In-EVM Solana State Verification
Proof System Description

Cherniaeva Alisa
a.cherniaeva@nil.foundation
=nil; Crypto3 (https://crypto3.nil.foundation)

Shirobokov Ilia
i.shirobokov@nil.foundation
=nil; Crypto3 (https://crypto3.nil.foundation)

November 11, 2021

1 Introduction

WIP

To prove Solana blockchain's state on the Ethereum Virtual Machine, we use Redshift SNARK[1]. RedShift is a transparent SNARK that uses PLONK[2] proof system but replaces the commitment scheme. The authors utilize FRI[3] protocol to obtain transparency for the PLONK system.

However, FRI cannot be straightforwardly used with the PLONK system. To achieve the required security level without huge overheads, the authors introduce *list polynomial commitment* scheme as a part of the protocol. For more details, we refer the reader to [1].

The original RedShift protocol utilizes the classic PLONK[2] system. To provide better performance, we generilize the original protocol for use with PLONK with custom gates [4], [5] and lookup arguments [6], [7].

2 RedShift Protocol

WIP

Notations:

$N_{\texttt{wires}}$	Number of wires ('advice columns')
$N_{\texttt{perm}}$	Number of wires that are included in the permutation argument
$N_{\texttt{sel}}$	Number of selectors used in the circuit
$N_{\texttt{const}}$	Number of constant columns
$N_{\texttt{lookups}}$	Number of lookups
$\mathbf{f}_i$	Witness polynomials, $0 \leq i < N_{\texttt{wires}}$
$\mathbf{f}_{c_i}$	Constant-related polynomials, $0 \leq i < N_{\texttt{const}}$
$\mathbf{gate}_i$	Gate polynomials, $0 \leq i < N_{\texttt{sel}}$
$\sigma(\text{col}: i, \text{row}: j) = (\text{col}: i', \text{row}: j')$	Permutation over the table

For details on polynomial commitment scheme and polynomial evaluation scheme, we refer the reader to [1].

Preprocessing:

1. $\mathcal{L}' = (\mathbf{q}_0, ..., \mathbf{q}_{N_{\mathtt{sel}}})$
2. Let ω be a 2^k root of unity
3. Let δ be a T root of unity, where $T \cdot 2^S + 1 = p$ with T odd and $k \leq S$
4. Compute $N_{\mathtt{perm}}$ permutation polynomials $S_{\sigma_i}(X)$ such that $S_{\sigma_i}(\omega^j) = \delta^{i'} \cdot \omega^{j'}$
5. Compute $N_{\mathtt{perm}}$ identity permutation polynomials: $S_{id_i}(X)$ such that $S_{id_i}(\omega^j) = \delta^i \cdot \omega^j$
6. Let $H = \{\omega^0, ..., \omega^n\}$ be a cyclic subgroup of $\mathbb{F}^*$
7. Let $Z(X) = \prod a \in H^*(X - a)$
8. Let A_i be a witness lookup columns and S_i be a table columns, $i = 0, .., m$.

Protocol (Prover):

1. Choose masking polynomials:

$$h_i(X) \leftarrow \mathbb{F}_{<k}[X] \text{ for } 0 \leq i < N_{\mathtt{wires}}$$

Remark: For details on choice of k, we refer the reader to [1].

2. Define new witness polynomials:

$$f_i(X) = \mathbf{f}_i(X) + h_i(X)Z(X) \text{ for } 0 \leq i < N_{\mathtt{wires}}$$

3. Send commitments to f_i to $\mathbf{V}$
4. Get $\theta \leftarrow \mathbb{F}$ from $\mathbf{V}$
5. Construct the witness lookup compression and table compression $S(\theta)$ and $A(\theta)$:

$A(\theta) = \theta^{m-1}A_0 + \theta^{m-2}A_1 + ... + \theta A_{m-2} + A_{m-1}$ // $S(\theta) = \theta^{m-1}S_0 + \theta^{m-2}S_1 + ... + \theta S_{m-2} + S_{m-1}$

6. Produce the permutation polynomials $S'(X)$ and $A'(X)$ such that:

   6.1 All the cells of column A' are arranged so that like-valued cells are vertically adjacent to each other.

   6.2 The first row in a sequence of values in A' is the row that has the corresponding value in S'.

7. Compute and send commitments to A' and S' to $\mathbf{V}$
8. Get $\beta, \gamma \leftarrow \mathbb{F}$ from $\mathbf{V}$
9. For $0 \leq i < N_{\mathtt{perm}}$

$$\begin{aligned} p_i &= f_i + \beta \cdot S_{id_i} + \gamma \\ q_i &= f_i + \beta \cdot S_{\sigma_i} + \gamma \end{aligned}$$

