A Framework for Interactive Argument Systems using Quasigroupic Homorphic Commitment

Luís Aguiar Brandão*

November 7, 2006

Abstract

Using a model based on *probabilistic functions* (PF), it's introduced the concept of *perfect zero knowledge* (PZK) *commitment scheme* (CS) allowing *quasigroupic homomorphic commitment* (QHC). Using QHC of $+_m$ (modular sum in $\mathbb{Z}_m$), application is considered in interactive argument systems (IAS) for several languages. In four of the examples – generalized nand ($[\overline{\wedge_{(\alpha)}}]$), string equality ($[=_{(m,\alpha,)}]$), string inequality ($[\neq_{(m,\alpha,)}]$) and graph three-colourations ($G3C$) – complexity improvements are obtained, in comparison to other established results. Motivation then arises to define a general framework for PZK-IAS for membership in language with committed alphabet ($MLCA$), such that the properties of soundness and PZK result from high-level parametrizable aspects. A general simulator is constructed for sequential and (most interestingly) for parallel versions of execution. It therefore becomes easier to conceptualize functionalities of this kind of IAS without the consideration of low level aspects of cryptographic primitives. The constructed framework is able to embrace PZK-CS allowing QHC of functions that are not themselves quasigroupic. Several theoretical considerations are made, namely recognizing a necessary requirements to demand on an eventual PZK-CS allowing QHC of some complete function in a Boolean sense.

*(E-mail) criptog@criptog.com, (Address) Estação Correios Miraflores, ap.1021, 1496-701-Algés, Portugal.

Contents

1 Introduction

Interactive Proof Systems (*IPS*) with *Zero Knowledge* (*ZK*), as introduced in [2], allow a prover P with unlimited computational power to convince a polynomially bounded verifier V that a given public element belongs to a certain language, without any other relevant information being transmitted. A variant of this notion – *Interactive Argument System* (*IAS*) – was introduced in [4], considering instead a polynomailly bounded P and a possibly unbounded V and prefering *computational* to *unconditional soundness* and *perfect* to *computational ZK*.

Related with these notions, *Commitment Schemes* (*CS*), "digital analogues of non-transparent sealed envelopes" ([10]), allow interesting functionalities. Already in [4] was developed a *CS* with unconditional security for bits (0 or 1), in the sense that *blobs* (published commitments) have no computational relation with the committed bit. It then becomes possible for P, by randomly selecting permutations of the truth table of function *Nand*, to convince V that some *blob* vector commits some satisfiable Boolean formula (and from that, trivially, any *NP*-language).

Meanwhile, the concept of "homomorphic encryption" (*HC*), allowing specific algebraic operations to be performed in encrypted data without necessity of decryption, formalizes the notion of "computing on encrypted bits" [4]. Several examples of cryptographic systems allowing *HC* have already been proposed (see examples in [13]).

In the beginning of this paper, with the intent of setting the basis for a framework to use, several concepts and respective notation are introduced by means of definitions, namely for *probability distribution* (*PD*), *probabilistic function* (*PF*), *perfect zero knowledge commitment scheme* (*PZK-CS*) and *quasigroupic homomorphic commitment* (*QHC*). Then, the concept of *perfect zero knowledge interactive argument system for membership in language with committed alphabet* (*PZK-IAS-MLCA*) is introduced as a way of arguing knowledge about a secret decommitment that codifies an element of some language.

Initial examples are given and in some cases are pointed out complexity improvements in comparison to other established results, namely for languages (with committed elements): generalized nand ($[\overline{\wedge_{(\alpha)}}]$), string equality ($[=_{(m,\alpha,)}]$), string inequality ($[\neq_{(m,\alpha,)}]$) and graph three-colourability (*G3C*). The similarities and differences in the set of examples motivates the construction of a general framework where all can fit as specific parametrizations.

An immediate benefit of such a highly parameterizable framework is the ability to reduce the description of requirements to a small number of high-level conditions that a *PZK-CS* and other paramenters should satisfy in order that a *PZK-IAS* is suitable for a given language. Soundness and *PZK* aspects are proven to exist as a consequence of the requirements. Moreover, the emphasis in high level aspects enables the conceptualization of protocols and its functionalities without accounting for low-level aspects of cryptographic primitives. General simulators are constructed for sequential and (most interestingly) for parallel versions of the *IAS*.

The framework is prepared to consider *PZK-CS* allowing *QHC* of functions that (in high level) are not themselves quasigroupic. In a inal consideration, a necessary requirement is identified that a *PZK-CS* must satisfy in order to allow *QHC* of some complete function in a Boolean sense, such as nand ($\bar{\wedge}$).

