

Attacking soundness for an optimization of the Gemini Polynomial Commitment Scheme

Lydia Garms¹ and Michael Livesey²

¹ Lydia.Garms@uk.ey.com

² Michael.Livesey@uk.ey.com
Ernst & Young, UK

Abstract. We demonstrate an attack on the soundness of a widely known optimization of the Gemini multilinear Polynomial Commitment Scheme (PCS). The attack allows a malicious prover to falsely claim that a multilinear polynomial takes a value of their choice, for any input point. We stress that the original Gemini multilinear PCS and HyperKZG, an adaptation of Gemini, are not affected by the attack.

1 Introduction

A Polynomial Commitment Scheme (PCS) [18] allows a prover to commit to a polynomial. The committer can then additionally provide opening proofs that the polynomial committed to evaluates to a particular output on a given input. PCSs are an important building block of many zk-SNARK proof systems [17,6,15]. This is because circuit arithmetizations, such as R1CS [16] or Plonk [15], allow the statement being proved to be written in terms of a series of statements involving polynomials, which can be proven via a Polynomial interactive oracle proof (IOP) [5]. A PCS allows a Polynomial-IOP to be compiled into a SNARK.

A multi-linear polynomial is a multivariate polynomial that is linear in each of its variables, i.e., no variable has degree 2 or higher. Instead of using a univariate PCS to compile a Polynomial-IOP into a SNARK, as in [23,15,14], a multilinear PCS can be used to compile a multilinear-IOP, as in [27,28,25,26,9,13]. One of the benefits of doing this is compatibility with the SumCheck protocol [22], which has a linear time prover. The SumCheck protocol allows one to prove the sum of evaluations of a multivariate polynomial with n variables over all inputs in a set H^n .

A range of univariate Polynomial Commitment Schemes have been constructed offering advantages and tradeoffs, such as succinctness, transparency and post quantum security, e.g., KZG [18], IPA [8,11,10], FRI [4,19,29] and DARK[12]. Likewise, a plethora of multilinear PCSs [24,30,11,29,21,9,20] have been constructed, offering similar advantages and trade-offs. Several works have focused on building multilinear PCSs generically with a univariate PCS as a building block [24,9,20], often resulting in some of the most practical schemes.

In particular, KZG [18] has the advantage of being the most succinct univariate scheme with the smallest opening proofs, although this comes at the cost

of trusted setup and requiring pairing friendly groups. Multilinear KZG commitments were introduced in [24]. However, for a n -linear polynomial, n pairings are required to verify the opening proofs, unlike the univariate scheme which has constant verification time. Virgo [29] reduces the verification of opening proofs to 1 pairing and $\Omega(n^2)$ field operations, but at the cost of proof sizes and proving times of $\Omega(2^n n)$.

The Gemini [9] and Zeromorph [20] schemes, using ZKG as the underlying univariate PCS (additionally using batching techniques from [7] for the case of Gemini), both reduce this verification time. They are the most concretely practical multilinear schemes when considering a combination of proof size and verification time. In Zeromorph verification requires $2n+2$ exponentiations in $\mathbb{G}_1$, 2 exponentiations in $\mathbb{G}_2$ and 3 pairings, whereas in Gemini, verification requires $n+2$ exponentiations in $\mathbb{G}_1$, 1 exponentiation in $\mathbb{G}_2$ and 3 pairings (2 pairings without the hiding property). However, Zeromorph has smaller proof sizes with only $n+3$ group elements, compared to Gemini with $n+2$ group elements and $2n$ field elements. Additionally, Zeromorph only requires an extra $\log(n)$ group exponentiations during proving to obtain zero-knowledge. The HyperKZG PCS [31] scheme adapts Gemini so that polynomials can be committed to in evaluation instead of coefficient form.

A blog post [3] introduced an optimization of the Gemini Polynomial Commitment Scheme. A proof for the zero-knowledge property was provided, but not a proof of the soundness property. The benefit of the optimization is that, for a multilinear opening proof, only $n+1$ instead of $2n$ univariate KZG openings had to be proved. When using the batched KZG techniques from [7], verification costs are not impacted when considering group operations, but the proof size is reduced by $n-1$ field elements because only $n+1$ polynomial evaluations now need to be included. When batching techniques from [7] are not used, the optimization also significantly reduces verification time. We believe that this version of Gemini is widely known and implemented [20,1,2].

