# On the security of the BCTV Pinocchio zk-SNARK variant

Ariel Gabizon

Zcash

**Abstract**

The main result of this note is a severe flaw in the description of the zk-SNARK of [BCTV14]. The flaw stems from including redundant elements in the CRS as compared to that of the original Pinocchio protocol [PHGR16] that it is vital not to expose. The flaw enables creating a proof of knowledge for *any* public input given a valid proof for *some* public input. We also provide a proof of security for the [BCTV14] zk-SNARK in the generic group model, when these elements are excluded from the CRS, provided a certain linear algebraic condition is satisfied by the QAP polynomials.

## 1 Introduction

Parno et. al [PHGR16] presented a zk-SNARK construction based on the breakthrough work of [GGPR13] that they called Pinocchio. Ben-Sasson et. al [BCTV14] presented a variant of Pinocchio with the advantage of shorter verification time and verification key length. However, [BCTV14] did not present a security proof for this variant, and in fact Parno [Par15] found an attack against the [BCTV14] SNARK and suggested to mitigate it by imposing a certain linear independence condition on some of the public instance polynomials. In this note, we show a more severe attack on [BCTV14] that takes advantage of redundant elements in the proving key of [BCTV14] that should have been omitted.

### 1.1 Impacted work

[BGG17] gave for the first time a proof of security for [BCTV14]. However, the proof has an error and in fact we discovered the attack while going over the proof of [BGG17]. Any paper that cites the [BCTV14] construction as is inherits the error; the ones we found are [BBFR15, Fuc18].

### 1.2 Notation

We will be working over bilinear groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_t$ each of prime order $p$, together with respective generators $g_1$, $g_2$ and $g_T$. These groups are equipped with a non-degenerate bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$, with $e(g_1, g_2) = g_T$. We write $\mathbb{G}_1$ and $\mathbb{G}_2$ additively, and $\mathbb{G}_t$ multiplicatively. For $a \in \mathbb{F}$, we denote $[a]_1 := a \cdot g_1, [a]_2 := a \cdot g_2$. We use the notation $\mathbf{G} := \mathbb{G}_1 \times \mathbb{G}_2$ and $\mathbf{g} := (g_1, g_2)$. Given an element $h \in \mathbf{G}$, we denote by $h_1(h_2)$ the $\mathbb{G}_1(\mathbb{G}_2)$ element of $h$. We denote by $\mathbb{G}_1 \setminus \{0\}, \mathbb{G}_2 \setminus \{0\}$ the non-zero elements of $\mathbb{G}_1, \mathbb{G}_2$ and denote $\mathbf{G}^* := \mathbb{G}_1 \setminus \{0\} \times \mathbb{G}_2 \setminus \{0\}$.

We recall the zk-SNARK of [BCTV14] as described in the paper. We assume familiary of quadratic arithmetic programs. We assume $\mathbb{F}$ is a field of (prime) order $p$. We denote by $\mathbb{F}_{<d}[X]$ the set of univariate polynomials of degree smaller than $d$ over the field $\mathbb{F}$.

We assume familiarity with Quadratic Arithmetic Programs (QAPs). See e.g., Section 2.3 in [Gro16] for definitions. We use similar notation to [BCTV14], denoting by $m$ the size of the QAP, $d$ the degree and $n$ the number of public inputs. More specifically, our QAP has the form $\left\{\{A_i(X), B_i(X), C_i(X)\}_{i\in[0..m]}, Z(X)\right\}$

where $A_i, B_i, C_i \in \mathbb{F}_{<d}[X]$ and $Z$ has degree $d$.

We proceed to describe the proving system of [BCTV14].

We assume we are already given a description of the groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$, the pairing $e$, and uniformly chosen generators $g_1\mathbb{G}_1, g_2\mathbb{G}_2$, and these are all public.

