Verificação Determinística com Incremental Merkle Tree On-Chain em Loterias Descentralizadas

Romulo de Moraes, Vinícius G. Pinto, Eder Gonçalves,
Roger Immich, Bruno L. Dalmazo
Universidade Federal do Rio Grande (FURG)
Universidade Federal do Rio Grande do Norte (UFRN)
{demoraes.romulo, vinicius.pinto, edergoncalves, dalmazo}@furg.br
roger@imd.ufrn.br

Abstract. *In large-scale lotteries, operational costs can make traditional approaches impractical. This paper proposes an architecture that addresses this limitation by introducing an on-chain Incremental Merkle Tree (IMT), which records a cryptographic commitment for each ticket purchase. During finalization, the contract requires that the submitted root matches the root derived from the IMT, preventing omission, address substitution, and fabricated sales in the commit phase, without requiring dispute windows or security deposits. We evaluate four implementations and show that the IMT-based variant preserves the favorable scalability of Merkle-based approaches while adding a constant overhead of ≈71k gas per purchase, regardless of the number of participants.*

Resumo. *Em loterias de grande porte, o custo pode inviabilizar abordagens tradicionais. Este artigo propõe uma arquitetura que supera essa limitação ao introduzir uma Incremental Merkle Tree (IMT) on-chain, que registra um compromisso criptográfico a cada compra de bilhete. Na finalização, o contrato exige que a raiz submetida coincida com a raiz derivada da IMT, impedindo omissão, substituição de endereços e vendas fabricadas na fase de commit, sem necessidade de disputa ou depósito de garantia. Avaliamos quatro implementações e mostramos que a variante com IMT mantém o escalonamento favorável das abordagens baseadas em Merkle, com sobrecusto constante de ≈71k gas por compra, independente do número de participantes.*

1. Introdução

O custo de operações que modificam o estado da Ethereum Virtual Machine (EVM) torna o registro individual de bilhetes em *arrays* e *mappings* economicamente inviável em loterias de grande porte [Antonopoulos and Wood 2018]. Em trabalho anterior, [Moraes et al. 2025] demonstrou que representar o conjunto de participantes por uma Merkle Tree, cuja raiz é armazenada *on-chain* e cujas folhas são reconstruídas *off-chain* a partir de eventos de compra, reduz o consumo total de *gas* em até três ordens de magnitude. Nesse modelo, participantes verificam sua inclusão apresentando uma prova composta pelos nós irmãos do caminho da folha até a raiz, verificada *on-chain* por meio de operações de hashing em pares ordenados.

A abordagem baseada em raiz calculada *off-chain*, embora escale bem, introduz uma lacuna de confiança: como a raiz pode ser submetida livremente pelo operador, um operador malicioso pode omitir compras legítimas, substituir endereços de compradores ou inserir vendas inexistentes sem que o contrato inteligente detecte inconsistências

durante o *commit*. Revisões do trabalho anterior destacaram essa vulnerabilidade e sugeriram mecanismos de contestação como forma de fortalecer a arquitetura.

Para superar essa vulnerabilidade, propomos um mecanismo determinístico de verificação no próprio *commit*. Mantemos uma Incremental Merkle Tree (IMT) *on-chain*, um mecanismo mais forte que esquemas de contestação clássicos [Kalodner et al. 2018], que registra, a cada compra, um compromisso criptográfico sobre (i) o intervalo de bilhetes adquirido e (ii) o endereço do comprador. Na finalização, o operador submete a raiz reconstruída *off-chain* a partir dos eventos públicos; o contrato a aceita apenas se coincidir com a raiz derivada das inserções realizadas *on-chain*, isto é, $\hat{r} = r_{\mathrm{IMT}}$, onde r_{IMT} é computado a partir das folhas inseridas nas transações de compra. Como as folhas são imutáveis após confirmação, qualquer desvio na reconstrução *off-chain*, altera a raiz submetida e faz a transação reverter por divergência em $\hat{r} = r_{\mathrm{IMT}}$. Então, a consistência do estado comprimido é verificada no *commit*, sem depender de esquemas típicos de contestação, como monitoramento externo, janela de disputa ou depósitos de garantia.

