David & Goliath Oblivious Affine Function Evaluation – Asymptotically Optimal Building Blocks for Universally Composable Two-Party Computation from a Single Untrusted Stateful Tamper-Proof Hardware Token

Nico Döttling Daniel Kraschewski Jörn Müller-Quade

nico.doettling@gmail.com, kraschew@ira.uka.de, mueller-quade@kit.edu

Abstract

In a seminal work, Katz (Eurocrypt 2007) showed that parties being able to issue tamper-proof hardware can implement universally composable secure computation without a trusted setup. Our contribution to the line of research initiated by Katz is a construction for general, information-theoretically secure, universally composable two-party computation based on a single stateful tamper-proof token. We provide protocols for multiple one-time memories, multiple commitments in both directions, and also bidirectional oblivious transfer. From this, general secure two-party computation (and even one-time programs) can be implemented by known techniques. Moreover, our protocols have asymptotically optimal communication complexity.

The central part of our work is a construction for oblivious affine function evaluation (OAFE), which can be seen as a generalization of the oblivious transfer primitive: Parametrized by a finite field $\mathbb{F}$ and a dimension k, the OAFE primitive allows a designated sender to choose an affine function $f : \mathbb{F} \to \mathbb{F}^k$, such that hidden from the sender a designated receiver can learn $f(x)$ for exactly one input $x \in \mathbb{F}$ of his choice. All our abovementioned results build upon this primitive and it may also be of particular interest for the construction of garbled arithmetic circuits.

Keywords: non-interactive secure computation, universal composability, tamper-proof hardware, information-theoretic security, oblivious transfer

Contents

1 Introduction

Tamper-proof hardware tokens allow information-theoretically secure protocols that are universally composable [Can01], they can be employed for protocols in the *globalized UC* framework [HMQU05, CDPW07], and they even allow for one-time programs, i.e. circuits that can be evaluated only once [GKR08]. However, almost all known protocols employing tamper-proof hardware are either indirect, i.e. the secure hardware is used to implement commitments or zero-knowledge proofs and additional computational assumptions must be used to obtain general secure two-party computation [Kat07, CGS08, DNW08, MS08, DNW09], or a large number of devices must be used [GKR08, GIS$^+$10]. However, issuing multiple independent tamper-proof devices requires much stronger isolation assumptions. Not only the communication between the devices and the issuer must be prevented, but also the many devices must be mutually isolated. This is especially difficult as the devices are not necessarily trusted—e.g., see [BKMN09] for the difficulty of isolating two devices in one location.

We present the first construction for universally composable, information-theoretically secure two-party computation (and even one-time programs) using only a single (untrusted) tamper-proof device that can keep a state. We also provide some impossibility results, which highlight optimality of our work in several respects. A preliminary version of our work appeared in [DKMQ11].

1.1 Related work

The idea of secure computation based on separation assumptions was introduced in [BOGKW88] to construct multi-prover interactive proof systems. In particular, [BOGKW88] proposes an unconditionally secure protocol for Rabin-OT [Rab81] between two provers and a verifier. Even though this result is not explicitly stated in the context of tamper-proof hardware[1] and is proven secure in a standalone, synchronous model, we suppose that an amplified variant of the protocol of [BOGKW88] can be proven UC-secure.

The idea of explicitly using tamper-proof hardware for cryptographic purposes was introduced by [GO96], where it was shown that tamper-proof hardware can be used for the purpose of software-protection. The interest in secure hardware and separation assumptions was renewed, when it was realized by [Kat07] that universally secure multi-party computation can be based on tamper-proof hardware tokens without additional setup assumptions. However, the tamper-proof hardware must suffice strong separation conditions, even though a more recent result showed that the assumptions about the physical separation can be relaxed to some extent [DNW08, DNW09].

Generally, the work on secure multi-party computation with tamper-proof hardware assumptions can be divided in works dealing with either stateful or stateless hardware-tokens. In [Kat07] a scenario is considered where all parties can create and issue stateful tamper-proof hardware tokens. Using additional number-theoretic assumptions, [Kat07] implements a reusable commitment functionality in this scenario. Subsequently, [MS08] improved upon [Kat07] by constructing information-theoretically secure commitments in an asymmetric scenario, where only one out of two parties is able to issue stateful tamper-proof hardware tokens. Another improvement upon [Kat07] was by [CGS08] with stateless tokens, but still bidirectional token exchange and use of enhanced trapdoor permutations (eTDP). [HMQU05] use (stateless) signature cards, issued by a trusted authority, to achieve universal composability with respect to global setup assumptions [CDPW07].

[1] The authors of [BOGKW88] mention that the provers in their protocol might be implemented as bank-cards.

In [FPS+11] it is shown how set intersection can be computed securely using a single untrusted tamper-proof hardware token and additional computational assumptions.

[GKR08] show that using a minimalistic stateful tamper-proof hardware assumption called *one-time memory* (OTM), a new cryptographic primitive called *one-time program* (OTP) can be implemented, i.e. programs that can be evaluated exactly once. An OTM can be seen as a non-interactive version of the $\binom{2}{1}$-string-OT functionality: The OTM sender stores two l-bit strings on the token and sends it to the receiver party, who can arbitrarily later choose to learn one (and only one) out of the two stored values (q.v. Figure 1).

Functionality $\mathcal{F}_{\mathrm{OTM}}$

Parametrized by a string length l. The variable *state* is initialized by *state* $\leftarrow$ `waiting`.

Creation:

- Upon receiving input (s_0, s_1) from Goliath, verify that *state* = `waiting` and $s_0, s_1 \in \{0,1\}^l$; else ignore that input. Next, update *state* $\leftarrow$ `sent`, record (s_0, s_1) and send (`sent`) to the adversary.
- Upon receiving a message (`Delivery`) from the adversary, verify that *state* = `sent`; else ignore that input. Next, update *state* $\leftarrow$ `delivered` and send (`ready`) to David.

Query:

- Upon receiving input (x) from David, verify that *state* = `delivered` and $x \in \{0,1\}$; else ignore that input. Next, update *state* $\leftarrow$ `queried` and output (s_x) to David.

When a party is corrupted, the adversary is granted unrestricted access to the channel between $\mathcal{F}_{\mathrm{OTM}}$ and the corrupted party, including the ability of deleting and/or forging arbitrary messages.

Figure 1: The ideal/hybrid functionality modeling a single one-time memory (OTM). Following [MS08], we call the token issuer "Goliath" and the receiver "David"; see also Section 2.3.1.

More recently, [Kol10] implemented string-OT with stateless tamper-proof tokens, but achieved only covert security [AL07]. A unified treatment of tamper-proof hardware assumptions is proposed by [GIS+10]. Important in the context of our work, they show that in a mutually mistrusting setting, trusted OTPs can be implemented statistically secure from a polynomial number of OTMs. In [GIMS10], statistically secure commitments and statistical zero-knowledge are implemented on top of a single stateless tamper-proof token. Furthermore, if tokens can be encapsulated into other tokens, general statistically secure multi-party computation is possible in this setting. [GIMS10] also show that unconditionally secure OT cannot be realized from stateless tamper-proof hardware alone. Finally, the latest result in this research field is by [CKS+14], that combine techniques of [GIS+10] and a previous version of our work [DKMQ11], resulting in a computationally secure, constant-round protocol for OT with unlimited token reusability. They only need stateless tokens and show black-box simulatability. However, this comes at the cost of bidirectional token exchange and the assumption that collision resistant hashfunctions (CRHF) and unique signatures or, equivalently, verifiable random functions (VRF) exist.

Except for [BOGKW88], all of the above schemes based on untrusted tamper-proof hardware either use additional complexity assumptions to achieve secure two-party computation [HMQU05, Kat07, MS08, GKR08, DNW08, DNW09, Kol10, CKS+14] or a large number of hardware tokens must be issued [GKR08, GIS+10].

1.2 Our contribution

In this paper we show that general, information-theoretically secure, universally composable two-party computation is possible in a setting where a single untrusted stateful tamper-proof hardware token is issued. The main challenge we have to deal with is to prevent a maliciously programmed token from encoding previous inputs in subsequent outputs. Note that using a *stateful* token is optimal in the sense that information-theoretically secure OT cannot be realized from stateless tokens, even if one allows bidirectional token exchange at arbitrary times in an interactive protocol [GIMS10]. Our protocols allow only limited token reuse, which we show to be unavoidable. However, all our constructions can be transformed into computationally secure solutions with unlimited token reusability. For this transformation the existence of a pseudorandom number generator (PRNG) is required, but only the token receiver needs to be computationally bounded.

As an abstraction for the primitives that can be implemented in our setting, we introduce a new primitive called *sequential one-time OAFE* (seq-ot-OAFE), where the acronym "OAFE" stands for *oblivious affine function evaluation*. Basically, seq-ot-OAFE allows the sender to pick some affine functions $f_1, \ldots, f_n$ and the receiver can evaluate each f_i on exactly one input x_i of his choice. However, similar to the OTM functionality the sender is not notified when the receiver chooses x_i and learns $f_i(x_i)$. We refer to Section 2.4 for a formal description of the seq-ot-OAFE functionality and its exact relation to OTMs and OT. Our implementation of seq-ot-OAFE consists of an interactive setup phase (including transmission of the token), which is independent of the parties' inputs. Then, secure computation of $f_i(x_i)$ requires one message from the sender to the receiver and one token query by the receiver. The number n of such secure evaluations is just limited by the amount of randomness stored on the token and the amount of randomness sent by the receiver during the setup phase. Unbounded reusability can be achieved by implementing a PRNG on the token (assuming a computationally bounded receiver) and allowing occasional random messages from the receiver to the sender. We also show that for a truly non-interactive solution it is necessary and sufficient that the sender issues *two* mutually isolated tokens.

Since the OT functionality can be written as the secure evaluation of an affine function, our results for general, statistically secure, universally composable two-party computation follow by the completeness of OT [Kil88, IPS08]. However, applying the straightforward reduction of string-OT to our seq-ot-OAFE protocol results in an OT construction with quadratic size of the messages. Therefore, we provide an alternative reduction, which yields linear communication complexity and thus an optimal construction for string-OT. This optimality and the option for unbounded token reuse are both improvements upon [DKMQ11], whose constructions and security proof rely on an a priori bound for the number of token queries even in the computationally bounded setting and whose protocols have quadratic communication complexity.

We also provide a commitment protocol with optimal communication complexity based on our seq-ot-OAFE construction. In the following, we briefly discuss and sum up the features of our protocols for OTMs, commitments, and string-OT.

OTMs: We propose an information-theoretically secure construction for an arbitrary polynomial number of OTM functionalities from a single tamper-proof token. The receiver can query these OTMs only in a predefined order, but this can be considered an advantage, since the OTP construction of [GIS$^+$10] still works, whereas the *out-of-order attacks* dealt with in [GIS$^+$10] are ruled out trivially. However, for implementation of a large number of OTMs we need that our token stores a comparably large amount of data and is capable of finite field

arithmetics. Now, if these OTMs are used to implement an OTP, one might wonder why not to implement the OTP directly on the token. There are at least three good reasons to implement an OTP via OTMs. Firstly, the token can be transferred a long time before the sender chooses which OTP to send. Secondly, via OTMs one can implement *trusted* OTPs [GIS+10], i.e. sender and receiver agree on a circuit to be evaluated and only their inputs for this circuit are kept secret. The crucial security feature of a trusted OTP is that even a corrupted sender cannot change the circuit. Thirdly, since our token only needs to store random values, we can compress its size using black-box access to a PRNG.

In particular, our OTM construction has the following features:

- many OTMs (arbitrary polynomial) by a single token; upper bound fixed at initialization
- implemented OTM instances only queriable in predefined order
- optimal round complexity: two rounds using one token or one round using two tokens (plus transfer of the token(s))
- optimal communication complexity (linear in number and size of implemented OTMs)
- information-theoretic security (but token program can be compressed by PRNG)

Commitments: Our commitment construction has the following features:

- bidirectional and reusable string-commitment functionality from a single token
- token program independent of parties' inputs (thus can be fixed by sender in advance)
- unlimited reusability at the cost of a minimal complexity assumption (PRNG)
- multiple commitments with $O(1)$ rounds by one token or non-interactively by two tokens
- optimal communication complexity (linear in number and size of commitments)

To the best of our knowledge, except for [MS08] all other constructions based on tamper-proof hardware have higher communication complexity than ours *and* either use stronger complexity assumptions or have $\omega(1)$ rounds. However, the construction of [MS08] is only unidirectional (from the token issuer to the receiver). For the opposite direction they only provide an approach that becomes inherently insecure, if composed concurrently with other instances of itself using the same token. They state it as an open problem to realize multiple commitments from the token receiver to the token sender, which is now solved by our work.

String-OT: Our OT protocol enjoys exactly the same features as our commitment protocol. Therefore, we omit an explicit itemization. Instead, by Figure 2 we compare our OT protocol with prior results in the literature.

At this point, it is important to mention that optimal communication complexity for only computationally secure OT is no great achievement at all. The string length of *any* computationally secure OT protocol can be polynomially extended by standard techniques, what accordingly improves its efficiency: The sender just uses the given OT protocol for transmission of two random PRNG seeds and announces the actual OT inputs one-time pad encrypted with the respective pseudorandomness. By this simple trick and some rescaling of the security parameter, one can transform any OT protocol with polynomial communication complexity into a protocol with linear (and thus optimal) communication complexity. However, we stress that nevertheless we present the first *information-theoretically* secure construction for multiple OT with optimal communication complexity based on reusable tamper-proof hardware. Moreover, note that an analogous approach for extending the string length of commitments or OTMs would destroy composability. We discuss this in further detail in Section 4.6.

	stateless tokens			stateful tokens (simulator needs to rewind)			
	[CGS08]	[GIS+10]	[CKS+14]	[GIS+10]	[DKMQ11]	this work	
tokens	2 (bidirect.)	$\Theta(k)$	2 (bidirect.)	$\Theta(k)$	1	1	1
rounds	$\Theta(k)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$
bits sent	?	$\Omega(k^2)$	$\Omega(k^2)$	$\Theta(k^2)$	$\Theta(k^2)$	$\Theta(k)$	$\Theta(k)$
assumptions	eTDP	CRHF	CRHF,VRF	none	none	none	PRNG
reusability	unbounded	none	unbounded	none	bounded	bounded	unbounded

Figure 2: UC-secure k-bit string-OT based on tamper-proof tokens; table partly borrowed from [CKS+14]. According to [CKS+14], the CRHF-based protocols can instead be based on one-way functions (equivalent to PRNGs), using $\Theta(k/\log k)$ rounds, and the VRF assumption in [CKS+14] can be traded for bounded token reusability. For [CGS08] an explicit estimation of the communication complexity is omitted, since they use the heavy machinery of general zero-knowledge proofs, signatures, etc., and their communication complexity depends on the actual implementation.

All our constructions also have very low computation complexity. Per implemented k-bit OTM/Commitment/OT all parties and the tamper-proof token have to perform $O(1)$ finite field operations (only additions and multiplications) with field size 2^k. Additionally, the protocol variants with unlimited token reusability require that the token issuer and the token each generates $\Theta(k)$ bits of pseudorandomness per OTM/Commitment/OT instance.

1.3 Outline of this paper

The rest of this paper is organized as follows. In Section 2 we introduce some notations (Section 2.1), give a short overview of the notion of security that we use (Section 2.2), describe how our tamper-proof hardware assumption is defined in that framework (Section 2.3), and introduce our new primitive (Section 2.4), which serves as the basic building block for all other constructions. In Section 3 we show that one can implement our new primitive UC-securely from the aforementioned tamper-proof hardware assumption. In Section 4 we discuss some variants and unobvious applications of our construction. At the end of Section 4, in Section 4.6, we also briefly discuss why an only computationally secure OT protocol with optimal communication complexity is not a noteworthy result, whereas the opposite is true for commitments and OTMs. In Section 5 we argue for some impossibility results that highlight optimality of our work in several respects, give a conclusion and suggest directions for improvements and future research.

2 Preliminaries

2.1 Notations

General stuff (finite fields, naturals and power sets): By $\mathbb{F}_q$ we denote the finite field of size q. The set of all naturals including zero is denoted by $\mathbb{N}$, without zero it is denoted by $\mathbb{N}_{>0}$. The power set of any set S is denoted by $\mathcal{P}(S)$.

Outer products: Given any field $\mathbb{F}$ and $k, l \in \mathbb{N}_{>0}$, we identify vectors in $\mathbb{F}^k$ by $(k\times 1)$-matrices, so that for all $x \in \mathbb{F}^k$ and $y \in \mathbb{F}^{1\times l}$ the matrix product $xy \in \mathbb{F}^{k\times l}$ is well-defined.

Complementary matrices: Given any field $\mathbb{F}$, some $k, l \in \mathbb{N}_{>0}$ with $k < l$ and any two matrices $C \in \mathbb{F}^{(l-k)\times l}$, $G \in \mathbb{F}^{k\times l}$, we say that G is *complementary* to C, if the matrix $M \in \mathbb{F}^{l\times l}$ gen-

erated by writing G on top of C has maximal rank in the sense that $\mathrm{rank}(M) = \mathrm{rank}(C) + k$. Note that, given any $C \in \mathbb{F}^{(l-k)\times l}$, $G \in \mathbb{F}^{k\times l}$, $x \in \mathbb{F}^l$, $y \in \mathbb{F}^k$ with G complementary to C, we can always find some $x' \in \mathbb{F}^l$, such that $Cx' = Cx$ and $Gx' = y$.

Random variables and uniform distribution: Throughout all formal proofs we will mostly denote random variables by bold face characters, e.g. $\mathbf{x}$. However, for ease of presentation and better readability, we will sometimes refrain from this general convention and just write them like non-random values, e.g. x. Let $\mathbf{x} \xleftarrow{\mathrm{r}} X$ denote that $\mathbf{x}$ is uniformly random over X.

Probabilities, expected values and entropy: We denote the probability of an event $\mathcal{E}$ by $\mathbb{P}[\mathcal{E}]$. The expected value of a random variable $\mathbf{x}$ is denoted by $\mathbb{E}(\mathbf{x})$, its Shannon entropy is denoted by $\mathbb{H}_1(\mathbf{x}) = -\sum_\alpha \mathbb{P}[\mathbf{x} = \alpha] \cdot \log_2 \mathbb{P}[\mathbf{x} = \alpha]$, and its collision entropy is denoted by $\mathbb{H}_2(\mathbf{x}) = -\log_2 \left(\sum_\alpha \big(\mathbb{P}[\mathbf{x} = \alpha]\big)^2\right)$.

Statistical distance: We denote the statistical distance of two given random variables $\mathbf{x}, \mathbf{y}$ by $\Delta(\mathbf{x}, \mathbf{y})$, using the following standard notion of statistical distance:

$$\Delta(\mathbf{x}, \mathbf{y}) \;=\; \tfrac{1}{2} \sum\nolimits_\alpha \big|\mathbb{P}[\mathbf{x} = \alpha] - \mathbb{P}[\mathbf{y} = \alpha]\big|$$

Correlation of random variables: We define the following measure for the correlation of random variables. Given any two random variables $\mathbf{x}, \mathbf{y}$ that may depend on each other, we set $\iota(\mathbf{x}, \mathbf{y}) := \Delta\big((\mathbf{x}, \mathbf{y}), (\tilde{\mathbf{x}}, \tilde{\mathbf{y}})\big)$ with $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ denoting independent versions of $\mathbf{x}$ and $\mathbf{y}$ respectively. Note that $\iota(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are statistically independent. Further note that there always exists an event $\mathcal{E}$, such that $\mathbb{P}[\mathcal{E}] = 1 - \iota(\mathbf{x}, \mathbf{y})$, and conditioned to $\mathcal{E}$ the random variables $\mathbf{x}$ and $\mathbf{y}$ are statistically independent.

2.2 Framework & notion of security

We state and prove our results in the Universal-Composability (UC) framework of [Can01]. In this framework, security is defined by comparison of an *ideal model* and a *real model*. The protocol of interest is running in the latter, where an adversary $\mathcal{A}$ coordinates the behavior of all corrupted parties. In the ideal model, which is secure by definition, an ideal functionality $\mathcal{F}$ implements the desired protocol task and a simulator $\mathcal{S}$ tries to mimic the actions of $\mathcal{A}$. An environment $\mathcal{Z}$ is plugged either to the ideal or the real model and has to guess, which model it is actually plugged to. A protocol Π is a *universally composable* (UC-secure) implementation of an ideal functionality $\mathcal{F}$, if for every adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$, such that for all environments $\mathcal{Z}$ the entire view of $\mathcal{Z}$ in the real model (with Π and $\mathcal{A}$) is statistically close to its view in the ideal model (with $\mathcal{F}$ and $\mathcal{S}$). In our case the adversarial entities $\mathcal{A}, \mathcal{S}$ and the environment $\mathcal{Z}$ are computationally unbounded and a hybrid functionality $\mathcal{F}_{\mathrm{wrap}}^{\mathrm{stateful}}$ models our tamper-proof hardware assumption (q.v. Section 2.3). Note that for convenience and better readability we use the notation of [Can01] a bit sloppy. E.g., throughout this paper we omit explicit notation of party and session IDs.

2.3 Modeling tamper-proof hardware

2.3.1 The hybrid functionality $\mathcal{F}_{\mathrm{wrap}}^{\mathrm{stateful}}$

Our formulation of general stateful tamper-proof hardware resembles the meanwhile standard definitions of [Kat07, MS08]. Following [MS08], we call the token issuer "Goliath" and the receiver

"David". This naming is also motivated by the fact that all computational versions of our protocols only need David's computing power to be polynomially bounded in the security parameter; Goliath (and even the token) may be arbitrary powerful.

To model tamper-proof hardware, we employ the $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ wrapper functionality (q.v. Figure 3). Goliath provides as input a Turing machine $\mathcal{M}$ to $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$. David can then query $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ on arbitrary input words w, whereupon $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ runs $\mathcal{M}$ on input w, sends the output that $\mathcal{M}$ produced to David and stores the new state of $\mathcal{M}$. Every time David sends a new query w' to $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$, it resumes simulating $\mathcal{M}$ with its most recent state, sends the output to David and updates the stored state of $\mathcal{M}$.

Functionality $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$

The variable *state* is initialized by *state* $\leftarrow$ `wait`.

Creation:

- Upon receiving a message $(\texttt{Create}, \mathcal{M}, b)$ from Goliath, where $\mathcal{M}$ is the program of a deterministic interactive Turing machine and $b \in \mathbb{N}$, verify that *state* = `wait`; else ignore that input. Next, initialize a simulated version of $\mathcal{M}$, store b, set *state* $\leftarrow$ `sent` and send (`created`) to the adversary.
- Upon receiving a message (`Delivery`) from the adversary, verify that *state* = `sent`; else ignore that input. Next, set *state* $\leftarrow$ `execute` and send (`ready`) to David.

Execution:

- Upon receiving a message $(\texttt{Run}, w)$ from David, where w is an input word, verify that *state* = `execute`; else ignore that input. Next, write w on the input tape of the simulated machine $\mathcal{M}$ and carry on running $\mathcal{M}$ for at most b steps, starting from its most recent state. When $\mathcal{M}$ halts (or b steps have passed) without generating output, send a special symbol $\perp$ to David; else send the output of $\mathcal{M}$.

When a party is corrupted, the adversary is granted unrestricted access to the channel between $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ and the corrupted party, including the ability of deleting and/or forging arbitrary messages.

Figure 3: The wrapper functionality by which we model stateful tamper-proof hardware sent from Goliath to David. Note that delivery of the token in the creation phase is scheduled by the adversary, whereas afterwards all communication between David and the token is immediate.

This captures the following properties one expects from tamper-proof hardware. On the one hand, Goliath is unable to revoke $\mathcal{M}$ once he has sent it to David. On the other hand, David can run $\mathcal{M}$ on inputs of his choice, but the program code and state of $\mathcal{M}$ are out of reach for him, due to the token's tamper-proofness. Note that $\mathcal{M}$ does not need a trusted source of randomness, as it can be provided with a sufficiently long hard-coded random tape. Thus, w.l.o.g. we can restrict $\mathcal{M}$ to be deterministic.

For formal reasons we require that Goliath not only specifies the program code of $\mathcal{M}$, but also an explicit runtime bound $b \in \mathbb{N}$. This just ensures that even a corrupted Goliath cannot make $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ run perpetually. As we will state and prove all our results without any computational assumptions regarding the token, a corrupted Goliath may choose b arbitrarily large. However, when Goliath is honest, we will only need that the number of computing steps performed by the token is polynomial in the security parameter. We will henceforth implicitly assume that an honest Goliath always adjusts the parameter b accordingly.

2.3.2 Real world meaning of our hardware assumption and proof techniques

All our constructions can be proven UC-secure. However, the respective simulator for a corrupted Goliath needs to rewind the token and thus has to know the token code. At first glance, it might seem a rather strong assumption that a corrupted token manufacturer always knows the internal program code of his tokens. How can such a party be prevented from just passing on a token received during another protocol from some uncorrupted token issuer?

We argue that tokens can be bound to the corresponding issuer IDs by not too unrealistic assumptions. The conceptually simplest (but a bit overoptimistic) way are standardized and unforgeable token cases, branded with the respective issuer ID, and that cannot be removed without destroying the token completely. However, we can go with a bit less rigorous assumptions. We just need that undetectable token encapsulation is infeasible (e.g., since the token's weight and size would be altered) and that every honestly programmed token initially outputs its manufacturer's ID. Then, only tokens of corrupted manufacturers can be successfully passed on. Since w.l.o.g. all corrupted parties collude, now every token issuer automatically knows the internal program code of all his issued and/or passed on tokens. Infeasibility of token encapsulation is also needed by [HMQU05, Kat07, MS08, GKR08].

We also argue that using a *stateful* token does not necessarily mean a categorical disadvantage compared to protocols based on *stateless* tokens. In the literature one can find the opposite point of view, usually motivated by *resetting attacks*. These attacks only affect stateful approaches, whereas stateless approaches stay secure. By a resetting attack a corrupted token receiver tries to rewind the token (e.g. by cutting off the power supply) and then run it with new input. Such an attack, if successful, would break security of all our protocols. However, as a countermeasure the tamper-proof token could delete its secrets or just switch to a special "dead state" when a resetting attempt is detected. For the technical realization we suggest, e.g., that the state information is stored as a code word of an error correcting code and the token does not work unless the stored state information is an error-free, non-trivial code word. Anyway, we consider a thorough investigation of this issue an interesting direction for future research.

2.4 Sequential one-time OAFE and its relation to OTMs and OT

All our constructions are based on the *oblivious affine function evaluation* (OAFE) primitive, which is parametrized by a field size q and a dimension k, allows the sender to choose an affine function parametrized by two vectors $a, b \in \mathbb{F}_q^k$ and the receiver to choose a preimage $x \in \mathbb{F}_q$, and outputs the $\mathbb{F}_q^k$-vector $y := ax + b$ to the receiver. The sender's output is empty. To make the parameters explicit, we write $\mathbb{F}_q^k$-OAFE. As one can see quite easily, $\mathbb{F}_2$-OAFE and OT can be reduced to each other without any overhead (q.v. Figure 4). Note that the reductions in Figure 4 also work perfectly for $\mathbb{F}_2^k$-OAFE and k-bit string-OT respectively.

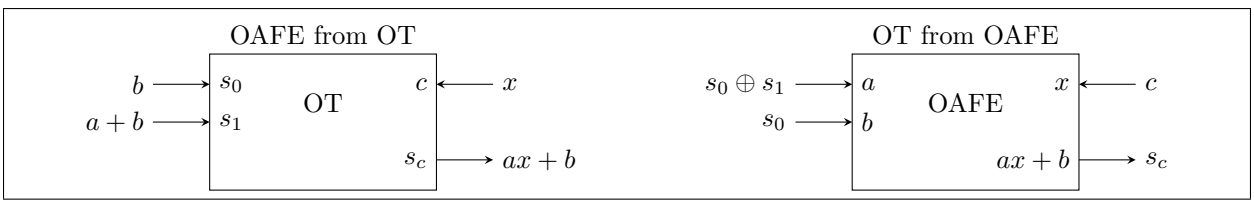

Figure 4: Reductions between bit-OT and $\mathbb{F}_2$-OAFE; protocols borrowed from [WW06].