1.1 Our contribution

We demonstrate in this work that the optimization of Gemini from [3] does not result in sound opening proofs. We describe an attack that will convince a verifier to accept an incorrect evaluation of a multilinear polynomial for any input. No significant extra computation is required by the malicious prover, compared to an honest prover. We emphasize that neither the original Gemini scheme nor the HyperKZG scheme is affected by this attack.

The intuition behind the attack is as follows. In the Gemini scheme n polynomials are committed to in the opening proof, say $f^{(0)}, f^{(1)}, f^{(n-1)}$, where $f^{(0)}$ is the original polynomial converted from multilinear to univariate form. The verifier recursively checks a relation involving $f^{(j)}(\rho)$, $f^{(j)}(-\rho)$ and $f^{(j+1)}(\rho^2)$, for $j = 0, \dots, n-2$ for a random challenge ρ . In the optimized version of the protocol, instead of using the same point ρ at each step, ρ^{2^j} is used as the challenge at step j . The recursive relations now involve $f^{(j)}(\rho^{2^j})$ and $f^{(j)}(-\rho^{2^j})$ and $f^{(j+1)}(\rho^{2^{j+1}})$, for $j = 0, \dots, n-2$. As $f^{(j)}(\rho^{2^j})$ for $j = 1, \dots, n-2$ can be written

in terms of $f^{(j-1)}(\rho^{2^{j-1}})$, $f^{(j-1)}(-\rho^{2^{j-1}})$, we can substitute this into the verification equation at the next step. Carrying on with this, we end up with one verification equation in terms of $f^{(0)}(\rho)$, $f^{(0)}(-\rho)$ and $f^{(j)}(-\rho^{2^j})$, for $j = 1, \dots, n-2$. This means that $n-1$ fewer openings are required in the underlying univariate PCS. However, as $f^{(j)}(\rho^{2^j})$ is never checked, the malicious prover can instead commit to polynomials $g^{(n-2)} = f^{(n-2)} + \epsilon^{(1)}$ and $g^{(n-1)} = f^{(n-1)} + \epsilon^{(2)}$. The polynomials $\epsilon^{(1)}$ and $\epsilon^{(2)}$ have been carefully chosen so that they cancel out in the verification equation and also will result in the verification passing for a malicious evaluation point of the prover's choice. If the prover was forced to open the polynomials at the positive powers of 2 of ρ this would not be possible, because the discrepancy of $\epsilon^{(1)}$ and $\epsilon^{(2)}$ would be detected.

Acknowledgments

We would like to thank Patrick Towa for helpful discussions and his useful feedback on this work.

2 Optimized Gemini Polynomial Commitment Scheme

The Gemini protocol is a univariate polynomial Interactive Oracle Proof (IOP), due to Bootle, Chiesa, Hu and Orrù [9], for the evaluations of multilinear polynomials. In this section, we outline the optimized Gemini protocol, originally proposed in [3]. Much of the background details are the same and the difference comes when we describe the IOP in §2.2.

Throughout this section, $\mathbb{F}$ will denote a field, n a positive integer and $\mathbf{u} := (u_0, \dots, u_{n-1})$ some element in $\mathbb{F}^n$.

As with the original Gemini protocol, the optimized version leverages the following $\mathbb{F}$ -vector-space isomorphism, which identifies multilinear polynomials in n variables with univariate polynomials of degree at most $2^n - 1$.

$$\begin{aligned} \varphi : \mathbb{F}[X_0, \dots, X_{n-1}]^{\leq 1} &\rightarrow \mathbb{F}[X]^{< 2^n} \\ f &\mapsto \sum_{i_0, \dots, i_{n-1} \in \{0,1\}} f(i_0, \dots, i_{n-1}) X^{i_{n-1} \cdot 2^{n-1} + \dots + i_0 \cdot 2^0}. \end{aligned}$$

2.1 Split and fold

Given an even positive integer N and a vector $\mathbf{A} := (a_0, \dots, a_{N-1}) \in \mathbb{F}^N$, we set

$$\begin{aligned} \mathbf{A}_0 &:= (a_0, a_2, \dots, a_{N-2}), \\ \mathbf{A}_1 &:= (a_1, a_3, \dots, a_{N-1}). \end{aligned}$$

