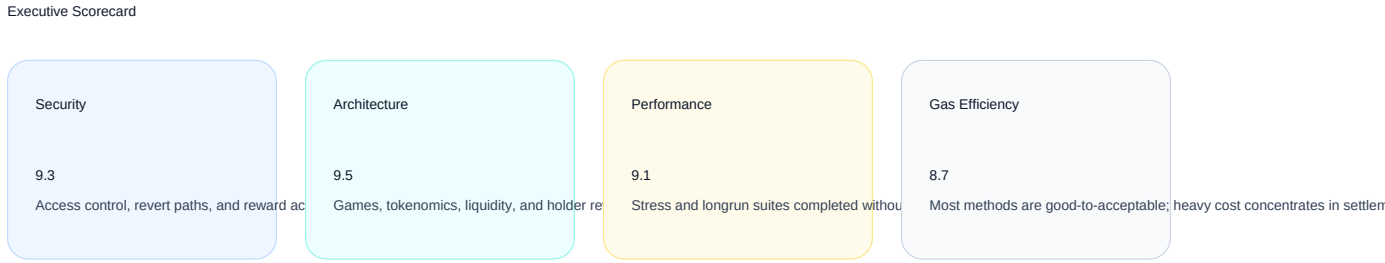
George Ecosystem Smart Contract Audit Report

Expanded investor-facing technical report built from supplied Hardhat test outputs, stress runs, and gas profiles. Styling is modeled after a polished institutional audit memo, while conclusions remain scoped only to the evidence provided.



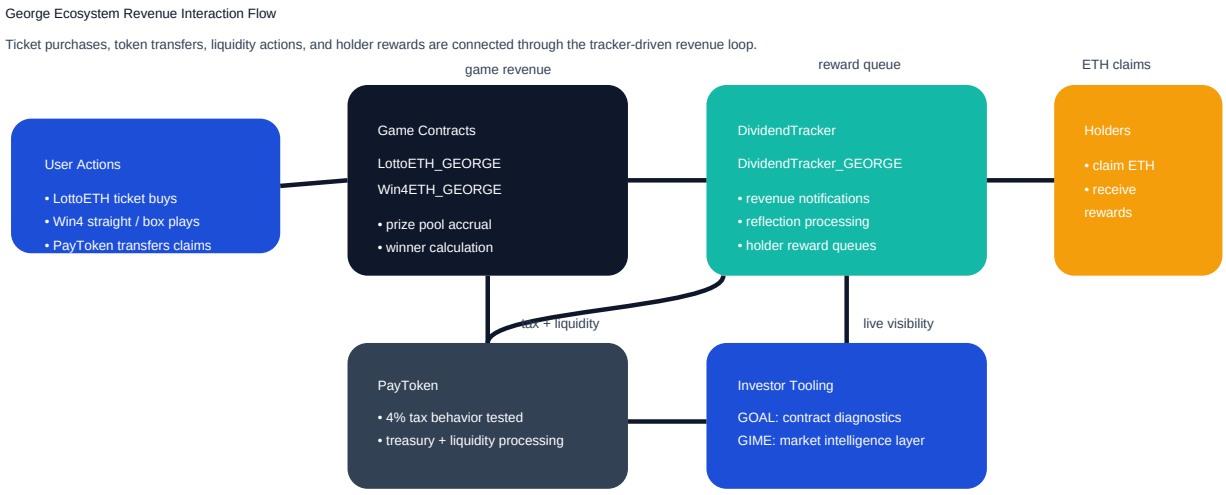
Scope LottoETH_GEORGE, Win4ETH_GEORGE, PayToken_GEORGE, DividendTracker_GEORGE	**Compiler** Solidity 0.8.26 Optimizer enabled, 200 runs
Evidence used 118 passing tests, longrun suite, stress suites, deployment gas table, method gas table	**Links** georgefoc.com goal.georgefoc.com gime.georgefoc.com x.com/GEORGE_FOC t.me/georgeFOC_PAY

Important scope note. This document is an evidence-backed presentation of the supplied tests and gas traces. It is **not** a substitute for a fresh, line-by-line manual code audit. Findings below are constrained to the observed behavior in the uploaded outputs.

1. Executive Summary

Across the supplied evidence, the George Ecosystem behaved as an integrated on-chain system rather than a collection of isolated contracts. LottoETH and Win4ETH forwarded revenue into the tracker, PayToken executed treasury/liquidity and reflection paths, and the holder reward pipeline remained alive across longrun and stress conditions.