We implement a variant of OAFE that we call "sequential one-time OAFE", or "seq-ot-OAFE" for short. By *one-time* OAFE we mean a primitive that works analogously to an OTM. The sender creates a token parametrized by $a, b \in \mathbb{F}_q^k$ and sends it to the receiver. Arbitrarily later the receiver may *once* input some $x \in \mathbb{F}_q$ of his choice into the token, whereupon the token outputs $y := ax + b$ and then terminates. *Sequential* one-time OAFE lets the sender send up to a polynomial number of single one-time OAFE tokens, but the receiver may only query them in the same order as they were sent. However, when the receiver has queried some of the tokens he already received, this does not vitiate the sender's ability to send some additional tokens, which in turn can be queried by the receiver afterwards, and so on. For a formal definition of the ideal seq-ot-OAFE functionality see Figure 5.

Functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some runtime bound n that is polynomially bounded in the security parameter $\lambda := k \log q$. The counters $j_{\mathsf{created}}, j_{\mathsf{sent}}, j_{\mathsf{queried}}$ are all initialized with 0.

Send phases:

- Upon receiving input (a, b, i) from Goliath, verify that $a, b \in \mathbb{F}_q^k$ and $i = j_{\mathsf{created}} + 1 \leq n$; else ignore that input. Next, update $j_{\mathsf{created}} \leftarrow i$, record (a, b, i) and send $(\texttt{created}, i)$ to the adversary.
- Upon receiving a message $(\texttt{Delivery}, i)$ from the adversary, verify that $i = j_{\mathsf{sent}} + 1 \leq j_{\mathsf{created}}$; else ignore that message. Next, update $j_{\mathsf{sent}} \leftarrow i$ and send $(\texttt{ready}, i)$ to David.

Choice phases:

- Upon receiving input (x, i) from David, verify that $x \in \mathbb{F}_q$ and $i = j_{\mathsf{queried}} + 1 \leq j_{\mathsf{sent}}$; else ignore that input. Next, update $j_{\mathsf{queried}} \leftarrow i$ and for the recorded tuple (a, b, i) compute $y \leftarrow ax + b$ and output (y, i) to David.

When a party is corrupted, the adversary is granted unrestricted access to the channel between $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$ and the corrupted party, including the ability of deleting and/or forging arbitrary messages.

Figure 5: The ideal functionality for sequential one-time OAFE (seq-ot-OAFE). Note that send and choice phases can be executed in mixed order with the only restriction that the i-th send phase must precede the i-th choice phase. Further note that David's notifications about Goliath's inputs in the send phases are scheduled by the adversary, whereas all messages in the choice phases are delivered immediately.

Note that the reduction protocols in Figure 4 can be adapted canonically to transform k-bit string-OTMs into $\mathbb{F}_2^k$-OAFE tokens and vice versa. Hence, using the seq-ot-OAFE functionality, a polynomial number of OTMs can be implemented very efficiently, but the receiver can query the single OTM tokens only in the same order as they were sent. However, the construction of [GIS+10] for trusted OTPs from OTMs still works, as there an honest receiver queries all OTM tokens in a fixed order anyway. Interestingly, the technical challenges dealt with in [GIS+10] arise from the fact that a malicious receiver might query the OTMs *out of order*. Moreover, the restriction to sequential access can be exploited to securely notify the sender that the receiver has already queried some OTM token. Therefor, every other OTM token is issued with purely random input from the sender and the receiver just announces his corresponding input-output tuple. A corrupted receiver

that tries to adversarially delay his OTM queries is caught cheating with overwhelming probability, as he has only a negligible chance to correctly guess the next check announcement. Thus, we can implement a polynomial number of OT instances that are perfectly secure against the OT sender and statistically secure against the OT receiver. Still, the receiver can query the single OT instances only in the same order as they were sent, but in fact this is already premised in most protocols that build on OT. Noting that OT and OAFE can be stored and reversed [Bea95, WW06, Wul07], we conclude that in the seq-ot-OAFE hybrid model OT can be implemented in both ways (from the token sender to the token receiver and vice versa).

Finally, a remark is in place. Even though seq-ot-OAFE can be used to implement several OTPs, the sequential nature of seq-ot-OAFE demands that those OTPs can only be executed in a predefined order. If one wishes to implement several OTPs that can be evaluated in random order, as many seq-ot-OAFE functionalities have to be issued. This seems an unavoidable limitation, as long as only one tamper-proof token is used. We discuss this in further detail in Section 5.5.

3 Sequential one-time OAFE from *one* tamper-proof token

3.1 Protocol construction

We want to implement seq-ot-OAFE, using only a single tamper-proof hardware token, which is not trusted by the receiver. As mentioned earlier, we call the token issuer "Goliath" and the receiver "David". The starting point for our construction is a trivial but insecure OAFE protocol with randomized sender inputs:

- Goliath chooses $2n$ random token parameters $r_1, s_1, \ldots, r_n, s_n \xleftarrow{\text{r}} \mathbb{F}_q^k$, programs the token with corresponding affine functions such that David's i-th token query x_i will be answered with $w_i := r_i x_i + s_i$, and sends the token to David.
- Arbitrarily later, David can query the token with his actual OAFE input and learn the corresponding output.

It is pretty obvious that a malicious David cannot cheat other than changing his inputs $x_1, \ldots, x_n$, which is legitimate. If, however, Goliath is corrupted, then there are two main problems. Firstly, token outputs could depend on *any previous* inputs. Secondly, there is no guarantee that each token output is indeed an *affine* function of the corresponding input. Though, the latter is only an issue if $q > 2$, as *every* function $f : \mathbb{F}_2 \to \mathbb{F}_2^k$ is affine: $f(x) = \big(f(0) + f(1)\big) \cdot x + f(0)$ for all $x \in \mathbb{F}_2$.

Our approach to solve these problems is letting Goliath announce some random linear hash values of the token's function parameters, which are then used by David for consistency checks. Hence, a malicious behaving of the token becomes detectable at the price of partly leaking secret information to David:

- Goliath chooses the i-th token parameters now from a larger[2] space, namely $r_i, s_i \xleftarrow{\text{r}} \mathbb{F}_q^{4k}$. Apart from the enlarged dimension nothing changes, i.e., the token still maps $x_i \mapsto r_i x_i + s_i$.
- Upon receiving the token, David announces a random check matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k \times 4k}$.

[2]We need to enlarge the dimension by a factor of 4 for technical reasons. We are not aware of any potential weakness of the protocol for any constant factor $1 + \alpha$ with $\alpha > 0$, but our proof techniques work only for $\alpha > 2$. In particular, the probability p' in the proof of Lemma 16 is not negligible if $\alpha \leq 2$.

- Goliath in turn announces $\tilde{r}_i := Cr_i$ and $\tilde{s}_i := Cs_i$.
- When David queries the token the i-th time, say he inputs $x_i \in \mathbb{F}_q$ and receives some output $w_i \in \mathbb{F}_q^{4k}$, he checks whether $Cw_i = \tilde{r}_i x_i + \tilde{s}_i$. If the check is not passed, David has caught Goliath cheating.

This way, we can implement some kind of "weak" OAFE, where the receiver additionally learns some linear projection of the sender's inputs, but by announcing $(\tilde{r}_i, \tilde{s}_i)$ Goliath has committed the token to affine behavior. Otherwise, if David's consistency check would be passed although the current token output w_i is not just an affine function of the latest input x_i, then the token could as well form collisions for the universal hash function C, of which it is oblivious. Moreover, we can nullify David's additional knowledge about (r_i, s_i) by multiplication with a matrix $G \in \mathbb{F}_q^{k \times 4k}$ that is *complementary* to C (cf. Section 2.1). When David just outputs Gw_i, we have implemented OAFE with random input (Gr_i, Gs_i) from Goliath and arbitrarily selectable input x_i from David. Finally, Goliath can derandomize his input to arbitrarily selectable $(a_i, b_i) \in \mathbb{F}_q^k \times \mathbb{F}_q^k$ by announcing $\tilde{a}_i := a_i - Gr_i$ and $\tilde{b}_i := b_i - Gs_i$. David then just has to replace his output by $y_i := Gw_i + \tilde{a}_i x_i + \tilde{b}_i$.

However, there is still a security hole left, as the token in round i might act honestly only on some specific input set $X_i \subsetneq \mathbb{F}_q$ or even only on some specific type of *input history*. Now, when David's inputs match this adversarially chosen specification, he will produce regular output; else an abort is caused with overwhelming probability (i.e. David produces default output). Such a behavior cannot be simulated in the ideal model, unless the simulator gathers some information about David's input. Thus, David must keep his real input x_i secret from the token (and as well from Goliath, of course). However, David's input must be reconstructible from the *joint* view of Goliath and the token, as otherwise a corrupted David could evaluate the affine function specified by Goliath's input (a_i, b_i) on more than one input x_i. Our way out of this dilemma is a linear secret sharing scheme, by which David shares his input x_i between Goliath and the token. The complete protocol now basically proceeds as follows:

- Goliath initializes the token with uniformly random parameters $r_i \xleftarrow{\text{r}} \mathbb{F}_q^{4k}$ and $S_i \xleftarrow{\text{r}} \mathbb{F}_q^{4k \times k}$.
- Upon receiving the token, David announces a random check matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k \times 4k}$ and a random share $h_i \xleftarrow{\text{r}} \mathbb{F}_q^k \setminus \{0\}$. David and Goliath also agree on some $G \in \mathbb{F}_q^{k \times 4k}$ complementary to C.
- Goliath announces the check information $\tilde{r}_i := Cr_i$ and $\tilde{S}_i := CS_i$ and the derandomization information $\tilde{a}_i := a_i - Gr_i$ and $\tilde{b}_i := b_i - GS_i h_i$, where $(a_i, b_i) \in \mathbb{F}_q^k \times \mathbb{F}_q^k$ is his OAFE input.
- David picks a random share $z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathbb{F}_q^{1 \times k} \mid \tilde{z} h_i = x_i\}$, where $x_i \in \mathbb{F}_q$ is his OAFE input. He inputs z_i into the token, whereupon the token has to compute and output $W_i := r_i z_i + S_i$. If $CW_i = \tilde{r}_i z_i + \tilde{S}_i$, David computes and outputs $y_i := GW_i h_i + \tilde{a}_i x_i + \tilde{b}_i$; else he aborts (or outputs some default value).

Now, neither Goliath nor the token can gather non-negligible information about David's OAFE input x_i. Given any set of token inputs $Z_i \subseteq \mathbb{F}_q^{1 \times k}$ adversarially chosen in advance, the hyperplanes $\{\tilde{z} \in \mathbb{F}_q^{1 \times k} \mid \tilde{z} h_i = x\}_{x \in \mathbb{F}_q}$ will partition Z_i into q subsets of roughly equal size, since h_i is chosen independently of Z_i. In other words, if the token behaves dishonestly on some input set $Z_i \subsetneq \mathbb{F}_q^{1 \times k}$, then the abort probability is practically independent of David's input x_i.

A formal description of the full protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ is given in Figure 6. We want to point out that in the formal protocol description we purposely do not exactly specify how the parameters k and q depend on the security parameter λ. We just demand that $k \cdot \log q \geq \lambda$. E.g., one can choose k to increase linearly in λ and q to be constant, or k constant and q exponential in λ. The former parameter choice results in the same message lengths (quadratic in λ) as in [DKMQ11], but with parameters chosen the latter way our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ has only linear and thus optimal communication complexity. We will exploit that in our asymptotically optimal constructions of OTMs, commitments, and string-OT.

Protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some runtime bound n that is polynomially bounded in the security parameter $\lambda := k \log q$. The setup phase is executed right at the start of the first send phase.

Setup phase:

i. Goliath chooses n random vectors $r_1, \ldots, r_n \xleftarrow{r} \mathbb{F}_q^{4k}$ and n random matrices $S_1, \ldots, S_n \xleftarrow{r} \mathbb{F}_q^{4k \times k}$, creates a token $\mathcal{T}$ with parameters $(r_1, S_1), \ldots, (r_n, S_n)$, and sends $\mathcal{T}$ to David via $\mathcal{F}_{\mathrm{wrap}}^{\mathrm{stateful}}$. The token also contains a counter j'_{queried} and Goliath maintains a counter j_{created}, both initialized with 0.

ii. Having received $\mathcal{T}$, David chooses n random vectors $h_1, \ldots, h_n \xleftarrow{r} \mathbb{F}_q^k \setminus \{0\}$ and a random matrix $C \xleftarrow{r} \mathbb{F}_q^{3k \times 4k}$, computes some $G \in \mathbb{F}_q^{k \times 4k}$ complementary to C, and sends $(C, G, h_1, \ldots, h_n)$ to Goliath. Furthermore, David initializes two counters $j_{\mathsf{queried}}, j_{\mathsf{sent}} \leftarrow 0$ and an initial flag $f_0 \leftarrow \top$.

iii. If Goliath finds G not complementary to C, he aborts the protocol.

Send phases:

1. Upon input (a_i, b_i, i) from the environment, Goliath verifies that $a_i, b_i \in \mathbb{F}_q^k$ and $i = j_{\mathsf{created}} + 1 \leq n$; else he ignores that input. Next, Goliath updates $j_{\mathsf{created}} \leftarrow i$, computes $\tilde{r}_i \leftarrow Cr_i$ and $\tilde{S}_i \leftarrow CS_i$ and $\tilde{a}_i \leftarrow a_i - Gr_i$ and $\tilde{b}_i \leftarrow b_i - GS_ih_i$, and sends $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ to David.
2. If $i = j_{\mathsf{sent}} + 1 \leq n$, David updates $j_{\mathsf{sent}} \leftarrow i$ and outputs $(\texttt{ready}, i)$; else he ignores that message.

Choice phases:

3. Upon input (x_i, i) from the environment, David verifies that $x_i \in \mathbb{F}_q$ and $i = j_{\mathsf{queried}} + 1 \leq j_{\mathsf{sent}}$; else he ignores that input. Next, he updates $j_{\mathsf{queried}} \leftarrow i$, chooses a random vector $z_i \xleftarrow{r} \{\tilde{z} \in \mathbb{F}_q^{1 \times k} \mid \tilde{z}h_i = x_i\}$ and inputs (z_i, i) into the token $\mathcal{T}$.
4. The token verifies that $z_i \in \mathbb{F}_q^{1 \times k}$ and $i = j'_{\mathsf{queried}} + 1 \leq n$; else it ignores that input. Next, the token updates $j'_{\mathsf{queried}} \leftarrow i$, computes $W_i \leftarrow r_iz_i + S_i$ and outputs W_i to David.
5. David verifies that $f_{i-1} = \top$ and $CW_i = \tilde{r}_iz_i + \tilde{S}_i$; if $W_i \notin \mathbb{F}_q^{4k \times k}$, it is treated as an encoding of the all-zero matrix in $\mathbb{F}_q^{4k \times k}$. If the check is passed, David sets $f_i \leftarrow \top$ and computes $y_i \leftarrow GW_ih_i + \tilde{a}_ix_i + \tilde{b}_i$; otherwise he sets $f_i \leftarrow \bot$ and $y_i \leftarrow 0$ $(\in \mathbb{F}_q^k)$. Then he outputs (y_i, i).

Figure 6: A protocol for sequential one-time OAFE, using *one* tamper-proof token. Note that several send and choice phases can be executed in mixed order with the only restriction that an honest David will not enter the i-th choice phase before the i-th send phase has been completed.

3.2 Limitations for the OAFE parameters k and q

As mentioned above, there is some freedom of choice how the parameters k and q in our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ depend on the security parameter λ. In particular, we need for our security proof that $k \cdot \log q \geq \lambda$ and $k \geq 5$. The latter condition, however, is an artifact from our proof techniques and probably not tight. If $k = 1$, the protocol is not UC-secure against a corrupted sender (see Remark 1 below), but for $2 \leq k \leq 4$ we are not aware of any potential attack (provided that q is sufficiently large). Still, $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ with $k < 5$ can be implemented from $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ with $k = 5$ straightforwardly and the reduction protocol itself has only linear overhead. Thus, the asymptotic optimality of our construction with $k = 5$ does directly carry over to the case that $k < 5$.

Remark 1. Our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ is not UC-secure against a corrupted Goliath, if $k = 1$.

Proof. The problem with $k = 1$ basically arises from the fact that in this case Goliath's shares h_i of David's inputs x_i are invertible field elements. Consider a maliciously programmed token that stops functioning after the first choice phase if $z_1 \in Z$ for some adversarially chosen $Z \subseteq \mathbb{F}_q$, e.g. with $|Z| = \frac{q}{2}$, and otherwise just follows the protocol. Since Goliath knows Z and learns h_1 during the protocol, he also knows exactly on which inputs x_1 the token breaks, namely if $x_1 h_1^{-1} \in Z$. In other words, it depends on x_1 if David's outputs $y_2, \ldots, y_n$ are all-zero or not. This is not simulatable in the ideal model, because the simulator gets absolutely no information about the uncorrupted David's inputs. □

3.3 Correctness of our construction

In a totally uncorrupted setting, simulation is straightforward. Since the simulator always is notified when the ideal Goliath receives input from the environment and the simulator also may arbitrarily delay the ideal David's corresponding ready-message, he can perfectly simulate any scheduling of the messages in the send phase. In turn, the choice phase cannot be influenced by the real model adversary and therefore can be simulated trivially. Furthermore, whenever in the real model David outputs some (y_i, i), it holds that $y_i = a_i x_i + b_i$, as one can verify as follows:

$$y_i \quad = \quad G \underbrace{(r_i z_i + S_i)}_{=W_i} h_i + \underbrace{(a_i - G r_i)}_{=\tilde{a}_i} x_i + \underbrace{b_i - G S_i h_i}_{=\tilde{b}_i} \quad = \quad G r_i \underbrace{(z_i h_i - x_i)}_{=0} + a_i x_i + b_i$$

Also note that in a totally uncorrupted setting David's consistency checks are always passed.

3.4 Security against a corrupted receiver

We show now security against a corrupted David. Basically, there are only two things he can do: follow the protocol honestly or query the token prematurely. We will refer to the former as the *regular case* and to the latter as the *irregular case*. A bit more formally, regarding some specific $i \in \{1, \ldots, n\}$ we speak of the *regular case* if David inputs (z_i, i) into the token *after* receiving $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ from Goliath, and we speak of the *irregular case* if David inputs (z_i, i) into the token *before* receiving $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ from Goliath. Note that, although David is corrupted, his messages to Goliath and his token queries still have to be well-formed, as otherwise they are just ignored. Hence, (C, G, h_i) and z_i are still well-defined and can be extracted from Goliath's and the token's view respectively. It turns out that due to the randomness of (r_i, S_i) every value seen by David, namely $\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, W_i$, is in both cases just uniformly random subject to the sole condition that in the end the correct result y_i can be computed. We formalize this by the next lemma.

Lemma 2. *In our protocol* $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$*, if Goliath is honest, then in the regular case (i.e., David inputs* (z_i, i) *into the token* after *receiving* $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ *from Goliath)* $\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i$ *are just uniformly random over* $C\mathbb{F}_q^{4k}, C\mathbb{F}_q^{4k\times 4k}, \mathbb{F}_q^k, \mathbb{F}_q^k$*, and the token output* W_i *is a uniformly random solution of the following linear equation system:*

$$CW_i = \tilde{r}_i z_i + \tilde{S}_i$$
$$GW_i h_i + \tilde{a}_i z_i h_i + \tilde{b}_i = a_i z_i h_i + b_i$$

In the irregular case (i.e., David inputs (z_i, i) *into the token* before *receiving* $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ *from Goliath) the token output* W_i *is just uniformly random over* $\mathbb{F}_q^{4k\times 4k}$*, and* $\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i$ *are a uniformly random solution of the abovementioned linear equation system. In particular, we have that* $\tilde{a}_i \xleftarrow{\mathrm{r}} \mathbb{F}_q^k$ *and* $\tilde{b}_i = (a_i z_i h_i + b_i) - (GW_i h_i + \tilde{a}_i z_i h_i)$*.*

Proof. We start off with the regular case, i.e. David first receives $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ from Goliath and later on inputs (z_i, i) into the token. In this case, Goliath just announces some $\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i$ uniformly at random (with support of $\tilde{r}_i$ and $\tilde{S}_i$ depending on C). The relation between $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i)$ and Goliath's OAFE input is perfectly blinded by the randomness (r_i, S_i) stored on the token. It remains to show that the token output W_i is uniformly random subject to the condition that $CW_i = \tilde{r}_i z_i + \tilde{S}_i$ and $GW_i h_i + \tilde{a}_i z_i h_i + \tilde{b}_i = a_i z_i h_i + b_i$. However, we can imagine that right before the computation of W_i the token's stored randomness S_i is uniformly resampled subject to the condition that $CS_i = \tilde{S}_i$ and $GS_i h_i = b_i - \tilde{b}_i$. This clearly does not change David's view at all. In other words, we can replace S_i by $S_i + S'$, where S' is uniformly random subject to the condition that $CS' = 0$ and $GS'h = 0$. Thereby, W_i is also replaced by $W_i + S'$ and hence becomes uniformly random subject to the sole condition that CW_i and $GW_i h_i$ are not changed. This means that W_i is uniformly random subject to the condition that $CW_i = \tilde{r}_i z_i + \tilde{S}_i$ and $GW_i h_i = (a_i - \tilde{a}_i) z_i h_i + b_i - \tilde{b}_i$. This concludes our proof for the regular case.

Now we consider the irregular case, i.e. the corrupted David inputs (z_i, i) into the token before receiving $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ from Goliath. In this case, the token's output W_i obviously is just uniformly random. It remains to show that the honest Goliath's announcement of $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i)$ is uniformly random subject to the condition that $CW_i = \tilde{r}_i z_i + \tilde{S}_i$ and $GW_i h_i + \tilde{a}_i z_i h_i + \tilde{b}_i = a_i z_i h_i + b_i$. However, we can imagine that right before the computation of $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i)$ the stored randomness (r_i, S_i) in Goliath's memory is uniformly resampled subject to the condition that $r_i z_i + S_i = W_i$. This clearly does not change David's view at all. In other words, we can replace (r_i, S_i) by $(r_i + r', S_i - r' z_i)$ with $r' \xleftarrow{\mathrm{r}} \mathbb{F}_q^{4k}$. Thereby, $(\tilde{r}_i, \tilde{S}_i)$ is replaced by $(\tilde{r}_i + Cr', \tilde{S}_i - Cr' z_i)$ and $(\tilde{a}_i, \tilde{b}_i)$ is replaced by $(\tilde{a}_i - Gr', \tilde{b}_i + Gr' z_i h_i)$. The former means that $(\tilde{r}_i, \tilde{S}_i)$ becomes uniformly random subject to the sole condition that $CW_i = \tilde{r}_i z_i + \tilde{S}_i$, and the latter means that $(\tilde{a}_i, \tilde{b}_i)$ becomes uniformly random subject to the sole condition that $GW_i h_i + \tilde{a}_i z_i h_i + \tilde{b}_i = a_i z_i h_i + b_i$. Note that $(\tilde{r}_i, \tilde{S}_i)$ and $(\tilde{a}_i, \tilde{b}_i)$ are statistically independent, since G is complementary to C. This concludes our proof for the irregular case. □

This lemma leads to a straightforward simulator construction in the UC framework. When the corrupted David queries the token *after* he already got the derandomization information $(\tilde{a}_i, \tilde{b}_i)$ from Goliath in the corresponding send phase, the simulator can revise the token's output W_i so that the check $CW_i \stackrel{?}{=} \tilde{r}_i z_i + \tilde{S}_i$ is still passed, but GW_i now matches a protocol run in the real model: When the token is to output W_i, the simulator has already seen both shares z_i, h_i

that are needed to extract David's input x_i. The simulator can then query the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$ on this input x_i, thus receiving y_i, and then revise W_i by some W' so that $CW' = CW_i$ and $y_i = GW'h_i + \tilde{a}_i x_i + \tilde{b}_i$. Note that existence of such a W' is always guaranteed, since G is complementary to C (i.e. especially G has full rank) and $h \neq 0$.

When the corrupted David queries the token *before* he got the derandomization information $(\tilde{a}_i, \tilde{b}_i)$ from Goliath in the corresponding send phase, the simulator can revise Goliath's announcement of the derandomization information so that it matches a protocol run in the real model: When Goliath is to announce the derandomization information $(\tilde{a}_i, \tilde{b}_i)$, the simulator has already seen both shares z_i, h_i that are needed to extract David's input x_i. The simulator can then query the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$ on this input x_i, thus receiving y_i, and then just revise $\tilde{b}_i$ so that $y_i = GW_i h_i + \tilde{a}_i x_i + \tilde{b}_i$.

Simulator $\mathcal{S}^{\text{David}}(\mathcal{A})$

- Set up an honest Goliath-machine $\mathcal{G}$; also set up simulated versions of $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ and the given real model adversary $\mathcal{A}$ (which especially impersonates the corrupted David). Wire the simulated machines $\mathcal{A}, \mathcal{G}, \mathcal{F}_{\text{wrap}}^{\text{stateful}}$ to each other and $\mathcal{A}$ to the environment right the way they would be wired in the real model with protocol $\Pi_{\text{OAFE}}^{\text{seq}-\text{ot}}$ (q.v. Figure 6 in Section 3.1).
- Upon receiving a message $(\texttt{created}, i)$ from the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$, reply with $(\texttt{Delivery}, i)$. Then, upon receiving $(\texttt{ready}, i)$ on behalf of the corrupted David, choose some random vectors $a_i, b_i \xleftarrow{\text{r}} \mathbb{F}_q^k$ and let $\mathcal{G}$ start the i-th send phase with input (a_i, b_i, i).
- Whenever $\mathcal{G}$ is to send some $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ to the corrupted David in Step 1 of $\Pi_{\text{OAFE}}^{\text{seq}-\text{ot}}$, extract the current state j of j'_{queried} from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$. If $j \geq i$, replace $\tilde{b}_i$ in the message $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ by $\tilde{b}'$, which is computed as follows; apart from that just carry on with simulation of the unmodified protocol.
  0. Extract G, h_i from the view of $\mathcal{G}$ and z_i, W_i from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$.
  1. Compute $x_i \leftarrow z_i h_i$ and (on behalf of the corrupted David) send (x_i, i) to the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$; let (y_i, i) denote the respective answer from $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$.
  2. Set $\tilde{b}' \leftarrow y_i - GW_i h_i - \tilde{a}_i x_i$.
- Whenever the token is to output some matrix W to the corrupted David in Step 4 of $\Pi_{\text{OAFE}}^{\text{seq}-\text{ot}}$, extract the current state i of j'_{queried} from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$. If $\mathcal{G}$ has already sent some message $(\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i, i)$ to the corrupted David in Step 1 of $\Pi_{\text{OAFE}}^{\text{seq}-\text{ot}}$, replace the token's output by W', which is computed as follows; apart from that just carry on with simulation of the unmodified protocol.
  0. Extract $C, G, \tilde{r}_i, \tilde{S}_i, h_i, \tilde{a}_i, \tilde{b}_i$ from the view of $\mathcal{G}$ and z_i from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$.
  1. Compute $x_i \leftarrow z_i h_i$ and (on behalf of the corrupted David) send (x_i, i) to the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$; let (y_i, i) denote the respective answer from $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$.
  2. Choose randomly $W' \xleftarrow{\text{r}} \left\{ \widetilde{W} \in \mathbb{F}_q^{4k \times k} \;\middle|\; C\widetilde{W} = \tilde{r}_i z_i + \tilde{S}_i \;\wedge\; G\widetilde{W} h_i + \tilde{a}_i x_i + \tilde{b}_i = y_i \right\}$.

Figure 7: The simulator program $\mathcal{S}^{\text{David}}(\mathcal{A})$, given an adversary $\mathcal{A}$ that corrupts David.

A formal description of this simulator is given in Figure 7. We conclude this section with the corresponding security theorem.

Theorem 3. *Let some arbitrary environment $\mathcal{Z}$ be given and some adversary $\mathcal{A}$ that corrupts David. Then the view of $\mathcal{Z}$ in the ideal model with ideal functionality $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ and simulator $\mathcal{S}^{\mathrm{David}}(\mathcal{A})$ is identically distributed to the view of $\mathcal{Z}$ in the real model with protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ and adversary $\mathcal{A}$.*

Proof. This directly follows by Lemma 2. □

3.5 Security against a corrupted sender

Before we give our simulator construction for a corrupted Goliath (q.v. Section 3.5.5), we first take a closer look at the problems we have to deal with, and introduce the respective solution tools (Sections 3.5.1-3.5.4).

3.5.1 Independence of the token view

We start our security considerations with showing that an honest David's token inputs $z_1, \ldots, z_n$ are statistically indistinguishable from uniform randomness. This is necessary for security, since otherwise David's OAFE inputs $x_1, \ldots, x_n$ would be non-negligibly correlated with the token view and a malicious token's behavior in the n-th choice phase could depend on $x_1, \ldots, x_{n-1}$.