Neste contexto, diferentemente de SMTs (Sparse Merkle Trees), a IMT permite definir uma profundidade compatível com o número esperado de compras. As principais contribuições deste trabalho são: (i) um mecanismo determinístico de verificação no *commit*, (ii) a formalização da consistência via IMT *on-chain*, e (iii) a demonstração de aplicabilidade em dApps que precisam escalar participantes mantendo integridade, como *airdrops* e votação.

2. Trabalhos Relacionados

A maioria das propostas de loterias descentralizadas concentra-se na geração verificável de aleatoriedade, negligenciando armazenamento e confiança no operador. Por exemplo, [Hire et al. 2023] propuseram uma loteria em Ethereum que automatiza registro e transferência via oráculo de aleatoriedade verificável, sem analisar custo de *gas* ou modelo de confiança. Já em [Pan et al. 2022], foi introduzido o FPLotto, combinando aleatoriedade verificável, interpolação de Lagrange e provas de conhecimento zero para preservar a privacidade dos participantes; criptograficamente robusto, mas com custo de verificação *on-chain* proibitivo em produção. Em [Zhang et al. 2024], foi proposto a SM2VRF com verificação em lote, demonstrando redução de 68% no tempo de prova, com compatibilidade limitada ao ecossistema Ethereum. Nenhum desses trabalhos aborda compressão de estado ou o problema de confiança no operador que calcula a árvore.

No domínio de comprometimentos, os *rollups* otimistas [Kalodner et al. 2018] presumem validade e permitem contestação dentro de uma janela de disputa. O mecanismo aqui proposto inverte essa lógica: a verificação ocorre durante a fase de vendas, tornando desnecessária qualquer janela de contestação. Em contraste com trabalhos anteriores, que negligenciam compressão de estado e dependem de modelos de confiança no operador ou disputas posteriores, nossa proposta garante consistência determinística no *commit*, sem depender de monitoramento externo ou depósitos de garantia.

3. Proposta

A proposta, denominada **LotteryFraudProof**, utiliza uma Incremental Merkle Tree (IMT) mantida *on-chain* para registrar um compromisso criptográfico a cada compra de bilhete. Isso impede manipulações do conjunto de participantes na finalização da loteria.

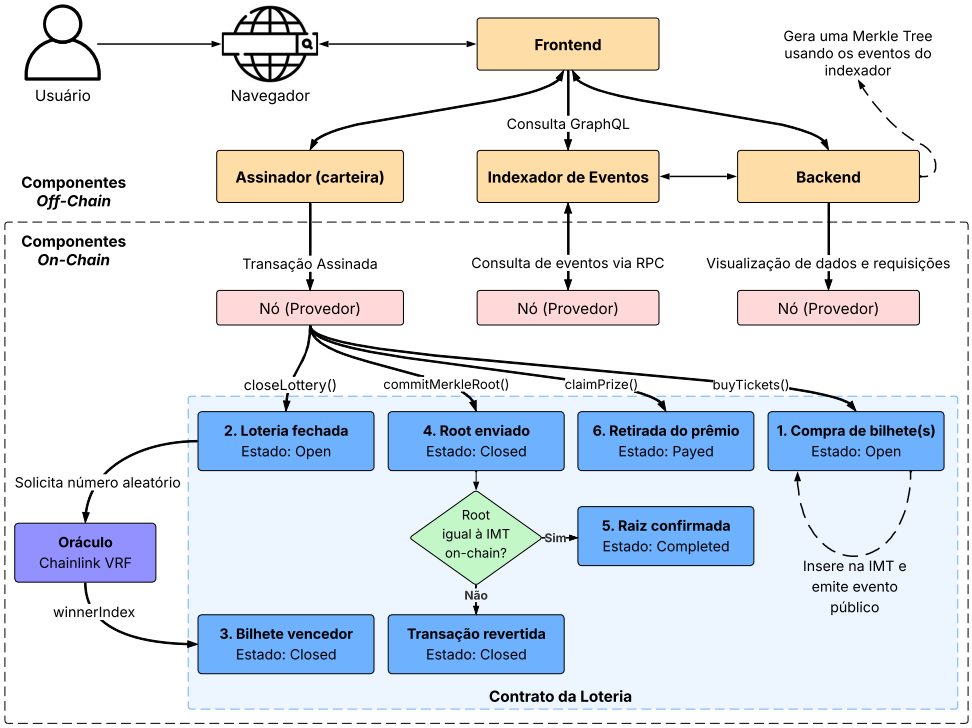

Figura 1. Diagrama de estados. A verificação de igualdade entre a raiz submetida e a raiz derivada da IMT é o ponto de controle central do mecanismo proposto.

