CONJURA · OPEN PROBLEM

Everlasting UC Commitment from Malicious PUFs Without a Common Reference String

Fully malicious token model, no trusted setup

Status. Statement: AI-written from Bernardo Magri, Giulio Malavolta, Dominique Schröder, Dominique Unruh, *Everlasting UC Commitments from Fully Malicious PUFs*, IACR Cryptology ePrint Archive 2022 (paper dated June 7, 2022 on the title page), not yet checked by a human. The CRS case is settled — the paper's Theorem 31 everlastingly UC-realizes the multiple-commitment functionality in the fully malicious token model with a PUF as the honest token, given a common reference string — and what is open is whether any protocol does so with no trusted setup at all; the paper poses this as a question and predicts no answer, so the affirmative form below is a framing choice and a refutation would be an impossibility theorem. Conjecture 1 also forbids every ideal functionality other than the token functionality, which strengthens the paper's wording — it asks only about removing the CRS — so as to exclude a PKI, signature cards or any other trusted setup as well.

Category. *Secure Computation & Universal Composability.*

Conjecture (informal). *Everlasting security asks that a protocol stay secure even after the computational assumptions it used are broken: the attacker is efficient while the protocol runs, and unbounded afterwards. Composable commitments with this guarantee are known to be impossible from a common reference string or a public-key infrastructure alone, so all constructions rely on a physical assumption, and until recently they all required the physical device to be honestly manufactured. This paper gives the first everlastingly composable commitment in a model where the attacker may build arbitrarily malicious hardware tokens, including tokens that swallow and wrap other tokens, and where the honest device is only a physically uncloneable function. But the construction still needs a trusted common reference string, used by the simulator to equivocate commitments, to simulate the oblivious transfers, and to extract. The authors point out that the usual trick for removing setup by leaning on a computationally hard problem does not survive here, because the environment eventually becomes unbounded and can tell a simulated transcript from a real one; and that the one known setup-free unconditional commitment from these devices breaks in their model as soon as the attacker may nest one device inside another. Whether a commitment scheme with this guarantee exists from such devices with no trusted setup at all is what they leave open.*

1 The setting

Everlasting (also called long-term) UC security, introduced by Müller-Quade and Unruh [MQU10], bounds the attacker to be efficient during the execution of a protocol and lets the distinguisher become unbounded afterwards. It is strictly stronger than computational UC security [Can01] and strictly weaker than statistical UC security. Müller-Quade and Unruh proved that everlasting UC commitments cannot be realized from a common reference string or a public-key infrastructure alone [MQU10], which is what forced this line of work onto physical assumptions: their own positive results assume *honestly generated* hardware tokens, and they left open whether malicious tokens suffice.

On the malicious-token side, Goyal, Ishai, Mahmoody and Sahai [GIMS10] obtain unconditional UC commitments and secure

computation from malicious tokens, but their honest tokens must encapsulate other tokens, which rules out physically uncloneable functions, since a PUF is a physical object and cannot swallow another one. Ostrovsky, Scafuro, Visconti and Wadia [OSVW13] introduced the malicious-PUF model; Damgård and Scafuro [DS13] built unconditionally secure UC commitments from PUFs *without* a CRS; and Badrinarayanan, Khurana, Ostrovsky and Visconti [BKOV17] exhibited an attack showing that the [DS13] construction breaks once the adversary may generate PUFs encapsulated inside other PUFs. The token-encapsulation model itself goes back to Chandran, Goyal and Sahai [CGS08], and the UC treatment of PUFs to Brzuska, Fischlin, Schröder and Katzenbeisser [BFSK11].