2 Preliminary Definitions

Some basic definitions and notation[1] are useful to the structures and results presented in this article.

Definition 2.1 (Probability Distribution) *A probability distribution (PD) over set Y is a function $g : Y \to [0,1]$ satisfying $\sum_{y \in Y} g(y) = 1$. $PD(Y)$ denotes the set of all PD over Y; $y \leftarrow [g] : Y$ indicates that variable y is assigned a value selected from Y using PD $g \in PD(Y)$.*

Definition 2.2 (Probabilistic Function) *A probabilistic function (PF) from set X to set Y is a function $\phi : X \to PD(Y)$. Let $\phi(x) \equiv \phi_x$. ϕ is* ***deterministic*** *if $(\forall x \in X)(\exists y \in Y)(\phi_x(y) = 1)$,* ***injective*** *if $(\forall x, x', y \in X, X, Y)(\phi_x(y)\,\phi_{x'}(y) > 0 \Rightarrow x = x')$ and* ***surjective*** *if $(\forall y \in Y)(\exists x \in X)(\phi_x(y) > 0)$. $\Re(X,Y)$ denotes the set of all PFs from X to Y.*

Definition 2.3 (Inverse of a *PF*) *Let $g \in PD(X)$ and $\phi \in \Re(X,Y)$. The $\langle g \rangle$-inverse of ϕ is a PF $\phi^{\langle -1,g \rangle} \in \Re(Y,X)$ satisfying ${\phi^{\langle -1,g \rangle}}_y(x) = g(x)\,\phi_x(y) / \sum_{x' \in X} g(x')\,\phi_{x'}(y)$ for all $x, y \in X, Y$ such that $g(x)\,\phi_x(y) \neq 0$ and ${\phi^{\langle -1,g \rangle}}_y(x) = 0$ if $g(x)\,\phi_x(y) = 0$. (Generalization to inverses of PFs with more than one argument of input or output is trivial and is thus considered implicit.)*

Informal Definition 2.4 (Computability and one-way-ness) *A PF ϕ or a PD g is computable if, given a set ($\Re$) of "available" PFs or PDs, it's possible to compute a PF ϕ' "similar" to ϕ or a PD g' "similar" to g, respectively. A PF ϕ is $\langle g \rangle$-one-way if it's computable but its $\langle g \rangle$-inverse is not. (Specific formal definitions of this concepts can be widely found in literature)*

As a first application of these definitions, commitment schemes (CS) will now be considered. Informally [8], a CS is a procedure by which a prover (P) compromises information to a verifier (V), without V being able to gain knowledge about it and without P being able to decommit dishonest information. Consider the following descriptions of steps of a particular CS.

Initialization Consider set K of security parameters, set N of public key parameters, set T of private key (trapdoor) parameters and family $\phi^{(0)} \in \Re(K, N \times T)$ of initialization PFs. Procedure: V chooses security parameter $k \in K$ and computes a pair $\langle n,t \rangle \leftarrow \left[{\phi^{(0)}}_k\right] : N \times T$ of public and private keys, univocally defining a one-way function $f^{(n,2)}$ with trapdoor t. Keeping secret the value of t, V publishes n and by means of some IPS or IAS, V convinces[2] P that n belongs to N and (for parallel versions) that it has the ability to invert $f^{(n,2)}$ (Consult Appendix B).

Codification and decodification Consider alphabet set Δ and, for each $n \in N$, codification set $X_{(n)}$, along with set $X = \bigcup_{n \in N} X_{(n)}$. Consider also family $\phi^{(1)} = \left\{\phi^{(n,1)} \in \Re\left(\Delta, X_{(n)}\right) : n \in N\right\}$ of *injective* and *surjective* codification PFs and family $f^{(1)} = \left\{f^{(n,1)} \in F\left(X_{(n)}, \Delta\right) : n \in N\right\}$ of *surjective* decodification functions, where each $\phi^{(n,1)}$ is a uniform inverse of $f^{(n,1)}$. Procedure: Each $\vec{d} \in \vec{\Delta}$ is codified by P as $\vec{x} \leftarrow \left[\phi^{(n,1)}\left(\vec{d}\right)\right] : X$ and each $\vec{x}$ is decodified by V as $\vec{d} = f^{(n,1)}(\vec{x})$.