A seguir, descrevemos o modelo de ameaça considerado e o funcionamento da IMT no protocolo.

3.1. Superfície de Ataque Considerada

O protocolo envolve três atores: (i) participantes, que compram bilhetes pagando ETH, (ii) um operador, responsável por gerenciar a loteria e reconstruir a Merkle Tree *off-chain*, e (iii) um oráculo de aleatoriedade verificável [Breidenbach et al. 2021]. O operador é considerado potencialmente malicioso, podendo tentar manipular o conjunto de participantes ao submeter a raiz final. Na arquitetura base, três ataques são relevantes: omissão de folha, substituição de endereço e fabricação de folha. A condição $\hat{r} = r_{\mathrm{IMT}}$ impede essas manipulações, pois qualquer divergência entre a reconstrução *off-chain* e as inserções registradas *on-chain* altera a raiz e faz a transação reverter.

Participantes também podem tentar reivindicar o prêmio com provas inválidas, o que é mitigado pela verificação *on-chain* da prova de inclusão na raiz comprometida. Por fim, assume-se que o adversário não quebra as propriedades criptográficas da função de hash utilizada, como resistência a colisões e a pré-imagem.

Tabela 1. Garantias de segurança por classe de ataque.

Ataque	**Arquitetura base**	**LotteryFraudProof**
Omissão de folha	Detectável; sem bloqueio	Impossível: ausência altera raiz *on-chain*.
Substituição de endereço	Detectável; sem bloqueio	Impossível: alteração propaga à raiz.
Fabricação de folha	Detectável; sem bloqueio	Impossível: folha fora da IMT gera raiz divergente.
Índice vencedor	Impossível (VRF antes do *commit*)	Idêntico
Prova Merkle falsa	Impossível (verificação *on-chain*)	Idêntico

3.2. Estrutura da IMT On-Chain

A IMT é uma árvore binária *append-only* de profundidade fixa D, definida para comportar o número máximo de compras. No protótipo, utiliza-se $D = 32$, suportando até 2^{32} folhas. Para cada compra, o contrato computa a folha $\ell_i = \mathrm{keccak256}(s_i, |, e_i, |, a_i)$, onde s_i e e_i são os índices do intervalo de bilhetes e a_i é o endereço do comprador, inserindo ℓ_i na IMT de forma atômica. A estrutura mantém um vetor de fronteira com até D nós intermediários e um vetor de nós nulos pré-computados para posições vazias, garantindo armazenamento $O(D)$, independente do número total de compras.

Para garantir consistência entre a raiz *on-chain* e a reconstruída *off-chain*, o protocolo utiliza a mesma função de combinação de hash em ambos os lados. A reconstrução *off-chain* utiliza as folhas derivadas dos eventos de compra e a mesma regra de combinação, produzindo uma raiz $\hat{r}$ que coincide com r_{IMT} quando todas as compras são consideradas. Assim, a condição $\hat{r} = r_{\mathrm{IMT}}$ funciona como teste de consistência entre o registro das compras e o compromisso submetido na finalização.

Cada inserção atualiza nós ao longo do caminho associado à nova folha. Em média, poucas posições são modificadas: metade das inserções encerra no primeiro nível e a probabilidade de atualizar k níveis é 2^{-k}. O recálculo da raiz utiliza apenas leituras e combinações de hash, com $O(D)$ acessos. Dessa forma, a IMT garante que qualquer tentativa de manipular o conjunto de participantes gere divergência de raiz e reversão da transação. A Figura 1 ilustra o fluxo de estados do contrato.

4. Avaliação e Discussão

Nesta seção avaliamos consumo de *gas*, escalabilidade e hipóteses de confiança em quatro implementações: a arquitetura base com Merkle Tree *off-chain*, a proposta com IMT *on-chain*, e duas variantes tradicionais de armazenamento baseadas em *array* e *mapping*.