The present paper unifies malicious tokens and PUFs in a single functionality that allows fully malicious, stateful, arbitrarily nesting tokens, and gives the first everlastingly UC-secure commitment scheme in that model with no honest-token encapsulation. The construction runs in the CRS model: the simulator uses the trapdoor behind the CRS to equivocate commitments, to simulate the oblivious transfers and to extract, and it must do all of this straight-line because rewinding is unavailable in UC. Two escape routes from the CRS are named and closed off. The technique of Orlandi, Ostrovsky, Rao, Sahai and Visconti [OOR+14], which replaces setup by a computationally hard problem, fails here because the environment eventually becomes unbounded and can separate a simulated trace from a real one — exactly the pressure everlasting security applies. And the setup-free construction of [DS13] is unavailable because of the [BKOV17] attack in the encapsulating-token model. The paper's other main result, an impossibility theorem for everlasting UC oblivious transfer (and hence secure computation) from non-erasable honest tokens, holds even in the presence of a trusted setup, so it does not bear on this question: it says nothing about commitment, which is strictly weaker than OT in this landscape.

Progress to date. On the positive side, the paper's own Theorem 31 achieves everlasting UC commitment in the fully malicious token model from PUFs, given a CRS, with building blocks — statistically hiding UC commitment, two-round statistically receiver-private UC oblivious transfer, statistical witness-indistinguishable argument of knowledge, strong randomness extractor, one-way permutation — instantiable from standard assumptions such as LWE. On the

negative side, everlasting UC commitment is impossible from a CRS or a PKI with no hardware at all, so any resolution must genuinely exploit the tokens. Two natural attacks on the problem are reported as failing: the technique of replacing setup by a computationally hard problem does not survive an unbounded environment, and the known setup-free unconditional PUF commitment breaks under encapsulated malicious PUFs.

2 Notation and parameters

$\lambda \in \mathbb{N}$ is the security parameter and negl denotes an unspecified negligible function, i.e. one that is eventually below $1/p$ for every positive polynomial p. *PPT* means probabilistic polynomial time in λ. For $x, x' \in \{0,1\}^n$, $\mathsf{hd}(x, x')$ is their Hamming distance. For two ensembles $X = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $Y = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ we write $X \approx Y$ to say that they are statistically indistinguishable, i.e. their statistical distance is negligible in λ. We write $x \leftarrow_{\$} \mathcal{X}$ for a uniform draw from a finite set $\mathcal{X}$ and $y \leftarrow A(\cdot)$ for the output of a randomized algorithm. The symbols γ and δ are the PUF noise and distance parameters of Definition 1, ℓ is the length of a committed message, $\mathcal{F}_{\mathsf{HToken}}^{\mathsf{PUFSamp},\mathsf{PUFEval}}$ is the hardware-token functionality of Definition 2, $\mathcal{F}_{\mathsf{MCOM}}^{S\to R,\ell}$ is the commitment functionality of Definition 3, and EXC' is the execution random variable of Definition 4. Saying that a protocol runs *in the $\mathcal{F}$-hybrid model* means its parties may invoke $\mathcal{F}$ and no other ideal functionality.

- λ: security parameter.
- ℓ: polynomial bounding the length of the committed message.
- δ: PUF noise bound; two evaluations of the same PUF on the same challenge agree up to Hamming distance δ, a distance between $\mathsf{rg}(\lambda)$-bit responses.
- γ: PUF distance parameter; the adversary is required to keep all its PUF queries at Hamming distance at least γ from the challenge it tries to predict, a distance between λ-bit challenges.
- rg: polynomial giving the bit length of a PUF response.

3 The conjecture

Definition 1 (PUF family). Let rg be a polynomial. A *PUF family* is a pair $\mathsf{PUF} = (\mathsf{PUFSamp}, \mathsf{PUFEval})$ of randomized algorithms, neither required to be efficient. On input 1^λ, $\mathsf{PUFSamp}$ outputs an index $\sigma \in \mathcal{I}_\lambda$; each σ determines, for every challenge $x \in \{0,1\}^\lambda$, a distribution $\mathcal{D}_\sigma(x)$ over $\{0,1\}^{\mathsf{rg}(\lambda)}$, and $\mathsf{PUFEval}(1^\lambda, \sigma, x)$ returns a sample drawn from $\mathcal{D}_\sigma(x)$. We require three properties.