W.l.o.g., we can assume that David's random tape is chosen after all other random tapes, i.e. we can consider everything to be deterministic except for David's random choice of $h_1, \ldots, h_n$ and $z_1, \ldots, z_n$. However, as we are aiming for universal composability, we must take into account that David's i-th OAFE input x_i might depend on everything that Goliath learned so far. In particular, Goliath knows $h_1, \ldots, h_n$ and there might have leaked some little information about $z_1, \ldots, z_{i-1}$ during past choice phases. Therefore, we have to model David's i-th OAFE input x_i as a function value $x_i(h_1, \ldots, h_n, z_1, \ldots, z_{i-1})$.

Lemma 4. *Let $\mathbb{F}_q$ be some arbitrary field of size $q \geq 2$ and let $k, n \in \mathbb{N}_{>0}$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \setminus \{0\}$. Further, for $i = 1, \ldots, n$ let any mapping $x_i : \mathcal{H}^n \times \mathcal{U}^{i-1} \to \mathbb{F}_q$ be given. Finally, for $i = 1, \ldots, n$ we define the following random variables:*

$$\mathbf{h}_i \xleftarrow{\mathrm{r}} \mathcal{H} \qquad \mathbf{z}_i \xleftarrow{\mathrm{r}} \left\{ z \in \mathcal{U} \;\middle|\; z\mathbf{h}_i = x_i(\mathbf{h}_1, \ldots, \mathbf{h}_n, \mathbf{z}_1, \ldots, \mathbf{z}_{i-1}) \right\} \qquad \mathbf{u}_i \xleftarrow{\mathrm{r}} \mathcal{U}$$

Then it holds that $\Delta\big((\mathbf{z}_1, \ldots, \mathbf{z}_n), (\mathbf{u}_1, \ldots, \mathbf{u}_n)\big) < \frac{1}{2}\sqrt{\exp\big(n \cdot q^{2-k}\big) - 1}$.

Proof. We show this by estimation techniques borrowed from a proof for the Leftover Hash Lemma [AB09, proof of Lemma 21.26]. Let $\vec{z} \in \mathbb{R}^{\mathcal{U}^n}$ denote the probability vector of $(\mathbf{z}_1, \ldots, \mathbf{z}_n)$ and let $\vec{u} \in \mathbb{R}^{\mathcal{U}^n}$ denote the probability vector of $(\mathbf{u}_1, \ldots, \mathbf{u}_n)$. Note that $\vec{z} - \vec{u}$ is orthogonal to $\vec{u}$:

$$\langle \vec{z} - \vec{u} \,|\, \vec{u} \rangle \;=\; \langle \vec{z} \,|\, \vec{u} \rangle - \langle \vec{u} \,|\, \vec{u} \rangle \;=\; \tfrac{\|\vec{z}\|_1}{|\mathcal{U}^n|} - \tfrac{\|\vec{u}\|_1}{|\mathcal{U}^n|} \;=\; \tfrac{1}{|\mathcal{U}^n|} - \tfrac{1}{|\mathcal{U}^n|} \;=\; 0$$

Let the $2n$-tuple of random variables $(\mathbf{h}'_1, \ldots, \mathbf{h}'_n, \mathbf{z}'_1, \ldots, \mathbf{z}'_n)$ be identically distributed as its unprimed counterpart $(\mathbf{h}_1, \ldots, \mathbf{h}_n, \mathbf{z}_1, \ldots, \mathbf{z}_n)$. The following equation system has exactly q^{k-2} different solutions $z \in \mathcal{U}$ if $\mathbf{h}_i$ and $\mathbf{h}'_i$ are linearly independent, and at most q^{k-1} solutions otherwise:

$$z\mathbf{h}_i = x_i(\mathbf{h}_1, \ldots, \mathbf{h}_n, \mathbf{z}_1, \ldots, \mathbf{z}_{i-1})$$
$$z\mathbf{h}'_i = x_i(\mathbf{h}'_1, \ldots, \mathbf{h}'_n, \mathbf{z}'_1, \ldots, \mathbf{z}'_{i-1})$$

Using the auxiliary random variable $\mathbf{m} := \#\big\{i \in \{1,\dots,n\} \;\big|\; \mathbf{h}_i \text{ and } \mathbf{h}'_i \text{ are linearly independent}\big\}$, we can thus estimate:

$$\mathbb{P}\big[(\mathbf{z}_1,\dots,\mathbf{z}_n) = (\mathbf{z}'_1,\dots,\mathbf{z}'_n) \;\big|\; \mathbf{m} = m\big] \;\leq\; \left(\frac{q^{k-2}}{q^{k-1}\cdot q^{k-1}}\right)^m \cdot \left(\frac{q^{k-1}}{q^{k-1}\cdot q^{k-1}}\right)^{n-m}$$

Further, we have that $\mathbb{P}[\mathbf{m} = m] = \binom{n}{m} \cdot \left(\frac{|\mathcal{H}|-(q-1)}{|\mathcal{H}|}\right)^m \cdot \left(\frac{q-1}{|\mathcal{H}|}\right)^{n-m}$ by construction. It follows:

$$\begin{aligned}
\|\vec{z}\|_2^2 \;&=\; \mathbb{P}\big[(\mathbf{z}_1,\dots,\mathbf{z}_n) = (\mathbf{z}'_1,\dots,\mathbf{z}'_n)\big] \\
&=\; \sum_{m=0}^{n} \mathbb{P}[\mathbf{m} = m] \cdot \mathbb{P}\big[(\mathbf{z}_1,\dots,\mathbf{z}_n) = (\mathbf{z}'_1,\dots,\mathbf{z}'_n) \;\big|\; \mathbf{m} = m\big] \\
&\leq\; \sum_{m=0}^{n} \binom{n}{m} \cdot \left(\frac{|\mathcal{H}|-(q-1)}{|\mathcal{H}|}\right)^m \cdot \left(\frac{q-1}{|\mathcal{H}|}\right)^{n-m} \cdot \left(\frac{q^{k-2}}{q^{k-1}\cdot q^{k-1}}\right)^m \cdot \left(\frac{q^{k-1}}{q^{k-1}\cdot q^{k-1}}\right)^{n-m} \\
&=\; \left(1 + \tfrac{(q-1)^2}{|\mathcal{H}|}\right)^n \cdot q^{-nk}
\end{aligned}$$

Using the Pythagorean Theorem, we can now estimate:

$$\|\vec{z}-\vec{u}\|_2^2 \;=\; \|\vec{z}\|_2^2 - \|\vec{u}\|_2^2 \;\leq\; \left(1 + \tfrac{(q-1)^2}{|\mathcal{H}|}\right)^n \cdot q^{-nk} - q^{-nk}$$

Since $\|\vec{v}\|_1 \leq \sqrt{m} \cdot \|\vec{v}\|_2$ for all $m \in \mathbb{N}$, $\vec{v} \in \mathbb{R}^m$, this yields:

$$\Delta\big((\mathbf{z}_1,\dots,\mathbf{z}_n),(\mathbf{u}_1,\dots,\mathbf{u}_n)\big) \;=\; \tfrac{1}{2}\|\vec{z}-\vec{u}\|_1 \;\leq\; \tfrac{1}{2}\sqrt{|\mathcal{U}^n|} \cdot \|\vec{z}-\vec{u}\|_2 \;\leq\; \tfrac{1}{2}\sqrt{\left(1 + \tfrac{(q-1)^2}{|\mathcal{H}|}\right)^n - 1}$$

To conclude our proof, we further estimate:

$$\left(1 + \tfrac{(q-1)^2}{|\mathcal{H}|}\right)^n \;=\; \exp\left(n \cdot \ln\left(1 + \tfrac{(q-1)^2}{|\mathcal{H}|}\right)\right) \;<\; \exp\left(\tfrac{n\cdot(q-1)^2}{|\mathcal{H}|}\right) \;<\; \exp\left(n \cdot q^{2-k}\right) \qquad \square$$

We will use this lemma not directly but for showing that the token functionality in the $(m+1)$-th choice phase can be considered to be independent of C and $h_{m+1},\dots,h_n$. So in the following corollary, the random variable $\mathbf{R}$ can be thought of as David's random choice of $(C, h_{m+1},\dots,h_n)$.

Corollary 5. *Let $\mathbb{F}_q$ be some arbitrary field of size $q \geq 2$ and let $k, m \in \mathbb{N}_{>0}$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \setminus \{0\}$. Further, let $\mathbf{R}$ be some arbitrary random variable with finite support $\mathcal{R}$. For $i = 1,\dots,m$ let any mapping $x_i : \mathcal{R} \times \mathcal{H}^m \times \mathcal{U}^{i-1} \to \mathbb{F}_q$ be given. Finally, for $i = 1,\dots,m$ we define the following random variables:*

$$\mathbf{h}_i \xleftarrow{\text{r}} \mathcal{H} \qquad\qquad \mathbf{z}_i \xleftarrow{\text{r}} \big\{z \in \mathcal{U} \;\big|\; z\mathbf{h}_i = x_i(\mathbf{R},\mathbf{h}_1,\dots,\mathbf{h}_m,\mathbf{z}_1,\dots,\mathbf{z}_{i-1})\big\}$$

Then it holds that $\iota\big(\mathbf{R},(\mathbf{z}_1,\dots,\mathbf{z}_m)\big) < \sqrt{\exp\big(mq^{2-k}\big) - 1}$.

Proof. By the Triangle Inequality (and the definition of ι, q.v. Section 2.1), we already have that $\iota\big(\mathbf{R},(\mathbf{z}_1,\dots,\mathbf{z}_m)\big) \leq 2 \cdot \Delta\big((\mathbf{R},\mathbf{z}_1,\dots,\mathbf{z}_m),(\mathbf{R},\mathbf{u}_1,\dots,\mathbf{u}_m)\big)$ for any random variables $\mathbf{u}_1,\dots,\mathbf{u}_m$ that are statistically independent from $\mathbf{R}$. Hence, our corollary directly follows by Lemma 4. $\square$

After all, note that independence between the token view and David's OAFE inputs $x_1, \ldots, x_n$ is only a starting point for our security proof. E.g., we have not used so far David's consistency check $CW_i \stackrel{?}{=} \tilde{r}_i z_i + \tilde{S}_i$ in the final step of the choice phases. Lemma 4 and Corollary 5 would still hold true without this consistency check, but the protocol would become susceptible to attacks where the token encodes z_{i-1} into W_i. Now, if this information about z_{i-1} is unveiled to Goliath, e.g. through the unveil message in a commitment protocol (q.v. Protocol $\Pi_{\text{COM}}^{\text{backward}}$ in Figure 25), he can reconstruct David's secret OAFE input x_{i-1}, although the token still learns nothing but uniform randomness. Thus, what we have shown so far can only be one core argument amongst several others.

3.5.2 Committing the token to affine behavior

In Section 3.1 we argued that David's check $CW_i \stackrel{?}{=} \tilde{r}_i z_i + \tilde{S}_i$ enforces affine behavior of the token, since otherwise the token could form collisions for the universal hash function C. However, this is only half the truth. In fact, with $\tau_i : \mathbb{F}_q^{1\times k} \to \mathbb{F}_q^{4k\times k}$ denoting the token functionality in the i-th choice phase, for each possible token input $z \in \mathbb{F}_q^{1\times k}$ there are always exactly q^{4k} different parameter tuples $(r, S) \in \mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}$, such that $\tau_i(z) = rz + S$. In particular, for all $z \in \mathbb{F}_q^{1\times k}$, $r \in \mathbb{F}_q^{4k}$ we can complement r to a matching parameter tuple by $S := \tau_i(z) - rz$. In total, there might exist up to q^{5k} different parameter tuples belonging to any image of τ_i and we must somehow rule out that there are too many collisions of the form $(Cr, CS) = (Cr', CS')$ with distinct $(r, S), (r', S')$.

Since the space of potential parameter tuples is that large, pure counting arguments (e.g. by considering the random matrix C as a 2-universal hash function) cannot be sufficient as long as the special structure of our problem is ignored: For example, consider the hypothetical case that every parameter tuple (r, S) has to be taken into account where each column of S equals r. Although this yields only q^{4k} different parameter tuples, we would always have that equivalence classes of q^k different parameter tuples do collide. However, this is exactly the size of the preimage space of τ_i. Thereby we just cannot rule out that τ_i is non-affine on every $Z \subseteq \mathbb{F}_q^{1\times k}$ with $|Z| > 1$, but enough parameter tuples collide so that $C \cdot \tau_i$ is affine on the complete input space $\mathbb{F}_q^{1\times k}$. Note also that this problem cannot be circumvent by enlarging the token input space to $\mathbb{F}_q^{1\times \alpha k}$ for some $\alpha > 1$, since in that case we can still argue analogously with the condition $|Z| > 1$ replaced by $|Z| > q^{(\alpha-1)k}$.

So, we explicitly have to exploit that the space of affine mappings $\mathbb{F}_q^{1\times k} \to \mathbb{F}_q^{4k\times k}$ has some specific structure. In fact, we only need the random matrix C to have some rank-preserving property when operating on the image space of τ_i. Given this (and a not too large overall abort probability in the current choice phase), we can show that τ_i is affine on *all* token inputs that do not cause an abort.

Lemma 6. *Let $\mathbb{F}_q$ be some finite field of size $q \geq 2$ and let $l, m, k \in \mathbb{N}_{>0}$. Let $\tau : \mathbb{F}_q^{1\times k} \to \mathbb{F}_q^{m\times k}$ be some arbitrary mapping and let $C \in \mathbb{F}_q^{l\times m}$, $\tilde{r} \in \mathbb{F}_q^{l}$, $\tilde{S} \in \mathbb{F}_q^{l\times k}$, $V := \big\{v \in \mathbb{F}_q^{1\times k} \;\big|\; C\cdot\tau(v) = \tilde{r}\cdot v + \tilde{S}\big\}$, such that $|V| > q$ and for all $v, v' \in V$ the following implications hold true:*

$$\begin{aligned} \operatorname{rank}\big(\tau(v) - \tau(v')\big) > 0 \quad &\Rightarrow \quad \operatorname{rank}\big(C\cdot\tau(v) - C\cdot\tau(v')\big) > 0 \\ \operatorname{rank}\big(\tau(v) - \tau(v')\big) > 1 \quad &\Rightarrow \quad \operatorname{rank}\big(C\cdot\tau(v) - C\cdot\tau(v')\big) > 1 \end{aligned}$$

Then there exists a unique tuple $(r, S) \in \mathbb{F}_q^m \times \mathbb{F}_q^{m\times k}$, such that $\tau(v) = r \cdot v + S$ for all $v \in V$. Further, for this unique tuple it holds that $(Cr, CS) = (\tilde{r}, \tilde{S})$.

Proof. We just need to show existence of (r, S); everything else follows straightforwardly. Moreover, if $\tilde{r} = 0$, the proof is trivial. In this case, since by assumption $\tau(v) = \tau(v')$ for all $v, v' \in V$ with $C \cdot \tau(v) = C \cdot \tau(v')$, we have that τ is constant on the entire input set V. So, w.l.o.g. let $\tilde{r} \neq 0$.

First of all, we now observe for all $v, v' \in V$ that $\text{rank}(\tau(v) - \tau(v')) \leq 1$, since else by the rank-preserving properties of C we had the contradiction that $1 < \text{rank}\big(C \cdot \tau(v) - C \cdot \tau(v')\big) = \text{rank}\big(\tilde{r} \cdot (v - v')\big) \leq 1$. Thereby, for all $v, v' \in V$ we find some $r \in \mathbb{F}_q^m$, $\bar{v} \in \mathbb{F}_q^{1 \times n}$, such that $\tau(v) - \tau(v') = r \cdot \bar{v}$. Moreover, we can always choose $\bar{v} := v - v'$, since $\tilde{r} \cdot (v - v') = Cr \cdot \bar{v}$ and we assumed that $\tilde{r} \neq 0$. Thus we have:

$$\forall\, v, v' \in V\ \exists\, r \in \mathbb{F}_q^m :\ \tau(v) - \tau(v') = r \cdot (v - v')$$

We will show now that r in fact is independent of v, v'. More precisely, we will show that for arbitrary $v, v', v'' \in V$ with linearly independent $v - v', v' - v''$ there always exists an $r \in \mathbb{F}_q^m$, such that $\tau(v) - \tau(v') = r \cdot (v - v')$ and $\tau(v') - \tau(v'') = r \cdot (v' - v'')$. It is sufficient to consider the case of linearly independent $v - v', v' - v''$, since $|V| > q$ by assumption and hence the affine span of V must have dimension 2 or higher; therefore for all $v, v', v'' \in V$ with linearly dependent $v - v', v' - v''$ there exists some $\hat{v} \in V$, such that $v - v', v' - \hat{v}$ are linearly independent and also are $\hat{v} - v', v' - v''$. So, let any $v, v', v'' \in V$, $r, r' \in \mathbb{F}_q^m$ be given with linearly independent $v - v', v' - v''$ and:

$$\begin{aligned} \tau(v) - \tau(v') &= r \cdot (v - v') \\ \tau(v') - \tau(v'') &= r' \cdot (v' - v'') \end{aligned}$$

Thereby follows:

$$\text{rank}\big(r \cdot (v - v') + r' \cdot (v' - v'')\big) \quad = \quad \text{rank}\big(\tau(v) - \tau(v'')\big) \quad \leq \quad 1$$

Since $v - v', v' - v''$ are linearly independent, this yields that r, r' must be linearly dependent. Hence, on the one hand we find some $\hat{r} \in \mathbb{F}_q^m$ and $\alpha, \alpha' \in \mathbb{F}_q$, such that $r = \alpha \hat{r}$ and $r' = \alpha' \hat{r}$. On the other hand, since $v, v' \in V$, we also have:

$$\tilde{r} \cdot (v - v') \quad = \quad C \cdot \big(\tau(v) - \tau(v')\big) \quad = \quad Cr \cdot (v - v')$$

Since $v - v' \neq 0$, this yields that $Cr = \tilde{r}$ and analogously it must hold that $Cr' = \tilde{r}$. Thus we have that $\alpha C \hat{r} = \alpha' C \hat{r} = \tilde{r}$. Since we assumed that $\tilde{r} \neq 0$, we can conclude that $\alpha = \alpha'$ and hence $r = r'$.

So, once we have shown that r is unique, we can finally pick some arbitrary $\tilde{v} \in V$ and set $S := \tau(\tilde{v}) - r \cdot \tilde{v}$, whereby for every $\tilde{v}' \in M$ it follows:

$$\tau(\tilde{v}') \quad = \quad \big(\tau(\tilde{v}') - \tau(\tilde{v})\big) + \tau(\tilde{v}) \quad = \quad \big(r \cdot (\tilde{v}' - \tilde{v})\big) + \big(r \cdot \tilde{v} + S\big) \quad = \quad r \cdot \tilde{v}' + S \qquad \square$$

To make this lemma applicable, yet we need to show that with overwhelming probability the random matrix C has the required rank-preserving properties. As a formal preparation we state our next technical lemma, where the matrix set $\mathcal{W}$ should be thought of as all possible token output differences $\tau_i(z) - \tau_i(z')$ and $l := 3k$ and $m := 4k$. Thereby, since $|\mathcal{W}| < |\text{image}(\tau_i)|^2 \leq q^{2k}$ and thus $q^{r-l}|\mathcal{W}| < q^{r-k}$, we get that the random matrix C has the required rank-preserving properties with overwhelming probability. For convenience, we state this lemma only for the case that the random matrix C is statistically independent from the token functionality τ_i (and $\mathcal{W}$ respectively). However, the error introduced by this assumption can be estimated by $\iota(C, \tau_i)$, and then Corollary 5 does apply.

Lemma 7. *Let $\mathbb{F}_q$ be some finite field of size $q \geq 2$ and let $l, m, k, r \in \mathbb{N}_{>0}$ with $r \leq \min(l, m, k)$. Then for arbitrary $\mathcal{W} \subseteq \mathbb{F}_q^{m \times k}$ and $\mathbf{C} \xleftarrow{\mathrm{r}} \mathbb{F}_q^{l \times m}$ it holds:*

$$\mathbb{P}\left[\exists\, W \in \mathcal{W} :\ \mathrm{rank}(W) \geq r > \mathrm{rank}(\mathbf{C}W)\right] \ <\ q^{r-l}|\mathcal{W}|$$

Proof. We first estimate the number of matrices in $\mathbb{F}_q^{l \times r}$ that have full rank r. Given $i \in \{1, \ldots, r\}$ and any matrix $C \in \mathbb{F}_q^{l \times i}$ with full rank i, there exist exactly $q^l - q^i$ columns in $\mathbb{F}_q^l$ (only the linear combinations of the columns of C are excluded) by which we can extend C to a matrix of dimension $l \times (i+1)$ and rank $i+1$. By induction on i follows:

$$\#\left\{C \in \mathbb{F}_q^{l \times r} \;\middle|\; \mathrm{rank}(C) = r\right\} \ =\ \prod_{i=0}^{r-1} q^l - q^i$$

Since the term $\prod_{i=0}^{r-1} q^l - q^i$ is a bit unhandy, we estimate it from below:

$$\prod_{i=0}^{r-1} q^l - q^i \ =\ q^{lr} \prod_{i=0}^{r-1} 1 - q^{i-l} \ \geq\ q^{lr}\left(1 - \sum_{i=0}^{r-1} q^{i-l}\right) \ =\ q^{lr}\left(1 - \frac{q^r - 1}{q^l(q-1)}\right) \ >\ q^{lr}\left(1 - q^{r-l}\right)$$

Now, let $W \in \mathbb{F}_q^{m \times k}$ be some arbitrary matrix with $\bar{r} := \mathrm{rank}(W) \geq r$. Further let $\bar{B} \in \mathbb{F}_q^{\bar{r} \times k}$, such that $\bar{B}$ only consists of linearly independent rows of W; i.e. especially $\bar{B}$ has full rank $\bar{r}$. Let $B \in \mathbb{F}_q^{m \times k}$, such that the first $\bar{r}$ rows of B are $\bar{B}$ and the rest of B is all-zero. Note that we can find an invertible matrix $M \in \mathbb{F}_q^{m \times m}$, such that $W = MB$. Hence we can estimate:

$$\begin{aligned}
\#\left\{C \in \mathbb{F}_q^{l \times m} \;\middle|\; \mathrm{rank}(CW) < r\right\} \ &=\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times m} \;\middle|\; \mathrm{rank}(CW) \geq r\right\} \\
&=\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times m} \;\middle|\; \mathrm{rank}(CMB) \geq r\right\} \\
&=\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times m} \;\middle|\; \mathrm{rank}(CB) \geq r\right\} \\
&=\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times \bar{r}} \;\middle|\; \mathrm{rank}(C\bar{B}) \geq r\right\} \cdot q^{l(m-\bar{r})} \\
&=\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times \bar{r}} \;\middle|\; \mathrm{rank}(C) \geq r\right\} \cdot q^{l(m-\bar{r})} \\
&\leq\ q^{lm} - \#\left\{C \in \mathbb{F}_q^{l \times r} \;\middle|\; \mathrm{rank}(C) = r\right\} \cdot q^{l(m-r)} \\
&<\ q^{lm} - q^{lm}\left(1 - q^{r-l}\right) \\
&=\ q^{lm+r-l}
\end{aligned}$$

Thereby, for arbitrary $W \in \mathbb{F}_q^{m \times n}$ and $\mathbf{C} \xleftarrow{\mathrm{r}} \mathbb{F}_q^{l \times m}$ we can conclude:

$$\mathbb{P}\left[\mathrm{rank}(W) \geq r > \mathrm{rank}(\mathbf{C}W)\right] \ <\ q^{r-l}$$

The assertion of our lemma now follows by the Union Bound. □

Basically, in each choice phase of our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ (with honest receiver David) we have now with overwhelming probability one of the following two cases:

- Either $\#\left\{z \in \mathbb{F}_q^{1 \times k} \;\middle|\; C \cdot \tau_i(z) = \tilde{r}_i z + \tilde{S}_i\right\} \leq q$, i.e. only few token inputs pass David's consistency check and thus the protocol is aborted with overwhelming probability,
- or there exist some $r \in \mathbb{F}_q^{4k}$, $S \in \mathbb{F}_q^{4k \times k}$, such that $\tau_i(z) = rz + S$ for all $z \in \mathbb{F}_q^{1 \times k}$ with $C \cdot \tau_i(z) = \tilde{r}_i z + \tilde{S}_i$, i.e. the token functionality is affine on all inputs that pass the consistency check.

3.5.3 Uniqueness of affine approximations of the token functionality

By our technical tools developed so far, we already have that the token functionality in each choice phase is piecewise affine, and the protocol is aborted if the affine pieces are too small. However, for our formal security proof we will need that the larger affine pieces yield an almost disjoint decomposition of the preimage space in the sense that only few inputs belong to more than one of these larger affine pieces. Otherwise, the simulator for a corrupted Goliath could fail with noticeable probability to extract the correct affine function parameters (a_i, b_i). This motivates our next lemma.

Lemma 8. *Let $\mathbb{F}_q$ be some finite field of size $q \geq 2$, let $\varepsilon > 0$ and let $k, l \in \mathbb{N}_{>0}$, such that $q^k \geq 2^{1/\varepsilon}$. Further, let $\tau : \mathbb{F}_q^{1\times k} \to \mathbb{F}_q^{l\times k}$ be an arbitrary mapping and let V' denote the set of all $v \in \mathbb{F}_q^{1\times k}$ for that exist more than one tuple $(r, S) \in \mathbb{F}_q^{l} \times \mathbb{F}_q^{l\times k}$ with the following property:*

$$\tau(v) = rv + S \quad \text{and} \quad \#\big\{\tilde{v} \in \mathbb{F}_q^{1\times k} \;\big|\; \tau(\tilde{v}) = r\tilde{v} + S\big\} \geq q^{(2/3+\varepsilon)k}$$

Then we have that $|V'| < q^{2k/3}$.

Proof. We call a mapping $\gamma : \mathbb{F}_q^{1\times k} \to \mathbb{F}_q^{l\times k}$ a *straight line*, if there exist $r \in \mathbb{F}_q^k$ and $S \in \mathbb{F}_q^{l\times k}$, such that $\gamma(v) = rv + S$ for all $v \in \mathbb{F}_q^{1\times k}$. Given two straight lines γ, γ', we call $v \in \mathbb{F}_q^{1\times k}$ an *intersection* of γ and γ', if $\gamma(v) = \gamma'(v)$. Given a straight line γ, we say that γ *intersects* with τ for m times, if $\tau(v) = \gamma(v)$ for exactly m different $v \in \mathbb{F}_q^{1\times k}$. Note that two straight lines are identical, iff they have two or more common intersections.

Now, let Γ denote the set of all straight lines that intersect with τ for at least $q^{(2/3+\varepsilon)k}$ times. Thus, V' is a subset of all intersections of distinct straight lines $\gamma, \gamma' \in \Gamma$. However, as two distinct straight lines may have no more than one common intersection, m straight lines have always less than m^2 intersections in total. Thus, if $|V'| \geq q^{2k/3}$, there would be more than $q^{k/3}$ straight lines in Γ, i.e. we could find some $\Gamma' \subseteq \Gamma$ with $|\Gamma'| = \lceil q^{k/3} \rceil$. However, this leads to a contradiction, as one can see as follows. Each of the straight lines in Γ' has less than $q^{k/3}$ intersections with all the other straight lines in Γ', what leaves more than $q^{(2/3+\varepsilon)k} - q^{k/3}$ intersections with τ that are not shared with other straight lines in Γ'. Hence, overall τ must have more than $\lceil q^{k/3} \rceil \cdot \big(q^{(2/3+\varepsilon)k} - q^{k/3}\big)$ of such non-shared intersections with straight lines in Γ', i.e. $\big|\tau(\mathbb{F}_q^{1\times k})\big| > q^{(1+\varepsilon)k} - q^{2k/3}$. Since $q^{\varepsilon k} \geq 2$ by assumption and thus $q^{(1+\varepsilon)k} - q^{2k/3} \geq q^k\big(2 - q^{-k/3}\big) > q^k$, this is impossible. □

3.5.4 Utilizing the Leftover Hash Lemma

We introduce now our final tool, a fairly technical partitioning argument, which will be needed to show that the abort behavior in the real model is indistinguishable from the abort behavior in the ideal model. Before we take a closer look at the technical details, we briefly recap the involved elements of our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ (q.v. Section 3.1):

- First of all, by programming the token the adversary commits to a disjoint decomposition of the i-th round token input space $\mathbb{F}_q^{1\times k}$, in the sense that each part of this decomposition corresponds to another affine token behavior in a later round.
- Next, the honest David announces a 2-universal hash function $h_i \xleftarrow{\mathrm{r}} \mathbb{F}_q^k \backslash \{0\}$.