For $N = 2^n$, we define $\mathbf{A}^{(0)} := \mathbf{A}$ and for $j = 0, \dots, n-1$,

$$\begin{aligned} \mathbf{A}^{(j+1)} &:= (1 - u_j) \cdot \mathbf{A}_0^{(j)} + u_j \cdot \mathbf{A}_1^{(j)} \\ &=: \left(a_0^{(j+1)}, \dots, a_{2^{n-1-j}-1}^{(j+1)} \right). \end{aligned}$$

That is, for all $i = 0, \dots, 2^{n-1-j} - 1$,

$$a_i^{(j+1)} = (1 - u_j) \cdot a_{2i}^{(j)} + u_j \cdot a_{2i+1}^{(j)}.$$

One can now prove, via induction on j , that

$$a_{i_{n-1} \cdot 2^{n-j-1} + \dots + i_j \cdot 2^0}^{(j)} = \sum_{\ell_0, \dots, \ell_{j-1} \in \{0,1\}} a_{i_{n-1} \cdot 2^{n-1} + \dots + i_j \cdot 2^j + \ell_{j-1} \cdot 2^{j-1} + \dots + \ell_0 \cdot 2^0} \cdot \prod_{k=0}^{j-1} L_{\ell_k, \{0,1\}}(u_k),$$

for all $j = 1, \dots, n$ and $i_j, \dots, i_{n-1} \in \{0,1\}$, where

$$L_{\ell_k, \{0,1\}}(X) := \ell_k X + (1 - \ell_k)(1 - X)$$

is the linear polynomial interpolating the points $(0, 1 - \ell_k)$, $(1, \ell_k)$. In particular,

$$v := a_0^{(n)} = \sum_{\ell_0, \dots, \ell_{n-1} \in \{0,1\}} a_{\ell_{n-1} \cdot 2^{n-1} + \dots + \ell_0 \cdot 2^0} \cdot \prod_{k=0}^{n-1} L_{\ell_k, \{0,1\}}(u_k).$$

Now set

$$f := \varphi^{-1} \left(\sum_{i=0}^{2^n-1} a_i X^i \right) \in \mathbb{F}[X_0, \dots, X_{n-1}]^{\leq 1}.$$

Then

$$f = \sum_{\ell_0, \dots, \ell_{n-1} \in \{0,1\}} a_{\ell_{n-1} \cdot 2^{n-1} + \dots + \ell_0 \cdot 2^0} \cdot \prod_{k=0}^{n-1} L_{\ell_k, \{0,1\}}(X_k),$$

as both assign

$$(\ell_0, \dots, \ell_{n-1}) \mapsto a_{\ell_{n-1} \cdot 2^{n-1} + \dots + \ell_0 \cdot 2^0},$$

for all $(\ell_0, \dots, \ell_{n-1}) \in \{0,1\}^n$. Therefore, $v = f(u_0, \dots, u_{n-1})$.

2.2 Interactive Oracle Proof

We continue with all our notation of §2.1. Given that φ transforms our multilinear polynomial f into a univariate polynomial, we require some univariate Polynomial Commitment Scheme. However, the exact choice of scheme is not important to us, as we can assume it operates with perfect soundness and still show that the optimized Gemini protocol is unsound. In the following protocol, we assume that each univariate polynomial can be perfectly queried at any given point.

COM(f, n) :

$$f^{(0)} \leftarrow \varphi(f) := \sum_{i=0}^{2^{n-j}-1} a_i X^i.$$

Send oracle $f^{(0)}$.

PROVE ($\mathbf{u} = (u_0, \dots, u_{n-1}), v, f$) $\rightarrow \mathbf{V_F}^{f^{(\cdot)}}(\mathbf{u}, v)$:

If $n > 1$

For $(j = 1, \dots, n-1) \left\{ f^{(j)} \leftarrow \sum_{i=0}^{2^{n-j}-1} a_i^{(j)} X^i \right\}$.

Send oracles $(f^{(1)}, \dots, f^{(n-1)})$.