Commitment and decommitment Consider for each $n \in N$ commitment set $Y_{(n)}$, set $Y = \bigcup_{n \in N} Y_{(n)}$ and family $f^{(2)} = \left\{f^{(n,2)} \in F\left(X_{(n)}, Y_{(n)}\right) : n \in N\right\}$ of *non-injective* committing functions. Procedure: To commit $\vec{x}$ (with decodification $\vec{d} = f^{(n,1)}(\vec{x})$), P calculates *blob* $\vec{y} = f^{(n,2)}(\vec{x})$ and sends it to V. To decommit $\vec{y}$, P simply sends x to V.

Procedure sketch: $(k \in K) \overset{\phi^{(0)}}{\to} (\langle n,t \rangle \in N \times T)$ and $(d \in \Delta) \underset{f^{(n,1)}}{\overset{\phi^{(n,1)}}{\rightleftarrows}} \left(x \in X_{(n)}\right) \overset{f^{(n,2)}}{\to} \left(y \in Y_{(n)}\right)$.

[1] Consult *Appendix* A for some notation.

[2] Note here the temporary exchange of roles – V is a prover and P a verifier.

Definition 2.5 (Perfect Zero Knowledge Commitment Scheme (*PZK-CS*)) *A PZK-CS is a CS procedure, with steps as defined above by sets K, N, T, Δ, X, Y and families $\phi^{(0)}$, $\phi^{(1)}$, $f^{(1)}$, $f^{(2)}$ of computable PFs and functions, satisfying the following properties:*
(Let $[...]^?$ denote a predicate returning 0 if ... is false and 1 if it's true.)

1. *PZK in blobs: Exists a PF $\phi^{(PZK)} \in \Re(N,Y)$ satisfying, for all $d,n,y \in \Delta, N, Y$, ${\phi^{(PZK)}}_n(y) = \sum_{x\in X_{(n)}} {\phi^{(n,1)}}_d(x)\left[f^{(n,2)}(x) =^? y\right]^?$ (i.e. blob $\vec{y}$ is independent of $\vec{d}$).*

2. *One-way-ness in $f^{(2)}$: For each $n, x \in N, X_{(n)}$ consider class $\bar{x} \equiv \left\{x' \in X_{(n)} : \wedge_{i\in\{1,2\}} f^{(n,i)}(x') = f^{(n,i)}(x)\right\}$, set $\overline{X_{(n)}} \equiv \left\{\bar{x} : x \in X_{(n)}\right\}$ of classes and family $\overline{f^{(2)}} = \left\{\overline{f^{(n,2)}} \in \Re\left(\overline{X_{(n)}}, Y_{(n)}\right) : n \in N\right\}$ of functions satisfying $\left(\forall x \in X_{(n)}\right)\left(\overline{f^{(n,2)}}(\bar{x}) = f^{(n,2)}(x)\right)$. $\overline{f^{(2)}}$ **is one-way** in the sense that, given $\langle x,y\rangle$ such that $y = f^{(n,2)}(x)$, it's **infeasible** – without a trapdoor t obtained by $\langle n,t\rangle \leftarrow \left[{\phi^{(0)}}_k\right]$ – to find x' such that $f^{(n,2)}(x') = f^{(n,2)}(x)$ and $f^{(n,1)}(x') \neq f^{(n,1)}(x)$, although it may be "fesible" to find $x' \neq x$ such that $x' \in \overline{x}$.*[3]

3. *Soundness: Exists a **computable** PF $\phi^{(SND)} \in \Re(N \times \langle X \times X\rangle \times Y \times \Delta, X)$ satisfying ${\phi^{(SND)}}_{n,\langle x^{(1)},x^{(2)}\rangle,y,d}(x) = \left[{\phi^{(n,1)}}_d(x) \times a_0(d,x,y) \middle/ a_1(d,y)\right] \times a_2\left(x^{(1)},x^{(2)}\right)$, for each $n, x^{(1)}, x^{(2)}, y, d \in N, X_{(n)}, X_{(n)}, Y, \Delta$, with $a_0(d,x,y) = \left[f^{(n,1)}(x) = d\right]^?\left[f^{(n,2)}(x) = y\right]^?$, $a_1(d,y) = \sum_{x'\in X_{(n)}} {\phi^{(n,1)}}_d(x') \times a_0(d,x',y)$ and $a_2\left(x^{(1)},x^{(2)}\right) = \left[f^{(n,2)}\left(x^{(1)}\right) = f^{(n,2)}\left(x^{(2)}\right)\right]^?\left[f^{(n,1)}\left(x^{(1)}\right) \neq f^{(n,1)}\left(x^{(2)}\right)\right]^?$ (i.e. a pair $\left\langle x^{(1)},x^{(2)}\right\rangle$ satisfying simultaneously $f^{(n,2)}\left(x^{(1)}\right) = f^{(n,2)}\left(x^{(2)}\right)$ and $f^{(n,1)}\left(x^{(1)}\right) \neq f^{(n,1)}\left(x^{(2)}\right)$ is a trapdoor element for $f^{(n,2)}$, because it allows its $\left\langle\phi^{(n,1)}(d)\right\rangle$-inversion $f^{(n,2)}$ for any $d \in \Delta$).*

