe to deploy."

REALITY

Parallel-run validation comparing new and old output exists specifically because **core banking errors are often invisible** until they've affected real balances.

09

SPEAK THE PROGRAM — KEY TERMS

CORE BANKING SYSTEM

The central platform maintaining the authoritative ledger of account balances and transactions that **every other banking application ultimately depends on**.

SYSTEM OF RECORD

The single authoritative data source for a category of information; for banking, the **core system's account and transaction data**.

PARALLEL RUN

Running new core banking logic alongside the existing system and comparing outputs before cutover, to **catch discrepancies before they affect real accounts**.

BATCH PROCESSING WINDOW

Scheduled overnight processing many core banking systems still rely on for **interest calculation, settlement, and statement generation**.

API/WRAPPER LAYER

The modern interface layer built around a legacy core to expose its functions to digital banking apps **without replacing the underlying ledger**.

VENDOR CONCENTRATION RISK

The security and operational dependency created when a large share of an industry **relies on a small number of critical platform vendors**.