MWC 2026 Q1-Q2 Review

AI Audit

This first half of the year we undertook a substantial project to use AI to audit the code base of MWC and find potential vulnerabilities and areas for improvement. The results from the audit helped us harden the node and wallet.

No consensus algorithms were changed; the work was just a broad defensive-hardening pass across the codebase. The specific areas addressed were:

- **Arithmetic safety.** Wherever overflow can't be ruled out at compile time, the code now uses checked or saturating arithmetic, with errors propagated up rather than silently wrapping.
- **Type conversions.** Conversions are now checked (with errors propagated), and the number of conversions is being reduced overall to clean up type inconsistencies.
- **I/O error handling.** Read/write operations that can fail now propagate their errors. As a consequence, several interfaces that previously couldn't fail — hashing, for example — now return fallible results.
- **Secret zeroization.** Secrets, nonces, and similar sensitive values are zeroized after use so that unclean copies aren't left on the heap, where they could in principle be recovered via swap files or other memory allocations.
- **Input validation.** Primary API functions now validate supplied data — public keys, commitments, and anything else that can be checked — to defend against data-manipulation attacks.
- **Dependent-data validation.** Beyond validating individual values, the code now verifies relationships between them — e.g. confirming that a rangeproof was actually generated for the commitment it's associated with.

- **secp edge cases.** Commitment values are represented internally as secrets, and a secret cannot be zero. That constraint surfaced a number of issues — including, with the current design, an inability to safely compute sums across separate threads.
- **PMMR data consistency.** The protocol here was previously very loose; it has been tightened substantially. Since the node still starts and runs correctly afterward, this is considered a net improvement.
- **Storage safety.** Three sub-areas:
    - Access controls. File-system access controls have been added.
    - Failure handling. The backend has no transaction support, so a multi-step update (e.g. add header → add block → add block sum) could fail partway through, leaving an inconsistent state. Locks and rollback points are now being used to make these updates effectively transactional.
    - Deadlock handling. Because locks were introduced, deadlocks are now possible, and each potential case is being addressed individually.

Node Updates

Arti Migration

The MWC Node was upgraded to Arti, the rust-native, embedded implementation of Tor, early this year, moving away from the older external Tor binary approach. This involved refactoring listeners so Tor connection logic could be reused inside the wallet, adding lifecycle tracking so Arti objects shut down and reboot cleanly, and switching the restart mechanism from flag-based to version-based. The Arti migration was part of a larger effort to support libraries and a more seamless wallet/node integration.

MWC Wallet Updates

The MWC Wallet was updated in tandem with the node. Updates included ownership-proof APIs (for viewing keys, slatepack/Tor addresses, and MWCMQS addresses), QT wallet integration hooks, a "reverted" transaction/output status to properly represent reorg-affected transactions, a secp library upgrade, NRD kernel support via slates, and optional Tor obfs4/Snowflake bridge support for censorship circumvention.

QT Wallet Updates

This year we continued improvements on the QT Wallet with UI updates to improve the usability and performance. We also streamlined the Slatepack transaction flow to make it easier, faster and more intuitive for users to send and receive interactive transactions.

Mining & App Updates

Mining Pools

A new mining pool was launched this year, providing additional redundancy to the MWC ecosystem. The pool is at pool.mwcmonitor.com.

MWC Monitor App

The MWC Monitor App was launched this year on web, iOS and Android. MWC Monitor provides real-time data from the MWC network including node data, transaction data, fork monitoring, and pool monitoring. It also contains a supply verifier for cryptographically verifying the total supply at any time.

Conclusion

We continue to harden the protocol and add tools and products for the ecosystem. Stay tuned for more updates in 2H 2026!