$\mathbf{V_F}^{f, f^{(0)}, f^{(1)}, \dots, f^{(n-1)}} :$

$\rho \leftarrow_{\$} \mathbb{F}$

$v_0 \leftarrow v$

If $n > 1$

For $j = n-1$ down to 1

Return 0 if $\rho^{2^j}(1 - u_j) + u_j = 0$

Query $f^{(j)}$ at $-\rho^{2^j}$

Compute

$$v_{n-j} \leftarrow \left(2 \cdot \rho^{2^j} \cdot v_{n-(j+1)} - \left(\rho^{2^j}(1 - u_j) - u_j \right) f^{(j)} \left(-\rho^{2^j} \right) \right) \cdot \left(\rho^{2^j}(1 - u_j) + u_j \right)^{-1}.$$

Query $f^{(0)}$ at ρ and $-\rho$

Return 1 if

$$2 \cdot \rho \cdot v_{n-1} = (1 - u_0) \cdot \rho \cdot \left(f^{(0)}(\rho) + f^{(0)}(-\rho) \right) + u_0 \cdot \left(f^{(0)}(\rho) - f^{(0)}(-\rho) \right)$$

and 0 otherwise.

Remark 1 We note that in the original Gemini protocol [9] the verifier queries $f^{(j)}$ at ρ and $-\rho$ and $f^{(j+1)}$ at ρ^2 , for each $j \in \{0, \dots, n-1\}$. (Here, $f^{(n)}$ is the constant multilinear polynomial v and the verifier just sets $f^{(n)}(\rho^2) := v$ rather than doing any queries.) The verifier then accepts if

$$f^{(j+1)}(\rho^2) = \frac{f^{(j)}(\rho) + f^{(j+1)}(-\rho)}{2} + \frac{f^{(j)}(\rho) - f^{(j)}(-\rho)}{2\rho},$$

for all $j \in \{0, \dots, n-1\}$. The optimized version of Gemini is an attempt to reduce the number of queries made to the $f^{(j)}$'s.

We now return to the optimized version of Gemini.

Completeness. As written above, the verifier rejects if it generates a value ρ such that $\rho^{2^j}(1 - u_j) + u_j = 0$ for some $j = 0, \dots, n-1$, i.e., a value ρ that is a 2^j -th root of $-u_j(1 - u_j)^{-1}$. However, the probability that ρ is a 2^j -th root of $-u_j(1 - u_j)^{-1}$ for some $j \in \{0, \dots, n-1\}$ is at most $\left(\sum_{j=0}^{n-1} 2^j\right) / |\mathbb{F}| = (N - 1) / |\mathbb{F}|$. Assuming that $N / |\mathbb{F}|$ is negligible (as a function of λ), this completeness error is negligibly close to 1.

Since, for all $j = 0, \dots, n-1$ and all $i = 0, \dots, 2^{n-j-1} - 1$,

$$a_i^{(j+1)} = (1 - u_j) \cdot a_{2i}^{(j)} + u_j \cdot a_{2i+1}^{(j)},$$

for any $x \in \mathbb{F}$,

$$\begin{aligned} 2 \cdot x \cdot f^{(j+1)}(x^2) &= (1 - u_j) \cdot x \cdot \left(f^{(j)}(x) + f^{(j)}(-x)\right) \\ &\quad + u_j \cdot \left(f^{(j)}(x) - f^{(j)}(-x)\right). \end{aligned}$$

That is because the polynomial $f_0^{(j)} := \sum_{i=0}^{2^{n-j-1}-1} a_{2i}^{(j)} X^i$ satisfies the relation

$$2f_0^{(j)}(X^2) = f^{(j)}(X) + f^{(j)}(-X).$$

Similarly, the polynomial $f_1^{(j)} := \sum_{i=0}^{2^{n-j-1}-1} a_{2i+1}^{(j)} X^i$ satisfies the relation

$$2Xf_1^{(j)}(X^2) = f^{(j)}(X) - f^{(j)}(-X).$$

This observation implies that, for all $j = n-1, \dots, 0$,

$$v_{n-(j+1)} = f^{(j+1)}(\rho^{2^{j+1}})$$

and that

$$\begin{aligned} 2 \cdot \rho \cdot v_{n-1} &= (1 - u_0) \cdot \rho \cdot \left(f^{(0)}(\rho) + f^{(0)}(-\rho)\right) \\ &\quad + u_0 \cdot \left(f^{(0)}(\rho) - f^{(0)}(-\rho)\right), \end{aligned}$$

which shows that the protocol is complete.

3 Attack on soundness

In this section, we outline the lack of soundness of the optimized Gemini protocol. For the avoidance of confusion the symbols used in the protocol outlined in §2.2 will denote the values when the protocol is performed honestly.