- All 118 tests in the provided consolidated output passed.
- The largest gas pressure sits in expected settlement-heavy paths such as **processWinnersBatch**, **purchaseBox**, and **calculateWinners**.
- The operational security model appears strong within tested scope: unauthorized actions revert, underfunded prize scenarios revert, reward claims block double collection, and swap failures fail closed rather than silently continuing.
- The architecture is investor-friendly because on-chain gameplay, token tax flow, liquidity support, and holder rewards all reinforce one another and can be surfaced by GOAL and GIME as live diagnostics.



Passing test coverage by suite

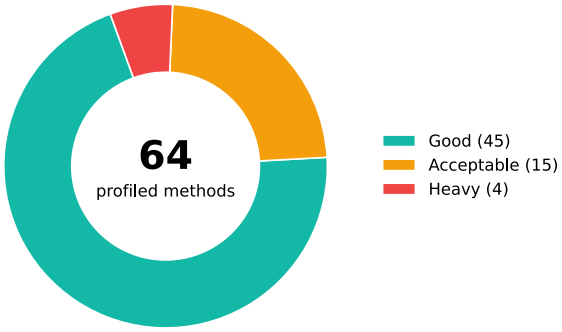
Suite	Passing tests	Interpretation
Integration: LottoETH_GEORGE -> DividendTracker_GEORGE	7	Validates revenue forwarding and tracker accumulation.
LottoETH_GEORGE	21	Lottery lifecycle, payout logic, revert paths, and rollover behavior.
PayToken_GEORGE	41	Tokenomics, tax, liquidity, reward distribution, exclusions, and admin paths.
George Ecosystem Longrun	3	Sustained combined rounds across games.
George Ecosystem Stress	3	Mixed high-load actions across many holders.
LottoETH_GEORGE Stress	3	Lotto-specific scale behavior.
Win4ETH_GEORGE Stress	3	Win4-specific scale behavior.
Win4ETH_GEORGE	25	Win4 game rules, pricing, claims, and rollover behavior.

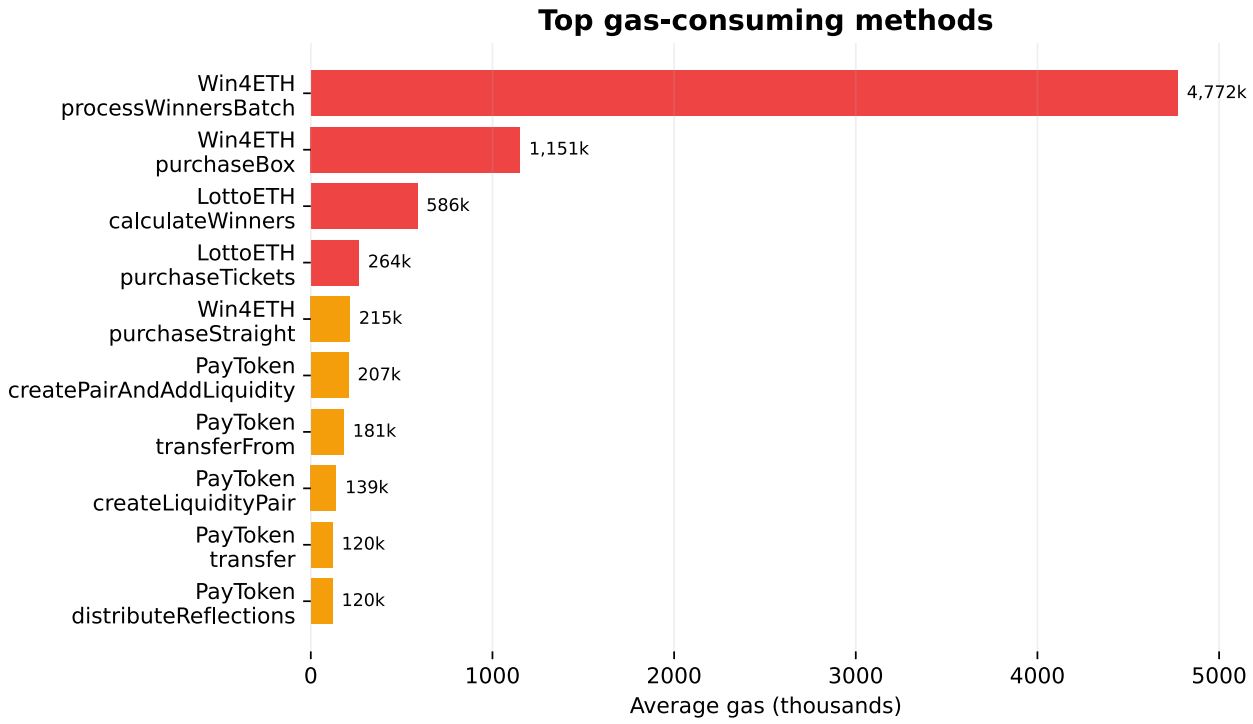
2. Gas Analysis & Transaction Efficiency

