A note on VRFs from Verifiable Functional Encryption

Saikrishna Badrinarayanan* Vipul Goyal † Aayush Jain ‡ Amit Sahai §

Abstract

Recently, Bitansky [Bit17] and Goyal et.al [GHKW17] gave generic constructions of selectively secure verifiable random functions(VRFs) from non-interactive witness indistinguishable proofs (NIWI) and injective one way functions. In this short note, we give an alternate construction of selectively secure VRFs based on the same assumptions as an application of the recently introduced notion of verifiable functional encryption [BGJS16]. Our construction and proof is much simpler than the ones in [Bit17, GHKW17], given previous work (most notably given the constructions of verifiable functional encryption in [BGJS16]).

*University of California, Los Angeles and Center for Encrypted Functionalities. Email: saikrishna@cs.ucla.edu

†Carnegie Mellon University. Email: vipul@cmu.edu

‡University of California, Los Angeles and Center for Encrypted Functionalities. Email: aayushjainiitd@gmail.com

§University of California, Los Angeles and Center for Encrypted Functionalities. Email: sahai@cs.ucla.edu. Research supported in part from a DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C- 0205. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government

1 Introduction

Verifiable random functions, introduced by Micali, Rabin and Vadhan[MRV99] are pseudorandom functions where it is possible to verify that a given output y corresponds to the correct evaluation of the function on any input x. Recently, Bitansky [Bit17] and Goyal et.al [GHKW17] gave generic constructions of selectively secure verifiable random functions(VRFs) from non-interactive witness indistinguishable proofs (NIWI) and injective one way functions. In this short note, we give an alternate construction of selectively secure VRFs based on the same assumptions as an application of the recently introduced notion of verifiable functional encryption [BGJS16]. Our construction and proof is much simpler than the ones in [Bit17, GHKW17], given previous work (most notably given the constructions of verifiable functional encryption in [BGJS16]).

Notation: Throughout the paper, let the security parameter be λ and let PPT denote a probabilistic polynomial time algorithm. We defer the description of non-interactive commitments, puncturable pseudorandom functions and non-interactive witness indistinguishable proofs (NIWI) to Appendix A. We define the notion of secret key verifiable functional encryption in Appendix B.

2 Verifiable Random Functions

A verifiable random function[MRV99] $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Prove}, \mathsf{Verify})$ consists of the following algorithms:

- $\mathsf{Gen}(\lambda)$. The setup algorithm takes as input a security parameter λ and outputs a secret key SK and public verification key $\mathsf{VK} \in \{0,1\}^{k(\lambda)}$.
- $\mathsf{Eval}(\mathsf{SK}, x)$. The evaluation algorithm takes as input the secret key SK and a message x and outputs a string $y \in \{0,1\}^{m(\lambda)}$.
- $\mathsf{Prove}(\mathsf{SK}, x)$. The prove algorithm takes as input the secret key SK and a message x. It produces a proof π that y is consistent with the verification key VK.
- $\mathsf{Verify}(\mathsf{VK}, \pi, x, y)$. The verification algorithm takes as input the verification key VK, a message x, a string y and a proof π. It outputs 1 if the proof verifies and 0 otherwise.

The scheme has the following properties:

Definition 1. *(Completeness) Informally, it states that given any input x, if y is generated by running the honest evaluation algorithm and π is generated by running the honest prove algorithm, the verification algorithm would output 1 always.*

Definition 2. *(Uniqueness) Informally, it states that for any input x and any verification key VK, there exists at most a single y for which there is an accepting proof π.*

Definition 3. *(Selective Indistinguishability) The selective security of a VRF scheme is captured by the following game.*

- *Adversary $\mathcal{A}$ on input the security parameter outputs a challenge x^*.*
- *The challenger $\mathcal{C}$ computes $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{Gen}(1^\lambda)$. VK is handed to the adversary.*