We denote by $g^{(0)}, \dots, g^{(n-1)}$ the polynomials sent by a dishonest prover $\mathbf{P}$. Since a commitment to $g^{(0)}$ is sent during the commit phase of the protocol, we assume $g^{(0)} = f^{(0)}$. We use w to denote $\mathbf{P}$'s claimed evaluation of f at $\mathbf{u}$ and $w_0, \dots, w_{n-1}$ the values calculated by the verifier in each round while engaged in the protocol with $\mathbf{P}$. In other words, $w =: w_0, \dots, w_{n-1}$ are what we call $v =: v_0, \dots, v_{n-1}$ when the protocol is performed honestly.

3.1 $n = 3$ case

Until further notice, we will fix $n = 3$. Let $\lambda \in \mathbb{F}$. We set $g^{(1)} := f^{(1)} + \epsilon^{(1)}$ and $g^{(2)} := f^{(2)} + \epsilon^{(2)}$, where

$$\begin{aligned}\epsilon^{(1)}(X) &:= 4\lambda X[(1 - u_1)X - u_1] \\ \epsilon^{(2)}(X) &:= 2\lambda[(1 - u_1)^2 X - u_1^2].\end{aligned}$$

Note that $\epsilon^{(i)}$ (and hence $g^{(i)}$) has degree that of $f^{(i)}$, for $i = 1, 2$. In addition, we set

$$w_0 := v_0 + 2\lambda[(1 - u_1)^2 u_2 - u_1^2(1 - u_2)].$$

Note that v_0 is the actual evaluation of the multi-linear polynomial at $(u_0, \dots, u_{n-1})$ and so does not depend on ρ .

We claim that the verifier will accept the proof with the above values. Since λ is arbitrary (and as long as $(1 - u_1)^2 u_2 - u_1^2(1 - u_2)$ is non-zero), we can open the multi-linear polynomial to any evaluation.

The verifier will calculate

$$\begin{aligned}w_1 &:= \frac{2\rho^4 w_0 - (\rho^4(1 - u_2) - u_2)g^{(2)}(-\rho^4)}{\rho^4(1 - u_2) + u_2} \\ &= \frac{2\rho^4 v_0 - (\rho^4(1 - u_2) - u_2)f^{(2)}(-\rho^4)}{\rho^4(1 - u_2) + u_2} \\ &\quad + \frac{4\lambda\rho^4[(1 - u_1)^2 u_2 - u_1^2(1 - u_2)] - (\rho^4(1 - u_2) - u_2)\epsilon^{(2)}(-\rho^4)}{\rho^4(1 - u_2) + u_2} \\ &= v_1 + \frac{4\lambda\rho^4[(1 - u_1)^2 u_2 - u_1^2(1 - u_2)] - (\rho^4(1 - u_2) - u_2)\epsilon^{(2)}(-\rho^4)}{\rho^4(1 - u_2) + u_2}.\end{aligned}$$

In the next round the verifier will calculate

$$w_2 := \frac{2\rho^2 w_1 - (\rho^2(1 - u_1) - u_1)g^{(1)}(-\rho^2)}{\rho^2(1 - u_1) + u_1}$$

and substituting our expressions for w_1 and $g^{(1)}$, gives

$$\begin{aligned}w_2 &= \frac{2\rho^2 v_1 - (\rho^2(1 - u_1) - u_1)f^{(1)}(-\rho^2)}{\rho^2(1 - u_1) + u_1} \\ &\quad + \frac{8\lambda\rho^6[(1 - u_1)^2 u_2 - u_1^2(1 - u_2)] - 2\rho^2(\rho^4(1 - u_2) - u_2)\epsilon^{(2)}(-\rho^4)}{(\rho^2(1 - u_1) + u_1)(\rho^4(1 - u_2) + u_2)} \\ &\quad - \frac{(\rho^2(1 - u_1) - u_1)\epsilon^{(1)}(-\rho^2)}{\rho^2(1 - u_1) + u_1} \\ &= v_2 + \frac{8\lambda\rho^6[(1 - u_1)^2 u_2 - u_1^2(1 - u_2)] - 2\rho^2(\rho^4(1 - u_2) - u_2)\epsilon^{(2)}(-\rho^4)}{(\rho^2(1 - u_1) + u_1)(\rho^4(1 - u_2) + u_2)} \\ &\quad - \frac{(\rho^2(1 - u_1) - u_1)\epsilon^{(1)}(-\rho^2)}{\rho^2(1 - u_1) + u_1}.\end{aligned}$$