Gas data was grouped into three investor-readable buckets: **Good** for average cost below 80k gas, **Acceptable** for 80k–250k gas, and **Heavy** above 250k gas. This makes it easy to separate ordinary user actions from batch settlement logic.

Grade	Methods	What it means
Good	45	Lightweight admin and holder operations, plus several reward and configuration paths.
Acceptable	15	Typical user actions and setup flows that are not unusually expensive.
Heavy	4	Complex gameplay settlement, winner processing, and multi-ticket batch logic.

Gas grading by average method cost





Key interpretation

• The heaviest observed method was **Win4ETH_GEORGE_Harness.processWinnersBatch** at ~7.10M average gas and up to ~10.32M in the consolidated run. This is operationally expensive but still below the 30M block limit shown in the report.

• The next major hotspots were **purchaseBox** (~1.15M average), **calculateWinners** (~586k average), and **purchaseTickets** (~264k average). These results match the design expectation that combinatorial gameplay and

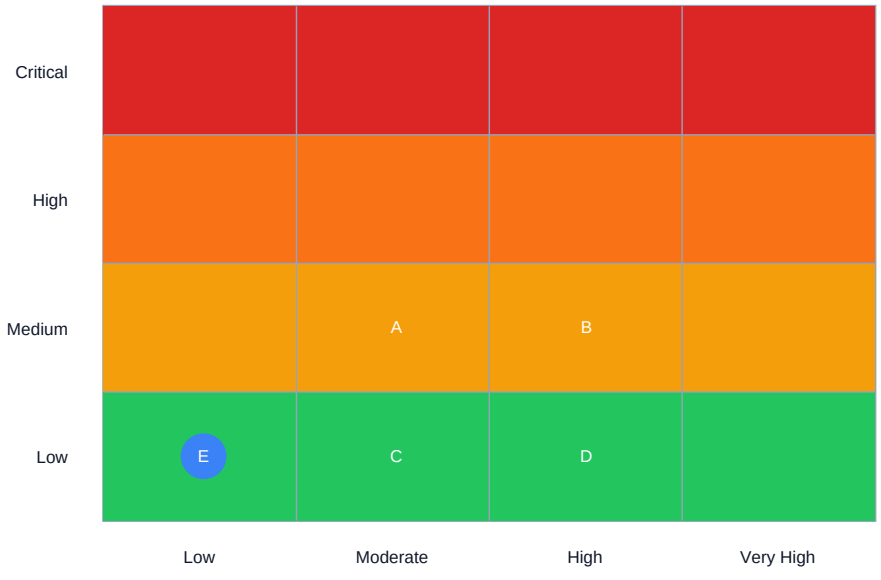
winner settlement cost more than ordinary token transfers.
• Core holder and admin operations remained efficient: **claimEthRewards** averaged ~51.9k gas, **authorizeLotteryContract** ~47.8k, and several tracker threshold setters remained near ~30k.

3. Security Posture & Findings

The supplied logs show a strong fail-closed posture. Under the observed scenarios, invalid claims, underfunded liabilities, prohibited admin calls, and swap failures reverted rather than partially executing. That is exactly the kind of behavior users and investors want to see in a revenue-backed game system.

Risk Matrix (tested-scope assessment)

Severity reflects operational impact within the supplied tests and gas traces, not an external line-by-line code audit.