- Then, Goliath announces some check information corresponding to one part of the disjoint decomposition he committed to. The protocol will be aborted, iff David's i-th token input z_i is not an element of this part.

- Finally, David gets his i-th OAFE input $x_i \in \mathbb{F}_q$ from the environment, inputs a share $z_i \xleftarrow{r} \{\tilde{z} \in \mathbb{F}_q^{1\times k} \mid \tilde{z}h_i = x_i\}$ into the token, and aborts the protocol if the token output does not match Goliath's check information.

On the one hand, since the environment w.l.o.g. knows how the corrupted Goliath programmed the token and what check information he announced, the environment can exactly determine the abort probability in the real model. On the other hand, since the simulator does not know the ideal David's OAFE input x_i, the abort probability in the ideal model is independent of x_i. Thus, we have to show that the abort probability in the real model also is not noticeably correlated with x_i. A bit more formally, the security proof basically boils down to the following problem, where $A(h)$ can be seen as the set of token inputs in the current stage that will not yield an abort:

- There are two adversarially chosen mappings, $x : \mathbb{F}_q^k \backslash \{0\} \to \mathbb{F}_q$ and $A : \mathbb{F}_q^k \backslash \{0\} \to \mathcal{P}(\mathbb{F}_q^{1\times k})$ (with $\mathcal{P}(\mathbb{F}_q^{1\times k})$ denoting the power set of $\mathbb{F}_q^{1\times k}$), such that for all $h, h' \in \mathbb{F}_q^k \backslash \{0\}$ the following implication holds true:

$$A(h) \neq A(h') \quad \Rightarrow \quad A(h) \cap A(h') = \emptyset$$

- There are the following three random variables:

$$\mathbf{h} \xleftarrow{r} \mathbb{F}_q^k \backslash \{0\} \qquad \mathbf{z} \xleftarrow{r} \{\tilde{z} \in \mathbb{F}_q^{1\times k} \mid \tilde{z}\mathbf{h} = x(\mathbf{h})\} \qquad \mathbf{u} \xleftarrow{r} \mathbb{F}_q^{1\times k}$$

- We have to show that $\Delta\big((\mathbf{z} \in A(\mathbf{h}), \mathbf{h}), (\mathbf{u} \in A(\mathbf{h}), \mathbf{h})\big)$ is negligible, where $\mathbf{z} \in A(\mathbf{h})$ denotes a predicate that is true iff $\mathbf{z}$ is an element of $A(\mathbf{h})$, and analogously for $\mathbf{u} \in A(\mathbf{h})$.

We address this problem by showing that $\mathbf{h}$ partitions $A(\mathbf{h})$ into parts of roughly equal size. Our starting point is the Leftover Hash Lemma, which we first recap for the sake of self-containedness (Lemma 9). If $x(\mathbf{h})$ and $A(\mathbf{h})$ were independent of $\mathbf{h}$, the Leftover Hash Lemma would already suffice to straightforwardly solve our problem. However, our problem is more complex and there seems no apparent way to directly apply the Leftover Hash Lemma. Nonetheless, we can utilize the Leftover Hash Lemma to get an estimation for the case that only $A(\mathbf{h})$ is independent of $\mathbf{h}$; q.v. Lemma 10. Finally, this estimation is used to develop our technical partitioning argument (Corollary 11).

Lemma 9 (Leftover Hash Lemma [BBR88, ILL89])**.** *Let $\mathcal{G}$ be a 2-universal class of functions $\mathcal{X} \to \mathcal{Y}$ and let $\mathbf{g} \xleftarrow{r} \mathcal{G}$, i.e. for any distinct $x, x' \in \mathcal{X}$ it holds that $\mathbb{P}\big[\mathbf{g}(x) = \mathbf{g}(x')\big] \leq \frac{1}{|\mathcal{Y}|}$. Further let $\mathbf{x} \in \mathcal{X}$ be some random variable with collision entropy $\mathbb{H}_2(\mathbf{x})$. Then, if $\mathbf{x}$ and $\mathbf{g}$ are independent, for the statistical distance between $\big(\mathbf{g}(\mathbf{x}), \mathbf{g}\big)$ and uniform randomness $(\mathbf{u}, \mathbf{g})$, i.e. $\mathbf{u} \xleftarrow{r} \mathcal{Y}$, it holds:*

$$\Delta\big((\mathbf{g}(\mathbf{x}), \mathbf{g}), (\mathbf{u}, \mathbf{g})\big) \;\leq\; \tfrac{1}{2}\sqrt{2^{-\mathbb{H}_2(\mathbf{x})} \cdot |\mathcal{Y}|}$$

Proof. We adapt the proof from [AB09, proof of Lemma 21.26]. Let $(\mathbf{g}', \mathbf{x}')$ be identically distributed as its unprimed counterpart $(\mathbf{g}, \mathbf{x})$. Thereby, when we treat the distribution of $\big(\mathbf{g}(\mathbf{x}), \mathbf{g}\big)$ as a probability vector $\vec{p} \in \mathbb{R}^{\mathcal{Y}\times\mathcal{G}}$, we get:

$$\begin{aligned}
\|\vec{p}\|_2^2 &= \mathbb{P}\big[(\mathbf{g}(\mathbf{x}), \mathbf{g}) = (\mathbf{g}'(\mathbf{x}'), \mathbf{g}')\big] \\
&= \mathbb{P}[\mathbf{g} = \mathbf{g}'] \cdot \mathbb{P}\big[\mathbf{g}(\mathbf{x}) = \mathbf{g}'(\mathbf{x}') \,\big|\, \mathbf{g} = \mathbf{g}'\big] \\
&= \mathbb{P}[\mathbf{g} = \mathbf{g}'] \cdot \mathbb{P}\big[\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{x}')\big] \\
&= \mathbb{P}[\mathbf{g} = \mathbf{g}'] \cdot \big(\mathbb{P}[\mathbf{x} = \mathbf{x}'] + \mathbb{P}[\mathbf{x} \neq \mathbf{x}'] \cdot \mathbb{P}\big[\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{x}') \,\big|\, \mathbf{x} \neq \mathbf{x}'\big]\big) \\
&\leq \mathbb{P}[\mathbf{g} = \mathbf{g}'] \cdot \big(\mathbb{P}[\mathbf{x} = \mathbf{x}'] + \mathbb{P}\big[\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{x}') \,\big|\, \mathbf{x} \neq \mathbf{x}'\big]\big) \\
&= |\mathcal{G}|^{-1} \cdot \Big(2^{-\mathbb{H}_2(\mathbf{x})} + \mathbb{P}\big[\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{x}') \,\big|\, \mathbf{x} \neq \mathbf{x}'\big]\Big) \\
&\leq |\mathcal{G}|^{-1} \cdot \Big(2^{-\mathbb{H}_2(\mathbf{x})} + |\mathcal{Y}|^{-1}\Big)
\end{aligned}$$

Now, let $\vec{u} \in \mathbb{R}^{\mathcal{Y}\times\mathcal{G}}$ denote the probability vector corresponding to the uniform distribution over $\mathcal{Y}\times\mathcal{G}$. Note that $\vec{p} - \vec{u}$ is orthogonal to $\vec{u}$:

$$\langle \vec{p} - \vec{u} \,|\, \vec{u} \rangle \;=\; \langle \vec{p} \,|\, \vec{u} \rangle - \langle \vec{u} \,|\, \vec{u} \rangle \;=\; \tfrac{\|\vec{p}\|_1}{|\mathcal{Y}\times\mathcal{G}|} - \tfrac{\|\vec{u}\|_1}{|\mathcal{Y}\times\mathcal{G}|} \;=\; \tfrac{1}{|\mathcal{Y}\times\mathcal{G}|} - \tfrac{1}{|\mathcal{Y}\times\mathcal{G}|} \;=\; 0$$

By the Pythagorean Theorem follows:

$$\|\vec{p} - \vec{u}\|_2^2 \quad=\quad \|\vec{p}\|_2^2 - \|\vec{u}\|_2^2 \quad\leq\quad |\mathcal{G}|^{-1} \cdot \Big(2^{-\mathbb{H}_2(\mathbf{x})} + |\mathcal{Y}|^{-1}\Big) - |\mathcal{Y}\times\mathcal{G}|^{-1} \quad=\quad |\mathcal{G}|^{-1} \cdot 2^{-\mathbb{H}_2(\mathbf{x})}$$

Finally, since $\|\vec{v}\|_1 \leq \sqrt{m} \cdot \|\vec{v}\|_2$ for all $m \in \mathbb{N}$, $\vec{v} \in \mathbb{R}^m$, we can conclude:

$$\Delta\big((\mathbf{g}(\mathbf{x}), \mathbf{g}), (\mathbf{u}, \mathbf{g})\big) \;=\; \tfrac{1}{2}\|\vec{p} - \vec{u}\|_1 \;\leq\; \tfrac{1}{2}\sqrt{|\mathcal{Y}\times\mathcal{G}|} \cdot \|\vec{p} - \vec{u}\|_2 \;\leq\; \tfrac{1}{2}\sqrt{2^{-\mathbb{H}_2(\mathbf{x})} \cdot |\mathcal{Y}|} \qquad \square$$

Lemma 10. *Let $\mathbb{F}_q$ be some finite field of size $q \geq 2$ and let $k \in \mathbb{N}_{>0}$. For each $\alpha \in \mathbb{F}_q$, $h \in \mathbb{F}_q^k$ let $Z_\alpha(h) := \big\{z \in \mathbb{F}_q^{1\times k} \;\big|\; zh = \alpha\big\}$. Further, let $x : \mathcal{H} \to \mathbb{F}_q$ be some arbitrary mapping. Then, for $\mathbf{h} \xleftarrow{\text{r}} \mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$ and arbitrary $A \subseteq \mathbb{F}_q^{1\times k}$ it holds:*

$$\mathbb{E}\,\Big|\big|A \cap Z_{x(\mathbf{h})}(\mathbf{h})\big| - \tfrac{1}{q}\big|A\big|\Big| \;\leq\; \sqrt{q \cdot |A|}$$

Proof. W.l.o.g., $A \neq \emptyset$. Let $\mathbf{a} \xleftarrow{\text{r}} A$ and $\mathbf{u} \xleftarrow{\text{r}} \mathbb{F}_q$. On the one hand, since $\mathbb{P}[a\mathbf{h} = a'\mathbf{h}] = \mathbb{P}[(a - a')\mathbf{h} = 0] = \frac{q^{k-1}-1}{q^k-1} < \frac{1}{|\mathbb{F}_q|}$ for all distinct $a, a' \in A$, we can estimate the statistical distance $\Delta\big((\mathbf{ah}, \mathbf{h}), (\mathbf{u}, \mathbf{h})\big)$ by the Leftover Hash Lemma (Lemma 9) as follows:

$$\Delta\big((\mathbf{ah}, \mathbf{h}), (\mathbf{u}, \mathbf{h})\big) \quad\leq\quad \tfrac{1}{2}\sqrt{2^{-\mathbb{H}_2(\mathbf{a})} \cdot |\mathbb{F}_q|} \quad=\quad \tfrac{1}{2}\sqrt{\tfrac{q}{|A|}}$$

On the other hand, we have:

$$\begin{aligned}
\Delta\big((\mathbf{ah}, \mathbf{h}), (\mathbf{u}, \mathbf{h})\big) &= \tfrac{1}{2}\sum\nolimits_{\alpha\in\mathbb{F}_q,\, h\in\mathcal{H}} \big|\mathbb{P}[\mathbf{ah} = \alpha \;\wedge\; \mathbf{h} = h] - \mathbb{P}[\mathbf{u} = \alpha \;\wedge\; \mathbf{h} = h]\big| \\
&= \tfrac{1}{2}\sum\nolimits_{\alpha\in\mathbb{F}_q,\, h\in\mathcal{H}} \mathbb{P}[\mathbf{h} = h] \cdot \big|\mathbb{P}[\mathbf{a}h = \alpha] - \mathbb{P}[\mathbf{u} = \alpha]\big| \\
&= \tfrac{1}{2}\sum\nolimits_{\alpha\in\mathbb{F}_q,\, h\in\mathcal{H}} \mathbb{P}[\mathbf{h} = h] \cdot \Big|\tfrac{|A\cap Z_\alpha(h)|}{|A|} - \tfrac{1}{q}\Big| \\
&\geq \tfrac{1}{2}\sum\nolimits_{h\in\mathcal{H}} \mathbb{P}[\mathbf{h} = h] \cdot \Big|\tfrac{|A\cap Z_{x(h)}(h)|}{|A|} - \tfrac{1}{q}\Big| \\
&= \tfrac{1}{2}\,\mathbb{E}\,\Big|\tfrac{|A\cap Z_{x(\mathbf{h})}(\mathbf{h})|}{|A|} - \tfrac{1}{q}\Big|
\end{aligned}$$

By the linearity of expected values follows:

$$\mathbb{E}\left(\left|A \cap Z_{x(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A\right|\right) \quad \leq \quad 2\,|A| \cdot \Delta\big((\mathbf{ah},\mathbf{h}),(\mathbf{u},\mathbf{h})\big) \quad \leq \quad \sqrt{q \cdot |A|} \qquad \square$$

Corollary 11. *Let $\mathbb{F}_q$ be some finite field of size $q \geq 2$ and let $k \in \mathbb{N}_{>0}$. Let $\mathcal{H} := \mathbb{F}_q^k \setminus \{0\}$ and let $\mathcal{R}, \mathcal{Q}$ be some arbitrary finite sets. Moreover, let some mapping $A : \mathcal{R} \times \mathcal{Q} \to \mathcal{P}(\mathbb{F}_q^{1\times k})$ be given, such that for all $\nu, \nu' \in \mathcal{R}$, $t \in \mathcal{Q}$ the following implication holds true:*

$$A(\nu,t) \neq A(\nu',t) \quad \Rightarrow \quad A(\nu,t) \cap A(\nu',t) = \emptyset$$

For each $(\alpha, h) \in \mathbb{F}_q \times \mathcal{H}$ let $Z_\alpha(h) := \big\{z \in \mathbb{F}_q^{1\times k} \;\big|\; zh = \alpha\big\}$. Finally, let $\mathbf{h} \xleftarrow{\mathrm{r}} \mathcal{H}$. Then for every random variable $\mathbf{t} \in \mathcal{Q}$ and arbitrary $\gamma \in \mathbb{R}_{>0}$ it holds:

$$\mathbb{P}\left[\exists\, \alpha \in \mathbb{F}_q,\, \nu \in \mathcal{R} : \left|\left|A(\nu,\mathbf{t}) \cap Z_\alpha(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu,\mathbf{t})\right|\right| > \gamma\right] \;\leq\; \frac{q^{k+1/2}}{\gamma^{3/2}} + \iota(\mathbf{h},\mathbf{t})$$

Proof. It obviously suffices to give a proof for the case that $\mathbf{t}$ and $\mathbf{h}$ are statistically independent, i.e. $\iota(\mathbf{h},\mathbf{t}) = 0$. Now, as $\mathbf{t}$ and $\mathbf{h}$ are independent, we may just assume that w.l.o.g. $\mathbb{P}[\mathbf{t} = t] = 1$ for some worst case constant $t \in \mathcal{Q}$. However, once we have fixed $\mathbf{t}$, we can consider α and ν as function values of $\mathbf{h}$, which we denote by $\alpha(\mathbf{h})$ and $\nu_{\mathbf{h}}$ respectively. Thereby, for each $h \in \mathcal{H}$ we can define the equivalence class $[h] := \big\{h' \in \mathcal{H} \;\big|\; A(\nu_{h'},t) = A(\nu_h,t)\big\}$. Further, let $\bar{\mathcal{H}} \subseteq \mathcal{H}$ denote a representative system for these equivalence classes, i.e. $\big|\bar{\mathcal{H}} \cap [h]\big| = 1$ for all $h \in \mathcal{H}$. Let some arbitrary $\gamma \in \mathbb{R}_{>0}$ be given. By construction we have:

$$\mathbb{P}\left[\left|\left|A(\nu_{\mathbf{h}},t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu_{\mathbf{h}},t)\right|\right| > \gamma\right] \;\leq\; \sum_{h\in\bar{\mathcal{H}}} \mathbb{P}\left[\left|\left|A(\nu_h,t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu_h,t)\right|\right| > \gamma\right]$$

Note that we can discard all summands with $\big|A(\nu_h,t)\big| < \gamma$ on the right side, since for any $X \subseteq \mathbb{F}_q^{1\times k}$, $\alpha \in \mathbb{F}_q$, $h \in \mathcal{H}$ it trivially holds that $\big||X \cap Z_\alpha(h)| - \frac{1}{q}|X|\big| < |X|$. With $\hat{\mathcal{H}} := \big\{h \in \bar{\mathcal{H}} \;\big|\; \gamma < |A(\nu_h,t)|\big\}$ we get:

$$\mathbb{P}\left[\left|\left|A(\nu_{\mathbf{h}},t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu_{\mathbf{h}},t)\right|\right| > \gamma\right] \;\leq\; \sum_{h\in\hat{\mathcal{H}}} \mathbb{P}\left[\left|\left|A(\nu_h,t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu_h,t)\right|\right| > \gamma\right]$$

However, since $\mathbb{E}(\mathbf{x}) \geq \gamma \cdot \mathbb{P}[\mathbf{x} \geq \gamma]$ for every random variable $\mathbf{x} \in \mathbb{R}$, we can estimate by Lemma 10 for all $\nu \in \mathcal{Q}$:

$$\mathbb{P}\left[\left|\left|A(\nu,t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu,t)\right|\right| \geq \gamma\right] \;\leq\; \tfrac{1}{\gamma}\sqrt{q \cdot |A(\nu,t)|}$$

It follows:

$$\mathbb{P}\left[\left|\left|A(\nu_{\mathbf{h}},t) \cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\right| - \tfrac{1}{q}\left|A(\nu_{\mathbf{h}},t)\right|\right| > \gamma\right] \;\leq\; \tfrac{1}{\gamma}\sum\nolimits_{h\in\hat{\mathcal{H}}} \sqrt{q \cdot |A(\nu_h,t)|}$$

Moreover, as a simple corollary of Jensen's Inequality it holds:

$$\sum\nolimits_{h\in\hat{\mathcal{H}}} \sqrt{\left|A(\nu_h,t)\right|} \;\leq\; \sqrt{|\hat{\mathcal{H}}| \cdot \sum\nolimits_{h\in\hat{\mathcal{H}}} \left|A(\nu_h,t)\right|}$$

Simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$

- Set up an honest David-machine $\mathcal{D}$; also set up a simulated version of $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ and the given real model adversary $\mathcal{A}$ (which especially impersonates the corrupted Goliath). Wire the simulated machines $\mathcal{A}, \mathcal{D}, \mathcal{F}_{\text{wrap}}^{\text{stateful}}$ to each other and $\mathcal{A}$ to the environment right the way they would be wired in the real model with protocol $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ (q.v. Figure 6 in Section 3.1). Further, initialize $f_0 \leftarrow \top$.
- Whenever $\mathcal{D}$ outputs $(\texttt{ready}, i)$, extract a snapshot $\mathcal{T}'$ of $\mathcal{T}$ (including its current internal state) from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ and extract $C, G, \tilde{r}_i, \tilde{S}_i, h_i, \tilde{a}_i, \tilde{b}_i$ from the view of $\mathcal{D}$. Then run the following extraction program:
  1. Pick a random vector $u_i \xleftarrow{\text{r}} \mathbb{F}_q^{1\times k}$ and input (u_i, i) into the token $\mathcal{T}$; let W_i denote the token's output (w.l.o.g. $W_i \in \mathbb{F}_q^{4k\times k}$). If $f_{i-1} = \bot$ or $CW_i \neq \tilde{r}_i u_i + \tilde{S}_i$, set $f_i \leftarrow \bot$ and go to Step 4 of this extraction program; else just set $f_i \leftarrow \top$.
  2. Pick a random vector $v_i \xleftarrow{\text{r}} \mathbb{F}_q^{1\times k}$ and input (v_i, i) into a copy of $\mathcal{T}'$; let W' denote the token's output (w.l.o.g. $W' \in \mathbb{F}_q^{4k\times k}$). Retry this step until $CW' = \tilde{r}_i v_i + \tilde{S}_i$ or q^k iterations have past; in the latter case give up, i.e. send a special message $(\star, i)$ to the environment and terminate. If afterwards $u_i = v_i$ or any row of the matrix $W_i - W'$ is linearly independent of $u_i - v_i$, also give up.
  3. Compute the unique vector $r_i \in \mathbb{F}_q^{4k}$, such that $W_i - W' = r_i(u_i - v_i)$, and set $S_i \leftarrow W_i - r_i u_i$. Then compute $a_i \leftarrow \tilde{a}_i + G r_i$ and $b_i \leftarrow \tilde{b}_i + G S_i h_i$.
  4. If $f_i = \top$, send (a_i, b_i, i) on behalf of the corrupted Goliath to the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq-ot}}$; else send $(0, 0, i)$.

  Finally, upon receiving $(\texttt{created}, i)$ from the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq-ot}}$, reply with $(\texttt{Delivery}, i)$.

Figure 8: The simulator program $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$, given an adversary $\mathcal{A}$ that corrupts the sender party.

Since by construction $\{A(\nu_h, t)\}_{h\in\hat{\mathcal{H}}}$ is a disjoint decomposition of some subset of $\mathbb{F}_q^{1\times k}$, we can further estimate:

$$\sum\nolimits_{h\in\hat{\mathcal{H}}} \big|A(\nu_h, t)\big| \;\leq\; \big|\mathbb{F}_q^{1\times k}\big| \;=\; q^k$$

Further note that by construction $|\hat{\mathcal{H}}| < \frac{1}{\gamma}\cdot q^k$. Putting things together, we have shown:

$$\mathbb{P}\left[\Big|\big|A(\nu_{\mathbf{h}}, t)\cap Z_{\alpha(\mathbf{h})}(\mathbf{h})\big| - \tfrac{1}{q}\big|A(\nu_{\mathbf{h}}, t)\big|\Big| > \gamma\right] \;<\; \frac{q^{k+1/2}}{\gamma^{3/2}} \qquad \square$$

3.5.5 The simulator for a corrupted Goliath

In each choice phase, the simulator for a corrupted Goliath has to extract the correct affine function parameters $(a_i, b_i) \in \mathbb{F}_q^k \times \mathbb{F}_q^k$ and send them to the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq-ot}}$ (q.v. Figure 5 in Section 2.4). Note that the simulator has no influence on the choice phases (he even is not activated at all), as long as David is not corrupted.

Our simulator for a corrupted Goliath is given in Figure 8. The high-level picture how this simulator works is as follows. The send phases are simulated straightforwardly: $\mathcal{Z}$ just interacts with a simulated version of the complete real model. When the i-th send phase is over, the simulator must extract a valid Goliath input (a_i, b_i, i), i.e. the simulator needs a description of the token functionality for the i-th choice phase. Therefor, the simulator first checks whether the token

acts honestly on random input. If the token's output appears faulty, the simulator henceforth gives default input $(0, 0, i)$ to the ideal functionality; otherwise he rewinds the token to the beginning of the i-th choice phase and inputs other vectors $v_i \in \mathbb{F}_q^{1\times k}$ until he can extract an affine function that describes the token behavior in this phase. Once having extracted this affine description of the token functionality, the simulator can easily compute the unique Goliath input (a_i, b_i, i) corresponding to this token functionality and the messages of the i-th send phase. Note that the running time of $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$ is not a priori polynomially bounded in the security parameter $\lambda := k \log q$, but there may be up to q^k simulated token queries in Step 2 of the simulator's extraction program. However, the *expected* number of iterations in that step is constant. We also refer to Section 5.4 for a discussion why this seems unavoidable.

Lemma 12. *Let some arbitrary environment $\mathcal{Z}$ be given and some adversary $\mathcal{A}$ that corrupts Goliath. Then the expected running time of the simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$ is polynomially bounded in the running time of $\mathcal{A}$ and the corresponding token $\mathcal{T}$. In particular, for each simulated send phase the expected number of iterations performed in Step 2 of the simulator's extraction program (q.v. Figure 8) is constant.*

Proof. When the simulator enters his extraction program, we can express by a variable p the probability that he picks some u_i passing the check in Step 1. Then, in each iteration of Step 2 with probability $1 - p$ he will pick some v_i that does not pass the check. Hence, if the simulator would not give up after q^k iterations but try on infinitely, we had the following probability that exactly t iterations are performed:

$$\begin{cases} 1-p & \text{for } t = 0 \\ p^2 \cdot (1-p)^{t-1} & \text{for } t > 0 \end{cases}$$

This yields the following upper bound for the expected number of iterations:

$$p^2 \cdot \sum_{t=1}^{\infty} t \cdot (1-p)^{t-1}$$

Note that w.l.o.g. $p > 0$, as otherwise Step 2 of the extraction program is not entered at all. However, if $p > 0$, we can use the formula for the expectation of a geometric distribution:

$$p \cdot \sum_{t=1}^{\infty} t \cdot (1-p)^{t-1} \;=\; \tfrac{1}{p} \qquad \text{i.e.} \qquad p^2 \cdot \sum_{t=1}^{\infty} t \cdot (1-p)^{t-1} \;=\; 1 \qquad \square$$

3.5.6 A sequence of hybrid games

We prove indistinguishability between the ideal model and the real model by a hybrid argument. In particular, we will show that for $l = 1, \ldots, n$ no environment can distinguish non-negligibly between some hybrid games Game_{l-1} and Game_l, where Game_0 and Game_n are indistinguishable from the ideal and real model respectively. Each hybrid game Game_l works like an ideal model with ideal functionality $\mathcal{F}'$ and (non-efficient) simulator $\mathcal{S}'_l(\mathcal{A})$. The functionality $\mathcal{F}'$ resembles the ideal functionality $\mathcal{F}_{\text{OAFE}}^{\text{seq-ot}}$, but the simulator learns the ideal David's inputs and may overwrite the corresponding outputs of $\mathcal{F}'$. For a formal description see Figure 9. Each simulator $\mathcal{S}'_l(\mathcal{A})$ overwrites the first l outputs, so that they exactly equal the first l David outputs in the real model. The remaining $n-l$ outputs are computed from an extracted affine description of the token functionality, very similar to the ideal model. For a formal description of the simulators $\mathcal{S}'_l(\mathcal{A})$ see Figure 10.

Functionality $\mathcal{F}'$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some runtime bound n that is polynomially bounded in the security parameter $\lambda := k \log q$. The counters $j_{\mathsf{created}}, j_{\mathsf{sent}}, j_{\mathsf{queried}}$ are all initialized with 0.

Send phases:

- Upon receiving input (a, b, i) from Goliath, verify that $a, b \in \mathbb{F}_q^k$ and $i = j_{\mathsf{created}} + 1 \leq n$; else ignore that input. Next, update $j_{\mathsf{created}} \leftarrow i$ and send $(\texttt{created}, i)$ to the simulator.
- Upon receiving a message $(\texttt{Delivery}, i)$ from the simulator, verify that $i = j_{\mathsf{sent}} + 1 \leq j_{\mathsf{created}}$; else ignore that message. Next, update $j_{\mathsf{sent}} \leftarrow i$ and send $(\texttt{ready}, i)$ to David.

Choice phases:

- Upon receiving input (x, i) from David, verify that $x \in \mathbb{F}_q$ and $i = j_{\mathsf{queried}} + 1 \leq j_{\mathsf{sent}}$; else ignore that input. Next, update $j_{\mathsf{queried}} \leftarrow i$, send $(\texttt{queried}, x, i)$ to the simulator and wait for the simulator's next message. Then, upon receiving $(\texttt{Reply}, y, i)$ from the simulator, output (y, i) to David.

When a party is corrupted, the simulator is granted unrestricted access to the channel between $\mathcal{F}'$ and the corrupted party, including the ability of deleting and/or forging arbitrary messages.

Figure 9: The ideal functionality for the hybrid games $\mathsf{Game}_0, \ldots, \mathsf{Game}_n$. The difference to the ideal functionality $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ (q.v. Figure 5) is that in the choice phases the simulator learns David's input x and may overwrite the respective output y. The inputs a, b in the send phase are just meaningless.