$\mathsf{BCTV}$ **key generation:**

1. Sample random $\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \gamma, \beta \in \mathbb{F}^*$
2. For $i \in [0..d]$ output $\mathsf{pk}_{H,i} := \left[\tau^i\right]_1$
3. For $i \in [0..m]$ output
   (a) $\mathsf{pk}_{A,i} := [\rho_A A_i(\tau)]_1$
   (b) $\mathsf{pk}'_{A,i} := [\rho_A \alpha_A A_i(\tau)]_1$,
   (c) $\mathsf{pk}_{B,i} := [\rho_B B_i(\tau)]_2$,
   (d) $\mathsf{pk}_{B',i} := [\rho_B \alpha_B B_i(\tau)]_1$,
   (e) $\mathsf{pk}_{C,i} := [\rho_A \rho_B C_i(\tau)]_1$,
   (f) $\mathsf{pk}_{C,i} := [\rho_A \rho_B C_i(\tau)]_1$
4. Output the additional verification key elements $([\alpha_A]_2, [\alpha_B]_1, [\alpha_C]_2, [\gamma]_2, [\beta\gamma]_1, [\beta\gamma]_2, [\rho_A\rho_B \cdot Z(\tau)]_2)$

$\mathsf{BCTV}$ **prover**

The prover has in his hand a QAP solution $(x_0 = 1, x_1, \ldots, x_m)$ that coincides with the public input $x = (x_1, \ldots, x_n)$ and satisfies the following: If we define $A := \sum_{i=0}^m x_i \cdot A_i$, $B := \sum_{i=0}^m x_i \cdot B_i$, and $C := \sum_{i=0}^m x_i \cdot C_i$; then the polynomial $P := A \cdot B - C$ will be divisble by the target polynomial $Z$, and $\mathsf{P}$ can compute the polynomial $H$ of degree at most $n$ with $P = H \cdot Z$. Given the proving key, $\mathsf{P}$ computes as linear combinations of the proving key elements

1. $\pi_A := [\rho_A A(\tau)]_1$, $\pi'_A := [\alpha_A \rho_A A(\tau)]_1$.
2. $\pi_B := [\rho_B B(\tau)]_2$, $\pi'_B := [\alpha_B \rho_B B(\tau)]_1$.
3. $\pi_C := [\rho_A \rho_B C(\tau)]_1$, $\pi'_C := [\alpha_C \rho_A \rho_B C(\tau)]_1$.
4. $\pi_K := [\beta(\rho_A A(\tau) + \rho_B B(\tau) + \rho_A \rho_B C(\tau))]_1$.
5. $\pi_H := [(P(\tau)/Z(\tau))]_1$.

and outputs $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$, BCTV **verifier**

Denote the "public input component"

$$\mathsf{PI}(x) := \mathsf{pk}_{A,0} + \sum_{i=1}^{n} x_i \mathsf{pk}_{A,i} = \left[A_0(\tau) + \sum_{i=1}^{n} x_i \rho_A A_i(\tau)\right]_1$$

The verifier, using pairings and the verification key, checks the following.

1. $e(\pi'_A, g_2) = e(\pi_A, [\alpha_A]_2)$.
2. $e(\pi'_B, g_2) = e([\alpha_B]_1, \pi_B)$.
3. $e(\pi'_C, g_2) = e(\pi_C, [\alpha_C]_2)$.
4. $e(\pi_K, [\gamma]_2) = e(\mathsf{PI}(x) + \pi_A + \pi_C, [\beta\gamma]_2) \cdot e([\beta\gamma]_1, \pi_B)$.
5. $e(\mathsf{PI}(x) + \pi_A, \pi_B) = e(\pi_C, g_2) \cdot e(\pi_H, [Z(s)\rho_A\rho_B]_2)$.

### 1.3 The attack

Note that the elements $\{\mathsf{pk}'_{A,i}\}_{i\in[0..n]}$ are not used at all by the honest verifier and prover, and thus could have been omitted from the key - we show here these elements allow to replace the public input arbitrarily when starting from a valid proof. Loosely speaking, we do this by adding a factor to $\pi_A$ that "switches" the public input the proof is arguing about. The first verifier check - the "knowledge check" for $\pi_A$, should catch us; but the redundant elements allow us to add the analogous factor to $\pi'_A$ and pass the check.

Suppose we are given a valid proof $\pi = (\pi_A, \pi_B, \pi_C, \pi'_A, \pi'_B, \pi'_C, \pi_H, \pi_K)$ for a public input $(x_1, \dots, x_n) \in \mathbb{F}^n$. Choose any $(x'_1, \dots, x'_n) \in \mathbb{F}^n$.