A Gas-heavy winner batches

High cost but expected for batch settlement

B External swap dependency

Reverts safely; still an operational dependency

C Owner privilege concentration

Admin control present; requires governance discipline

D Underfunded pool protection

Gracefully reverts when liabilities are short

E Toolchain support note

Solidity 0.8.26 stack traces partially limited in Hardhat

ID	Finding	Severity	Evidence-backed interpretation	Recommendation
A	Winner settlement and box-purchase paths are gas heavy.	Medium	High complexity is expected, but batch methods may become user-friction points if transaction conditions tighten.	Consider pagination / bounded batches, or explicit operational runbooks for settlement cadence.
B	Swap-dependent treasury and prize processing introduces operational dependency on router behavior.	Medium	Tests show these paths revert safely when swaps fail, which is positive.	Preserve fail-closed logic and expose router/slippage settings in admin monitoring.
C	Owner and admin controls are powerful.	Low	Tests confirm access control, but the privileged surface still matters in production governance.	Use multisig, time delays, and operational segregation for key setters and emergency functions.
D	Underfunded prize states are explicitly protected by reverts.	Low	This is a positive control, not a vulnerability, but it should be monitored operationally.	Provide dashboard alerts when prize liabilities approach funding thresholds.
E	Hardhat notes limited support for Solidity 0.8.26 stack traces.	Info	This affects debugging ergonomics rather than contract behavior.	Keep compiler/toolchain versions pinned and retest when upstream support improves.

Control observations

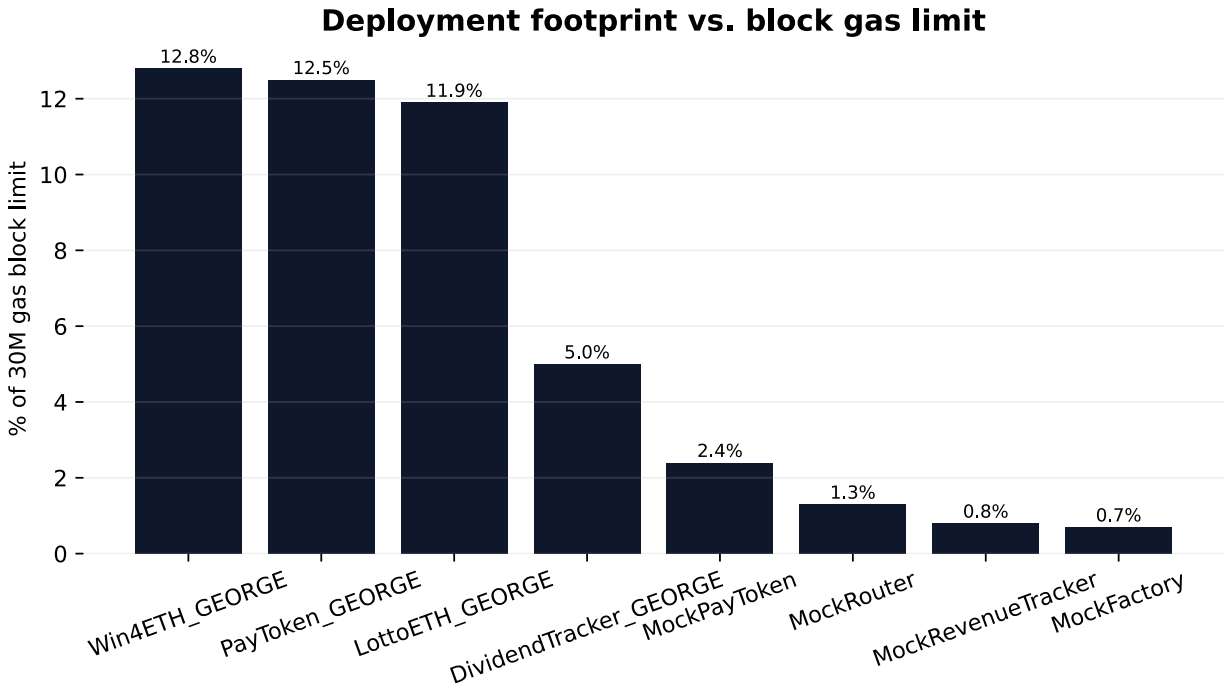
- Unauthorized owner-only setters were tested to revert.
- Claim protection logic blocked double claim scenarios.
- Prize and house accounting remained compatible with tracker forwarding.
- Excluded and ineligible holders did not receive rewards incorrectly.
- Zero-amount and boundary conditions were explicitly tested.