- *Adversary can now adaptively make queries of the form $x_i \neq x^*$. On each query, $\mathcal{C}$ computes $y_i \leftarrow \mathsf{Eval}(\mathsf{SK}, x)$ and $\pi_i \leftarrow \mathsf{Prove}(\mathsf{SK}, x)$. Adversary is given (y_i, π_i).*
- *$\mathcal{C}$ samples a bit $b \in \{0,1\}$. If $b = 0$, $\mathcal{C}$ sets $y^* = \mathsf{Eval}(K, x^*)$ and otherwise sets it as a random string in the co-domain of the PRF. y^* is given to $\mathcal{A}$.*
- *Adversary $\mathcal{A}$ then outputs a bit b'. It wins the game if $b' = b$.*

We say that VRF *is selectively secure if for any polynomial time adversary $\mathcal{A}$, the adversary wins with a negligible probability in the game described above.*

We refer the reader to [Bit17] for more formal definitions of the above properties.

3 Construction

Let $\mathsf{VFE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec}, \mathsf{VerifyCT}, \mathsf{VerifyK})$ be a secret-key verifiable functional encryption scheme secure against single ciphertext and unbounded key queries. Such a scheme can be instantiated using NIWI and injective one-way functions by applying the verifiable FE transformation in [BGJS16] to the FE construction of [GVW12, SS10]. Let $\mathsf{PRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Punc})$ be a puncturable PRF. Now we describe our construction for VRF.

$\mathsf{Gen}(1^\lambda)$: On input the security parameter, first run $\mathsf{PRF.Gen}(1^\lambda) \to K$. Also run $\mathsf{Setup}(1^\lambda) \to (\mathsf{PP}_0, \mathsf{MSK})$. Compute $\mathsf{Enc}(\mathsf{PP}_0, \mathsf{MSK}, K) \to \mathsf{PP}_1$ Output $\mathsf{VK} = (\mathsf{PP}_0, \mathsf{PP}_1)$ and $\mathsf{SK} = (K, \mathsf{MSK}, \mathsf{PP}_0, \mathsf{PP}_1)$

$\mathsf{Eval}(\mathsf{SK}, x)$: On input x and $\mathsf{SK} = (K, \mathsf{MSK}, \mathsf{PP}_0, \mathsf{PP}_1)$, output $\mathsf{PRF.Eval}(K, x) \to y$.

$\mathsf{Prove}(\mathsf{SK}, x)$: On input $\mathsf{SK} = (K, \mathsf{MSK}, \mathsf{PP}_0, \mathsf{PP}_1)$ and x, compute and output $\mathsf{KeyGen}(\mathsf{PP}_0, \mathsf{MSK}, U_x) \to \pi$. Here U_x represents the function $\mathsf{PRF.Eval}(\cdot, x)$.

$\mathsf{Verify}(\mathsf{VK}, \pi, x, y)$: On input $\mathsf{VK} = (\mathsf{PP}_0, \mathsf{PP}_1)$, π, x and y checks that $\mathsf{Dec}(\mathsf{PP}_0, \pi, \mathsf{PP}_1) = y$. It outputs 1 if the check passes, otherwise it outputs 0.

We observe the following two properties of the above scheme:

Completeness. Completeness follows from the correctness of the puncturable PRF and the correctness of the verifiable FE scheme.

Uniqueness. Uniqueness follows from the verifiability of the verifiable FE scheme.

We describe the security proof in the next section.

4 Security Proof

Theorem 1. *Assuming* PRF *is a secure puncturable PRF and* VFE *is a secure secret-key verifiable functional encryption scheme against single ciphertext and unbounded key queries, the construction of* VRF *in the above section is a selectively secure verifiable random function.*

Proof. We now list the hybrids where the first hybrid corresponds to the real security game when $b = 0$ and the last hybrid corresponds to the game when $b = 1$. We show that these hybrids are computationally close.