For the protocol to pass, we need $w_2 = v_2$, as this is checked by querying $g^{(0)} = f^{(0)}$. This holds if and only if

$$8\lambda\rho^6[(1-u_1)^2u_2 - u_1^2(1-u_2)] - 2\rho^2(\rho^4(1-u_2) - u_2)\epsilon^{(2)}(-\rho^4) - (\rho^2(1-u_1) - u_1)\epsilon^{(1)}(-\rho^2)(\rho^4(1-u_2) + u_2) = 0. \quad (2)$$

We evaluate each term in (2) individually. Firstly,

$$\begin{aligned} & 2\rho^2(\rho^4(1-u_2) - u_2)\epsilon^{(2)}(-\rho^4) \\ &= 4\lambda\rho^2(\rho^4(1-u_2) - u_2)[-(1-u_1)^2\rho^4 - u_1^2] \\ &= -4\lambda\rho^{10}(1-u_1)^2(1-u_2) \\ &\quad - 4\lambda\rho^6u_1^2(1-u_2) + 4\lambda\rho^6(1-u_1)^2u_2 + 4\lambda\rho^2u_1^2u_2. \end{aligned} \quad (3)$$

Also,

$$\begin{aligned} & (\rho^2(1-u_1) - u_1)\epsilon^{(1)}(-\rho^2)(\rho^4(1-u_2) + u_2) \\ &= -4\lambda\rho^2(\rho^2(1-u_1) - u_1)[-(1-u_1)\rho^2 - u_1](\rho^4(1-u_2) + u_2) \\ &= 4\lambda\rho^2(\rho^2(1-u_1) - u_1)[(1-u_1)\rho^2 + u_1](\rho^4(1-u_2) + u_2) \\ &= 4\lambda\rho^2(\rho^4(1-u_1)^2 - u_1^2)(\rho^4(1-u_2) + u_2) \\ &= 4\lambda\rho^{10}(1-u_1)^2(1-u_2) + 4\lambda\rho^6(1-u_1)^2u_2 - 4\lambda\rho^6u_1^2(1-u_2) \\ &\quad - 4\lambda\rho^2u_1^2u_2. \end{aligned} \quad (4)$$

Finally, plugging in (3) and (4) we have that (2) holds.

3.2 General case

In this section we briefly discuss the general $n > 2$ case. Since we know the protocol lacks soundness in the $n = 3$ case, for the general case, the dishonest prover behaves similarly to the $n = 3$ case for the first two rounds and then proceeds “honestly” in the remaining $n - 3$ rounds.

More specifically, for any $\lambda \in \mathbb{F}$ and a claimed evaluation of

$$w_0 := v_0 + 2\lambda[(1-u_{n-2})^2u_{n-1} - u_{n-2}^2(1-u_{n-1})],$$

the prover sets $g^{(i)} := f^{(i)}$, for $1 \leq i \leq n - 3$, $g^{(n-2)} := f^{(n-2)} + \varepsilon^{(1)}$ and $g^{(n-1)} := f^{(n-1)} + \varepsilon^{(2)}$. Similarly to §3.1, the verifier will calculate

$$\begin{aligned} w_1 = v_1 + & \frac{4\lambda\rho^{2^{n-1}}[(1-u_{n-2})^2u_{n-1} - u_{n-2}^2(1-u_{n-1})]}{\rho^{2^{n-1}}(1-u_{n-1}) + u_{n-1}} \\ & - \frac{(\rho^{2^{n-1}}(1-u_{n-1}) - u_{n-1})\epsilon^{(2)}(-\rho^{2^{n-1}})}{\rho^{2^{n-1}}(1-u_{n-1}) + u_{n-1}}, \end{aligned}$$

and $w_2 = v_2$. Since $g^{(i)} := f^{(i)}$ for $1 \leq i \leq n - 3$, $w_i = v_i$, for all $3 \leq i \leq n - 1$. In particular, $w_{n-1} = v_{n-1}$ and the verifier will accept.