4. Contract Roles, Logic Design & Revenue Engine

Contract	Role in ecosystem	What the evidence shows
LottoETH_GEORGE	Primary multi-round ETH lottery engine	Starts new games, sells tickets, draws numbers, calculates winners, finalizes liabilities, supports rollover, and interacts with the tracker without breaking house/prize accounting.
Win4ETH_GEORGE	High-throughput straight and box game engine	Processes straight and combo box logic, supports forced winning number tests, handles winner claims, and rolls into subsequent games.
PayToken_GEORGE	Revenue token, liquidity processor, reflection forwarder	Applies tax behavior, forwards reflections, auto-processes treasury/liquidity, manages LP setup, and exposes holder reward claims.
DividendTracker_GEORGE	Reward distribution and eligibility state engine	Accepts revenue notifications, processes reflections and rewards, and enforces caller permissions around distribution paths.

Revenue engine narrative

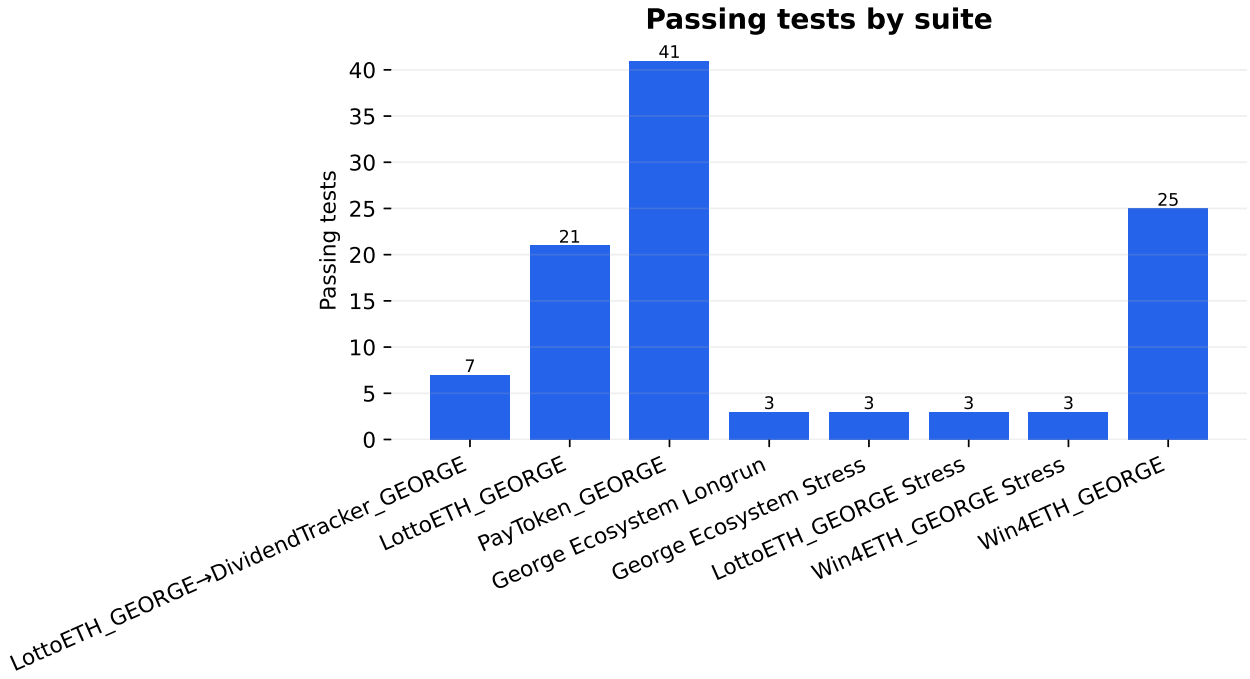
- **Gameplay revenue** originates from LottoETH and Win4ETH user activity.
- **Tracker state** receives holder allocations and keeps reward accumulation alive across repeated play.
- **PayToken** adds the tokenomics layer: tax forwarding, liquidity support, and direct reward claims.
- **GOAL and GIME** can turn these on-chain mechanics into a user-facing visibility layer by exposing contract state, reward status, slippage controls, and market context.