Hyb_0:

- $\mathcal{A}$ on input the security parameter outputs a challenge x^*.
- Challenger samples a VFE key $(\mathsf{PP}_0, \mathsf{MSK})$ and a PRF key K and outputs $VK = (\mathsf{PP}_0, \mathsf{PP}_1 = \mathsf{VFE.Enc}(\mathsf{PP}_0, \mathsf{MSK}, K))$. It sends VK to the adversary.
- Each query $x_i \neq x^*$ is handled as in the algorithm. Namely, compute $y_i \leftarrow \mathsf{PRF.Eval}(K, x_i)$ and compute $\pi_i \leftarrow \mathsf{VFE.KeyGen}(\mathsf{PP}_0, \mathsf{MSK}, U_{x_i})$.
- Adversary is now given $y^* = \mathsf{PRF.Eval}(K, x^*)$ and then it outputs b'. It wins the game if $b' = 0$.

Hyb_1: This hybrid is the same as the previous one except that key K is punctured at x^* and then the punctured key K_{x^*} is used to compute VK and the other responses. The challenge is given out as $y^* = \mathsf{PRF.Eval}(K, x^*)$.

Hyb_2: This hybrid is the same as the previous one except that y^* is sampled uniformly from the co-domain of the PRF.

Hyb_3: This hybrid is the same as the previous one except that the original key K is used to compute VK and other responses.

We now provide short indistinguishability arguments:

1. Hyb_0 is indistinguishable from Hyb_1 due to the selective security of the VFE and correctness property of PRF.
2. Hyb_1 is indistinguishable from Hyb_2 due to the security of the puncturable PRF.
3. Hyb_2 is indistinguishable from Hyb_3 due to the selective security of the VFE and correctness property of PRF.

□

Remark on Adaptive Security: We note that the construction described above can also be made to achieve adaptive security by following the overall approach given in [Bit17, GHKW17]. Specifically, adaptive security can be achieved by using a (single-key) adaptive constrained PRF for admissible hash function constraints instead of a puncturable PRF.

References

[BGJS16] Saikrishna Badrinarayanan, Vipul Goyal, Aayush Jain, and Amit Sahai. Verifiable functional encryption. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information*

Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II, pages 557–587, 2016.

[Bit17] Nir Bitansky. Verifiable random functions from non-interactive witness-indistinguishable proofs. Cryptology ePrint Archive, Report 2017/018, 2017. http://eprint.iacr.org/2017/018.

[Blu81] Manuel Blum. Coin flipping by telephone. In *Advances in Cryptology: A Report on CRYPTO 81, CRYPTO 81, IEEE Workshop on Communications Security, Santa Barbara, California, USA, August 24-26, 1981.*, pages 11–15, 1981.

[BOV07] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.

[BP15] Nir Bitansky and Omer Paneth. Zaps and non-interactive witness indistinguishability from indistinguishability obfuscation. In *TCC*, 2015.

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, pages 280–300, 2013.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. A generic approach to constructing and proving verifiable random functions. Cryptology ePrint Archive, Report 2017/021, 2017. http://eprint.iacr.org/2017/021.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for NIZK. In *CRYPTO*, 2006.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO*, 2012.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 120–130, 1999.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In *CCS*, 2010.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 475–484, 2014.

A Preliminaries

A.1 Commitment Schemes

A commitment scheme Com is a PPT algorithm that takes as input a string x and randomness r and outputs $c \leftarrow \mathsf{Com}(x, r)$. A perfectly binding and computationally hiding commitment scheme must satisfy the following properties:

- **Perfectly Binding**: Two different strings cannot have the same commitment.
  More formally, $\forall x_1 \neq x_2, r_1, r_2,\ \mathsf{Com}(x_1; r_1) \neq \mathsf{Com}(x_2; r_2)$.

- **Computational Hiding**: For all strings x_0 and x_1 (of the same length), for all *non-uniform* PPT adversaries $\mathcal{A}$, we have that:
  $|Pr[\mathcal{A}(\mathsf{Com}(x_0) = 1] - Pr[\mathcal{A}(\mathsf{Com}(x_1) = 1]| \leq \mathsf{negl}(\lambda)$

In our constructions, we will use a standard non interactive perfectly binding and computationally hiding commitment scheme. Such a commitment scheme can be based on injective one way functions[Blu81].

