Verifiable Random Functions Explainer

Overview

- This is an explainer of what a verifiable random function (VRF). It will also discuss unique signatures, as that is a related concept.
- Top section has the definition and properties of VRFs.
- Then we continue into explaining different papers: Micali/Rabin/Vadhan, Lysyanskaya, BFM88, BDMP91, FLS99, Hohenberger/Waters, It Wasn't Me, PWHNVRG17, Dodis/Yampolskiy
    - We'll explain their main theorems, main contributions, and any useful insights
    - These papers all have a VRF/Unique Signature construction, so we'll describe (at high level) the construction

Background

Verifiable random functions (VRFs) [MRV99] provides a pseudorandom output along with a publicly verifiable proof of that output's correctness. The prover owns a VRF secret key $\sk$ and a public key $\pk$. On an input x the owner of the VRF secret key can calculate both $F_{\sk}(X) = y$ and $P_{\sk} (x,y) = \pi$, where F is a pseudorandom function and P is a proving function. Anyone can use the proof π and the public key $\pk$ to check that F_{sk} was indeed computed correctly.

One idea to make a VRF is to use a pseudorandom function (PRF) along with a non-interactive zero-knowledge (NIZK) proof of the correctness of the output. But, NIZKs need to use a common reference string (CRS). A trusted party must generate the CRS and make it available to all parties in the system. It is preferable that VRFs be possible to make in the plain model; that is, with only the use of heuristic hardness assumptions.

VRFs in the plain model appear in many different works, with a variety of different hardness assumptions (see table \ref{table:comparisons}). In general, the hardness assumptions have an RSA flavor \cite{FOCS:MicRabVad99,etc}, a DH-DDH flavor~\cite{CRYPTO:Lys02,etc}, bilinear groups (which still have a DH flavor)~\cite{EC:HohWat10,etc}.

Table of Comparisons

Scheme	Has Strong Unique Provability?	Hardness assumption

FOCS:MicRabVad99		RSA'-s(k) (RSA with large primes)
Lysyanskaya02, CRYPTO		very-many-very-hard-DH
PKC:DodYam05		q-DHI, q-DBDHI
CCS:BonMonRag10		O(m)-BDH
EC:HohWat10		O(mQ)-BDHE assumption
C:Yamada17		Bilinear maps (see DodYam05, BonMonRag10)
TCC:Bitansky17	NO	NIWIs (general primitive!)
TCC:Jager15	YES	
AC:Katsumata17		Bilinear maps
TCC:HofJag15		
Rosie18, CANS		
Kohl19, PKC	YES	

Definition

- Let G, F,V be polynomial-time algorithms, where:
    - G: function generator. G(1^k) → pk, sk
    - F = (F_1, F_2): function evaluator. F_1(SK,x)= val, F_2(SK,x) = proof
    - V(PK, x, v, proof) → YES/NO. The function verifier.
- Properties:
    - Domain-range correctness
    - Complete provability
    - Unique provability: For every PK, x, v_1, v_2, proof_1, proof_2, where v_1 != v_2, either for 1 or 2: Pr[V(PK, x,v,proof)=YES] > 1- 2^{-\Omega(k)}
    - Residual pseudorandomness: Pseudorandomness ensures that when someone who does not know SK sees a VRF output val without its corresponding VRF proof pi, then val is indistinguishable from a random value.

Pseudorandomness defined using efficient statistical tests for functions

- A forecast of key topics or texts that will appear in the review

- Potentially, a description of how you found sources and how you analyzed them for inclusion and discussion in the review (more often found in published, standalone literature reviews than in lit review sections in an article or research paper)

Description of Selected Papers

Micali/Rabin/Vadhan [MRV99]

Assume that the RSA function with large prime exponents cannot be inverted in polynomial time. Then there exists a VRF from {0,1}* into {0,1}

They show a VRF from a VUF, and that you can get a VUF from RSA' (the problem described above).

To show how to go from unpredictability to pseudorandomness, they use the Goldreich-Levin hardcore bit. Given a VUF, f, the VRF f' is such that f'(x) = <f(x), r>, where r is a random string that is part of the public key.

They use a GGM "tree-like" construction.