Corollary 13. *Given any adversary $\mathcal{A}$ that corrupts Goliath, our hybrid game Game_0 with simulator $\mathcal{S}_0'(\mathcal{A})$ is statistically indistinguishable from the ideal model with simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$, and our hybrid game Game_n with simulator $\mathcal{S}_n'(\mathcal{A})$ is perfectly indistinguishable from the real model with adversary $\mathcal{A}$.*

Proof. We have perfect indistinguishability between Game_n and the real model just by construction. It is also straightforward to see that Game_0 is perfectly indistinguishable from the ideal model conditioned to the event that the simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$ does not reach the iteration bound q^k in Step 2 of his extraction program. However, by Lemma 12 this iteration bound is only reached with negligible probability and thus Game_0 is statistically indistinguishable from the ideal model. □

3.5.7 Transformation of successive hybrid games into an indistinguishability game

For our security proof we have to show that from the environment's view any successive hybrid games $\mathsf{Game}_{l-1}, \mathsf{Game}_l$, parametrized with the finite vector space $\mathbb{F}_q^k$ and runtime bound n, are statistically indistinguishable. Our approach is to transform these hybrid games into an indistinguishability game $\Gamma_0(\mathbb{F}_q, n, l)$, so that every environment $\mathcal{Z}$ that can distinguish Game_{l-1} from Game_l corresponds to a player that wins in $\Gamma_0(\mathbb{F}_q, n, l)$ with some non-negligible advantage. This approach allows us to successively modify the obtained indistinguishability game $\Gamma_0(\mathbb{F}_q, n, l)$, so that it becomes feasible to derive a maximum winning probability from which we can then infer a negligible upper bound for the statistical distance between $\mathcal{Z}$'s views in Game_{l-1} and Game_l. The intuition behind this sequence of indistinguishability games can be sketched as follows.

Simulator $\mathcal{S}'_l(\mathcal{A})$

- Set up an honest David-machine $\mathcal{D}$; also set up a simulated version of $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ and the given real model adversary $\mathcal{A}$ (which especially impersonates the corrupted Goliath). Wire the simulated machines $\mathcal{A}, \mathcal{D}, \mathcal{F}_{\text{wrap}}^{\text{stateful}}$ to each other and $\mathcal{A}$ to the environment right the way they would be wired in the real model. Further, initialize $f_0 \leftarrow \top$.
- Whenever $\mathcal{D}$ outputs $(\texttt{ready}, i)$, choose any $a, b \in \mathbb{F}_q^k$ and send (a, b, i) on behalf of the corrupted Goliath to the functionality $\mathcal{F}'$. Then, upon receiving $(\texttt{created}, i)$ from $\mathcal{F}'$, reply with $(\texttt{Delivery}, i)$.
- Upon receiving $(\texttt{queried}, x_i, i)$ from the functionality $\mathcal{F}'$, extract the token function τ_i of the current choice phase from the view of the simulated $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$, in the sense that on input $(\tilde{v}, i)$ the token $\mathcal{T}$ currently would output $\tau_i(\tilde{v})$. If $\tau_i(\tilde{v}) \notin \mathbb{F}_q^{4k \times k}$ for some $\tilde{v} \in \mathbb{F}_q^{1 \times k}$, treat this as an encoding of the all-zero matrix; thus w.l.o.g. $\tau_i : \mathbb{F}_q^{1 \times k} \to \mathbb{F}_q^{4k \times k}$. Further, extract $C, G, \tilde{r}_i, \tilde{S}_i, h_i, \tilde{a}_i, \tilde{b}_i$ from the view of $\mathcal{D}$. Then, if $i > l$, run the following extraction program:
  1. Pick randomly $u_i \xleftarrow{\text{r}} \mathbb{F}_q^{1 \times k}$ and input (u_i, i) into the token $\mathcal{T}$, thus progressing its internal state.
  2. If $f_{i-1} = \top$ and $C \cdot \tau_i(u_i) = \tilde{r}_i u_i + \tilde{S}_i$, set $f_i \leftarrow \top$; otherwise set $f_i \leftarrow \bot$.
  3. If $f_i = \top$, pick a random vector $v_i \xleftarrow{\text{r}} \left\{\tilde{v} \in \mathbb{F}_q^{1 \times k} \;\middle|\; C \cdot \tau_i(\tilde{v}) = \tilde{r}_i \tilde{v} + \tilde{S}_i\right\}$. Then, if $v_i = u_i$ or any row of the matrix $\tau_i(u_i) - \tau_i(v_i)$ is linearly independent of $u_i - v_i$, send $(\star, i)$ to the environment and terminate; else compute $y_i \leftarrow (\tilde{a}_i + G r_i) x_i + \tilde{b}_i + G S_i h_i$, where $r_i \in \mathbb{F}_q^{4k}$ is the unique vector with $\tau_i(u_i) - \tau_i(v_i) = r_i \cdot (u_i - v_i)$, and $S_i := \tau_i(u_i) - r_i u_i$.
     If $f_i = \bot$, just set $y_i \leftarrow 0$ (such that $y_i \in \mathbb{F}_q^k$).
  4. Send $(\texttt{Reply}, y_i, i)$ to $\mathcal{F}'$.

  If $i \leq l$, just run the following program instead:
  1. Pick a random vector $z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathbb{F}_q^{1 \times k} \mid \tilde{z} h_i = x_i\}$ and input (z_i, i) into the token $\mathcal{T}$, thus progressing its internal state.
  2. If $f_{i-1} = \top$ and $C \cdot \tau_i(z_i) = \tilde{r}_i z_i + \tilde{S}_i$, set $f_i \leftarrow \top$; otherwise set $f_i \leftarrow \bot$.
  3. If $f_i = \top$, compute $y_i \leftarrow G \cdot \tau_i(z_i) \cdot h_i + \tilde{a}_i x_i + \tilde{b}_i$.
     If $f_i = \bot$, just set $y_i \leftarrow 0$ (such that $y_i \in \mathbb{F}_q^k$).
  4. Send $(\texttt{Reply}, y_i, i)$ to $\mathcal{F}'$.

Figure 10: The simulator program $\mathcal{S}'_l(\mathcal{A})$ for the hybrid game Game_l, given an adversary $\mathcal{A}$ that corrupts Goliath. The hybrid games Game_n and Game_0 are indistinguishable from the real model with adversary $\mathcal{A}$ and the ideal model with simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$ respectively.

$\Gamma_0(\mathbb{F}_q, n, l)$: This is just a straightforward reformulation of what the environment $\mathcal{Z}$ sees in the hybrid games Game_{l-1} and Game_l respectively.

$\Gamma_1(\mathbb{F}_q, n, l)$: We make the player a bit stronger by giving him more direct access to the internal game state.

$\Gamma_2(\mathbb{F}_q, n, l, \varepsilon)$: We exploit that the token is somehow committed to affine behavior (cf. Section 3.5.2). This allows us to unify the way, David's outputs are computed in the hybrid real part and the hybrid ideal part: Basically, David's outputs in the hybrid real part are now also computed from an extracted affine approximation of the token functionality. The additional game parameter ε is introduced for technical reasons; it will be needed later to apply Lemma 8.

Game $\Gamma_0(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tau_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times k}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\text{r}} \mathcal{H}$, a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$ and some $G \in \mathbb{F}_q^{k\times 4k}$ complementary to C.
3. A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly.
4. For $i = 1, \ldots, l - d$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $\tilde{a}_i, \tilde{b}_i \in \mathbb{F}_q^k$ and $x_i \in \mathbb{F}_q$.
   (b) Let $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, chosen secretly.
   (c) If $C \cdot \tau_j(w_1, \ldots, w_j) \neq \tilde{r}_j w_j + \tilde{S}_j$ for any $j \leq i$, the player $\mathcal{K}$ learns $y_i := 0 \in \mathbb{F}_q^k$.
   Otherwise, $\mathcal{K}$ learns $y_i := G \cdot \tau_i(w_1, \ldots, w_i) \cdot h_i + \tilde{a}_i x_i + \tilde{b}_i$.
5. For $i = l - d + 1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $\tilde{a}_i, \tilde{b}_i \in \mathbb{F}_q^k$ and $x_i \in \mathbb{F}_q$.
   (b) Let $w_i := u_i \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
   (c) If $C \cdot \tau_j(w_1, \ldots, w_j) \neq \tilde{r}_j w_j + \tilde{S}_j$ for any $j \leq i$, the player $\mathcal{K}$ learns $y_i := 0 \in \mathbb{F}_q^k$.
   Otherwise, a random vector $v_i \xleftarrow{\text{r}} \left\{\tilde{v} \in \mathcal{U} \;\middle|\; C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i \tilde{v} + \tilde{S}_i\right\}$ is chosen secretly. Then, if $v_i = w_i$ or any row of the matrix $\tau_i(w_1, \ldots, w_i) - \tau_i(w_1, \ldots, w_{i-1}, v_i)$ is linearly independent of $w_i - v_i$, the player $\mathcal{K}$ receives a special message $(\star, i)$ and the game is aborted in the sense that Step 6 follows next; else $\mathcal{K}$ learns $y_i := (\tilde{a}_i + Gr_i)x_i + \tilde{b}_i + GS_i h_i$, where $r_i \in \mathbb{F}_q^{4k}$ is the unique vector with $\tau_i(w_1, \ldots, w_i) - \tau_i(w_1, \ldots, w_{i-1}, v_i) = r(w_i - v_i)$, and $S_i := \tau_i(w_1, \ldots, w_i) - r_i w_i$.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 11: Definition of a stand-alone indistinguishability game that captures the difference between the hybrid games Game_{l-1} and Game_l. The player's view in the indistinguishability game $\Gamma_0(\mathbb{F}_q^k, n, l)$ corresponds straightforwardly to the environment's view in the hybrid game Game_{l-d}, where d is the secret challenge bit from Step 3 of $\Gamma_0(\mathbb{F}_q^k, n, l)$. Thus, the statistical distance between the environment's view in Game_{l-1} and its view in Game_l is upper bounded by 2δ, where $\frac{1}{2} + \delta$ is the maximum winning probability in the indistinguishability game $\Gamma_0(\mathbb{F}_q^k, n, l)$.

$\Gamma_3(\mathbb{F}_q, n, l, \varepsilon)$: We replace the simulator's abort message $(\star, i)$, q.v. Step 2 in Figure 8. This corresponds to a simulator modification, so that he may not give up any more, but instead switches to the mode where David henceforth produces default (all-zero) output.

$\Gamma_4(\mathbb{F}_q, n, l, \varepsilon)$: We exploit that the token functionality for most inputs can be approximated by no more than one affine function (cf. Section 3.5.3). This allows us to consider the extracted affine function parameters as token outputs rather than approximations of the token functionality.

$\Gamma_5(\mathbb{F}_q, n, l)$: We no longer only *consider* the extracted affine function parameters as token outputs; now they are.

$\Gamma_6(\mathbb{F}_q, n, l)$**:** We make the player stronger. We let him learn the first $l-1$ token inputs and let him choose the last $n-l$ token inputs. Thus only the stage l, which is the only stage in which Game_{l-1} differs from Game_l, stays out of control of the player.

$\Gamma_7(\mathbb{F}_q, n, l)$**:** We just exploit that several variables have become obsolete, and get rid of them.

$\Gamma_8(\mathbb{F}_q, n, l)$**:** We get rid of the challenge matrix C. The player must now exactly forecast token outputs rather than only linear projections.

$\Gamma_9(\mathbb{F}_q, n, l)$**:** We make it explicit that w.l.o.g. the player follows a deterministic strategy, solely depending on what he learns during the game run. This is the final version of our indistinguishability game.

We give now a detailed description of our indistinguishability game (Figure 11) and its relation to Game_{l-1} and Game_l (Lemma 14). Then we successively transform it and show how this affects the winning probability.

Lemma 14. *Let $\frac{1}{2}+\delta$ be the maximal winning probability in the game $\Gamma_0(\mathbb{F}_q^k, n, l)$. Then the statistical distance between the environment's view in Game_{l-1} and its view in Game_l, both parametrized with the finite vector space $\mathbb{F}_q^k$ and runtime bound n, is upper bounded by 2δ.*

Proof. The proof is absolutely straightforward. Basically, we just have to show how the player $\mathcal{K}$ in $\Gamma_0(\mathbb{F}_q^k, n, l)$ can perfectly emulate the hybrid game Game_{l-d} for an environment $\mathcal{Z}$, where d is the secret challenge bit that $\mathcal{K}$ finally tries to guess. Our player $\mathcal{K}$ just works as follows:

- Setup a simulated version of the given environment $\mathcal{Z}$ and the complete hybrid game Game_l.
- As soon as in the game Game_l the token $\mathcal{T}$ is fixed, specify the mappings $\tau_1, \ldots, \tau_n$ such that the token functionality in the i-th choice phase on input history $(w_1, 1), \ldots, (w_{i-1}, i-1)$ implements the function $(w_i, i) \mapsto \tau_i(w_1, \ldots, w_i)$. This is Step 1 of $\Gamma_0(\mathbb{F}_q^k, n, l)$.
- Always overwrite the simulated David's random choices of $C, G, h_1, \ldots, h_n$ by the respective values learned in Step 2 of $\Gamma_0(\mathbb{F}_q^k, n, l)$.
- Whenever some $\tilde{r}_i, \tilde{S}_i, \tilde{a}_i, \tilde{b}_i$ are to be chosen in Step 4a or Step 5a of $\Gamma_0(\mathbb{F}_q^k, n, l)$, just take the respective values from the simulated David's view. Analogously take x_i from the view of the simulated functionality $\mathcal{F}'$.
- Whenever the simulated functionality $\mathcal{F}'$ outputs some (y_i, i), overwrite y_i by the respective value learned in Step 4c or Step 5c of $\Gamma_0(\mathbb{F}_q^k, n, l)$ respectively.
- Upon receiving a special message $(\star, i)$, just forward it to the simulated environment $\mathcal{Z}$ and stop the simulated hybrid game.

It is straightforward to see that this way the player $\mathcal{K}$ perfectly emulates a view of $\mathcal{Z}$ in the hybrid game Game_{l-d}; this is just how we constructed the game $\Gamma_0(\mathbb{F}_q^k, n, l)$.

Now, let the random variable $\mathbf{view}_{\mathcal{Z}}$ denote this emulated view of $\mathcal{Z}$, and let the random variable $\mathbf{d}$ denote the secret challenge bit that $\mathcal{K}$ tries to guess in Step 6 of $\Gamma_0(\mathbb{F}_q^k, n, l)$. Since by assumption the player $\mathcal{K}$ wins the game $\Gamma_0(\mathbb{F}_q^k, n, l)$ at most with probability $\frac{1}{2}+\delta$, it must hold for every predicate P that $\mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}}) = 0 \ \wedge \ \mathbf{d} = 0\big] + \mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}}) = 1 \ \wedge \ \mathbf{d} = 1\big] \leq \frac{1}{2} + \delta$.

Furthermore, note that there does exist a predicate P such that we can write the statistical distance $dist_l$ between the views of $\mathcal{Z}$ in Game_{l-1} and Game_l as follows:

$$dist_l \;=\; \overbrace{\mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}})=1 \;\big|\; \mathbf{d}=1\big]}^{=\mathbb{P}[P(\text{view of } \mathcal{Z} \text{ in } \mathsf{Game}_{l-1})=1]} - \overbrace{\mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}})=1 \;\big|\; \mathbf{d}=0\big]}^{=\mathbb{P}[P(\text{view of } \mathcal{Z} \text{ in } \mathsf{Game}_{l})=1]}$$

Thus we can conclude:

$$dist_l \;=\; \underbrace{\mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}})=1 \;\big|\; \mathbf{d}=1\big]}_{=2\cdot\mathbb{P}[P(\mathbf{view}_{\mathcal{Z}})=1\,\wedge\,\mathbf{d}=1]} - \Big(1 - \underbrace{\mathbb{P}\big[P(\mathbf{view}_{\mathcal{Z}})=0 \;\big|\; \mathbf{d}=0\big]}_{=2\cdot\mathbb{P}[P(\mathbf{view}_{\mathcal{Z}})=0\,\wedge\,\mathbf{d}=0]}\Big) \;\leq\; 2\delta \qquad \square$$

Game $\Gamma_1(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tau_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times k}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\text{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$.
3. A challenge bit $d \xleftarrow{\text{r}} \{0,1\}$ is chosen secretly.
4. For $i = 1, \ldots, l-d$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) Let $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, chosen secretly.
   (c) If $C \cdot \tau_i(w_1, \ldots, w_i) \neq \tilde{r}_i w_i + \tilde{S}_i$, the player $\mathcal{K}$ receives a special message $(\bot, i)$ and the game is aborted in the sense that Step 6 follows next.
   Otherwise, the player $\mathcal{K}$ learns $\tau_i(w_1, \ldots, w_i) \cdot h_i$.
5. For $i = l-d+1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) Let $w_i := u_i \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
   (c) If $C \cdot \tau_i(w_1, \ldots, w_i) \neq \tilde{r}_i w_i + \tilde{S}_i$, the player $\mathcal{K}$ receives a special message $(\bot, i)$ and the game is aborted in the sense that Step 6 follows next.
   Otherwise, a random vector $v_i \xleftarrow{\text{r}} \big\{\tilde{v} \in \mathcal{U} \;\big|\; C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\big\}$ is chosen secretly. Then, if $v_i = w_i$ or any row of the matrix $\tau_i(w_1, \ldots, w_i) - \tau_i(w_1, \ldots, w_{i-1}, v_i)$ is linearly independent of $w_i - v_i$, the player $\mathcal{K}$ receives a special message $(\star, i)$ and the game is aborted in the sense that Step 6 follows next; else $\mathcal{K}$ learns $r_i x_i + S_i h_i$, where $r_i \in \mathbb{F}_q^{4k}$ is the unique vector with $\tau_i(w_1, \ldots, w_i) - \tau_i(w_1, \ldots, w_{i-1}, v_i) = r(w_i - v_i)$, and $S_i := \tau_i(w_1, \ldots, w_i) - r_i w_i$.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0,1\}$. He wins the game, if $\tilde{d} = d$.

Figure 12: First transformation of our stand-alone indistinguishability game. There are two differences to the game $\Gamma_0(\mathbb{F}_q^k, n, l)$. Firstly, where $\Gamma_0(\mathbb{F}_q^k, n, l)$ in Step 4c or Step 5c switched to a mode such that the player $\mathcal{K}$ henceforth only receives all-zero outputs, now $\mathcal{K}$ is notified about that by a special message $(\bot, i)$ and the game is aborted. Secondly, $\mathcal{K}$ now directly learns $\tau_i(w_1, \ldots, w_i) \cdot h_i$ in Step 4c and $r_i x_i + S_i h_i$ in Step 5c instead of the corresponding image of $\vartheta \mapsto G\vartheta + \tilde{a}_i x_i + \tilde{b}_i$. These game modifications just make $\mathcal{K}$ strictly stronger and $G, (\tilde{a}_1, \tilde{b}_1), \ldots, (\tilde{a}_n, \tilde{b}_n)$ obsolete.

Game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$

Parametrized by a finite vector space $\mathbb{F}_q^k$, some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$, and $\varepsilon \in \mathbb{R}_{>0}$ such that $q^{(2/3+\varepsilon)k} \geq q$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tau_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times k}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\text{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$.
3. A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly.
4. For $i = 1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) If $i \leq l - d$, let $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, else let $w_i := u_i \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
   (c) If $C \cdot \tau_i(w_1, \ldots, w_i) \neq \tilde{r}_i w_i + \tilde{S}_i$, the player $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 5 follows next.
   (d) In the following cases the player $\mathcal{K}$ receives a special message $(\star, i)$ and the game is aborted in the sense that Step 5 follows next:
      i. $C \cdot \tau_i(w_1, \ldots, w_i) = \tilde{r}_i w_i + \tilde{S}_i$ and $\#\{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\} \leq q^{(2/3+\varepsilon)k}$.
      ii. There exist some $W, W' \in \tau_i(w_1, \ldots, w_{i-1}, \mathcal{U})$, such that $\operatorname{rank}(CW - CW') \leq 1$ and $\operatorname{rank}(W - W') > \operatorname{rank}(CW - CW')$.

   Otherwise, $\mathcal{K}$ learns $r_i x_i + S_i h_i$, where $(r_i, S_i) \in \mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}$ is the unique tuple such that $\tau_i(w_1, \ldots, w_{i-1}, v) = r_i v + S_i$ for all $v \in \{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\}$.
5. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 13: Second transformation of our stand-alone indistinguishability game. The difference to the game $\Gamma_1(\mathbb{F}_q^k, n, l)$ is the now uniform way to compute outputs for $\mathcal{K}$. Note that the tuple (r_i, S_i) in Step 4d is well-defined by Lemma 6.

Lemma 15. *The maximum winning probability in the game $\Gamma_0(\mathbb{F}_q^k, n, l)$ is upper bounded by the maximum winning probability in the game $\Gamma_1(\mathbb{F}_q^k, n, l)$.*

Proof. This holds trivially, since the player in $\Gamma_1(\mathbb{F}_q^k, n, l)$ is strictly stronger than in $\Gamma_0(\mathbb{F}_q^k, n, l)$. □

Lemma 16. *The probability that the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is aborted in Step 4d, is upper bounded by:*

$$n \cdot \left(q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\right) + \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1}$$

Proof. Let some arbitrary player $\mathcal{K}$ be given and let the random variables $\mathbf{C}, \mathbf{w}_1, \ldots, \mathbf{w}_n$ represent the same-named values in the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$. It is straightforward to see that for each stage $i \in \{1, \ldots, n\}$ the case 4(d)i occurs at most with probability $p := q^{(2/3+\varepsilon)k}/q^{k-1}$. Further, for each stage $i \in \{1, \ldots, n\}$ we have by Lemma 7 that the case 4(d)ii occurs at most with probability $p' := (q^{1-3k} + q^{2-3k}) \cdot |\mathcal{U}|^2$, if $\mathbf{C}$ and $(\mathbf{w}_1, \ldots, \mathbf{w}_{i-1})$ are statistically independent. Thus, by the Union Bound we can estimate the overall probability that the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is aborted in Step 4d by $n \cdot (p + p') + \iota\big(\mathbf{C}, (\mathbf{w}_1, \ldots, \mathbf{w}_n)\big)$. Estimating $\iota\big(\mathbf{C}, (\mathbf{w}_1, \ldots, \mathbf{w}_n)\big)$ by Corollary 5 yields:

$$n \cdot (p + p') + \iota\big(\mathbf{C}, (\mathbf{w}_1, \ldots, \mathbf{w}_n)\big) \;<\; n \cdot \left(q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\right) + \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1} \qquad \square$$

Game $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$

Parametrized by a finite vector space $\mathbb{F}_q^k$, some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$, and $\varepsilon \in \mathbb{R}_{>0}$ such that $q^{(2/3+\varepsilon)k} \geq q$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tau_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times k}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\text{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$.
3. A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly.
4. For $i = 1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) If $i \leq l - d$, let $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, else let $w_i := u_i \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
   (c) In the following cases the player $\mathcal{K}$ receives a special message $(\bot, i)$ and the game is aborted in the sense that Step 5 follows next:
      i. It holds that $C \cdot \tau_i(w_1, \ldots, w_i) \neq \tilde{r}_i w_i + \tilde{S}_i$.
      ii. It holds that $\#\{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\} \leq q^{(2/3+\varepsilon)k}$.
      iii. There exist some $W, W' \in \tau_i(w_1, \ldots, w_{i-1}, \mathcal{U})$, such that $\operatorname{rank}(CW - CW') \leq 1$ and $\operatorname{rank}(W - W') > \operatorname{rank}(CW - CW')$.

   Otherwise, $\mathcal{K}$ learns $r_i x_i + S_i h_i$, where $(r_i, S_i) \in \mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}$ is the unique tuple such that $\tau_i(w_1, \ldots, w_{i-1}, v) = r_i v + S_i$ for all $v \in \{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\}$.
5. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 14: Third transformation of our stand-alone indistinguishability game. The only difference to the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is that the abort message $(\star, i)$ was replaced by $(\bot, i)$.

Lemma 17. *The statistical distance between $\mathcal{K}$'s view in the game $\Gamma_1(\mathbb{F}_q^k, n, l)$ and $\mathcal{K}$'s view in the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is upper bounded by:*

$$n \cdot \left(q^{-(2/3+\varepsilon)k} + q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\right) + \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1}$$

Proof. Let some arbitrary player $\mathcal{K}$ for the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ be given. Note that the only difference to the game $\Gamma_1(\mathbb{F}_q^k, n, l)$ is the computation of $\mathcal{K}$'s output in Step 4d, which is now the same for $i \leq l - d$ and $i > l - d$. Since by Lemma 16 we already have an estimation for the abort probability in Step 4d, it suffices to consider the case that $\mathcal{K}$ actually learns $r_i x_i + S_i h_i$. If $i \leq l - d$, we can just argue that $r_i x_i + S_i h_i = (r_i w_i + S_i)h_i = \tau_i(w_1, \ldots, w_i) \cdot h_i$ by construction, and thus the player $\mathcal{K}$ receives exactly the same as he would have received in Step 4c of $\Gamma_1(\mathbb{F}_q^k, n, l)$. For $i > l - d$, we exploit the following facts:

- If the game is not aborted afore, in Step 4d of $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ the player $\mathcal{K}$ learns $r_i x_i + S_i h_i$, where $(r_i, S_i) \in \mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}$ is the unique tuple such that $\tau_i(w_1, \ldots, w_{i-1}, v) = r_i v + S_i$ for all $v \in \{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\}$.
- Thus, for every $v \in \{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\}$ either $v = w_i$, or r_i is the unique vector with $\tau_i(w_1, \ldots, w_i) - \tau_i(w_1, \ldots, w_{i-1}, v_i) = r(w_i - v_i)$ and it holds that $S_i = \tau_i(w_1, \ldots, w_i) - r_i w_i$.

Game $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$

Parametrized by a finite vector space $\mathbb{F}_q^k$, some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$, and $\varepsilon \in \mathbb{R}_{>0}$ such that $q^{(2/3+\varepsilon)k} \geq q$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tau_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times k}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\text{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$.
3. A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly.
4. For $i = 1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) If $i \leq l - d$, let $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, else let $w_i := u_i \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
   (c) If there exists a unique tuple $(r_i, S_i) \in \mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}$ such that $\tau_i(w_1, \ldots, w_i) = r_i w_i + S_i$ and $\#\{\tilde{v} \in \mathcal{U} \mid \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = r_i\tilde{v} + S_i\} > q^{(2/3+\varepsilon)k}$, and for this unique tuple it holds that $(Cr_i, CS_i) = (\tilde{r}_i, \tilde{S}_i)$, then $\mathcal{K}$ learns $r_i x_i + S_i h_i$.
   Otherwise, $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 5 follows next.
5. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 15: Fourth transformation of our stand-alone indistinguishability game. The only difference to $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$ is the way the tuple (r_i, S_i) is computed in Step 4c.

- Moreover, since $\#\{\tilde{v} \in \mathcal{U} \mid C \cdot \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = \tilde{r}_i\tilde{v} + \tilde{S}_i\} > q^{(2/3+\varepsilon)k}$ for non-aborted stages, a uniformly random v may equal w_i with probability less than $q^{-(2/3+\varepsilon)k}$.

Putting these three observations together, we can conclude that with probability higher than $q^{-(2/3+\varepsilon)k}$ in Step 5c of $\Gamma_1(\mathbb{F}_q^k, n, l)$ the same tuple (r_i, S_i) and hence the same output $r_i x_i + S_i h_i$ would be generated as in Step 4d of $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$. Thus, conditioned to the event that $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is not aborted in Step 4d, the statistical distance between $\mathcal{K}$'s view in $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ and $\mathcal{K}$'s view in $\Gamma_1(\mathbb{F}_q^k, n, l)$ is upper bounded by $n \cdot q^{-(2/3+\varepsilon)k}$. Finally, we just have to add the estimation from Lemma 16 to get the claimed upper bound for the statistical distance between $\mathcal{K}$'s view in $\Gamma_1(\mathbb{F}_q^k, n, l)$ and $\mathcal{K}$'s view in $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ without any conditions. □

Corollary 18. *The statistical distance between $\mathcal{K}$'s view in the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ and $\mathcal{K}$'s view in the game $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$ is upper bounded by:*

$$n \cdot \left(q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\right) + \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1}$$

Proof. The only difference between $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ and $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is that the abort message $(\star, i)$ in Step 4d of $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ was replaced by $(\perp, i)$. Thus, the statistical distance between $\mathcal{K}$'s respective views is just the probability that the game $\Gamma_2(\mathbb{F}_q^k, n, l, \varepsilon)$ is aborted in Step 4d. We already estimated this abort probability by the claimed term in Lemma 16. □

Game $\Gamma_5(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{1, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\tau}_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{\mathrm{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\mathrm{r}} \mathbb{F}_q^{3k\times 4k}$.
3. A challenge bit $d \xleftarrow{\mathrm{r}} \{0, 1\}$ is chosen secretly.
4. For $i = 1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$, $x_i \in \mathbb{F}_q$.
   (b) If $i \leq l - d$, let $w_i := z_i \xleftarrow{\mathrm{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, else let $w_i := u_i \xleftarrow{\mathrm{r}} \mathcal{U}$, chosen secretly.
   (c) If $C \cdot \tilde{\tau}_i(w_1, \ldots, w_i) = (\tilde{r}_i, \tilde{S}_i)$, then $\mathcal{K}$ learns $(r_i, S_i) := \tilde{\tau}_i(w_1, \ldots, w_i)$.
   Otherwise, $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 5 follows next.
5. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 16: Fifth transformation of our stand-alone indistinguishability game. There are two differences to the game $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$, which just make the player $\mathcal{K}$ strictly stronger. Firstly, the player $\mathcal{K}$ directly learns (r_i, S_i) instead of only $r_i x_i + S_i h_i$ in Step 4c. Secondly, the tuple (r_i, S_i) in Step 4c is no longer generated deterministically from τ_i and $w_1, \ldots, w_i$ by the game, but the player $\mathcal{K}$ may specify an arbitrary mapping $\tilde{\tau}_i$ instead that directly generates (r_i, S_i) from $w_1, \ldots, w_i$.