10. Define:

$$\begin{aligned} p'(X) &= \prod_{0 \leq i < N_{\mathtt{perm}}} p_i(X) \in \mathbb{F}_{<N_{\mathtt{perm}} \cdot n}[X] \\ q'(X) &= \prod_{0 \leq i < N_{\mathtt{perm}}} q_i(X) \in \mathbb{F}_{<N_{\mathtt{perm}} \cdot n}[X] \end{aligned}$$

11. Compute $P(X), Q(X) \in \mathbb{F}_{<n+1}[X]$, such that:

$$\begin{aligned} P(\omega) &= Q(\omega) = 1 \\ P(\omega^i) &= \prod_{1 \leq j < i} p'(\omega^i) \text{ for } i \in 2, \ldots, n+1 \\ Q(\omega^i) &= \prod_{1 \leq j < i} q'(\omega^i) \text{ for } i \in 2, \ldots, n+1 \end{aligned}$$

12. Compute and send commitments to P and Q to $\mathbf{V}$

13. Compute permutation product column:

$$V(\omega^i) = \frac{(\theta^{m-1}A_0(\omega^i)+\theta^{m-2}A_1(\omega^i)+...+\theta A_{m-2}(\omega^i)+A_{m-1}(\omega^i)+\beta)\cdot(\theta^{m-1}S_0(\omega^i)+\theta^{m-2}S_1(\omega^i)+...+\theta S_{m-2}(\omega^i)+S_{m-1}(\omega^i)+\gamma)}{(A'(\omega^i)+\beta)(S'(\omega^i)+\gamma)}$$
$$V(1) = V(\omega^{N_{\texttt{lookups}}}) = 1$$

14. Compute and send commitments to V to $\mathbf{V}$

15. Get $\alpha_0, \dots, \alpha_5 \leftarrow \mathbb{F}$ from $\mathbf{V}$

16. Define polynomials ($F_0, \dots, F_4$ - copy-satisfability):

$$F_0(X) = L_1(X)(P(X) - 1)$$
$$F_1(X) = L_1(X)(Q(X) - 1)$$
$$F_2(X) = P(X)p'(X) - P(X\omega)$$
$$F_3(X) = Q(X)q'(X) - Q(X\omega)$$
$$F_4(X) = L_n(X)(P(X\omega) - Q(X\omega))$$
$$F_5(X) = \sum_{0 \le i < N_{\texttt{sel}}} (\mathbf{q}_i(X) \cdot \texttt{gate}_i(X)) + \sum_{0 \le i < N_{\texttt{const}}} (\mathbf{f}_{c_i}(X)) + PI(X)$$

17. For the lookup:

    17.1 Two selectors q_{last} and q_{blind} are used, where $q_{last} = 1$ for t last blinding rows and $q_{blind} = 1$ on the row in between the usable rows and the blinding rows.

    17.2 $F_6(X) = L_0(X)(1 - V(X))$

    17.3 $F_7(X) = q_{last} \cdot (V(X)^2 - V(X))$

    17.4 $F_8(X) = (1 - (q_{last} + q_{blind})) \cdot (V(\omega X)(A'(X) + \beta)(S'(X) + \gamma) - V(X)(\theta^{m-1}A_0(X) + ... + A_{m-1}(X) + \beta)(\theta^{m-1}S_0(X) + ... + S_{m-1}(X) + \gamma))$

    17.5 $F_9(X) = L_0(X) \cdot (A'(X) - S'(X))$

    17.6 $F_{10}(X) = (1 - (q_{last} + q_{blind})) \cdot (A'(X) - S'(X)) \cdot (A'(X) - A'(\omega^{-1}X))$

18. Compute:

$$F(X) = \sum_{i=0}^{10} \alpha_i F_i(X)$$
$$T(X) = \frac{F(X)}{Z(X)}$$

19. Split $T(X)$ into separate polynomials $T_0(X), ..., T_{N_{\texttt{perm}}}(X)$

20. Send commitments to $T_0(X), ..., T_{N_{\texttt{perm}}}(X)$ to $\mathbf{V}$

21. Get $y \leftarrow \mathbb{F}/H$ from $\mathbf{V}$

22. Run evaluation scheme with the committed polynomials and y

23. Send proof π to $\mathbf{V}$

2.1 Non-Interactive Verification

1. Let $f_{0,\texttt{comm}}, \dots, f_{N_{\texttt{wires}},\texttt{comm}}$ be commitments to $f_0(X), \dots, f_{N_{\texttt{wires}}}(X)$

2. transcript = setup_values$||f_{0,\texttt{comm}}|| \dots ||f_{N_{\texttt{wires}},\texttt{comm}}$