4.1. Setup e Configuração do Ambiente

Os experimentos foram conduzidos utilizando Hardhat 2.22.19 e compilador Solidity 0.8.28 com otimizador desativado, de modo a isolar o comportamento estrutural de cada implementação e evitar otimizações específicas do compilador que poderiam favorecer variantes particulares. O ambiente de execução consistiu em um processador AMD R5 5500, 16 GB de RAM e sistema operacional Ubuntu 22.04.

4.2. Estudo de caso 1

O cenário base simula dois compradores dividindo o total de bilhetes. A Figura 2 apresenta o consumo de *gas* de 5 a 100.000 bilhetes. As duas variantes baseadas em Merkle mantêm custo praticamente constante, enquanto as abordagens tradicionais escalam linearmente, ultrapassando 10^7 gas rapidamente. Para 1.000 bilhetes e dois compradores, o custo total da arquitetura base é ≈491k gas e o da variante com IMT é ≈636k gas. O sobrecusto de ≈145k gas (aproximadamente 71k por compra) concentra-se na fase de aquisição, pois a fase de *commit* realiza essencialmente leituras e combinações de hash. Para referência, uma transferência ERC-20 padrão consome aproximadamente 21k gas, assim, o sobrecusto introduzido pela IMT equivale a cerca de 3,4 transferências adicionais por compra.

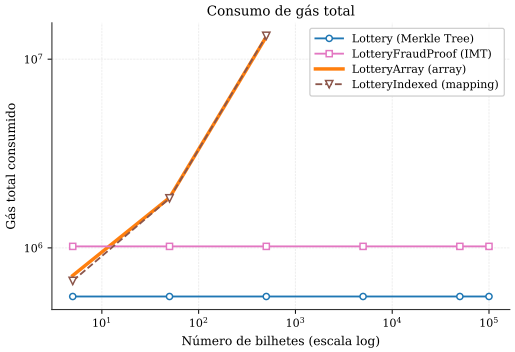

Figura 2. Consumo de *gas* (tickets)

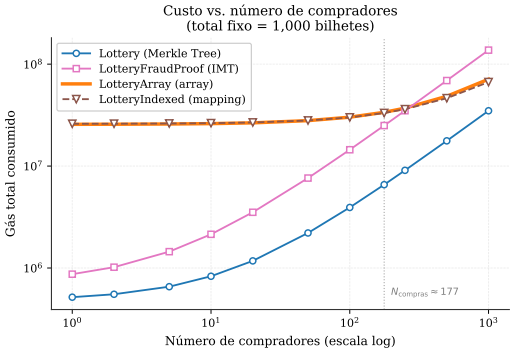

Figura 3. Consumo de *gas* (buyers)

4.3. Estudo de caso 2

Enquanto o primeiro estudo de caso avalia escalabilidade em função do número de bilhetes, o segundo explora o impacto da fragmentação de compras entre diferentes participantes. Nesse sentido, para refutar a hipótese de que o benchmark favorece a IMT ao usar apenas dois compradores, fixamos 1.000 bilhetes e variamos o número de compradores de 1 a 1.000 (Figura 3). O custo da variante IMT cresce linearmente com o número de compras, enquanto o custo das abordagens tradicionais é determinado exclusivamente pelo total de bilhetes. Para até ≈177 compradores em 1.000 bilhetes, a variante IMT permanece mais barata, delimitando claramente o regime de aplicabilidade: loterias com compras em lote ou onde a garantia de integridade justifica o sobrecusto.

4.4. Discussão

Os resultados confirmam que a IMT desloca parte do custo para a entrada do sistema, adicionando um pequeno sobrecusto por compra para eliminar a necessidade de confiar no operador durante o *commit*.

Tabela 2. Comparação qualitativa das estratégias de armazenamento. Seja T o total de bilhetes, n o número de compras (folhas) e D a profundidade da IMT.