Lemma 19. *The statistical distance between $\mathcal{K}$'s view in the game $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$ and $\mathcal{K}$'s view in the game $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$ is upper bounded by $n \cdot q^{1-k/3}$, if $q^k \geq 2^{1/\varepsilon}$.*

Proof. The only difference between $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$ and $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$ is in the computation of the tuple (r_i, S_i). It is straightforward to verify (see also Lemma 6) that by construction in Step 4c of $\Gamma_3(\mathbb{F}_q^k, n, l, \varepsilon)$ it always holds: Either the game is aborted, or $\tau_i(w_1, \ldots, w_i) = r_i w_i + S_i$ and $\#\{\tilde{v} \in \mathcal{U} \mid \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = r_i\tilde{v} + S_i\} > q^{(2/3+\varepsilon)k}$ and $(Cr_i, CS_i) = (\tilde{r}_i, \tilde{S}_i)$. Thus, we just have to estimate the probability that there exists some other tuple $(r', S') \in (\mathbb{F}_q^{4k} \times \mathbb{F}_q^{4k\times k}) \setminus \{(r_i, S_i)\}$ with $\tau_i(w_1, \ldots, w_i) = r'w_i + S'$ and $\#\{\tilde{v} \in \mathcal{U} \mid \tau_i(w_1, \ldots, w_{i-1}, \tilde{v}) = r'\tilde{v} + S'\} > q^{(2/3+\varepsilon)k}$. Now given that $q^k \geq 2^{1/\varepsilon}$, we have by Lemma 8 that only for less than $q^{2k/3}$ different choices of w_i there may exist such a second tuple (r', S'). For each stage $i \in \{1, \ldots, n\}$, since w_i is chosen uniformly random with support size q^{k-1} or larger, we can hence upper bound the probability that such a second tuple (r', S') exists by $q^{2k/3}/q^{k-1}$. Thus, our lemma follows by the Union bound. □

Lemma 20. *The maximum winning probability in the game $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$ is upper bounded by the maximum winning probability in the game $\Gamma_5(\mathbb{F}_q^k, n, l)$.*

Proof. This holds trivially, since the player in $\Gamma_5(\mathbb{F}_q^k, n, l)$ is strictly stronger than in $\Gamma_4(\mathbb{F}_q^k, n, l, \varepsilon)$. □

Game $\Gamma_6(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. (a) For each $i \in \{l, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\tau}_i : \mathcal{U}^i \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$.
   (b) For each $i \in \{l+1, \ldots, n\}$ the player $\mathcal{K}$ chooses some $u_i \in \mathcal{U}$.
2. The player $\mathcal{K}$ learns n random vectors $h_1, \ldots, h_n \xleftarrow{r} \mathcal{H}$ and a random matrix $C \xleftarrow{r} \mathbb{F}_q^{3k\times 4k}$.
3. For $i = 1, \ldots, l-1$: The player $\mathcal{K}$ chooses some $x_i \in \mathbb{F}_q$ and learns $w_i := z_i \xleftarrow{r} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$.
4. (a) A challenge bit $d \xleftarrow{r} \{0,1\}$ is chosen secretly, and $\mathcal{K}$ chooses some $\tilde{r}_l \in \mathbb{F}_q^{3k}$, $\tilde{S}_l \in \mathbb{F}_q^{3k\times k}$, $x_l \in \mathbb{F}_q$.
   (b) If $d = 0$, let $w_l := z_l \xleftarrow{r} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_l = x_l\}$, else let $w_l := u_l \xleftarrow{r} \mathcal{U}$, chosen secretly.
   (c) If $C \cdot \tilde{\tau}_l(w_1, \ldots, w_l) = (\tilde{r}_l, \tilde{S}_l)$, then $\mathcal{K}$ learns $(r_l, S_l) := \tilde{\tau}_l(w_1, \ldots, w_l)$.
   Otherwise, $\mathcal{K}$ receives a special message $(\perp, l)$ and the game is aborted in the sense that Step 6 follows next.
5. For $i = l+1, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$. Let $w_i := u_i$.
   (b) If $C \cdot \tilde{\tau}_i(w_1, \ldots, w_i) = (\tilde{r}_i, \tilde{S}_i)$, then $\mathcal{K}$ learns $(r_i, S_i) := \tilde{\tau}_i(w_1, \ldots, w_i)$.
   Otherwise, $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 6 follows next.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0,1\}$. He wins the game, if $\tilde{d} = d$.

Figure 17: Sixth transformation of our stand-alone indistinguishability game. There are two differences to the game $\Gamma_5(\mathbb{F}_q^k, n, l)$, which just make the player $\mathcal{K}$ strictly stronger. Firstly, the last $n-l$ "token inputs" $w_{l+1}, \ldots, w_n$ are no longer chosen uniformly at random, but the player $\mathcal{K}$ may choose them at the start of the game in Step 1b. Secondly, in the first $l-1$ stages the game may no longer be aborted, and the player $\mathcal{K}$ directly learns w_i instead of only $\tilde{\tau}_i(w_1, \ldots, w_i)$, which makes the mappings $\tilde{\tau}_1, \ldots, \tilde{\tau}_{l-1}$ obsolete.

Lemma 21. *The maximum winning probability in the game* $\Gamma_5(\mathbb{F}_q^k, n, l)$ *is upper bounded by the maximum winning probability in the game* $\Gamma_6(\mathbb{F}_q^k, n, l)$.

Proof. This holds trivially, since the player in $\Gamma_6(\mathbb{F}_q^k, n, l)$ is strictly stronger than in $\Gamma_5(\mathbb{F}_q^k, n, l)$. □

Lemma 22. *The games* $\Gamma_6(\mathbb{F}_q^k, n, l)$ *and* $\Gamma_7(\mathbb{F}_q^k, n, l)$ *have the same maximum winning probability.*

Proof. This holds trivially, since the changes from $\Gamma_6(\mathbb{F}_q^k, n, l)$ to $\Gamma_7(\mathbb{F}_q^k, n, l)$ are just cosmetic. □

Game $\Gamma_7(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{l, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\tau}_i : \mathcal{U}^l \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$.
2. The player $\mathcal{K}$ learns l random vectors $h_1, \ldots, h_l \xleftarrow{r} \mathcal{H}$ and a random matrix $C \xleftarrow{r} \mathbb{F}_q^{3k\times 4k}$.
3. For $i = 1, \ldots, l-1$: The player $\mathcal{K}$ chooses some $x_i \in \mathbb{F}_q$ and learns $w_i := z_i \xleftarrow{r} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$.
4. (a) A challenge bit $d \xleftarrow{r} \{0, 1\}$ is chosen secretly, and $\mathcal{K}$ chooses some $x_l \in \mathbb{F}_q$.
   (b) If $d = 0$, let $w_l := z_l \xleftarrow{r} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_l = x_l\}$, else let $w_l := u_l \xleftarrow{r} \mathcal{U}$, chosen secretly.
5. For $i = l, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $\tilde{r}_i \in \mathbb{F}_q^{3k}$, $\tilde{S}_i \in \mathbb{F}_q^{3k\times k}$.
   (b) If $C \cdot \tilde{\tau}_i(w_1, \ldots, w_l) = (\tilde{r}_i, \tilde{S}_i)$, then $\mathcal{K}$ learns $(r_i, S_i) := \tilde{\tau}_i(w_1, \ldots, w_l)$.
   Otherwise, $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 6 follows next.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 18: Seventh transformation of our stand-alone indistinguishability game. This is just a "cleaned" version of $\Gamma_6(\mathbb{F}_q^k, n, l)$. Firstly, instead of letting the player $\mathcal{K}$ choose the last $n - l$ "token inputs" $w_{l+1}, \ldots, w_n$ at the start of the game explicitly, they are now implicitly hard-coded. Secondly, the meanwhile obsolete random vectors $h_{l+1}, \ldots, h_n$ are omitted. Thirdly, we moved $\mathcal{K}$'s choice of $(\tilde{r}_l, \tilde{S}_l)$ and the subsequent output generation from Step 4 to Step 5.

Lemma 23. *The maximum winning probability in the game $\Gamma_7(\mathbb{F}_q^k, n, l)$ and the maximum winning probability in the game $\Gamma_8(\mathbb{F}_q^k, n, l)$ differ at most by:*

$$n \cdot q^{1-k} + \sqrt{\exp\bigl(n \cdot q^{2-k}\bigr) - 1}$$

Proof. Let some arbitrary player $\mathcal{K}$ be given and let the random variables $\mathbf{C}, \mathbf{w}_1, \ldots, \mathbf{w}_l$ denote the same-named values in the game $\Gamma_7(\mathbb{F}_q^k, n, l)$. First of all, we just arbitrarily fix the random coins of $\mathcal{K}$ and hence get some fixed mappings $\tilde{\tau}_l, \ldots, \tilde{\tau}_n : \mathcal{U}^l \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$ in Step 1 of $\Gamma_7(\mathbb{F}_q^k, n, l)$. Now note that, if $\mathbf{C}M \neq \mathbf{C}M'$ for all distinct $M, M' \in \tilde{\tau}_i(\mathbf{w}_1, \ldots, \mathbf{w}_{l-1}, \mathcal{U})$, then $\tilde{\tau}_i(\mathbf{w}_1, \ldots, \mathbf{w}_l)$ is completely determined by $\bigl(\mathbf{C}, \mathbf{w}_1, \ldots, \mathbf{w}_{l-1}, \mathbf{C} \cdot \tilde{\tau}_i(\mathbf{w}_1, \ldots, \mathbf{w}_l)\bigr)$ and the specification of $\tilde{\tau}_i$. Thus, conditioned to the event that $\mathbf{C}M \neq \mathbf{C}M'$ for all distinct $M, M' \in \tilde{\tau}_i(\mathbf{w}_1, \ldots, \mathbf{w}_{l-1}, \mathcal{U})$ for all $i \in \{l, \ldots, n\}$ in both games $\Gamma_7(\mathbb{F}_q^k, n, l)$ and $\Gamma_8(\mathbb{F}_q^k, n, l)$, we can straightforwardly transform a player for $\Gamma_7(\mathbb{F}_q^k, n, l)$ into a player for $\Gamma_8(\mathbb{F}_q^k, n, l)$ with exactly the same winning probability. In other words, the maximum winning probability in the game $\Gamma_7(\mathbb{F}_q^k, n, l)$ may differ from the maximum winning probability in the game $\Gamma_8(\mathbb{F}_q^k, n, l)$ at most by the probability that $\mathbf{C}M = \mathbf{C}M'$ for some distinct $M, M' \in \tilde{\tau}_i(\mathbf{w}_1, \ldots, \mathbf{w}_{l-1}, \mathcal{U})$ with $i \in \{l, \ldots, n\}$. However, by Lemma 7 and the Union Bound we can estimate this probability by $(n - l + 1) \cdot q^{1-3k} \cdot \bigl|\mathcal{U}\bigr|^2 + \iota\bigl(\mathbf{C}, (\mathbf{w}_1, \ldots, \mathbf{w}_{l-1})\bigr)$. Further, by Corollary 5 we have that $\iota\bigl(\mathbf{C}, (\mathbf{w}_1, \ldots, \mathbf{w}_{l-1})\bigr) < \sqrt{\exp\bigl((l-1)q^{2-k}\bigr) - 1}$. Together this yields the claimed estimation. □

Game $\Gamma_8(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. For each $i \in \{l, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\tau}_i : \mathcal{U}^l \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$.
2. The player $\mathcal{K}$ learns l random vectors $h_1, \ldots, h_l \xleftarrow{\text{r}} \mathcal{H}$ and a random matrix $C \xleftarrow{\text{r}} \mathbb{F}_q^{3k\times 4k}$.
3. For $i = 1, \ldots, l-1$: The player $\mathcal{K}$ chooses some $x_i \in \mathbb{F}_q$ and learns $w_i := z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$.
4. (a) A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly, and $\mathcal{K}$ chooses some $x_l \in \mathbb{F}_q$.
   (b) If $d = 0$, let $w_l := z_l \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i\}$, else let $w_l := u_l \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
5. For $i = l, \ldots, n$:
   (a) The player $\mathcal{K}$ chooses some $r_i \in \mathbb{F}_q^{4k}$, $S_i \in \mathbb{F}_q^{4k\times k}$.
   (b) If $\tilde{\tau}_i(w_1, \ldots, w_l) = (r_i, S_i)$, then $\mathcal{K}$ is notified about that by a special message $(\top, i)$. Otherwise, $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 6 follows next.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 19: Eighth transformation of our stand-alone indistinguishability game. The only difference to $\Gamma_7(\mathbb{F}_q, n, l)$ is that in Step 5b the player $\mathcal{K}$ now must exactly forecast $\tilde{\tau}_i(w_1, \ldots, w_l)$ rather than only the linear projection $C \cdot \tilde{\tau}_i(w_1, \ldots, w_l)$.

Lemma 24. *The games $\Gamma_8(\mathbb{F}_q^k, n, l)$ and $\Gamma_9(\mathbb{F}_q^k, n, l)$ have the same maximum winning probability.*

Proof. This holds trivially, since w.l.o.g. we only need to consider deterministic players. □

Lemma 25. *The maximum winning probability in the game $\Gamma_9(\mathbb{F}_q^k, n, l)$ is upper bounded by:*

$$\tfrac{1}{2} + n \cdot \left(2q^{(4-k)/3} + q \cdot \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1}\right)$$

Proof. W.l.o.g. we consider a deterministic player $\mathcal{K}$, i.e. the mappings $\tilde{\tau}_i, x_i, \tilde{\sigma}_i$ are all fixed. Let the random variables $\mathbf{h}_1, \ldots, \mathbf{h}_n, \mathbf{z}_1, \ldots, \mathbf{z}_{l-1}, \mathbf{w}, \mathbf{d}$ represent the same-named random values in the game $\Gamma_9(\mathbb{F}_q^k, n)$, i.e. it holds:

$$\mathbf{d} \xleftarrow{\text{r}} \{0, 1\} \qquad \mathbf{h}_1, \ldots, \mathbf{h}_l \xleftarrow{\text{r}} \mathcal{H} \qquad \mathbf{z}_i \xleftarrow{\text{r}} \left\{z \in \mathcal{U} \;\middle|\; z\mathbf{h}_i = x_i(\mathbf{h}_1, \ldots, \mathbf{h}_l, \mathbf{z}_1, \ldots, \mathbf{z}_{i-1})\right\}$$

For convenience we set:

$$\mathbf{H} := (\mathbf{h}_1, \ldots, \mathbf{h}_l) \qquad \mathbf{H}' := (\mathbf{h}_1, \ldots, \mathbf{h}_{l-1}) \qquad \mathbf{T} := (\mathbf{z}_1, \ldots, \mathbf{z}_{l-1}, \mathbf{w}) \qquad \mathbf{T}' := (\mathbf{z}_1, \ldots, \mathbf{z}_{l-1})$$

Further, let the random variable $\mathbf{m} \in \{l-1, \ldots, n\}$ represent the index of the latest stage where the game is not aborted; i.e. $\tilde{\tau}_i(\mathbf{T}) = \tilde{\sigma}_i(\mathbf{H}, \mathbf{T}')$ for all $i \in \{l, \ldots, \mathbf{m}\}$, and $\tilde{\tau}_{\mathbf{m}+1}(\mathbf{T}) \neq \tilde{\sigma}_{\mathbf{m}+1}(\mathbf{H}, \mathbf{T}')$ if not $\mathbf{m} = n$. Note that $\mathcal{K}$'s complete view can be deterministically reconstructed from $(\mathbf{H}, \mathbf{T}', \mathbf{m})$

Game $\Gamma_9(\mathbb{F}_q^k, n, l)$

Parametrized by a finite vector space $\mathbb{F}_q^k$ and some $n, l \in \mathbb{N}_{>0}$ with $l \leq n$. Let $\mathcal{U} := \mathbb{F}_q^{1\times k}$ and $\mathcal{H} := \mathbb{F}_q^k \backslash \{0\}$. The player $\mathcal{K}$ is computationally unbounded.

1. (a) For each $i \in \{1, \ldots, l\}$ the player $\mathcal{K}$ specifies some mapping $x_i : \mathcal{H}^l \times \mathcal{U}^{i-1} \to \mathbb{F}_q$.
   (b) For each $i \in \{l, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\tau}_i : \mathcal{U}^l \to \mathbb{F}_q^{4k\times(1+k)} \cup \{\perp\}$.
   (c) For each $i \in \{l, \ldots, n\}$ the player $\mathcal{K}$ specifies some mapping $\tilde{\sigma}_i : \mathcal{H}^l \times \mathcal{U}^{l-1} \to \mathbb{F}_q^{4k\times(1+k)}$.
2. The player $\mathcal{K}$ learns l random vectors $h_1, \ldots, h_l \xleftarrow{\text{r}} \mathcal{H}$.
3. For $i = 1, \ldots, l-1$: The player $\mathcal{K}$ learns $z_i \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_i = x_i(h_1, \ldots, h_l, z_1, \ldots, z_{i-1})\}$.
4. (a) A challenge bit $d \xleftarrow{\text{r}} \{0, 1\}$ is chosen secretly.
   (b) If $d = 0$, let $w \xleftarrow{\text{r}} \{\tilde{z} \in \mathcal{U} \mid \tilde{z}h_l = x_l(h_1, \ldots, h_n, z_1, \ldots, z_{l-1})\}$, else let $w \xleftarrow{\text{r}} \mathcal{U}$, chosen secretly.
5. For $i = l, \ldots, n$: If $\tilde{\tau}_i(z_1, \ldots, z_{l-1}, w) = \tilde{\sigma}_i(h_1, \ldots, h_l, z_1, \ldots, z_{l-1})$, then $\mathcal{K}$ is notified about that by a special message $(\top, i)$; else $\mathcal{K}$ receives a special message $(\perp, i)$ and the game is aborted in the sense that Step 6 follows next.
6. The player $\mathcal{K}$ computes and outputs a guess bit $\tilde{d} \in \{0, 1\}$. He wins the game, if $\tilde{d} = d$.

Figure 20: Final transformation of our stand-alone indistinguishability game. The difference to the game $\Gamma_8(\mathbb{F}_q^k, n, l)$ is that the player $\mathcal{K}$ must specify in Step 1 how all his future choices will depend on the information gathered so far, and the meanwhile obsolete random matrix C is omitted.

and $\mathcal{K}$'s program code. Thus, with the random variable $\tilde{\mathbf{d}} \in \{0, 1\}$ representing $\mathcal{K}$'s final guess, we have:

$$\begin{aligned}
&\mathbb{P}\big[\tilde{\mathbf{d}} = \mathbf{d}\big] \\
&= \mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 0\big] \cdot \mathbb{P}\big[\mathbf{d} = 0\big] + \mathbb{P}\big[\tilde{\mathbf{d}} = 1 \,\big|\, \mathbf{d} = 1\big] \cdot \mathbb{P}\big[\mathbf{d} = 1\big] \\
&= \tfrac{1}{2}\Big(\mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 0\big] + \mathbb{P}\big[\tilde{\mathbf{d}} = 1 \,\big|\, \mathbf{d} = 1\big]\Big) \\
&= \tfrac{1}{2}\Big(\mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 0\big] + 1 - \mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 1\big]\Big) \\
&\leq \tfrac{1}{2} + \tfrac{1}{2}\Big|\mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 0\big] - \mathbb{P}\big[\tilde{\mathbf{d}} = 0 \,\big|\, \mathbf{d} = 1\big]\Big| \\
&\leq \tfrac{1}{2} + \tfrac{1}{2}\sum\nolimits_{H,T',m}\Big|\mathbb{P}\big[(\mathbf{H}, \mathbf{T}', \mathbf{m}) = (H, T', m) \,\big|\, \mathbf{d} = 0\big] - \mathbb{P}\big[(\mathbf{H}, \mathbf{T}', \mathbf{m}) = (H, T', m) \,\big|\, \mathbf{d} = 1\big]\Big| \quad (1)
\end{aligned}$$

Now, for $H \in \mathcal{H}^l$, $T' \in \mathcal{U}^{l-1}$, $m \in \{l-1, \ldots, n\}$ we define the following sets:

$$\begin{aligned}
A_m(H, T') &:= \big\{\tilde{v} \in \mathcal{U} \mid \forall\, j \in \{l, \ldots, m\} :\ \tilde{\tau}_j(T', \tilde{v}) = \tilde{\sigma}_j(H, T')\big\} \\
\bar{A}_m(H, T') &:= A_m \backslash A_{m+1} \quad \text{with the convention that } A_{n+1}(H, T') = \emptyset
\end{aligned}$$

The intuition behind this is that $A_m(\mathbf{H}, \mathbf{T}')$ consists of all token inputs for stage l, such that the game is not aborted before stage m. Accordingly, $\bar{A}_m(\mathbf{H}, \mathbf{T}')$ consists of all token inputs for stage l, such that stage m is the latest non-aborted stage. In other words, it holds:

$$\begin{aligned}
A_m(H, T') &= \big\{w \in \mathcal{U} \mid (\mathbf{H}, \mathbf{T}', \mathbf{w}) = (H, T', w) \ \Rightarrow\ \mathbf{m} \geq m\big\} \\
\bar{A}_m(H, T') &= \big\{w \in \mathcal{U} \mid (\mathbf{H}, \mathbf{T}', \mathbf{w}) = (H, T', w) \ \Rightarrow\ \mathbf{m} = m\big\}
\end{aligned}$$

Further, for all $h \in \mathcal{H}$, $\alpha \in \mathbb{F}_q$ we define:

$$Z_\alpha(h) := \{\tilde{z} \in \mathcal{U} \mid zh = \alpha\}$$

Note that $\mathbf{w} \xleftarrow{r} Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l)$ if $\mathbf{d} = 0$, and $\mathbf{w} \xleftarrow{r} \mathcal{U}$ if $\mathbf{d} = 1$. Hence, given $H := (h_1, \ldots, h_l) \in \mathcal{H}^l$, $T' \in \mathcal{U}^{l-1}$, $m \in \{l-1, \ldots, n\}$, we can compute:

$$\begin{aligned}
&\Big|\mathbb{P}\big[\mathbf{m} = m \;\big|\; (\mathbf{d}, \mathbf{H}, \mathbf{T}') = (0, H, T')\big] \;-\; \mathbb{P}\big[\mathbf{m} = m \;\big|\; (\mathbf{d}, \mathbf{H}, \mathbf{T}') = (1, H, T')\big]\Big| \\
&= \;\Big|\mathbb{P}\big[\mathbf{w} \in \bar{A}_m(H, T') \;\big|\; (\mathbf{d}, \mathbf{H}, \mathbf{T}') = (0, H, T')\big] \;-\; \mathbb{P}\big[\mathbf{w} \in \bar{A}_m(H, T') \;\big|\; (\mathbf{d}, \mathbf{H}, \mathbf{T}') = (1, H, T')\big]\Big| \\
&= \;\left|\frac{\big|Z_{x_l(H,T')}(h_l) \cap \bar{A}_m(H, T')\big|}{\big|Z_{x_l(H,T')}(h_l)\big|} - \frac{\big|\bar{A}_m(H, T')\big|}{|\mathcal{U}|}\right| \\
&= \;q^{1-k} \cdot \Big|\big|Z_{x_l(H,T')}(h_l) \cap \bar{A}_m(H, T')\big| - \tfrac{1}{q}\big|\bar{A}_m(H, T')\big|\Big|
\end{aligned}$$

Plugging this into (1), we get:

$$\mathbb{P}[\tilde{\mathbf{d}} = \mathbf{d}] \;\leq\; \tfrac{1}{2} + \tfrac{q^{1-k}}{2} \sum_{m=l-1}^{n} \mathbb{E}\,\Big|\big|Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l) \cap \bar{A}_m(\mathbf{H}, \mathbf{T}')\big| - \tfrac{1}{q}\big|\bar{A}_m(\mathbf{H}, \mathbf{T}')\big|\Big|$$

Now we exploit that $\big|\bar{A}_m(H, T')\big| = \big|A_m(H, T') \setminus A_{m-1}(H, T')\big| = \big|A_m(H, T')\big| - \big|A_{m-1}(H, T')\big|$ by construction and analogously $\big|Z \cap \bar{A}_m(H, T')\big| = \big|Z \cap A_m(H, T')\big| - \big|Z \cap A_{m-1}(H, T')\big|$ for every $Z \subseteq \mathcal{U}$. Using this and the Triangle Inequality, we can derive:

$$\mathbb{P}[\tilde{\mathbf{d}} = \mathbf{d}] \;\leq\; \tfrac{1}{2} + q^{1-k} \sum_{m=l-1}^{n+1} \mathbb{E}\,\Big|\big|Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l) \cap A_m(\mathbf{H}, \mathbf{T}')\big| - \tfrac{1}{q}\big|A_m(\mathbf{H}, \mathbf{T}')\big|\Big|$$

I.e., we just lost the factor $\frac{1}{2}$ in front of the big sum and in return could replace each $\bar{A}_m$ by A_m. Moreover, since always $A_{l-1}(\mathbf{H}, \mathbf{T}') = \mathbb{F}_q^{1\times k}$ and $A_{n+1}(\mathbf{H}, \mathbf{T}') = \emptyset$ by definition, the first and last summand of the expression above are always zero and can be discarded; i.e. it holds:

$$\mathbb{P}[\tilde{\mathbf{d}} = \mathbf{d}] \;\leq\; \tfrac{1}{2} + q^{1-k} \sum_{m=l}^{n} \mathbb{E}\,\Big|\big|Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l) \cap A_m(\mathbf{H}, \mathbf{T}')\big| - \tfrac{1}{q}\big|A_m(\mathbf{H}, \mathbf{T}')\big|\Big| \tag{2}$$

Now we exploit that $\big\{A_m(H, T')\big\}_{H \in \mathcal{H}^l}$ can be considered as a disjoint decomposition of some subset of $\mathbb{F}_q^{1\times k}$, since by construction we have:

$$A_m(H_1, T') \neq A_m(H_2, T') \quad \Rightarrow \quad A_m(H_1, T') \cap A_m(H_2, T') = \emptyset$$

Thus, for arbitrary $\gamma \in \mathbb{R}_{>0}$ by Corollary 11 follows:

$$\mathbb{P}\left[\exists\, \alpha \in \mathbb{F}_q,\, H \in \mathcal{H}^l : \Big|\big|Z_\alpha(\mathbf{h}_l) \cap A_m(H, \mathbf{T}')\big| - \tfrac{1}{q}\big|A_m(H, \mathbf{T}')\big|\Big| > \gamma\right] \;\leq\; \frac{q^{k+1/2}}{\gamma^{3/2}} + \iota(\mathbf{h}_l, \mathbf{T}')$$

We instantiate α in this inequality by $x_l(\mathbf{H}, \mathbf{T}')$ and H by $\mathbf{H}$, which yields:

$$\mathbb{P}\left[\Big|\big|Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l) \cap A_m(\mathbf{H}, \mathbf{T}')\big| - \tfrac{1}{q}\big|A_m(\mathbf{H}, \mathbf{T}')\big|\Big| > \gamma\right] \;\leq\; \frac{q^{k+1/2}}{\gamma^{3/2}} + \iota(\mathbf{h}_l, \mathbf{T}')$$

Since $\mathbb{E}(\mathbf{x}) = \int_0^\infty \mathbb{P}[\mathbf{x} > \gamma]\,\mathrm{d}\gamma$ for every real-valued random variable $\mathbf{x} \in \mathbb{R}_{\geq 0}$, this directly implies:

$$\mathbb{E}\left|\left|Z_{x_l(\mathbf{H},\mathbf{T}')}(\mathbf{h}_l) \cap A_m(\mathbf{H},\mathbf{T}')\right| - \tfrac{1}{q}\left|A_m(\mathbf{H},\mathbf{T}')\right|\right| \;\leq\; \int_0^{q^k} \min\{1, \tfrac{q^{k+1/2}}{\gamma^{3/2}}\} + \iota(\mathbf{h}_l,\mathbf{T}')\;\mathrm{d}\gamma$$

$$=\; q^{(2k+1)/3} + \int_{q^{(2k+1)/3}}^{q^k} \tfrac{q^{k+1/2}}{\gamma^{3/2}}\;\mathrm{d}\gamma + q^k \cdot \iota(\mathbf{h}_l,\mathbf{T}') \;=\; 2q^{(2k+1)/3} - q^{(k+1)/2} + q^k \cdot \iota(\mathbf{h}_l,\mathbf{T}')$$

Moreover, by Corollary 5 we have that $\iota(\mathbf{h}_l,\mathbf{T}') < \sqrt{\exp\big((l-1)q^{2-k}\big) - 1}$. Using (2), we conclude:

$$\begin{aligned}\mathbb{P}[\tilde{\mathbf{d}} = \mathbf{d}] \;&<\; \tfrac{1}{2} + q^{1-k}\cdot(n-l+1)\cdot\left(2q^{(2k+1)/3} - q^{(k+1)/2} + q^k\cdot\sqrt{\exp\big((l-1)\cdot q^{2-k}\big)-1}\right)\\ &<\; \tfrac{1}{2} + n\cdot\left(2q^{(4-k)/3} + q\cdot\sqrt{\exp\big(n\cdot q^{2-k}\big)-1}\right)\end{aligned}$$ □

3.5.8 Concluding the security proof

We can now finally conclude our security proof by just putting things together. We first sum up what we know so far about successive hybrid games; then we conclude this whole section with our final security theorem.