Questions

1. How does their RSA assumption compare with the standard? Is it problematic somehow?
2. What is a Markov argument, and why do you need it to prove how to get from unpredictability to pseudorandomness? Similarly: Goldreich-Levin reconstruction algorithm.
3. In fact, what is the difference between unpredictability and pseudo randomness?
4. If using a NIZK, if you had the seed owner pick the crs R, why does it break the soundness of the system? Similarly, if the verifier selects R, why does it break the zero-knowledge property of the system?
5. You can turn a probabilistic signature scheme into a deterministic one if the signer uses a GGM pseudorandom oracle (what is?) to replace the randomness used. How does this work? Why is this not enough to get unique provability?
6. Why is proposition 2, increasing the input length, necessary?

Lysyanskaya (Unique Signatures focus)

- Uses the Very -Many-DH-very Hard assumption
- Also has propositions to show how to construct VRFs from VUFs (like MRV, above)
- Tree-like construction.

- message space consists of codewords of an error correcting code that can correct a constant fraction of errors.
- root of tree labeled w/ g, Generator of a group where DH hard, DDH easy
- They call it the “PRF made public” paradigm.
- unique signature is also VUF using the very-many-DH-very-hard assumption.
-

Questions

1. VRFS can be viewed as a commitment to an exponential number of bits. What does that mean?
2. What is the separation between VRF and unique signatures? Is US much harder than VRF for some reason?

PWHNVRG17

- Has an elliptic curve based VRF
- Considers a VRF proof as as a “full-domain hash” construction. The output is simply the cryptographic hash of the VRF proof. (at least in the RSA version)
- Replaces the RSA-style assumption with an ECDSA assumption.
    - It doesn’t port over right away from RSA to ECDSA, because instead of having a deterministic

Dodis/Yampolskiy

- Note that prior work: Use an inefficient Goldreich-Levin hardcore bit [MRV99, Lys02]
- Construct a verifiable unpredictable function (VUF), whose output is hard to guess but not necessarily random. Use Goldreich-Levin bit to convert a VUF into a VRF.
- Also: Inputs need to be encoded in a special way [MRV99, Lys02, Dod03]. [MRV99]: inputs are first mapped into primes[Lys02, Dod03]: inputs are mapped to codewords of an error-correcting code
- First paper to avoid using the VUF/Goldreich-Levin, instead goes directly, doesn’t need special encoding.
- Uses bilinear groups, 2 assumptions
- ¨ q-DHI assumption: given $(g, g^x, \ldots, g^{(xq)})$, it is hard to compute $g^{1/x}$ [MSK02]
- ¨ q-DBDHI assumption: given $(g, g^x, \ldots, g^{(xq)})$,   it is hard to distinguish $e(g,g)^{1/x}$ from random [BB04]
-
-

Hohenberger/Waters

- Based on bilinear functions (like DY, above)
- Proof by partition, “all-but-one” partition

TCC:Jager15

(copied from their paper): VRFs were introduced by Micali, Rabin, and Vadhan [29], along with verifiable unpredictable functions (VUFs), a generic conversion from VUFs to VRFs based on Goldreich-Levin hard-core predicates [22], and a VUF-construction (with small input space) based on the RSA assumption. Specific, number-theoretic constructions of VRFs can be found in [29,28,16,17,1,25,9]. Note that most of these constructions either do not achieve full adaptive security for large input spaces, or are based on much stronger, interactive complexity assumptions. In particular, the VRF construction of Dodis [16] with outer error-correcting code is based on a q-type assumption (there called the sf-DDH assumption of order q) with q = O(log k), but this assumption is interactive. We wish to avoid interactive assumptions to prevent circular arguments, as explained by Naor [32].

Questions:

1. What does it mean that most constructions do not achieve “full adaptive security for large input spaces”?
2. Why does achieving full adaptive security relate to using interactive complexity assumptions?
3. Why do interactive assumptions cause circular arguments?

Niehues PKC 21

(2021-217)

- Uses Yamada’s VRF
- Not actually a paper about a new VRF, but discusses meta-reduction techniques

Yamada's VRF (CRYPTO '17)

affine functions & balancing
q -DBDHI assumption.