References

1. <https://github.com/han0110/plonkish>, accessed 13th February, 2025.
2. https://github.com/AztecProtocol/barretenberg/tree/c9c89fa6d3a925f5eef77abf44029dbf939c0ecd/cpp/src/barretenberg/commitment_schemes/gemini, accessed 13th February, 2025.
3. Gemini. <https://hackmd.io/@adrian-aztec/BJxoyeCqj>, accessed 13th February, 2025.
4. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast reed-solomon interactive oracle proofs of proximity. In: Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.) ICALP 2018. LIPIcs, vol. 107, pp. 14:1–14:17. Schloss Dagstuhl (Jul 2018)
5. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 31–60. Springer, Heidelberg (Oct / Nov 2016)
6. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Proceedings of the 3rd innovations in theoretical computer science conference. pp. 326–349 (2012)
7. Boneh, D., Drake, J., Fisch, B., Gabizon, A.: Efficient polynomial commitment schemes for multiple points and polynomials. Cryptology ePrint Archive, Report 2020/081 (2020), <https://eprint.iacr.org/2020/081>
8. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 327–357. Springer, Heidelberg (May 2016)
9. Bootle, J., Chiesa, A., Hu, Y., Orrù, M.: Gemini: Elastic SNARKs for diverse environments. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 427–457. Springer, Heidelberg (May / Jun 2022)
10. Bowe, S., Grigg, J., Hopwood, D.: Halo: Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021 (2019), <https://eprint.iacr.org/2019/1021>
11. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society Press (May 2018)
12. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 677–706. Springer, Heidelberg (May 2020)
13. Chen, B., Bünz, B., Boneh, D., Zhang, Z.: HyperPlonk: Plonk with linear-time prover and high-degree custom gates. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part II. LNCS, vol. 14005, pp. 499–530. Springer, Heidelberg (Apr 2023)
14. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, P., Ward, N.P.: Marlin: Pre-processing zkSNARKs with universal and updatable SRS. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part I. LNCS, vol. 12105, pp. 738–768. Springer, Heidelberg (May 2020)
15. 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://eprint.iacr.org/2019/953>

16. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (May 2013)
17. Groth, J.: Short non-interactive zero-knowledge proofs. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 341–358. Springer, Heidelberg (Dec 2010)
18. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194. Springer, Heidelberg (Dec 2010)
19. Kattis, A., Panarin, K., Vlasov, A.: RedShift: Transparent SNARKs from list polynomial commitment IOPs. Cryptology ePrint Archive, Report 2019/1400 (2019), <https://eprint.iacr.org/2019/1400>
20. Kohrita, T., Towa, P.: Zeromorph: Zero-knowledge multilinear-evaluation proofs from homomorphic univariate commitments. *Journal of Cryptology* 37(4), 38 (2024)
21. Lee, J.: Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part II. LNCS, vol. 13043, pp. 1–34. Springer, Heidelberg (Nov 2021)
22. Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. *Journal of the ACM (JACM)* 39(4), 859–868 (1992)
23. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 2111–2128. ACM Press (Nov 2019)
24. Papamanthou, C., Shi, E., Tamassia, R.: Signatures of correct computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 222–242. Springer, Heidelberg (Mar 2013)
25. Setty, S.: Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 704–737. Springer, Heidelberg (Aug 2020)
26. Setty, S., Lee, J.: Quarks: Quadruple-efficient transparent zkSNARKs. Cryptology ePrint Archive, Report 2020/1275 (2020), <https://eprint.iacr.org/2020/1275>
27. Wahby, R.S., Tzialla, I., shelat, a., Thaler, J., Walfish, M.: Doubly-efficient zk-SNARKs without trusted setup. In: 2018 IEEE Symposium on Security and Privacy. pp. 926–943. IEEE Computer Society Press (May 2018)
28. Xie, T., Zhang, J., Zhang, Y., Papamanthou, C., Song, D.: Libra: Succinct zero-knowledge proofs with optimal prover computation. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 733–764. Springer, Heidelberg (Aug 2019)
29. Zhang, J., Xie, T., Zhang, Y., Song, D.: Transparent polynomial delegation and its applications to zero knowledge proof. In: 2020 IEEE Symposium on Security and Privacy. pp. 859–876. IEEE Computer Society Press (May 2020)
30. Zhang, Y., Genkin, D., Katz, J., Papadopoulos, D., Papamanthou, C.: A zero-knowledge version of vsql. Cryptology ePrint Archive (2017)
31. Zhao, J., Setty, S., Cui, W.: Micronova: Folding-based arguments with efficient (on-chain) verification. Cryptology ePrint Archive (2024)