1. (δ-reproducibility) For $\sigma \leftarrow \mathsf{PUFSamp}(1^\lambda)$, $x \leftarrow_\$ \{0,1\}^\lambda$, $y \leftarrow \mathsf{PUFEval}(1^\lambda, \sigma, x)$ and an independent $y' \leftarrow \mathsf{PUFEval}(1^\lambda, \sigma, x)$, $\Pr[\mathsf{hd}(y, y') \le \delta(\lambda)] \ge 1 - \mathsf{negl}(\lambda)$.

2. ((γ, δ)-unpredictability) For every adversary $\mathcal{A}$,

$$\Pr\Big[\, \mathsf{hd}(y, y') \le \delta(\lambda) \;\wedge\; \forall q \in Q :\ \mathsf{hd}(q, x) \ge \gamma(\lambda) \,\Big] \le \mathsf{negl}(\lambda),$$

   where $\sigma \leftarrow \mathsf{PUFSamp}(1^\lambda)$, $x \leftarrow_\$ \{0,1\}^\lambda$, $y \leftarrow \mathcal{A}^{\mathsf{PUFEval}(1^\lambda, \sigma, \cdot)}(x)$, $y' \leftarrow \mathsf{PUFEval}(1^\lambda, \sigma, x)$, and Q is the list of all oracle queries made by $\mathcal{A}$.

3. (lazy sampler) There is a polynomial-time stateful interactive Turing machine pair $(\mathsf{MSamp}, \mathsf{MEval})$ such that for every $n = \mathrm{poly}(\lambda)$ and every sequence $(x_1, \ldots, x_n)$ of challenges, the tuples $(Y_1, \ldots, Y_n)$ and $(Y'_1, \ldots, Y'_n)$ are identically distributed, where $\mathsf{st} \leftarrow \mathsf{MSamp}(1^\lambda)$ and $Y_i \leftarrow \mathsf{MEval}(x_i)$ (with MEval carrying state across calls), and $\sigma \leftarrow \mathsf{PUFSamp}(1^\lambda)$, $Y'_i \leftarrow \mathsf{PUFEval}(1^\lambda, \sigma, x_i)$.

Definition 2 (Fully malicious hardware-token functionality $\mathcal{F}_{\mathsf{HToken}}^{\mathsf{HTSamp}, \mathsf{M}_{\mathsf{honest}}}$). The functionality is parameterized by an algorithm HTSamp, a PPT Turing machine $\mathsf{M}_{\mathsf{honest}}$ and a polynomial $p(\lambda)$ bounding the running time of $\mathsf{M}_{\mathsf{honest}}$ (the functionality itself is not required to be efficient, since it places no bound on malicious token code). It runs with parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ and adversary $\mathcal{A}$, and maintains a list $\mathcal{L}$ of tokens; each token

tok has a unique identifier tok.id $\in \{0,1\}^\lambda$, an internal state tok.st, code tok.M, a list tok.children of tokens embedded inside it, an owner tok.owner, and a boolean tok.honest. It answers:

- (`create`) from $P \in \mathcal{P}$: create tok with fresh tok.id, tok.honest := true, tok.owner := P, tok.children := $\emptyset$, tok.M := $\mathsf{M}_{\mathsf{honest}}$ and (tok.st, pubinfo) $\leftarrow \mathsf{HTSamp}(1^\lambda)$; add it to $\mathcal{L}$ and return (tok.id, pubinfo) to P.
- (`createmal`, M, st, children) from $\mathcal{A}$, provided $\mathcal{A}$ owns every token in children: create tok with fresh identifier, arbitrary code M, arbitrary initial state st, the given children (whose owner becomes *embedded*), tok.honest := false and tok.owner := $\mathcal{A}$; return tok.id to $\mathcal{A}$.
- (`handover`, id, P_j) from the current owner P_i: set tok.owner := P_j and notify P_j.
- (`query`, id, q) from the current owner P: run $(a, \mathrm{st}') \leftarrow$ tok.M(tok.st, q), where tok.M is an oracle machine with one oracle per child, each oracle query being answered by evaluating that child recursively in the same way; set tok.st := st′ and return a to P.
- (`readout`, id) from $\mathcal{A}$, for a token with tok.honest = false owned by $\mathcal{A}$: return tok.st to $\mathcal{A}$.
- (`openup`, id) from $\mathcal{A}$, for a token with tok.honest = false owned by $\mathcal{A}$: remove tok from $\mathcal{L}$ and reassign each child's owner to a party of $\mathcal{A}$'s choice.

At the end of the execution the long-term output tape of the functionality records all information held by every token that is owned by $\mathcal{A}$, or owned by a token owned by $\mathcal{A}$ through any number of layers. Writing $\mathcal{F}_{\mathsf{HToken}}^{\mathsf{PUFSamp},\mathsf{PUFEval}}$ means the instantiation in which honest tokens are PUFs: HTSamp = PUFSamp and $\mathsf{M}_{\mathsf{honest}}$ is the (stateful, lazily sampled) machine computing $\mathsf{PUFEval}(1^\lambda, \sigma, \cdot)$.

Definition 3 (Multiple commitment functionality). Let S (sender) and R (receiver) be two parties and ℓ a polynomial. Upon a command $(\texttt{commit}, \mathsf{sid}, x)$ from S with $x \in \{0,1\}^{\ell(\lambda)}$, the functionality $\mathcal{F}_{\mathsf{MCOM}}^{S \to R, \ell}$ sends $(\texttt{committed}, \mathsf{sid})$ to R. Upon a command $(\texttt{unveil}, \mathsf{sid})$ from S, it sends $(\texttt{unveiled}, \mathsf{sid}, x)$ to R with the matching sid. Further commands $(\texttt{commit})$ or $(\texttt{unveil})$ with the same sid are ignored.

Definition 4 (Everlasting UC realization with long-term tapes). Work in the UC framework, in which a protocol, an adversary $\mathcal{A}$ and an environment $\mathcal{Z}$ are interactive Turing machines and $\mathcal{Z}$ outputs an arbitrary string at the end. Every instance of a machine in the protocol, other than $\mathcal{Z}$ and $\mathcal{A}$, is given an additional output tape called its *long-term output tape*. When $\mathcal{Z}$ produces its output m, the adversary $\mathcal{A}$ is invoked once more, this time with all long-term tapes, and produces an output a; set $\mathrm{EXC}'_{\pi,\mathcal{A},\mathcal{Z}}(\lambda, z) := (m, a)$. A protocol π *everlastingly UC-realizes* an ideal protocol ρ if for every PPT adversary $\mathcal{A}$ there is a PPT simulator $\mathcal{S}$ such that for every PPT environment $\mathcal{Z}$,

$$\{\mathrm{EXC}'_{\pi,\mathcal{A},\mathcal{Z}}(\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^{\mathrm{poly}(\lambda)}} \approx \{\mathrm{EXC}'_{\rho,\mathcal{S},\mathcal{Z}}(\lambda, z)\}_{\lambda \in \mathbb{N}, z \in \{0,1\}^{\mathrm{poly}(\lambda)}}.$$

Corruption is *static*: $\mathcal{Z}$ may issue corruption requests only before the protocol begins.

