Issue	Auditor	Severity	Description & Notes	Fix PR / commit	Status	Duplicate of	Verdict
C01 #2 Validator signature with zero timestamp can always be replayed	Three Sigma	Critical Medium	No check for timestamp != 0 in `updateCollateral`	PR: https://github.com/MZero-Labs/protocol/pull/114 Commit in main: https://github.com/MZero-Labs/protocol/commit/b2c421c132cf6af6a18860ad17285b900be83163	Resolved	https://gist.github.com/0xmonsoon/447ab8db4e80bbbbab2b93d25f1b2a6d	Fixed. However, the recommendation to ignore signatures with a timestamp below `_minterStates[minter_].updateTimestamp` was not implemented.
H01 #25 MToken's total principal invariant can be broken	Three Sigma	High Medium ?	unchecked total earning principal addition in `MToken._addEarningAmount`	PR: https://github.com/MZero-Labs/protocol/pull/126 Commit in main: https://github.com/MZero-Labs/protocol/commit/74523a8e77be8654b902baaca41201d135c72190 Since this PR has been superseded, probably better to show the diff to later version of main	Resolved	https://github.com/threesigmaxyz/mzero-labs-8-1-2024-issues-external/issues/23	Fixed with the merge of PR 151.
H02 #21 Lack of deadline in PowerToken.buy can lead to user's cashToken being distributed through ZeroToken holders	Three Sigma	High Low	Add expiration time for POWER buy order	PR: https://github.com/MZero-Labs/ttg/pull/216 Commit in main: https://github.com/MZero-Labs/ttg/commit/3fb74e72f03d45e7ec56f5c420143bcb627ac206	Resolved		Fixed by checking against a new `expiryEpoch_` parameter.
M01 #27 Creating a new proposal in StandardGovernor may reach a state of permanent DoS	Three Sigma	Medium Low	Power target supply overflow, missing cast	PR: https://github.com/MZero-Labs/ttg/pull/217 Commit in main: https://github.com/MZero-Labs/ttg/commit/c2131a5fb1dc1d8f4eb12874165016b90885db2d	Resolved		Fixed by upcasting `_targetSupply`.

M03 #1 A sufficiently large collateral may break the maximum owed M calculation	Three Sigma	Medium Low	Multiplication within unchecked block can lead to overflow	PR: https://github.com/MZero-Labs/protocol/pull/117 Commit in main: https://github.com/MZero-Labs/protocol/commit/06b6cb1593da5baffa60d50fb75a0767dd754a6b	Resolved		Fixed; however, the comments look wrong: it should be 650_000.
C01 Attacker can double it's PowerToken balance and voting power every time Reset event occurs	Certora	Critical	Double bootstrap was happening in self-delegation or self-transfer Delegator == delegatee Sender == recipient	PR: https://github.com/MZero-Labs/ttg/pull/215 Commit in main: https://github.com/MZero-Labs/ttg/commit/c16400216827b6a6a5485823c2c283c35cb49e75	Resolved		Fixed -- instances where the same user was being bootstrapped twice have been removed.
H01 Past Voting Power isn't read from bootstrap after sync	Certora	High		Multiple Prs	Resolved		Fixed: now, in the case where bootstrap epoch is passed but account hasn't synced yet, will get bootstrap balance for voting power.
M01 Code Restricts Execution of Proposal to 1Epoch	Chainsecurity	Medium Info, inconsistency between papers	The function StandardGovernor.execute() tries to execute all proposals voted in the last two epochs. However, the function StandardGovernor.state() returns the status Succeeded only for proposals voted in the previous epoch. The state Expired is returned for older proposals, hence stopping them from being executed. This behavior conflicts the whitepaper and the code comments in function execute() which state that a successful proposal can be executed during the next 2 epochs: // Proposals have voteStart=N and voteEnd=N, and can be executed only during epochs N+1	PR: https://github.com/MZero-Labs/ttg/pull/206 Commit in main: https://github.com/MZero-Labs/ttg/commit/22bd53c50b4224217a4548d85c9ac8615f49f31f	Resolved		Fixed by explicitly restricting the execution to one epoch.

			and N+2.				
M02 EIP-712 Dynamic Types	Chainsecurity	Medium Medium?	The EIP-712 is not fully compliant with the standard. It must encode dynamic types	PRs: https://github.com/MZero-Labs/protocol/pull/120 https://github.com/MZero-Labs/ttg/pull/208 Commits in main: https://github.com/MZero-Labs/protocol/commit/287efbd7d4c8ba0fbbc14f1dae71c494d6875b70 https://github.com/MZero-Labs/ttg/commit/5d6ba760fc764617f3997e7db060ea06e79ff476	Resolved		Fixed.
C01 Any action that moves delegation to address(0) will cause that user's funds to be locked.	Prototech	Critical	Delegation to 0x0 leads to inability to re-delgate or transfer tokens	PR: https://github.com/MZero-Labs/ttg/pull/214 Commit in main: https://github.com/MZero-Labs/ttg/commit/27c751f6177850751053c011b3a0327896db3e44	Resolved		Fixed by adding voting power to delegator when `newDelegatee_` is zero.
C03 PowerToken: Delegation and transfer fails when actor.balance > actor.votes	Prototech	Critical	Identified a scenario where actor balance > actor votes. In this scenario, delegation and transfer above the vote amount but within the available balance fails with an overflow. **Resolution:** This issue happens when actor.balance > delegatee.votes not when actor.balance > actor.votes Duplicate. This is also caused by the same bug as Prototech C01.	PR: https://github.com/MZero-Labs/ttg/pull/214 Commit in main: https://github.com/MZero-Labs/ttg/commit/27c751f6177850751053c011b3a0327896db3e44	Resolved		Fixed. Duplicate of Prototech C01.