A.2 NIWI Proofs

We will be extensively using non-interactive witness indistinguishable proofs NIWI as provided by [GOS06].

Definition 4. *A pair of PPT algorithms $(\mathcal{P}, \mathcal{V})$ is a* NIWI *for an NP relation $\mathcal{R}_{\mathcal{L}}$ if it satisfies:*

1. *Completeness: for every $(x, w) \in \mathcal{R}_{\mathcal{L}}$, $Pr[\mathcal{V}(x, \pi) = 1 : \pi \leftarrow \mathcal{P}(x, w)] = 1$.*

2. *(Perfect) Soundness: Proof system is said to be perfectly sound if there for every $x \notin L$ and $\pi \in \{0,1\}^*$*
   $Pr[\mathcal{V}(x, \pi) = 1] = 0$.

3. *Witness indistinguishability: for any sequence $\mathcal{I} = \{(x, w_1, w_2) : w_1, w_2 \in \mathcal{R}_{\mathcal{L}}(x)\}$*
   $\{\pi_1 : \pi_1 \leftarrow \mathcal{P}(x, w_1)\}_{(x,w_1,w_2)\in\mathcal{I}} \approx_c \{\pi_2 : \pi_2 \leftarrow \mathcal{P}(x, w_2)\}_{(x,w_1,w_2)\in\mathcal{I}}$

[GOS06] provides a construction of perfectly sound non-interactive witness indistinguishable proofs based on the decisional linear (DLIN) assumption. [BOV07] also provides perfectly sound proofs (although less efficient) under a complexity theoretic assumption, namely that Hitting Set Generators against co-non deterministic circuits exist. [BP15] construct NIWI from one-way permutations and indistinguishability obfuscation.

A.3 Puncturable Psuedorandom Functions

A PRF $F : \mathcal{K}_{k\in\mathbb{N}} \times \mathcal{X} \rightarrow \mathcal{Y}_{k\in\mathbb{N}}$ is a puncturable pseudorandom function [BW13, SW14] if there is an additional key space $\mathcal{K}_p$ and three polynomial time algorithms $(F.\mathsf{setup}, F.\mathsf{eval}, F.\mathsf{puncture})$ as follows:

- $F.\mathsf{setup}(1^k)$ a randomized algorithm that takes the security parameter k as input and outputs a description of the key space $\mathcal{K}$, the punctured key space $\mathcal{K}_p$ and the PRF F.

- $F.\mathsf{puncture}(K, x)$ is a randomized algorithm that takes as input a PRF key $K \in \mathcal{K}$ and $x \in \mathcal{X}$, and outputs a key $K\{x\} \in \mathcal{K}_p$.

- $F.\mathsf{Eval}(K, x')$ is a deterministic algorithm that takes as input a punctured key $K\{x\} \in \mathcal{K}_p$ and $x' \in \mathcal{X}$. Let $K \in \mathcal{K}$, $x \in \mathcal{X}$ and $K\{x\} \leftarrow F.\mathsf{puncture}(K, x)$.

The primitive satisfies the following properties:

1. **Functionality is preserved under puncturing:** For every $x^* \in \mathcal{X}$,

$$Pr[F.\mathsf{eval}(K\{x^*\}, x) = F(K, x)] = 1$$

here probability is taken over randomness in sampling K and puncturing it.