Set

$$\eta_A := \pi_A + \sum_{i=1}^{n} (x_i - x'_i)\mathsf{pk}_{A,i}$$

$$\eta'_A := \pi'_A + \sum_{i=1}^{n} (x_i - x'_i)\mathsf{pk}'_{A,i}$$

We claim that $\pi^* := (\eta_A, \eta'_A, \pi_B, \pi'_B, \pi_C, \pi'_C, \pi_K, \pi_H)$ is a valid proof for public input $x' = (x'_1, \dots, x'_n)$.

Note that the verifier checks with public input $x'$ and proof $\pi^*$ are

1. $e(\eta'_A, g_2) = e(\eta_A, [\alpha_A]_2)$.
2. $e(\pi'_B, g_2) = e([\alpha_B]_1, \pi_B)$.
3. $e(\pi'_C, g_2) = e(\pi_C, [\alpha_C]_2)$.
4. $e(\pi_K, [\gamma]_2) = e(\mathsf{PI}(x') + \eta_A + \pi_C, [\beta\gamma]_2) \cdot e([\beta\gamma]_1, \pi_B)$.
5. $e(\mathsf{PI}(x') + \eta_A, \pi_B) = e(\pi_C, g_2) \cdot e(\pi_H, [Z(s)\rho_A\rho_B]_2)$.

We show that the five verifier equations all hold.

1. The check $e(\eta'_A, g_2) = e(\eta_A, [\alpha_A]_2)$; this is where the redundant elements crucially come into play.

   We have
$$\eta'_A = \pi'_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}$$
   So
$$e(\eta'_A, g_2) = \epsilon(\pi'_A, g_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}, g_2\right)$$
   Since $\pi$ is a valid proof, this is:
$$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}'_{A,i}, g_2\right)$$
   Using $\mathsf{pk}'_{A,i} = \alpha_A \cdot \mathsf{pk}_{A,i}$ for every $i$,
$$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\alpha_A \cdot \left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}\right), g_2\right)$$
   Using bi-linearity of the pairing:
$$= e(\pi_A, [\alpha_A]_2) \cdot e\left(\sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}, [\alpha_A]_2\right)$$
$$= e\left(\pi_A + \sum_{i=1}^{n}(x_i - x'_i)\mathsf{pk}_{A,i}, [\alpha_A]_2\right) = e(\eta_A, [\alpha_A]_2).$$

2. The second and third checks involve the unchanged $\pi_B, \pi'_B, \pi_C, \pi'_C$ and thus pass since $\pi$ was accepting.

3. The fourth and fifth equations are also identical in $\pi$ and $\pi^*$. The only difference is that the later replaces the term $\mathsf{PI}(x) + \pi_A$ with $\mathsf{PI}(x') + \eta_A$. And
$$\mathsf{PI}(x)+\pi_A = \mathsf{pk}_{A,0}+\sum_{i=1}^{n} x_i\mathsf{pk}_{A,i}+\pi_A = \mathsf{pk}_{A,0}+\sum_{i=1}^{n} x'_i\mathsf{pk}_{A,i}+\sum_{i=1}^{n}(x_i-x'_i)\mathsf{pk}_{A,i}+\pi_A = \mathsf{PI}(x')+\eta_A.$$


## Acknowledgements

We thank Sean Bowe for useful discussions.


## References


[BBFR15] M. Backes, M. Barbosa, D. Fiore, and R. M. Reischuk. ADSNARK: nearly practical and privacy-preserving proofs on authenticated data. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 271–286, 2015.

[BCTV14] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, pages 781–796, 2014.

[BGG17] S. Bowe, A. Gabizon, and M. D. Green. A multi-party protocol for constructing the public parameters of the pinocchio zk-snark. *IACR Cryptology ePrint Archive*, 2017:602, 2017.

[Fuc18] G. Fuchsbauer. Subversion-zero-knowledge snarks. In *Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part I*, pages 315–347, 2018.

[GGPR13] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 626–645, 2013.

[Gro16] J. Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, pages 305–326, 2016.

[Par15] B. Parno. A note on the unsoundness of vntinyram's SNARK. *IACR Cryptology ePrint Archive*, 2015:437, 2015.

[PHGR16] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: nearly practical verifiable computation. *Commun. ACM*, 59(2):103–112, 2016.