H01 ERC3009 validAfter and validBefore are incorrectly implemented as inclusive	Prototech	High	ERC3009 authorizations must not be valid *at* timestamp and must be non inclusive	PR: https://github.com/MZero-Labs/common/pull/13 Commit in main: https://github.com/MZero-Labs/common/commit/ed02a2c94bb22df03b93fe397e73caa2aef5d955	Resolved		Fixed.
H02 MToken.mint() can overflow totalNonEarningSupply and principalOfTotalEarningSupply	Prototech	High	https://github.com/MZero-Labs/protocol/blob/3499f50ff3382729f3e59565b19386ba61ef8e36/src/MToken.sol#L217	Multiple prs Properly resolved here: https://github.com/MZero-Labs/protocol/pull/143/files#diff-6d16a12288164b2eb7971f1325b337cb7ab8909b0fb939f78524943ee2f93d2bR205-R226	Resolved		Fixed. Duplicate of Three Sigma H01.
H03 PowerToken: Inflation rounding creates deviation in account balances and total supply.	Prototech	High	Several regressions were discovered that indicate that the total sum of user balances after inflation do not equal the total supply. This is likely due to the necessity of rounding down on the 10% epoch inflation. Example regressions: PASS: test_regression_invariant_P_B1_4d72c83e_failure() test_regression_invariant_P_B1_bc6627bc_failure() PASS test_regression_invariant_P_B1_b				Unknown.

			a29ad51_failure() 				
M01 High Mint Ratio and High Collateral can cause Uint112 overflow in UpdateCollateral	Prototech	Medium	Cap on Mint ratio	PR: https://github.com/MZero-Labs/protocol/pull/146 Commit in main: https://github.com/MZero-Labs/protocol/commit/f24d34a77d5c8082529df40f4c2f90587025f150	Resolved	Independent L01	Fixed.
M03 resetToTokenHolders() functions will brick the new vote token if the bootstrap token's pastTotalSupply(epoch) returns 0	Prototech	Medium	Check for bootstrap totalSupply != 0	PR: https://github.com/MZero-Labs/ttg/pull/236 Commit in main: https://github.com/MZero-Labs/ttg/commit/62290374c54d752a422ba559be52be64aea9c91a	Resolved		Fixed.
M04 PowerToken: Account balances can exceed total supply.	Prototech	Medium	It appears that `markParticipation` can cause user balances to exceed `totalSupply`. The following regression shows `Actor7` receives an extra `100` tokens (or 10%) after self delegating and getting a `markParticipation` call. It was not clear to us exactly where this happens or why and would strongly recommend further investigation. test_regression_invariant_P_B1_5cd1d968_failure() 	test_regression_invariant_P_B1_5cd1d968_failure() Issue: C04 Account balances can exceed total supply Duplicate. Caused by the same bug as Prototech C01. Tests fails at line: https://gist.github.com/brianmcmichael/95a09be043d82d88a027b777dceb47e1#file-gistfile1-txt-L11 Call sequence lines: https://gist.github.com/brianmcmichael/95a09be043d82d88a027b777dceb47e1#file-gistfile1-txt-L716-L724 When markParticipation is called for Actor 7, the total supply is not increased as the voting power of	Resolved		Fixed. Prototech C01

				Actor 7 is zero (because voting power has been given to address zero). But the balanceOf function, add unrealised inflation to the balance of Actor 7			
M05 Invariant P_VD2 failure: Actor votes do not match delegated balance.	Prototech	Medium	test_regression_invariant_P_VD2_dc8c60c1_failure In this regression, we arrive at a state where the actor has 1000 tokens delegated to them but 0 voting power in the Voting Epoch.	test_regression_invariant_P_VD2_dc8c60c1_failure() Issue: M05 Actor votes do not match delegated balance. Duplicate. This is also caused by the same bug as Prototech C01. Tests fails at the second last line: _powerTokenHandler.delegate(2, 151614098927087407634265825876666786217000614717236199529083); Call sequence lines: https://gist.github.com/brianmcmichael/be5e9703d91fc94c99ca1774e3a84084#file-test_regression_invariant_p_vd2_dc8c60c1_failure-L1299-L1306	Resolved		Fixed. Prototech C01.
H01 - Validator signatures can be double counted	OpenZeppelin	High	Issue introduced while gas optimizing. Signature order check was bypassed if an unapproved validator sig was passed to the function.	PR: https://github.com/MZero-Labs/protocol/pull/136 Commit in main: https://github.com/MZero-Labs/protocol/commit/5e8a69d1a5b4af2c71c12da8b0b9a352f65a1ce4	Resolved		Fixed.
M01 - Missing Approval event in permit implementation	Kirill	Medium	*Approval* event is not emitted in *permit()*.	PR: https://github.com/MZero-Labs/common/pull/18 Commit in main: https://github.com/MZero-Labs/common/commit/81e204cb69ed8189b11102e5a661d63d55ef4858	Resolved		Fixed.