2. **Psuedo-randomness at punctured point:** For any poly size distinguisher D, there exists a negligible function $\mu(\cdot)$, such that for all $k \in \mathbb{N}$ and $x^* \in \mathcal{X}$,

$$| Pr[D(x^*, K\{x^*\}, F(K, x^*)) = 1] - Pr[D(x^*, K\{x^*\}, u) = 1] | \leq \mu(k)$$

where $K \leftarrow F.\mathsf{Setup}(1^\mathsf{k})$, $K\{x^*\} \leftarrow F.\mathsf{puncture}(K, x^*)$ and $u \xleftarrow{\$} \mathcal{Y}_k$

B Verifiable Secret Key Functional Encryption

In this section, we define verifiable secret key functional encryption. Let $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ denote ensembles where each $\mathcal{X}_\lambda$ and $\mathcal{Y}_\lambda$ is a finite set. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ denote an ensemble where each $\mathcal{F}_\lambda$ is a finite collection of functions, and each function $f \in \mathcal{F}_\lambda$ takes as input a string $x \in \mathcal{X}_\lambda$ and outputs $f(x) \in \mathcal{Y}_\lambda$. Similar to a secret key functional encryption scheme, a verifiable secret key functional encryption scheme $\mathsf{VFE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec}, \mathsf{VerifyCT}, \mathsf{VerifyK})$ consists of the following polynomial time algorithms:

- $\mathsf{Setup}(1^\lambda)$. The setup algorithm takes as input the security parameter λ and outputs the public parameters PP and the master secret key MSK.
- $\mathsf{Enc}(\mathsf{PP}, \mathsf{MSK}, x) \rightarrow \mathsf{CT}$. The encryption algorithm takes as input a message $x \in \mathcal{X}_\lambda$, the public parameters PP and the master secret key MSK. It outputs a ciphertext CT.
- $\mathsf{KeyGen}(\mathsf{PP}, \mathsf{MSK}, f) \rightarrow \mathsf{SK}_f$. The key generation algorithm takes as input a function $f \in \mathcal{F}_\lambda$, the public parameters PP and the master secret key MSK. It outputs a function secret key SK_f.
- $\mathsf{Dec}(\mathsf{PP}, f, \mathsf{SK}_f, \mathsf{CT}) \rightarrow y$ or $\perp$. The decryption algorithm takes as input the public parameters PP, a function f, a function secret key SK_f and a ciphertext CT. It either outputs a string $y \in \mathcal{Y}$ or $\perp$. Informally speaking, PP is given to the decryption algorithm for verification purpose.
- $\mathsf{VerifyCT}(\mathsf{PP}, \mathsf{CT}) \rightarrow 1/0$. Takes as input the public parameters PP and a ciphertext CT. It outputs 0 or 1. Intuitively, it outputs 1 if CT was correctly generated using the master secret key MSK for some message x.
- $\mathsf{VerifyK}(\mathsf{PP}, f, \mathsf{SK}) \rightarrow 1/0$. Takes as input the public parameters PP, a function f and a function secret key SK. It outputs either 0 or 1. Intuitively, it outputs 1 if SK was correctly generated as a function secret key for function f.

The scheme has the following properties:

Definition 5. *(Correctness) A verifiable secret key functional encryption scheme* VFE *for* $\mathcal{F}$ *is correct if for all* $f \in \mathcal{F}_\lambda$ *and all* $x \in \mathcal{X}_\lambda$

$$\Pr\left[\begin{array}{c}(\mathsf{PP},\mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{SK}_f \leftarrow \mathsf{KeyGen}(\mathsf{PP},\mathsf{MSK},f) \\ \mathsf{Dec}(\mathsf{PP},f,\mathsf{SK}_f,\mathsf{Enc}(\mathsf{PP},\mathsf{MSK},x)) = f(x)\end{array}\right] = 1$$

Definition 6. *(Verifiability) A verifiable secret key functional encryption scheme* VFE *for* $\mathcal{F}$ *is verifiable if, for all* $\mathsf{PP} \in \{0,1\}^*$*, for all* $\mathsf{CT} \in \{0,1\}^*$*, there exists* $x \in \mathcal{X}$ *such that for all* $f \in \mathcal{F}$ *and* $\mathsf{SK} \in \{0,1\}^*$*, there exists* $f \in \mathcal{F}$ *such that: if*

$$\mathsf{VerifyCT}(\mathsf{PP},\mathsf{CT}) = 1 \text{ and } \mathsf{VerifyK}(\mathsf{PP},f,\mathsf{SK}) = 1$$

then

$$\Pr\Big[\mathsf{Dec}(\mathsf{PP},f,\mathsf{SK},\mathsf{CT}) = f(x)\Big] = 1$$

.