Corollary 26. *For any $l \in \{1,\dots,n\}$, the hybrid games Game_{l-1} and Game_l are statistically indistinguishable, if $k \geq 5$. More particular, the statistical distance between the environment's respective views is negligible in the security parameter $\lambda := k\log q$, if only $k \geq 5$.*

Proof. For $i = 0,\dots,9$, let δ_i denote the player's advantage in the respective indistinguishability game; i.e. the maximum winning probability in the game $\Gamma_i(\mathbb{F}_q^k, n, l)$, or $\Gamma_i(\mathbb{F}_q^k, n, l, \varepsilon)$ respectively, is $\frac{1}{2} + \delta_i$. By Lemma 14, the statistical distance between the environment's views in Game_{l-1} and Game_l is upper bounded by $2\delta_0$. Furthermore, given any $\varepsilon \in \mathbb{R}_{>0}$ with $q^{(2/3+\varepsilon)k} \geq q$, it holds:

$\delta_0 \leq \delta_1$	by Lemma 15
$\delta_1 \leq \delta_2 + n\cdot\big(q^{-(2/3+\varepsilon)k} + q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\big) + \sqrt{\exp\big(n\cdot q^{2-k}\big)-1}$	by Lemma 17
$\delta_2 \leq \delta_3 + n\cdot\big(q^{1-(1/3-\varepsilon)k} + q^{1-k} + q^{2-k}\big) + \sqrt{\exp\big(n\cdot q^{2-k}\big)-1}$	by Corollary 18
$\delta_3 \leq \delta_4 + n\cdot q^{1-k/3}, \quad \text{if } q^k \geq 2^{1/\varepsilon}$	by Lemma 19
$\delta_4 \leq \delta_5$	by Lemma 20
$\delta_5 \leq \delta_6$	by Lemma 21
$\delta_6 = \delta_7$	by Lemma 22
$\delta_7 \leq \delta_8 + n\cdot q^{1-k} + \sqrt{\exp\big(n\cdot q^{2-k}\big)-1}$	by Lemma 23
$\delta_8 = \delta_9$	by Lemma 24
$\delta_9 \leq n\cdot\left(2q^{(4-k)/3} + q\cdot\sqrt{\exp\big(n\cdot q^{2-k}\big)-1}\right)$	by Lemma 25

Now, let $\varepsilon := \frac{1}{12}$ and let $k \geq 5$, which especially yields that $q^{(2/3+\varepsilon)k} \geq q$ and allows us to estimate:

$$q^{-(2/3+\varepsilon)k},\; q^{1-(1/3-\varepsilon)k},\; q^{1-k},\; q^{2-k},\; q^{1-(1/3-\varepsilon)k},\; q^{1-k/3},\; q^{(4-k)/3} \;\leq\; q^{-k/5}$$

Further let $q^k \geq n^{25/3}$. This, together with $k \geq 5$, allows us to estimate:

$$q \cdot \sqrt{\exp\left(n \cdot q^{2-k}\right) - 1} \;\leq\; q \cdot \sqrt{\exp\left(q^{2-22k/25}\right) - 1} \;<\; q \cdot \sqrt{4q^{2-22k/25}} \;=\; 2q^{2-11k/25} \;\leq\; 2q^{-k/5}$$

Putting things together, we have shown that the statistical distance between the environment's views in the hybrid games Game_{l-1} and Game_l is upper bounded by $(13n+3) \cdot \exp(-\lambda/5)$, where $\lambda := k \log q$ is the security parameter and we need that $\exp(\lambda) \geq \max\left(2^{12}, n^{25/3}\right)$. □

Theorem 27. *Let some arbitrary environment $\mathcal{Z}$ be given and some adversary $\mathcal{A}$ that corrupts Goliath. Then the view of $\mathcal{Z}$ in the ideal model with ideal functionality $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ and simulator $\mathcal{S}^{\mathrm{Goliath}}(\mathcal{A})$ is statistically indistinguishable (with security parameter $\lambda := k \log q$) from the view of $\mathcal{Z}$ in the real model with protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ and adversary $\mathcal{A}$, if only $k \geq 5$.*

Proof. By Corollary 26 we have that the statistical distance between the environment's views in successive hybrid games $\mathsf{Game}_{l-1}, \mathsf{Game}_l$ is negligible in the security parameter λ, if only $k \geq 5$. By the Union Bound, we can conclude that the statistical distance between the environment's views in Game_0 and Game_n may be at most by a factor n bigger, and hence is still negligible. Finally, by Corollary 13 we have that Game_0 is statistical indistinguishable from the ideal model, and Game_n is perfectly indistinguishable from the real model. Thus, the ideal ideal model and the real model must be statistical indistinguishable. □

4 Applications and variants of our construction

We present now how the claimed optimal constructions for multiple OTMs, Commitments, and OT do work. The respective constructions are all UC-secure. At the end of this section, in Section 4.6, we also discuss how the communication complexity of computationally secure OT protocols can be amortized by standard techniques and why this does not work for commitments or OTMs.

4.1 Unidirectional OT and OTMs with optimal communication complexity

As discussed in Section 2.4, one can reduce k-bit string-OT and $\mathbb{F}_2^k$-OAFE to each other without any overhead. However, our construction for seq-ot-OAFE has communication complexity $\Theta(k^2 \log q)$ per implemented instance of $\mathbb{F}_q^k$-OAFE. I.e., by the aforementioned reduction approach we would end up with a quadratic communication complexity, as it happened in [DKMQ11]. In contrast, if k is constant and q grows exponentially in the security parameter, we have only a communication complexity of $O(\log q)$ for each implemented instance of $\mathbb{F}_q^k$-OAFE, which obviously is optimal. Therefore, it is desirable to implement l-bit string-OT by a constant number of $\mathbb{F}_{2^l}^d$-OAFE instances with constant dimension d. We present such a reduction protocol in Figure 21; our construction needs only a single instance of $\mathbb{F}_{2^l}^2$-OAFE and the protocol idea is as follows. The $\mathbb{F}_{2^l}^2$-OAFE primitive allows the sender to specify two affine functions $f_0, f_1 : \mathbb{F}_{2^l} \to \mathbb{F}_{2^l}$, such that the receiver can evaluate both functions only once and only simultaneously on the same input. Thus, if the sender announces his OT-inputs s_0 and s_1 encrypted with $f_0(0)$ and $f_1(1)$ respectively, then the receiver can learn at most one of the values needed for decryption of s_0 and s_1. One can even go without transmitting any ciphertexts: The sender just has to choose f_0, f_1, such that $f_0(0) = s_0$ and $f_1(1) = s_1$, whereas $f_0(1)$ and $f_1(0)$ are completely random.

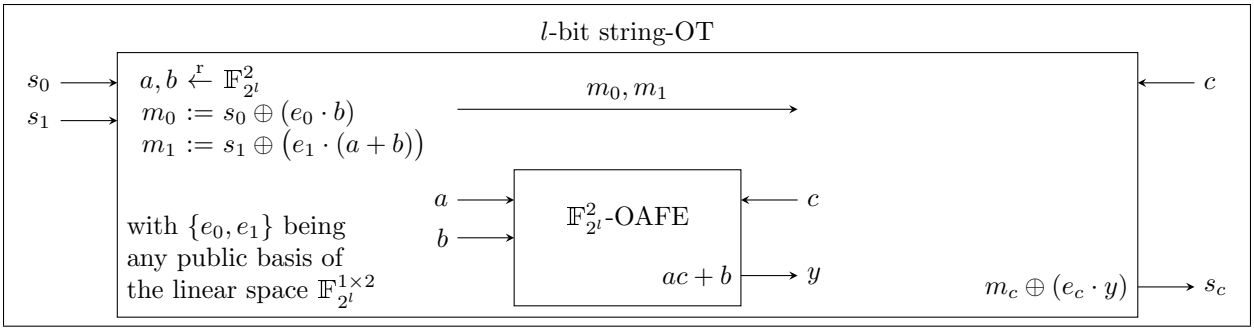

Figure 21: Reduction of l-bit string-OT to $\mathbb{F}_{2^l}^2$-OAFE. Note that the transmission of m_0 and m_1 is not essential; instead the sender can just choose (a, b) subject to the condition that $e_0 \cdot b = s_0$ and $e_1 \cdot (a + b) = s_1$.

The protocol in Figure 21 is perfectly UC-secure and works also for implementation of sequentially queriable OTM tokens from seq-ot-OAFE. Thus, in the outcome we have a construction for sequentially queriable $\log(q)$-bit OTM tokens, using only $\Theta(\log q)$ bits of communication per implemented OTM token, which is obviously optimal. To the best of our knowledge, our approach is the first to implement statistically secure OT (or OTMs respectively) with optimal communication complexity, while based only on untrusted tamper-proof hardware.

Note that our protocols with linear communication complexity also have very low computation complexity. Per implemented $\log(q)$-bit string-OT (or $\log(q)$-bit OTM respectively) every party (and in particular the exchanged token) has only to perform $O(1)$ finite field operations with field size q, which is considerably faster than, e.g., something based on modular exponentiation.

4.2 Achieving optimal communication complexity for bidirectional OT

We have shown above how one can implement unidirectional string-OT (from the token issuer to the token receiver) with optimal communication complexity, using our protocol $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ as a building block. Implementing string-OT in the other direction with still optimal communication complexity turns out a bit more challenging. The starting point for our construction is the protocol in Figure 22 for reversing the direction of a given $\mathbb{F}_q$-OAFE primitive. Note that this protocol is

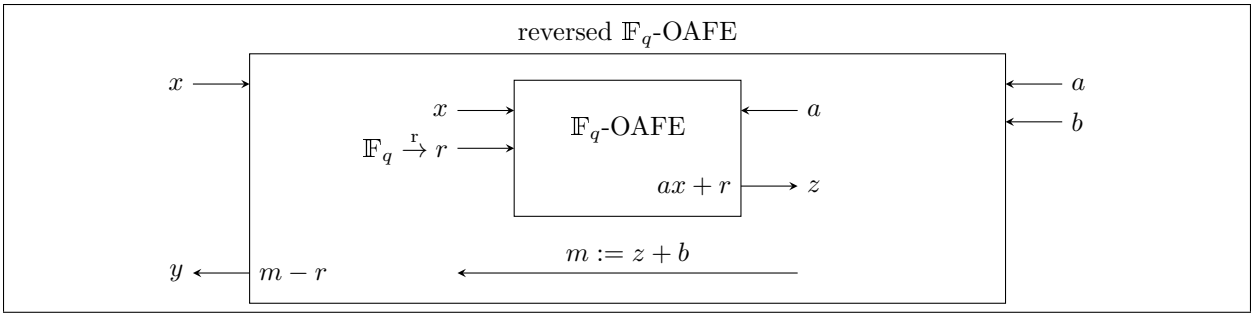

Figure 22: Basic approach for reversing the direction of a given $\mathbb{F}_q$-OAFE primitive; protocol taken from [WW06]. Note that this protocol is not UC-secure, unless input of a into the underlying $\mathbb{F}_q$-OAFE instance is enforced before the receiver outputs y; otherwise a corrupted sender can maliciously delay his choice of a (and b).

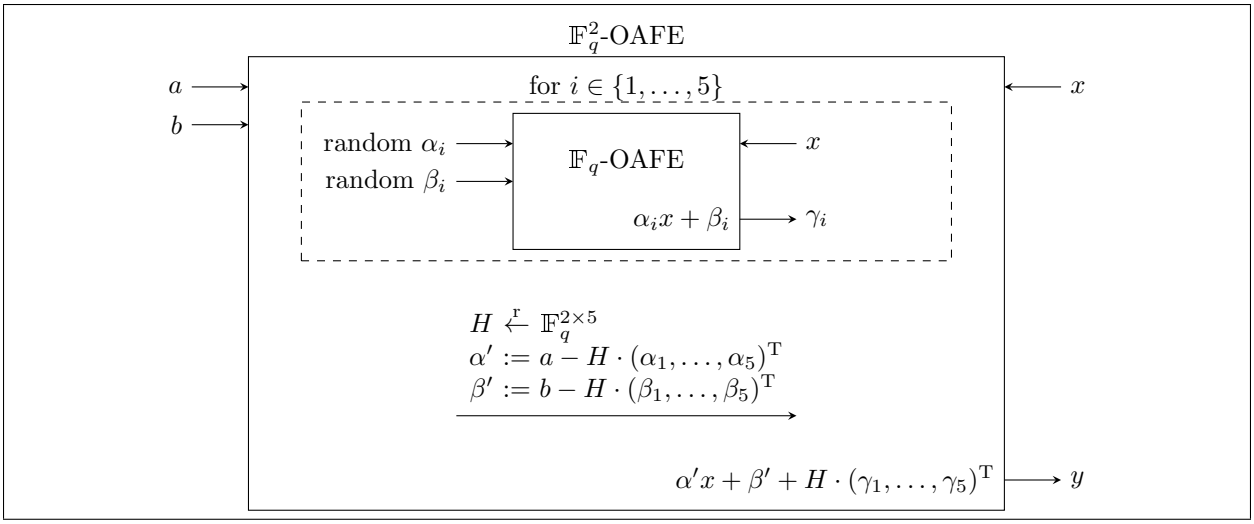

Figure 23: UC-secure implementation of $\mathbb{F}_q^2$-OAFE from five instances of $\mathbb{F}_q$-OAFE; protocol taken from [DKMQ12]. Additional measures must be taken so that H is not announced before the receiver has provided some input to all five underlying $\mathbb{F}_q$-OAFE instances in the dashed box; otherwise the protocol is inherently insecure, as shown in [DKMQ12, Lemma 1].

not UC-secure, since a corrupted sender can cause the receiver to output some y before a and b are fixed: The corrupted sender can just send a random $m \in \mathbb{F}_q$ and arbitrarily later input some $a \in \mathbb{F}_q$ of his choice into the underlying $\mathbb{F}_q$-OAFE instance (and then compute $b := m - z$). This breaches UC-security, since an ideal version of the reversed $\mathbb{F}_q$-OAFE primitive would not send y to the receiver before the sender's inputs a and b are fixed. However, in our case this problem can be solved very easily: Since our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ implements sequentially queriable OAFE instances, it suffices to use every other OAFE instance for a check announcement, i.e. both parties just input randomness and the receiver has to announce his input-output tuple.

However, the approach in Figure 22 does not work for $\mathbb{F}_q^k$-OAFE with $k > 1$, and we need $\mathbb{F}_q^2$-OAFE for our aimed at OT protocol. Thus, a construction for $\mathbb{F}_q^k$-OAFE from some instances of $\mathbb{F}_q$-OAFE would come in very handy. In [DKMQ12] one can find such a construction and a security proof for the case that $k \log q$ increases polynomially in the security parameter. For the sake of self-containedness we recap in Figure 23 the approach of [DKMQ12] with $k = 2$. By combining this with the protocol for reversed OAFE (Figure 22) and some optimization in the number of $\mathbb{F}_q$-OAFE instances used for check announcements, we end up with the protocol depicted in Figure 24.

By plugging the protocol of Figure 24 on top of $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, we get sequentially queriable $\mathbb{F}_q^2$-OAFE from the token receiver to the token sender with an overall communication complexity of $O(\log q)$ per implemented $\mathbb{F}_q^2$-OAFE instance. So, we can finally apply the construction from Section 4.1 to get string-OT with optimal communication complexity also in the direction from the token receiver to the token issuer.

4.3 Efficient protocol for string-commitments in both directions

Based on our approach for bidirectional OT, one can also construct a bidirectional and reusable commitment functionality from one tamper-proof token. However, the generic reduction via OT has suboptimal communication complexity, which can be circumvent by a direct reduction to

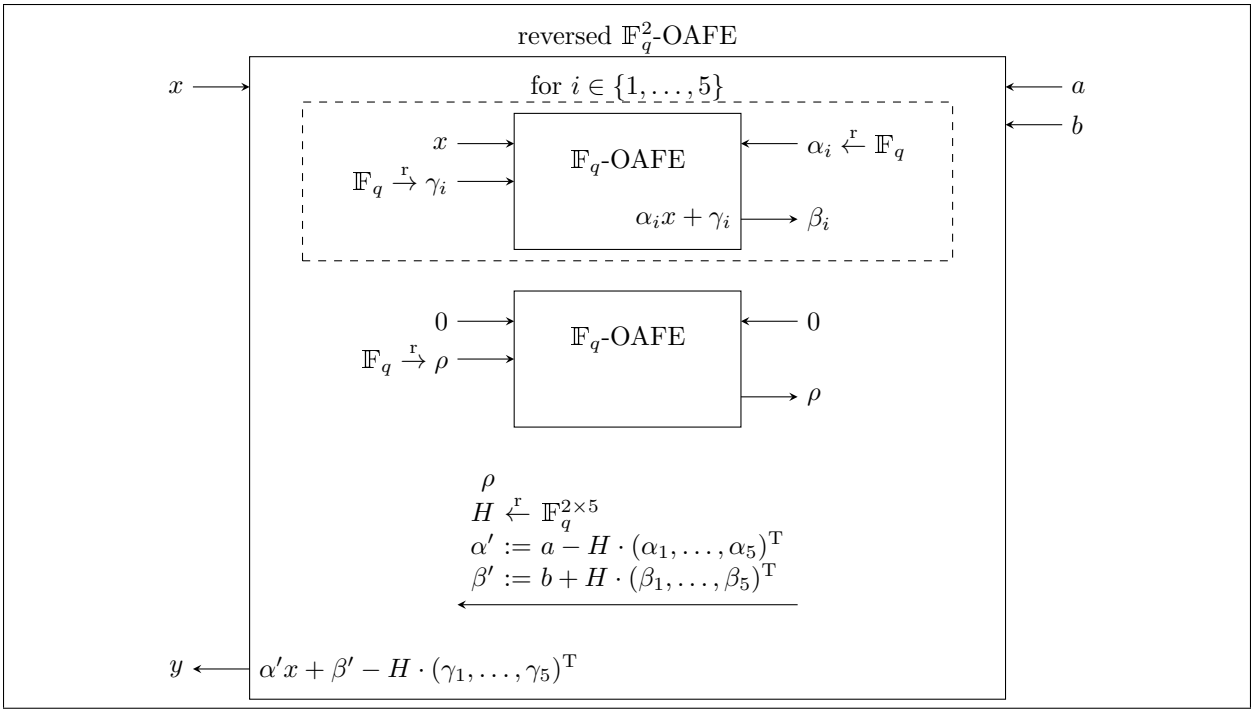

Figure 24: Combined protocol for UC-secure reversed $\mathbb{F}_q^2$-OAFE from six sequentially queriable instances of $\mathbb{F}_q$-OAFE. Note that the receiver must not output y unless ρ was announced correctly by the sender.

$\mathbb{F}_{2^l}$-OAFE, where l is the commitment length. The simple idea is as follows. If Goliath wants to commit to some $s \in \mathbb{F}_{2^l}$, he randomly picks $b \xleftarrow{r} \mathbb{F}_{2^l}$, David picks $x \xleftarrow{r} \mathbb{F}_{2^l}$, and via OAFE David learns $y := sx + b$. Due to the blinding by b, this reveals no information about s to David. Goliath can unveil the commitment just by sending (s, b). If he wants to equivocate the commitment, he needs to find some $(s', b') \neq (s, b)$ with $s'x + b' = sx + b$, which is equivalent to guessing x correctly and thus may happen at most with probability 2^{-l}.

If David wants to commit to some $s \in \mathbb{F}_{2^l}$, he just uses s as his OAFE input and Goliath chooses random inputs $a, b \xleftarrow{r} \mathbb{F}_{2^l}$. I.e., David learns $y := as + b$. To unveil, David sends (s, y) to Goliath. If he wants to equivocate the commitment, he needs to find some $(s', y') \neq (s, y)$ with $as' + b = y'$, which is equivalent to guessing (a, b) correctly. Since y contains only l bits of information about (a, b), this may happen at most with probability 2^{-l}. However, note that if this approach is based on $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ rather than some interactive OAFE implementation, additional measures must be taken to enforce that David actually queries the token in the commit phase. This can be done, exploiting the sequential nature of $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$: the next OAFE instance is just used with random inputs and David has to announce his corresponding input-output tuple. See Figure 25 for the formal details.

Furthermore note that Step 2 of the protocol $\Pi_{\mathrm{COM}}^{\mathrm{backward}}$, if based on $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, can be made non-interactive, so that the commit phase consists only of a single massage from David to Goliath. The reason is that in the send phase of $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ Goliath only sends some check information $(\tilde{r}_i, \tilde{S}_i)$ and derandomization information $(\tilde{a}_i, \tilde{b}_i)$. The former can be moved to the setup phase, because it is independent of the parties' inputs, and the latter can just be omitted, because Goliath's OAFE inputs are uniformly random anyway.

Protocol $\Pi_{\mathrm{COM}}^{\mathrm{forward}}$ (Goliath is the committing/unveiling party)

Parametrized by a string length l, which also serves as security parameter, and some runtime bound n that is polynomially bounded in l. All parties have access to a hybrid functionality $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ parametrized by the finite vector space $\mathbb{F}_{2^l}^1$ and with runtime bound n. Bit strings of length l and elements of $\mathbb{F}_{2^l}$ are identified with each other. The counter j, held by Goliath, is initialized with 0.

Commit phases:

1. Upon input $(\texttt{Commit}, s_i, i)$ from the environment, Goliath verifies that $s_i \in \{0,1\}^l$ and $i = j+1 \leq n$; else he ignores that input. Next, Goliath updates $j \leftarrow i$, chooses some random $b_i \xleftarrow{\mathrm{r}} \mathbb{F}_{2^l}$ and sends (s_i, b_i, i) to $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$.
2. David, upon receiving the message $(\texttt{ready}, i)$ from $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, picks some random $x_i \xleftarrow{\mathrm{r}} \mathbb{F}_{2^l}$. He sends (x_i, i) to $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, receives some (y_i, i) and outputs $(\texttt{committed}, i)$.

Unveil phases:

3. Upon input $(\texttt{Unveil}, i)$ from the environment, Goliath verifies that $i \leq j$; else he ignores that input. Next, Goliath sends (s_i, b_i, i) to David.
4. David verifies that $s_i x_i + b_i = y_i$. If the check is passed, he outputs (s_i, i); otherwise he outputs $(\perp, i)$.

Protocol $\Pi_{\mathrm{COM}}^{\mathrm{backward}}$ (David is the committing/unveiling party)

Parametrized by a string length l, which also serves as security parameter, and some runtime bound n that is polynomially bounded in l. All parties have access to a hybrid functionality $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ parametrized by the finite vector space $\mathbb{F}_{2^l}^1$ and with runtime bound $2n$. Bit strings of length l and elements of $\mathbb{F}_{2^l}$ are identified with each other. The counter j, held by David, is initialized with 0.

Commit phases:

1. Upon input $(\texttt{Commit}, s_i, i)$ from the environment, David verifies that $s_i \in \{0,1\}^l$ and $i = j+1 \leq n$; else he ignores that input. Next, David updates $j \leftarrow i$ and sends (i) to Goliath.
2. Goliath randomly picks $a_i, b_i, c_i, d_i \xleftarrow{\mathrm{r}} \mathbb{F}_{2^l}$ and sends $(a_i, b_i, 2i-1)$ and $(c_i, d_i, 2i)$ to $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$.
3. David, after receiving the messages $(\texttt{ready}, 2i-1)$ and $(\texttt{ready}, 2i)$ from $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, sends $(s_i, 2i-1)$ and $(0, 2i)$ to $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$. He receives some $(y_i, 2i-1)$ and $(r_i, 2i)$ and announces (r_i, i) to Goliath.
4. Goliath outputs $(\texttt{committed}, i)$.

Unveil phases:

5. Upon input $(\texttt{Unveil}, i)$ from the environment, David verifies that $i \leq j$; else he ignores that input. Next, David sends (s_i, y_i, i) to Goliath.
6. Goliath verifies that $r_i = d_i$ and $y_i = a_i s_i + b_i$. If both checks are passed, he outputs (s_i, i); otherwise he outputs $(\perp, i)$.

Figure 25: Asymptotically optimal protocols for UC-secure string-commitments in the seq-ot-OAFE hybrid model. Since the $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ primitive is only invoked in the commit phases, the same instance of $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ can be used for both protocols simultaneously; the parties just have to keep track of which data belongs to which protocol.

4.4 Computational solution for unlimited reusability of a memory-limited token

Our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ guarantees perfect security against a corrupted David (cf. Section 3.4). However, to achieve this, the token needs to be able to store $\Theta(nk^2 \log q)$ bits of information. This contradicts the idea of a tamper-proof hardware token being a small and simple device. In [MS08] it was noted, that if David is computationally bounded, then the functions stored on the token could be chosen to be pseudorandom [GGM86, HILL99]. The same is true for our construction. It suffices that the token stores a succinct seed of length $\Theta(k \log q)$ for a pseudorandom number generator F. Upon input (z_i, i) the token can compute the next pseudorandom value $(r_i, S_i) = F(i)$ and output $W_i = r_i z_i + S_i$.

Moreover, in such a setting we can achieve unbounded token reusability at the price of occasional random messages from David to Goliath. More specifically, we do not need our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ and the ideal functionality $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ to be parametrized by an explicit runtime bound n any more, but David has to send fresh random shares h_i to Goliath once the shares from the initialization phase have been used up. Our security proofs are still valid. Perfect security against a corrupted David is traded for computational security but otherwise directly carries over to the modified construction. Statistical security against a corrupted Goliath is just maintained, because David's computational boundedness implies still a polynomial upper bound for the number of token queries and a corrupted Goliath only becomes weaker if he learns his shares $h_1, \ldots, h_n$ of David's inputs $x_1, \ldots, x_n$ not all at once at the beginning of the protocol.

4.5 Truly non-interactive solution with two tokens

Our approach still needs David to send some message to Goliath, and in the computational variant with unlimited token reusability this even has to happen repeatedly. We will show in Section 5.2 that one cannot implement $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ from a single instance of $\mathcal{F}_{\mathrm{wrap}}^{\mathrm{stateful}}$ without any further interaction between sender and receiver. However, there is a generic non-interactive protocol for $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$, if *two* instances of $\mathcal{F}_{\mathrm{wrap}}^{\mathrm{stateful}}$ are in place, i.e. Goliath issues two tamper-proof tokens and David can trust that they are mutually isolated. Then, the second token can play Goliath's role in our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ with random inputs a_i and b_i. As Goliath knows the second token's random coins, derandomization of his inputs can be done as follows: If Goliath wants to replace the random input tuple (a_i, b_i) by some arbitrarily chosen (a'_i, b'_i), he just sends $(a'_i - a_i, b'_i - b_i, i)$ to David, who then has to replace his output y_i by $y'_i := y_i + (a'_i - a_i)x_i + (b'_i - b_i)$.

Note that based on the two-token protocol that implements $\mathcal{F}_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ with random Goliath inputs, Step 2 of $\Pi_{\mathrm{COM}}^{\mathrm{backward}}$ (q.v. Figure 25) can also be made truly non-interactive, as Goliath does not need to derandomize any of his inputs. All other protocols become non-interactive straightforwardly.