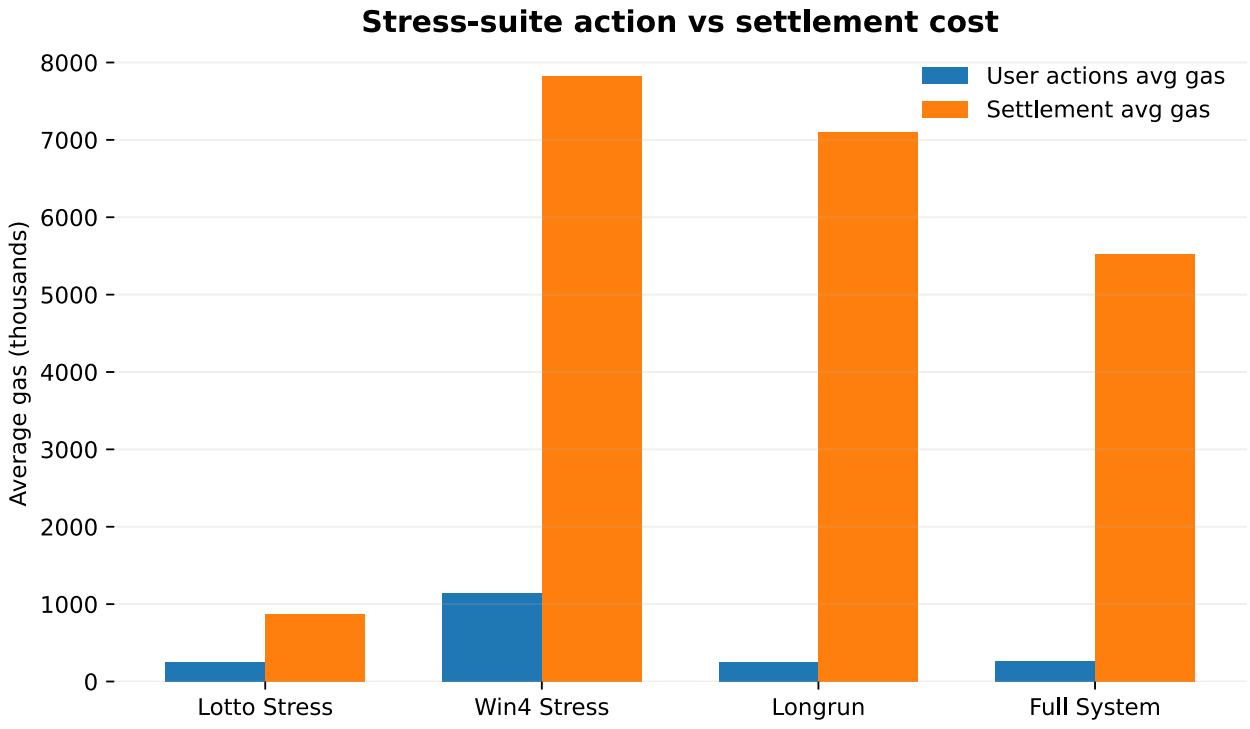


Deployment footprint for the primary George contracts sits near ~5% to ~12.8% of the stated 30M block limit, which is substantial but not abnormal for feature-rich gaming and tokenomics contracts.

5. Stress, Longrun & Scalability Findings

The separate stress and longrun runs reinforce the consolidated report: repeated rounds, heavy ticket creation, and tracker accumulation continued without reported failure in the uploaded logs. Performance pressure is real in settlement-heavy methods, but behavior remained consistent.





• The Lotto-specific stress output showed ~257k average gas on purchaseTickets and ~878k average on calculateWinners during that run.
• The Win4-specific stress output showed ~1.15M average gas on purchaseBox and ~7.83M average on processWinnersBatch.
• The longrun combined suite preserved tracker accumulation across repeated cross-game rounds.
• The full-system stress run completed three passing tests and kept processRewards, ticket creation, and cross-game action flow alive.

6. Interactive Dashboard Blueprint (GOAL / GIME aligned)

The accompanying Astro dashboard package turns this report into a live, screen-first audit experience. It includes KPI cards, severity grading, gas charts, revenue flow visuals, contract summaries, link buttons, and Chart.js-ready datasets that can later be bound to live chain data or API responses.

Panel	Purpose
Executive KPI row	Security, architecture, performance, gas, total tests, critical findings.
Risk matrix section	Severity grid plus remediation list.
Gas profiling section	Top methods, grade distribution, deployment footprint, and stress comparison.
Contract cards	LottoETH, Win4ETH, PayToken, DividendTracker with role summaries.
Revenue flow section	User → games → tracker → holders → GOAL/GIME visibility.
External links rail	georgefoc.com, GOAL, GIME, X, Telegram.

Links included in the dashboard package

- https://georgefoc.com
- https://goal.georgefoc.com
- https://gime.georgefoc.com
- https://x.com/GEORGE_FOC
- https://t.me/georgeFOC_PAY

Source basis

All figures and conclusions in this report were derived from the uploaded Hardhat outputs supplied in this conversation, including the consolidated 118-test report and the separate stress/longrun/full-system outputs.