Remark:

B.1 Indistinguishability based Security

The indistinguishability based security notion for message hiding in a verifiable secret key functional encryption is similar to the security notion for message hiding of a secret key functional encryption scheme. For completeness, we define it below. We also consider a {full/selective} CCA secure variant where the adversary, in addition to the security game described below, has access to a decryption oracle which takes a ciphertext and a function as input and decrypts the ciphertext with an honestly generated key for that function and returns the output. The adversary is allowed to query this decryption oracle for all ciphertexts of his choice except the challenge ciphertext itself. We define the security notion for message hiding in a verifiable secret key functional encryption scheme using the following game $(\mathsf{Full}-\mathsf{IND})$ message hiding between a challenger and an adversary.
Setup Phase: The challenger runs the setup algorithm and generates $(\mathsf{PP},\mathsf{MSK}) \leftarrow \mathsf{VFE.Setup}(1^\lambda)$. The challenger then hands over the public parameters PP to the adversary. The adversary has access to two oracles which it can query in any interleaved fashion.
Function Secret Key Oracle: The adversary makes function secret key queries by submitting functions $\mathsf{f} \in \mathcal{F}_\lambda$. The challenger responds by giving the adversary the corresponding function secret key $\mathsf{SK_f} \leftarrow \mathsf{vFE.KeyGen}(\mathsf{PP},\mathsf{MSK},\mathsf{f})$.
Encryption Oracle: The adversary queries with a message $m \in \mathcal{X}_\lambda$ and gets back a ciphertext $\mathsf{CT} \leftarrow \mathsf{vFE.Enc}(\mathsf{PP},\mathsf{MSK},\mathsf{m})$.
Challenge Phase: The adversary chooses two messages $(\mathsf{m_0},\mathsf{m_1})$ of the same size (each in $\mathcal{X}_\lambda$) such that for all queried functions f to the function secret key oracle, it holds that $\mathsf{f}(\mathsf{m_0}) = \mathsf{f}(\mathsf{m_1})$. The challenger selects a random bit $\mathsf{b} \in \{0,1\}$ and sends a ciphertext $\mathsf{CT} \leftarrow \mathsf{VFE.Enc}(\mathsf{PP},\mathsf{MSK},\mathsf{m_b})$ to the adversary.
Function Secret Key Oracle: The adversary may submit additional function queries $\mathsf{f} \in \mathcal{F}_\lambda$ as long as they do not violate the constraint described above. That is, for all queries f, it must hold that $\mathsf{f}(\mathsf{m_0}) = \mathsf{f}(\mathsf{m_1})$.
Encryption Oracle: The adversary may submit additional message queries.
Guess: The adversary submits a guess b' and wins if $b' = b$. The adversary's advantage in this

game is defined to be $2 * |\mathsf{Pr}[\mathsf{b} = \mathsf{b}^{'}] - 1/2|$.

We also define the *selective* security game, which we call $(\mathsf{sel} - \mathsf{IND})$ where the adversary outputs the challenge message pair even before seeing the public parameters.

Definition 7. *A secret key verifiable functional encryption scheme* VFE *is* { selective, fully } *secure message hiding if all polynomial time adversaries have at most a negligible advantage in the* $\{\mathsf{Sel} - \mathsf{IND}, \mathsf{Full} - \mathsf{IND}\}$ *message hiding security game.*

One can also consider (q_1, q_2) secure secret-key VFE where the adversary is allowed upto q_1 and q_2 queries to the oracles respectively. Our results are general and directly extend to these restricted security models also.