3. $\theta = H(\text{transcript})$

4. Let $A'_{\texttt{comm}}, S'_{\texttt{comm}}$ be commitments to $A'(X), S'(X)$.

5. transcript = transcript$||A'_{\texttt{comm}}||S'_{\texttt{comm}}$

6. $\beta, \gamma = H(\text{transcript})$

7. Let $P_{\texttt{comm}}, Q_{\texttt{comm}}, V_{i,\texttt{comm}}$ be commitments to $P(X), Q(X), V(X)$.

8. transcript $=$ transcript$||P_{\texttt{comm}}||Q_{\texttt{comm}}||V_{\texttt{comm}}$

9. $\alpha_0, \ldots, \alpha_5 = H(\text{transcript})$

10. Let $T_{0,\texttt{comm}}, ..., T_{N_{\texttt{perm},\texttt{comm}}}$ be commitments to $T_0(X), ..., T_{N_{\texttt{perm}}}(X)$

11. transcript $=$ transcript$||T_{0,\texttt{comm}}||...||T_{N_{\texttt{perm},\texttt{comm}}}$

12. $y = H_{\mathbb{F}/H}(\text{transcript})$

13. Run evaluation scheme verification with the committed polynomials and y to get values $f_i(y), P(y), P(y\omega), Q(y), Q(y\omega), T_j(y), A'(y), S'(y), V(y), A'(y\omega^{-1}, V(y\omega))$.
Remark: Depending on the circuit, evaluation can be done also on $f_i(y\omega), f_i(y\omega^{-1})$ for some i.

14. Calculate:

$$\begin{gathered}
F_0(y) = L_1(y)(P(y) - 1) \\
F_1(y) = L_1(y)(Q(y) - 1) \\
p'(y) = \prod p_i(y) = \prod f_i(y) + \beta \cdot S_{id_i}(y) + \gamma \\
F_2(y) = P(y)p'(y) - P(y\omega) \\
q'(y) = \prod q_i(y) = \prod f_i(y) + \beta \cdot S_{\sigma_i}(y) + \gamma \\
F_3(y) = Q(y)q'(y) - Q(y\omega) \\
F_4(y) = L_n(y)(P(y\omega) - Q(y\omega)) \\
F_5(y) = \sum_{0 \leq i < N_{\texttt{sel}}} (\mathbf{q}_i(y) \cdot \texttt{gate}_i(y)) + \sum_{0 \leq i < N_{\texttt{const}}} (\mathbf{f}_{c_i}(y)) + PI(y) \\
T(y) = \sum_{0 \leq j < N_{\texttt{perm}+1}} y^{n \cdot j} T_j(y) \; F_6(y) = L_0(y)(1 - V(y)) \\
F_7(y) = q_{last} \cdot (V(y)^2 - V(y)) \\
F_8(y) = (1 - (q_{last} + q_{blind})) \cdot (V(\omega y)(A'(y) + \beta)(S'(y) + \gamma) - V(y)(\theta^{m-1} A_0(y) + ... + A_{m-1}(y) + \\
\beta)(\theta^{m-1} S_{i,0}(y) + ... + S_{m-1}(y) + \gamma)) \\
F_9(y) = L_0(y) \cdot (A'(y) - S'(y)) \\
F_{10}(y) = (1 - (q_{last} + q_{blind})) \cdot (A'(y) - S'(y)) \cdot (A'(y) - A'(\omega^{-1} y))
\end{gathered}$$

15. Check the identity:

$$\sum_{i=0}^{10} \alpha_i F_i(y) = Z(y)T(y)$$

3 Optimizations

WIP

References

1. Kattis A., Panarin K., Vlasov A. RedShift: Transparent SNARKs from List Polynomial Commitment IOPs. Cryptology ePrint Archive, Report 2019/1400. 2019. https://ia.cr/2019/1400.

2. Gabizon A., Williamson Z. J., Ciobotaru O. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. Cryptology ePrint Archive, Report 2019/953. 2019. https://ia.cr/2019/953.

3. Fast Reed-Solomon interactive oracle proofs of proximity / E. Ben-Sasson, I. Bentov, Y. Horesh et al. // 45th international colloquium on automata, languages, and programming (icalp 2018) / Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.

4. Gabizon A., Williamson Z. J. Proposal: The Turbo-PLONK program syntax for specifying SNARK programs. https://docs.zkproof.org/pages/standards/accepted-workshop3/proposal-turbo_plonk.pdf.

5. PLONKish Arithmetization - The halo2 book. https://zcash.github.io/halo2/concepts/arithmetization.html.

6. Gabizon A., Williamson Z. J. plookup: A simplified polynomial protocol for lookup tables. Cryptology ePrint Archive, Report 2020/315. 2020. https://ia.cr/2020/315.

7. Lookup argument - The halo2 book. https://zcash.github.io/halo2/design/proving-system/lookup.html.