# Verifying Snapping Mechanism

January 20, 2020

In order to verify the differential privacy proeprty of an implementation of the snapping mechanism [5], we follow the logic rules designed from [1] and the floating point error semantics from [7, 4, 2, 6].

## 1 Preliminary Definitions

**Definition 1 (Laplce mechanism [3])**
Let $\epsilon > 0$. The Laplace mechanism $\mathcal{L}_\epsilon$: $\mathbb{R} \to \mathsf{Distr}(\mathbb{R})$ is defined by $\mathcal{L}(t) = t + v$, where $v \in \mathbb{R}$ is drawn from the Laplace distribution $\mathsf{laplce}(\frac{1}{\epsilon})$.

## 2 Syntax

Following are the syntax of the system. The circled operators are rounded operation in floating point computation.

$$
\begin{array}{llcl}
\text{Expr.} & e & ::= & c \mid x \mid f(x) \mid e_1 \oplus e_2 \mid e_1 \otimes e_2 \mid e_1 \ominus e_2 \mid e_1 \oslash e_2 \mid \text{ⓛⓝ}(e) \mid x \xleftarrow{\$} \mu \\
\text{Value} & v & ::= & c \\
\text{Distribution} & \mu & ::= & \mathsf{laplce} \mid \mathsf{unif} \mid \mathtt{bernoulli} \\
\text{Error} & err & ::= & (e_1, e_2) \\
\text{Condition} & \Phi & ::= & \mathtt{true} \mid \mathtt{false} \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2
\end{array}
$$

**Definition 2** ($\mathsf{Snap}(a) : A \to \mathsf{Distr}(B)$)
The ideal Snapping mechanism $\mathsf{Snap}(a)$ is defined as:

$$u \xleftarrow{\$} \mu; y = \text{ⓛⓝ}(u) \oslash \epsilon; s \xleftarrow{\$} \{-1, 1\}; z = s \otimes y; x = f(a); w = x \oplus z; w' = \lfloor w \rceil_\Lambda; r = \mathsf{clamp}_B(w')$$

where $f$ is the query function over input $a \in A$, $\epsilon$ is the privacy budget, $B$ is the clampping bound and $\Lambda$ is the rounding argument satisfying $\lambda = 2^k$ where $2^k$ is the smallest power of 2 greater or equal to the $\frac{1}{\epsilon}$.

## 3 Semantics

The big step semantics with floating point computation error are shown in Figure. 1.

$$\frac{(e_1, err, \Phi) \Downarrow (v_1, err_1, \Phi_1) \qquad (e_2, err, \Phi) \Downarrow (v_2, err_2, \Phi_2)}{(e_1 \oplus e_2, err, \Phi) \Downarrow (v_1 + v_2, err_1 \uplus err_2 \uplus err, \Phi_1 \wedge \Phi_2 \wedge \Phi)}\ \text{PLUS} \qquad \cdots$$

Figure 1: Semantics with Floating Point Error

$$\dfrac{\dfrac{\dfrac{\dfrac{u \in \left[ⓔ^{\epsilon \otimes (x \ominus \frac{\lambda}{2} \ominus f(a))}, ⓔ^{\epsilon \otimes (x \oplus \frac{\lambda}{2} \ominus f(a))}\right) \sim u' \in \left[ⓔ^{\epsilon \otimes (x \ominus \frac{\lambda}{2} \ominus f(a'))}, ⓔ^{\epsilon \otimes (x \oplus \frac{\lambda}{2} \ominus f(a'))}\right)}{\cdots}}{\mathsf{Snap}''(a) \in [x \ominus \frac{\lambda}{2} \ominus f(a), x \oplus \frac{\lambda}{2} \ominus f(a)) \sim_{\_} \mathsf{Snap}''(a') \in [x \ominus \frac{\lambda}{2} \ominus f(a'), x \oplus \frac{\lambda}{2} \ominus f(a'))}}{\mathsf{Snap}'(a) \in [x \ominus \frac{\lambda}{2}, x \oplus \frac{\lambda}{2}) \sim_{\_} \mathsf{Snap}'(a') \in [x \ominus \frac{\lambda}{2}, x \oplus \frac{\lambda}{2})}}{\mathsf{Snap}(a) = x \sim_{\_} \mathsf{Snap}(a') = x}$$

Figure 2: Derivation of two $\mathsf{Snap}$ mechanisms: $\mathsf{Snap}(a)$, $\mathsf{Snap}(a')$

# 4 Soundness Theorems

**Theorem 1 (The $\mathsf{Snap}$ mechanism is $\epsilon-$differentially private)**
Consider $\mathsf{Snap}(a)$ defined as before, if $\mathsf{Snap}(a) = x$ given database $a$ and privacy parameter $\epsilon$, then its actual privacy loss is bounded by $\epsilon + 12x\epsilon\eta + 2\eta$

*Proof.* the proof is developed by 3 steps.

- Given $\mathsf{Snap}(a) = x$ and parameter $\epsilon$, we consider $a'$ be the adjacent database of $a$ satisfying $|f(a) - f(a')| \leq 1$. We first have the derivation for both the $\mathsf{Snap}(a) = x$ and $\mathsf{Snap}(a') = 1$ in parallel.

  The derivations differ in value of x:

  - $x = B$
  - $x \in (B, \lfloor f(a) \rceil_\Lambda)$ The derivation of this case is shown in Figure. 2
  - $\cdots$

- Following the semantics in Figure 1, we have following evaluation results.

$$u \in \left[(v_1, err_1, \Phi_1), (v_2, err_2, \Phi_2)\right) \ \sim\ u' \in \left[(v_1', err_1', \Phi_1'), (v_2', err_2', \Phi_2')\right)$$

- given that the probability is equivalent to the length of the range, we have the ratio between $u$ and $u'$ is bounded by:

$$\frac{u}{u'} \leq \frac{v_2 + err_2 - (v_1 - err_1)}{v_2' - err_2' - (v_1' + err_1')} \leq \epsilon + 12x\epsilon\eta + 2\eta$$

  By the AxUnif rule, we have the actual privacy loss is bounded by the same value.

□

## References


[1] Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. Proving differential privacy via probabilistic couplings. In *LICS 2016*.

[2] H. Becker, N. Zyuzin, R. Monat, E. Darulova, M. O. Myreen, and A. Fox. A verified certificate checker for finite-precision error bounds in coq and hol4. In *2018 Formal Methods in Computer Aided Design (FMCAD)*, 2018.

[3] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*, 2016.

[4] Matthieu Martel. Semantics of roundoff error propagation in finite precision calculations. *Higher-Order and Symbolic Computation*, 2006.

[5] Ilya Mironov. On significance of the least significant bits for differential privacy. In *CCS 2012*, 2012.

[6] Mariano Moscato, Laura Titolo, Aaron Dutle, and César A. Muñoz. Automatic estimation of verified floating-point round-off errors via static analysis. In Stefano Tonetta, Erwin Schoitsch, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security*, 2017.

[7] Tahina Ramananandro, Paul Mountcastle, Benoundefinedt Meister, and Richard Lethin. A unified coq framework for verifying c programs with floating-point computations. In *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs (CPP)*. Association for Computing Machinery, 2016.