Conjecture 1 (everlasting UC commitment from fully malicious PUFs with no trusted setup). *There exist a polynomial ℓ, functions $\gamma, \delta : \mathbb{N} \to \mathbb{N}$, a PUF family* $\mathsf{PUF} = (\mathsf{PUFSamp}, \mathsf{PUFEval})$ *that is δ-reproducible, (γ, δ)-unpredictable and admits a lazy sampler, and a two-party protocol* Π *between a sender S and a receiver R whose honest parties are PPT, such that*

$$\Pi \textit{ everlastingly UC-realizes } \mathcal{F}_{\mathsf{MCOM}}^{S \to R, \ell} \textit{ in the } \mathcal{F}_{\mathsf{HToken}}^{\mathsf{PUFSamp}, \mathsf{PUFEval}}\textit{-hybrid model}$$

in the sense of Definition 4, against static corruption of either party, where Π *invokes no ideal functionality other than* $\mathcal{F}_{\mathsf{HToken}}^{\mathsf{PUFSamp},\mathsf{PUFEval}}$*; in particular the parties are given no common reference string and no other trusted setup, and every string they use is either sampled by a party during the execution or obtained from* $\mathcal{F}_{\mathsf{HToken}}^{\mathsf{PUFSamp},\mathsf{PUFEval}}$.

Remark 1 (what a resolution would look like). The source poses this as a question rather than as a conjecture with a direction, and Conjecture 1 records the affirmative direction so that the statement is provable or refutable; a refutation is an impossibility theorem ruling out every such Π. An affirmative resolution would in practice be conditional on computational assumptions — the paper's own building blocks come from LWE — which is legitimate for everlasting security but means the existence claim would be proved conditionally rather than absolutely. Any proposed construction has to answer the obstruction the paper names: how a straight-line simulator equivocates without a trapdoor, against an environment that becomes unbounded once the run is over.

4 Bibliography

[BFSK11] C. Brzuska, M. Fischlin, H. Schröder, S. Katzenbeisser. *Physically uncloneable functions in the universal composition framework.* Advances in Cryptology — CRYPTO 2011, LNCS 6841, pages 51–70, Springer, 2011.

[BKOV17] S. Badrinarayanan, D. Khurana, R. Ostrovsky, I. Visconti. *Unconditional UC-secure computation with (stronger-malicious) PUFs.* Advances in Cryptology — EUROCRYPT 2017, Part I, LNCS 10210, pages 382–411, Springer, 2017.

[Can01] R. Canetti. *Universally composable security: A new paradigm for cryptographic protocols.* 42nd Annual Symposium on Foundations of Computer Science, pages 136–145, IEEE Computer Society Press, 2001.

[CGS08] N. Chandran, V. Goyal, A. Sahai. *New constructions for UC secure computation using tamper-proof hardware.* Advances in Cryptology — EUROCRYPT 2008, LNCS 4965, pages 545–562, Springer, 2008.

[DS13] I. Damgård, A. Scafuro. *Unconditionally secure and universally composable commitments from physical assumptions.* Advances in Cryptology — ASIACRYPT 2013, Part II, LNCS 8270, pages 100–119, Springer, 2013.

[GIMS10] V. Goyal, Y. Ishai, M. Mahmoody, A. Sahai. *Interactive locking, zero-knowledge PCPs, and unconditional cryptography.* Advances in Cryptology — CRYPTO 2010, LNCS 6223, pages 173–190, Springer, 2010.

[MQU10] J. Müller-Quade, D. Unruh. *Long-term security and universal composability.* Journal of Cryptology, 23(4):594–671, 2010.

[OOR+14] C. Orlandi, R. Ostrovsky, V. Rao, A. Sahai, I. Visconti. *Statistical concurrent non-malleable zero knowledge.* TCC 2014: 11th Theory of Cryptography Conference, LNCS 8349, pages 167–191, Springer, 2014.

[OSVW13] R. Ostrovsky, A. Scafuro, I. Visconti, A. Wadia. *Universally composable secure computation with (malicious) physically uncloneable functions.* Advances in Cryptology — EUROCRYPT 2013, LNCS 7881, pages 702–718, Springer, 2013.