Método	**Estado *on-chain***	**Compra**	**Confiança no *commit***
Array	Endereço por bilhete ($O(T)$)	Cresce com bilhetes por compra	Nenhuma; estado explícito
Mapping	Endereço por bilhete ($O(T)$)	Cresce com bilhetes por compra	Nenhuma; estado explícito
Merkle *off-chain*	Apenas raiz ($O(1)$)	Baixo; depende de eventos	Operador pode submeter raiz inválida
IMT *on-chain*	Fronteira + nós nulos ($O(D)$)	Cresce com nº de compras	Nenhuma; raiz deve coincidir com IMT

A Tabela 2 resume qualitativamente os efeitos da proposta em armazenamento, custo e hipóteses de confiança. A principal diferença está entre uma raiz *off-chain* livre e uma raiz validada pela IMT. No primeiro caso, o contrato não distingue uma raiz correta de uma manipulada; no segundo, a própria blockchain restringe os valores aceitos no *commit*. Como raízes inválidas são rejeitadas, a finalização pode ser executada por qualquer parte interessada, reduzindo o risco de travamento por omissão do operador. No protótipo, ela é iniciada pelo operador, mas a segurança decorre da verificação $\hat{r} = r_{\mathrm{IMT}}$. Outro fator é a escolha dos parâmetros. A profundidade D define o número máximo de compras e o custo de reconstrução da raiz. Embora $D = 32$ atenda a muitas aplicações, cenários com milhões de compras podem exigir ajustes. Nesse contexto, soluções de

segunda camada podem reduzir custos de armazenamento, tornando a abordagem mais viável em cenários com maior fragmentação de compras.

Por fim, o mecanismo reduz a necessidade de contestação, mas não elimina totalmente questões de vivacidade. Falhas ou atrasos no oráculo de aleatoriedade podem interromper o progresso do protocolo. Por isso, implementações práticas devem considerar políticas de recuperação e *timeouts* para lidar com falhas de infraestrutura.

5. Considerações Finais

Este artigo apresentou uma extensão que elimina a suposição de confiança no operador durante o *commit* do conjunto de participantes. Ao registrar cada compra como uma folha inserida *on-chain* em uma IMT e impor a igualdade entre a raiz submetida e a derivada dessas inserções, o protocolo impede omissão, substituição e fabricação de participantes na finalização, sem depender de janela de disputa ou monitoramento externo. Para poucos compradores com compras em lote, o sobrecusto de $\approx$71k gas por compra é marginal face aos ganhos de escalabilidade; para muitos compradores únicos, o custo cresce linearmente com o número de transações, tornando a decisão de adoção sensível ao padrão de uso.

Como trabalhos futuros, pretende-se avaliar a arquitetura em redes de segunda camada, onde custos de escrita em armazenamento tendem a ser menores; formalizar propriedades de segurança e vivacidade do protocolo sob modelos de falha realistas; e investigar variantes baseadas em SMT em cenários que demandem provas de não-inclusão.

Referências

Antonopoulos, A. M. and Wood, G. (2018). *Mastering Ethereum: Building Smart Contracts and DApps*. O'Reilly Media.

Breidenbach, L., Cachin, C., Chan, B., Coventry, A., Ellis, S., Juels, A., Koushanfar, F., Miller, A., Magauran, B., and et al., D. M. (2021). Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. *Chainlink Labs*, 1:1–136.

Hire, A., Lanjewar, H., Haridas, P., Jadhav, M., and Rane, M. (2023). Decentralized lottery using blockchain. In *Proceedings of the 2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN)*, pages 1035–1041. IEEE.

Kalodner, H., Goldfeder, S., Chen, X., Weinberg, S. M., and Felten, E. W. (2018). Arbitrum: Scalable, private smart contracts. In *27th USENIX Security Symposium*. USENIX Association.

Moraes, R., Bubolz, A. G., Ayres, D. G., Pinto, V. G., and Dalmazo, B. L. (2025). Arquitetura híbrida para loterias em blockchain com compressão de estado via merkle tree. In *Anais da XXII Escola Regional de Redes de Computadores (ERRC 2025)*, pages 8–13, Porto Alegre, RS, Brasil. SBC.

Pan, Y., Zhao, Y., Liu, X., Wang, G., and Su, M. (2022). Fplotto: A fair blockchain-based lottery scheme for privacy protection. In *Proceedings of the 2022 IEEE International Conference on Blockchain (Blockchain)*, pages 21–28. IEEE.

Zhang, Y., Yang, J., Lei, H., Bao, Z., Lu, N., Shi, W., and Chen, B. (2024). Verifiable random function schemes based on sm2 digital signature algorithm and its applications for committee elections. *IEEE Open Journal of the Computer Society*.