4.6 A note on optimal communication complexity

The string length of any computationally secure OT protocol can be polynomially extended by standard techniques (cf. protocol $\Pi_{\mathrm{OT}}^{\mathrm{enlarge}}$ in Figure 26). It is straightforward to show UC-security of this approach. Hence, optimal communication complexity of the computational versions of our OT solution is not a noteworthy result. However, applying an analogous transformation to commitments or OTMs would destroy UC-security (see Remark 28 below) and we are not aware of any universally composable amortization techniques for these primitives that do not come along with additional setup assumptions.

Protocol $\Pi_{\mathrm{OT}}^{\mathrm{enlarge}}$

Parametrized by two security parameters k and l with $k > l$, a hybrid functionality $\mathcal{F}_{\mathrm{OT}}$ for l-bit string-OT and a PRNG function F with seed length l and output length k, i.e. $F : \{0,1\}^l \to \{0,1\}^k$.

1. Upon input (s_0, s_1) from the environment, the sender verifies that $s_0, s_1 \in \{0,1\}^k$; else that input is ignored. Next, the sender chooses two random seeds $\tilde{s}_0, \tilde{s}_1 \xleftarrow{\mathrm{r}} \{0,1\}^l$ and inputs $(\tilde{s}_0, \tilde{s}_1)$ into $\mathcal{F}_{\mathrm{OT}}$.
2. Upon input x from the environment, the receiver verifies that $x \in \{0,1\}$; else that input is ignored. Next, the receiver inputs x into $\mathcal{F}_{\mathrm{OT}}$, thus receiving $\tilde{s}_x$.
3. The sender, after being notified that the receiver also did provide some input to $\mathcal{F}_{\mathrm{OT}}$, announces $r_0 := s_0 \oplus F(\tilde{s}_0)$ and $r_1 := s_1 \oplus F(\tilde{s}_1)$.
4. The receiver computes and outputs $s_x = r_x \oplus F(\tilde{s}_x)$.

Protocol $\Pi_{\mathrm{COM}}^{\mathrm{enlarge}}$

Parametrized by two security parameters k and l with $k > l$, a hybrid functionality $\mathcal{F}_{\mathrm{COM}}$ for l-bit string-commitment and a PRNG function F with seed length l and output length k, i.e. $F : \{0,1\}^l \to \{0,1\}^k$.

Commit phase:

1. Upon input $(\texttt{Commit}, c)$ from the environment, the sender verifies that $c \in \{0,1\}^k$; else that input is ignored. Next, the sender chooses some random $\tilde{s} \xleftarrow{\mathrm{r}} \{0,1\}^l$, commits to s via $\mathcal{F}_{\mathrm{COM}}$ and sends $r := c \oplus F(\tilde{s})$ to the receiver.
2. The receiver outputs (`committed`).

Unveil phase:

3. Upon input (`Unveil`) from the environment, the sender unveils $\tilde{s}$.
4. If the unveil is successful, the receiver computes and outputs $r \oplus F(\tilde{s})$; otherwise it outputs $\perp$.

Protocol $\Pi_{\mathrm{OTM}}^{\mathrm{enlarge}}$

Parametrized by two security parameters k and l with $k > l$, a hybrid functionality $\mathcal{F}_{\mathrm{OTM}}$ for l-bit OTM and a PRNG function F with seed length l and output length k, i.e. $F : \{0,1\}^l \to \{0,1\}^k$.

Creation:

1. Upon input (s_0, s_1) from the environment, the sender verifies that $s_0, s_1 \in \{0,1\}^k$; else that input is ignored. Next, the sender chooses two random seeds $\tilde{s}_0, \tilde{s}_1 \xleftarrow{\mathrm{r}} \{0,1\}^l$, sends $(\tilde{s}_0, \tilde{s}_1)$ via $\mathcal{F}_{\mathrm{OTM}}$ to the receiver and announces $r_0 := s_0 \oplus F(\tilde{s}_0)$ and $r_1 := s_1 \oplus F(\tilde{s}_1)$.
2. The receiver outputs (`ready`).

Query:

3. Upon input x from the environment, the receiver verifies that $x \in \{0,1\}$; else that input is ignored. Next, the receiver inputs x into $\mathcal{F}_{\mathrm{OTM}}$, thus receiving $\tilde{s}_x$, and computes and outputs $s_x = r_x \oplus F(\tilde{s}_x)$.

Figure 26: Straightforward approaches for enlarging the string length of some given OT, commitment or OTM functionality, using a PRNG. The protocol $\Pi_{\mathrm{OT}}^{\mathrm{enlarge}}$ is UC-secure, but $\Pi_{\mathrm{COM}}^{\mathrm{enlarge}}$ and $\Pi_{\mathrm{OTM}}^{\mathrm{enlarge}}$ are not (q.v. Remark 28).

Remark 28. The protocols $\Pi_{\text{COM}}^{\text{enlarge}}$ and $\Pi_{\text{OTM}}^{\text{enlarge}}$ in Figure 26 are not UC-secure.

Proof. We just show that $\Pi_{\text{COM}}^{\text{enlarge}}$ is not UC-secure. For $\Pi_{\text{OTM}}^{\text{enlarge}}$ one can argue analogously. Consider a passively corrupted receiver that just hands over every message to the environment. For the real model, this means that in the commit phase the environment learns some k-bit string r and in the unveil phase it learns a seed $s \in \{0,1\}^l$, such that $r \oplus F(s)$ is the honest sender's input c. Now, if the environment chooses the honest sender's input $c \in \{0,1\}^k$ uniformly at random, this is not simulatable in the ideal model. The simulator has to choose r before he learns c. Thus, using a simple counting argument, the probability that there exists any seed $s \in \{0,1\}^l$ with $r \oplus F(s) = c$ can be upper bounded by 2^{l-k}. In other words, the simulation fails at least with probability $1 - 2^{l-k}$. □

5 No-go arguments & conclusion

In this section we conclude our work by a short summery of what we achieved so far, what further improvement opportunities are left open, and which drawbacks of our work seem unavoidable (or at least hard to circumvent). We start with the negative aspects; they highlight that our results are quite close to optimal. Though, we give rather intuitive arguments than full formal proofs.

5.1 Impossibility of unlimited token reuse without computational assumptions

Our first negative result is that tokens with a limited amount of entropy can only be used to implement a limited number of statistically secure OTs. To show this we will only consider passively secure protocols and show stronger statements. Namely, given a token that can store ℓ bits of randomness, we cannot hope to instantiate more than $\ell/2$ bit-OTs between David and Goliath using this token. For passive security this is optimal. Given that the token behaves honestly, we can implement bit-OT from Goliath to David by using the token as a *selective decrypter*: Goliath one-time-pad encrypts his OT-inputs, sends them to David, and David can ask the token for one of the keys and decrypt his output. The correctness and privacy properties of this protocol follow immediately.

Now, for our impossibility argumentation assume we were given k bit-OTs between Goliath and David. In the semi-honest setting, implementing k-bit string-OT is then trivial: David just inputs the same choice-bit into each bit-OT. We thus only need to show that there is no protocol Π that implements a single k-bit string-OT using a token with at most ℓ bits of randomness, where k is significantly larger than $\ell/2$. With the above said, we can also conclude that there exists no protocol realizing k bit-OTs using a token with significantly less than $2k$ bits of randomness.

Assume we were given a correct and statistically receiver-private protocol Π that implements k-bit string-OT in the $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$-hybrid model, where the sender of the OT is also the sender of the token. We first provide an extractor $\mathsf{Ext}_c(\tau, \sigma)$ which computes the most likely David output, given David's choice bit c, a transcript τ of all messages between David and Goliath, and the token random tape σ. Notice that τ only contains the messages sent and received by David, not his complete view. Our extractor does the following. It iterates through all possible random tapes r for David, of which there are at most $2^{poly(k)}$, since we require an honest David to be efficient. For each such random tape r, $\mathsf{Ext}_c(\tau, \sigma)$ checks if r is consistent with the message transcrip τ and the token random tape σ. More precisely, $\mathsf{Ext}_c(\tau, \sigma)$ simulates David with input c and random tape r, and a token with random tape σ. For each message m sent by this simulated David to Goliath,

$\mathsf{Ext}_c(\tau,\sigma)$ checks whether m appears in the transcript τ at the appropriate position. If for any message m this is not the case, the random tape r is discarded. If m is identical to the message in τ, then $\mathsf{Ext}_c(\tau,\sigma)$ answers the message by the simulated David with the next message of Goliath in τ. In the end, the simulated David will produce an output s. $\mathsf{Ext}_c(\tau,\sigma)$ stores these David outputs in a list. After all possible random tapes r are iterated, $\mathsf{Ext}_c(\tau,\sigma)$ checks which output s appears most frequently in the list of David outputs and then outputs this s. By the correctness property of Π it must hold with overwhelming probability that $\mathsf{Ext}_c(\tau,\sigma) = s_c$, where σ and τ are generated by a real run of Π with Goliath input (s_0, s_1), David input c, and fresh random tapes for both of them.

However, it must also hold with overwhelming probability that $\mathsf{Ext}_{1-c}(\tau,\sigma) = s_{1-c}$, as otherwise an unbounded Goliath could simply compute $\big(\mathsf{Ext}_0(\tau,\sigma), \mathsf{Ext}_1(\tau,\sigma)\big)$, compare it with (s_0, s_1) and thereby learn David's choice bit c. Thus, for real runs of Π it holds with overwhelming probability that $(s_0, s_1) = \big(\mathsf{Ext}_0(\tau,\sigma), \mathsf{Ext}_1(\tau,\sigma)\big)$, regardless of Goliath's input distribution. In other words, the Shannon entropy $\mathbb{H}_1(s_0, s_1 \,|\, \tau, \sigma) =: \nu$ is always negligible.

Now we turn to show that a transcript τ alone may contain only negligible information about Goliath's input (s_0, s_1). This will conclude our argumentation, since together with the negligibility of $\mathbb{H}_1(s_0, s_1 \,|\, \tau, \sigma)$ it will yield that σ may have only negligibly less entropy than (s_0, s_1). Assume that Π is also statistically sender-private and consider protocol runs with uniformly random Goliath input (s_0, s_1). If we set $c = 0$, it must hold that $\mathbb{H}_1(s_1 \,|\, \tau, s_0) \geq k - \mu$ for some negligible μ, because of the sender-privacy property of Π. Especially, it must hold in this case that $\mathbb{H}_1(s_1 \,|\, \tau) \geq k - \mu$. Analogously, in the case of $c = 1$ it must hold that $\mathbb{H}_1(s_0 \,|\, \tau) \geq \mathbb{H}_1(s_0 \,|\, \tau, s_1) \geq k - \mu'$ for some negligible μ'. However, as an unbounded Goliath can compute $\mathbb{H}_1(s_0 \,|\, \tau = \tilde{\tau})$ and $\mathbb{H}_1(s_1 \,|\, \tau = \tilde{\tau})$ for any actually observed message transcript $\tilde{\tau}$, it must just hold that $\mathbb{H}_1(s_0 \,|\, \tau)$ and $\mathbb{H}_1(s_1 \,|\, \tau)$ are negligibly close to k in either case; otherwise Goliath could distinguish both cases and the receiver-privacy of Π would be broken. Say $\mathbb{H}_1(s_0 \,|\, \tau) \geq k - \mu''$ and $\mathbb{H}_1(s_1 \,|\, \tau) \geq k - \mu'''$ in both cases. Thus, if $c = 0$, we can estimate:

$$\mathbb{H}_1(s_0, s_1 \,|\, \tau) \;=\; \mathbb{H}_1(s_1 \,|\, \tau, s_0) + \mathbb{H}_1(s_0 \,|\, \tau) \;\geq\; (k - \mu) + (k - \mu'') \;=\; 2k - (\mu + \mu'')$$

Analogously, if $c = 1$, we can estimate:

$$\mathbb{H}_1(s_0, s_1 \,|\, \tau) \;=\; \mathbb{H}_1(s_0 \,|\, \tau, s_1) + \mathbb{H}_1(s_1 \,|\, \tau) \;\geq\; (k - \mu') + (k - \mu''') \;=\; 2k - (\mu' + \mu''')$$

Hence, we find some negligible ν', such that $\mathbb{H}_1(s_0, s_1 \,|\, \tau) \geq 2k - \nu'$ for any distribution of David's input c. Remember that we assumed Goliath's input (s_0, s_1) to be uniformly random. Since $\mathbb{H}_1(s_0, s_1 \,|\, \tau, \sigma) = H_1(s_0, s_1, \sigma \,|\, \tau) - \mathbb{H}_1(\sigma \,|\, \tau) \geq \mathbb{H}_1(s_0, s_1 \,|\, \tau) - \mathbb{H}_1(\sigma)$, we can conclude:

$$\ell \;=\; \mathbb{H}_1(\sigma) \;\geq\; \mathbb{H}_1(s_0, s_1 \,|\, \tau) - \mathbb{H}_1(s_0, s_1 \,|\, \tau, \sigma) \;\geq\; 2k - \nu' - \nu$$

I.e., since ν and ν' are negligible, there cannot exist a correct and statistically secure protocol Π that implements k-bit string-OT using a token with at most ℓ bits of randomness, where k is significantly larger than $\ell/2$.

5.2 Lower bounds for the rounds of interaction between David and Goliath

Our protocol $\Pi^{\text{seq}-\text{ot}}_{\text{OAFE}}$ needs two rounds of interaction between David and Goliath after the token is transmitted. We show now that this is a tight lower bound, even if one wishes to implement only

two OT instances (or two sequentially queriable OTMs) from one token. Our proof also holds with respect to computational assumptions, as long as David is subject to the same restrictions as the token. If just the token has to run in polynomial time but David is computationally unbounded, there might exist UC-secure protocols for $\mathcal{F}_{\text{OAFE}}^{\text{seq}-\text{ot}}$ from $\mathcal{F}_{\text{wrap}}^{\text{stateful}}$ relative to some computational assumption.

Similar to the impossibility result of [CF01], we first assume existence of a UC-protocol and then turn the simulator for a corrupted David into an unsimulatable attack by Goliath. However, first of all note that any protocol with only one round of interaction after the token transmission can be turned into a protocol without any interaction after the token transmission: If Goliath sends a message after the token transmission, he can as well implement the message on the token, and if David sends a message after the token transmission, it can just be omitted, because it does not affect any further computation. So, we only have to consider the case that after the token transmission there is no more communication between David and Goliath. By UC-security against a corrupted David, we have that David's inputs can be extracted from his communication with Goliath and the token. Since all communication between David and Goliath takes place before transmission of the token, a malicious Goliath can store a complete transcript of that communication on the token. But now, the token can use the simulator program for a corrupted David to compute David's inputs from the stored transcript and its communication with David. This allows a malicious token to encode David's first input into the second output, which is not simulatable in the ideal model.

5.3 Lower bounds for David's communication overhead

Even our refined construction for l-bit string-OT (q.v. Section 4.1) needs that David inputs $\Theta(l)$ bits into the token. One could wonder, if it is possible to implement multiple instances of OT from reusable tamper-proof tokens, such that for each implemented instance of OT the communication complexity for David is constant. We argue that this seems very improbable. The main argument is that a corrupted Goliath can correctly guess David's token inputs for the first OT instances with some constant probability. Thus, he can maliciously create the tokens, so that they immediately shut down if David's first token inputs do not match Goliath's guess. Thereby, when Goliath learns that the protocol was *not* aborted, he can reconstruct David's first OT input. Such a protocol cannot be UC-secure, since in the ideal model the abort probability may not depend on Davids inputs. Moreover, the whole argumentation still seems valid, even if we allow that David inputs polylogarithmically many bits per OT into the tokens.

5.4 Impossibility of polynomially bounded simulation runtime

The running time of our simulator $\mathcal{S}^{\text{Goliath}}(\mathcal{A})$ for a corrupted sender is not a priori polynomially bounded (cf. Section 3.5). Instead, we have only a polynomial bound for the *expected* running time (q.v. Lemma 12). The same problem occurred in [MS08] and they stated it as an open problem to find a protocol with strict polynomial-time simulation. We argue that such a protocol seems very hard to find, unless computational assumptions are used.

Since information-theoretically secure OT cannot be realized from stateless tokens, as shown by [GIMS10], it suffices to consider stateful solutions. However, simulatability is only possible if a corrupted sender's inputs can be extracted from his messages sent to the receiver and the program code of the token(s). The straightforward approach of extraction is to rewind the token, but as the token may act honestly only on some fraction of inputs, the simulator will have to rewind the

token repeatedly. In particular, a corrupted token issuer can choose some arbitrary probability p, such that the token acts honestly only with this probability p. Unless p is negligible, this will necessitate a simulator that can rewind the token for about $\frac{1}{p}$ times. Since p may be effectively chosen by the adversary (and thus by the environment) during runtime, strict polynomial-time simulation with repeated token rewinding seems impossible. Moreover, we are not aware of any information-theoretic approach (i.e. without computational assumptions) that would allow us to avoid repeated token rewinding.

5.5 Impossibility of random access solutions

Via our protocol $\Pi_{\mathrm{OAFE}}^{\mathrm{seq-ot}}$ one can implement sequentially queriable OTM tokens from a single piece of untrusted tamper-proof hardware (cf. Section 3.1 and Section 2.4). We discuss now, why it seems impossible to implement multiple OTMs that the token receiver can access in arbitrary order. The main argument is that a corrupted token issuer can try to let the token work only for the first OTM query and then shut down. This is not simulatable in the ideal model, since the simulator does not learn which OTM is queried first—the decision which OTM to query first even might be made not until the interactive part of the protocol is over.

In particular, the attack idea is as follows. Given any hypothetical protocol for random access OTMs from a single token, let b denote a lower bound of token queries that are needed for the first OTM access and let B denote an upper bound. W.l.o.g., b and B are polynomially bounded in the security parameter. The corrupted token issuer randomly picks $j \xleftarrow{\mathrm{r}} \{b, \ldots, B\}$ and programs the token such that it shuts down after the j-th query. Now, with probability $\frac{1}{B-b+1}$ the receiver will be able to access only the very OTM that is queried first. Note that this probability is independent of the access order to the implemented OTMs. Further note that by this attack it cannot happen that the OTM accessed first is malformed and any other is not. For the simulator this means an unsolvable dilemma. With non-negligible probability, all but one of the sent OTMs must be malformed and the non-malformed OTM must always be the one that is accessed first.

5.6 Conclusion & improvement opportunities

In this paper, we showed that exchange of a single untrusted tamper-proof hardware token is sufficient for general, universally composable, information-theoretically secure two-party computation. Our approach is the first to implement several widely used primitives (namely string-commitments, string-OT and sequentially queriable OTMs) at optimal rates. Moreover, our constructions have very low computation complexity. As a drawback, our information-theoretically secure protocols have only limited token reusability, but can be transformed straightforwardly into computationally secure protocols with unlimited token reusability. The computational assumption needed is the weakest standard assumption in cryptography, namely the existence of a pseudorandom number generator, and beyond that we only need the receiver party to be computationally bounded. After all, we consider our work a substantial gain towards practical secure two-party computation, but still want to point out some issues that in our view need some further improvement.

Smaller constants for better practicability. Even though we achieve asymptotically optimal communication complexity, there are some nasty constants left that might make our protocols somewhat slow in practice. In particular, for every l-bit string-OT (or l-bit OTM respectively) the token has to compute and output an $\mathbb{F}_{2^l}^{20\times 5}$-matrix, i.e. we have a blow-up factor of 100. This

enormous factor results from two technical artifacts. Firstly, we were only able to prove that our protocol $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ securely realizes $\mathbb{F}_q^k$-OAFE, if $k \geq 5$ (cf. Section 3.5.8). In contrast, we only need $\mathbb{F}_{2^l}^2$-OAFE for our optimal l-bit string-OT protocol (cf. Section 4.1) and are not aware of any potential attack against $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ with $k = 2$. Secondly, for technical reasons we need that David chooses a check matrix C of dimension $3k \times 4k$ in Step ii of the setup phase (q.v. Figure 6) and later computes a check value CW_i from the i-th token output W_i, i.e. we especially need that W_i has dimension $4k \times k$. However, we are not aware of any potential attack, if only $C \in \mathbb{F}_q^{\alpha k \times (1+\alpha)k}$ with constant $\alpha > 0$. Now, if we choose $\alpha = \frac{1}{2}$ and $k = 2$, this means that David chooses a check matrix C of dimension 1×3 and the token just needs to compute and output $\mathbb{F}_{2^l}^{3\times 2}$-matrices. In other words, we believe that the blow-up factor can be reduced from 100 to 6 just by more sophisticated proof techniques and a slight modification of the protocol.

Less interaction. Our protocol $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ is non-interactive in the sense that there is no message from David to Goliath after the setup phase. However, this approach comes along with two drawbacks. Firstly, the message from David to Goliath in the setup phase is quite large. Secondly, David either needs to know an upper bound for the number of upcoming send phases, what clearly rules out unlimited token reusability, or we must allow that David occasionally sends some fresh randomness to Goliath. As a solution for both drawbacks we suggest that David just sends a random seed of a PRNG. We believe (but were not able to prove) that this does not breach security, as long as Goliath and the token are computationally bounded.

More realistic hardware assumptions. For security of our protocol $\Pi_{\text{OAFE}}^{\text{seq-ot}}$ against a corrupted Goliath we need that the tamper-proof token in David's hands and the token issuer Goliath are perfectly isolated from each other. This assumption is questionable, since one cannot prevent Goliath from placing a very powerful communication device near David's lab. At least, this will enable Goliath to send some messages to the token. However, we hold the view that the token's transmitting power can be reliably bounded by its weight and size, so that it cannot send any messages back to Goliath. Still, even a unidirectional channel from Goliath to the token suffices to break our protocols.

Therefore, we propose a two-token solution (namely that of Section 4.5), where one token just plays Goliath's role of the original protocol. As long as neither token can *send* any message, the tokens are mutually isolated and everything seems well except for one subtle issue: Goliath can change the behavior of the tokens during runtime and thus change his OAFE inputs without being noticed. However, this may be considered unavoidable in real world applications, since a very similar attack could also be mounted if adversarially issued tokens contain clocks.

Closing the gap between primitives and general secure two-party computation. By our approach we implement OT (and OTMs respectively) via some quite general $\mathbb{F}_q^k$-OAFE functionality. However, $\mathbb{F}_q^k$-OAFE is strictly stronger than OT in the sense that in general many OT instances and a quite sophisticated protocol are needed to implement $\mathbb{F}_q^k$-OAFE, whereas l-bit string-OT can be implemented rather straightforwardly from a single instance of $\mathbb{F}_2^l$-OAFE or $\mathbb{F}_{2^l}^2$-OAFE (cf. Section 2.4 and Section 4.1 respectively). This raises the question, whether one could base general secure two-party computation directly on $\mathbb{F}_q^k$-OAFE rather than OT, e.g. via (garbled) arithmetic circuits [Cle91, CFIK03, AIK11], and thereby possibly reduce the computational overhead. More generally, one could also try to implement other sorts of functions directly on the tamper-proof hardware.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[AIK11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In Rafail Ostrovsky, editor, *Proceedings of FOCS 2011*, pages 120–129. IEEE, 2011.

[AL07] Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. In Salil P. Vadhan, editor, *Theory of Cryptography, Proceedings of TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 137–156. Springer, 2007.

[BBR88] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, 1988.

[Bea95] Donald Beaver. Precomputing oblivious transfer. In Don Coppersmith, editor, *Advances in Cryptology, Proceedings of CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 1995.

[BKMN09] Julien Brouchier, Tom Kean, Carol Marsh, and David Naccache. Temperature attacks. *IEEE Security & Privacy*, 7(2):79–82, 2009.

[BOGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of STOC 1988*, pages 113–131. ACM, 1988.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of FOCS 2001*, pages 136–145, 2001. Revised version online available at `http://eprint.iacr.org/2000/067`.

[CDPW07] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In Salil P. Vadhan, editor, *Theory of Cryptography, Proceedings of TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 61–85. Springer, 2007.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology, Proceedings of CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 2001.

[CFIK03] Ronald Cramer, Serge Fehr, Yuval Ishai, and Eyal Kushilevitz. Efficient multi-party computation over rings. In Eli Biham, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 596–613. Springer, 2003.

[CGS08] Nishanth Chandran, Vipul Goyal, and Amit Sahai. New constructions for UC secure computation using tamper-proof hardware. In Nigel P. Smart, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 545–562. Springer, 2008.

[CKS+14] Seung Geol Choi, Jonathan Katz, Dominique Schröder, Arkady Yerukhimovich, and Hong-Sheng Zhou. (Efficient) universally composable oblivious transfer using a minimal number of stateless tokens. In Yehuda Lindell, editor, *Theory of Cryptography, Proceedings of TCC 2014*, volume 8349 of *Lecture Notes in Computer Science*, pages 638–662. Springer, 2014.

[Cle91] Richard Cleve. Towards optimal simulations of formulas by bounded-width programs. *Computational Complexity*, 1:91–105, 1991.

[DKMQ11] Nico Döttling, Daniel Kraschewski, and Jörn Müller-Quade. Unconditional and composable security using a single stateful tamper-proof hardware token. In Yuval Ishai, editor, *Theory of Cryptography, Proceedings of TCC 2011*, volume 6597 of *Lecture Notes in Computer Science*, pages 164–181. Springer, 2011.

[DKMQ12] Nico Döttling, Daniel Kraschewski, and Jörn Müller-Quade. Statistically secure linear-rate dimension extension for oblivious affine function evaluation. In Adam Smith, editor, *Information Theoretic Security, Proceedings of ICITS 2012*, volume 7412 of *Lecture Notes in Computer Science*, pages 111–128. Springer, 2012.

[DNW08] Ivan Damgård, Jesper Buus Nielsen, and Daniel Wichs. Isolated proofs of knowledge and isolated zero knowledge. In Nigel P. Smart, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 509–526. Springer, 2008.

[DNW09] Ivan Damgård, Jesper Buus Nielsen, and Daniel Wichs. Universally composable multiparty computation with partially isolated parties. In Omer Reingold, editor, *Theory of Cryptography, Proceedings of TCC 2009*, volume 5444 of *Lecture Notes in Computer Science*, pages 315–331. Springer, 2009.

[FPS+11] Marc Fischlin, Benny Pinkas, Ahmad-Reza Sadeghi, Thomas Schneider, and Ivan Visconti. Secure set intersection with untrusted hardware tokens. In *Proceedings of the 11th international conference on Topics in cryptology: CT-RSA 2011*, CT-RSA'11, pages 1–16, Berlin, Heidelberg, 2011. Springer-Verlag.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GIMS10] Vipul Goyal, Yuval Ishai, Mohammad Mahmoody, and Amit Sahai. Interactive locking, zero-knowledge PCPs, and unconditional cryptography. In Tal Rabin, editor, *Advances in Cryptology, Proceedings of CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 173–190. Springer, 2010.

[GIS+10] Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In Daniele Micciancio, editor, *Theory of Cryptography, Proceedings of TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 308–326. Springer, 2010.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. One-time programs. In David Wagner, editor, *Advances in Cryptology, Proceedings of CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2008.

[GO96] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, 1996.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HMQU05] Dennis Hofheinz, Jörn Müller-Quade, and Dominique Unruh. Universally composable zero-knowledge arguments and commitments from signature cards. In *Proceedings of the 5th Central European Conference on Cryptology MoraviaCrypt 2005*, 2005.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstract). In *Proceedings of STOC 1989*, pages 12–24. ACM, 1989.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *Advances in Cryptology, Proceedings of CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008.

[Kat07] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In Moni Naor, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 115–128. Springer, 2007.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings of STOC 1988*, pages 20–31. ACM, 1988.

[Kol10] Vladimir Kolesnikov. Truly efficient string oblivious transfer using resettable tamper-proof tokens. In Daniele Micciancio, editor, *Theory of Cryptography, Proceedings of TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 327–342. Springer, 2010.

[MS08] Tal Moran and Gil Segev. David and Goliath commitments: UC computation for asymmetric parties using tamper-proof hardware. In Nigel P. Smart, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 527–544. Springer, 2008.

[Rab81] Michael O. Rabin. How to exchange secrets by oblivious transfer. Technical report, Aiken Computation Laboratory, Harvard University, 1981.

[Wul07] Jürg Wullschleger. Oblivious-transfer amplification. In Moni Naor, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 555–572. Springer, 2007.

[WW06] Stefan Wolf and Jürg Wullschleger. Oblivious transfer is symmetric. In Serge Vaudenay, editor, *Advances in Cryptology, Proceedings of EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 222–232. Springer, 2006.