Some types of *CS* enable the possibility of "homomorphic commitment" (consult [3] for the concept of "computing on encrypted bits" and [13] for a list of examples). After the two following auxiliary definitions, a similar though more specialized notion – quasigroupic homomorphic commitment (*QHC*) – is presented, with relation to the just defined *PZK-CS* structure.

Informal Definition 2.6 (Relevant arguments of input – function γ) *Consider a positive integer α and a function $\xi \in F(X^\alpha, X)$. By definition, $\gamma(\xi) \subseteq \mathbb{Z}_\alpha$ is the set of indices of relevant arguments of input of ξ, i.e. the arguments whose input is susceptible of influencing the output. Note: each index is given as the number of the argument less 1, thus for an input with α arguments, the respective indices run from 0 to $\alpha - 1$. (consult appendix B for a formal definition of γ).*

Definition 2.7 (Quasigroupic arguments) [4] *Consider sets $S_{(0)}, ..., S_{(\alpha)}$ with equal cardinality. A function $f \in F\left(\times_{j\in\mathbb{Z}_\alpha} S_{(j)}, S_{(\alpha)}\right)$ is said to be quasigroupic in argument with index $j \in \mathbb{Z}_\alpha$ if $\left(\forall \overrightarrow{s^{(1)}} \in \times_{j\in\mathbb{Z}_\alpha} S_{(j)}\right) S_{(\alpha)} = \left\{f\left(\overrightarrow{s^{(2)}}\right) : \left({s^{(2)}}_j \in \times_{j\in\mathbb{Z}_\alpha} S_{(j)}\right) \wedge \left(\wedge_{k\in\mathbb{Z}_{\alpha''}:k\neq j}\, {s^{(1)}}_k = {s^{(2)}}_k\right)\right\}$.*

Definition 2.8 (Quasigroupic Homomorphic Commitment (*QHC*)) *A PZK-CS is said to allow QHC of function $\diamondsuit \in F(\Delta^\alpha, \Delta)$ over alphabet Δ if for every $n \in N$ exists at least one function $\xi \in F\left({X_{(n)}}^\alpha, X_{(n)}\right)$ such that:*

1. *$f^{(n,1)} : \left\langle X_{(n)}, \xi\right\rangle \rightarrow \langle\Delta, \diamondsuit\rangle$ is an homomorphism, i.e. $f^{(n,1)}(\xi(\vec{x})) = \diamondsuit\left(f^{(n,1)}(\vec{x})\right)$.*

[3]For the sake of simplicity it's avoided the formal definition of family of one-way functions. It's intentional however that no dependence exists with an assumption that $P \neq NP$. All that is needed is that exist functions whose computational evaluation is sufficiently more "easy" that the computation of it's inverse.

[4]This definition is inspired in the concept of quasigroup: A pair $\langle S, *\rangle$ of a set S and a binary operation $*$ is a quasigroup if for each a and b in S there exist unique elements x and y in S such that $a * x = b$ and $y * a = b$.

2. *Exists a function* $\psi \in F\left(Y_{(n)}{}^{\alpha}, Y_{(n)}\right)$ *such that* $f^{(n,2)} : \langle X_{(n)}, \xi \rangle \rightarrow \langle Y_{(n)}, \psi \rangle$ *is an homomorphism, i.e.* $f^{(n,2)}(\xi(\vec{x})) = \psi\left(f^{(n,2)}(\vec{x})\right)$.

3. *For every* $\vec{d} \in \Delta^{\alpha}$*, consider element* $d' \equiv \diamondsuit\left(\vec{d}\right)$*, set* $X_{(n,d)} \equiv \left\{x \in X_{(n)} : f^{(n,1)}(x) = d\right\}$ *and function* $\overline{\xi^{\left(\vec{d}\right)}} : \times_{j \in \mathbb{Z}_{\alpha}} X_{(n,d_j)} \rightarrow X_{(n,d')}$ *defined by* $\left(\forall \vec{x} \in \times_{j \in \mathbb{Z}_{\alpha}} X_{(n,d_j)}\right)\left(\overline{\xi^{\left(\vec{d}\right)}}(\vec{x}) = \xi(\vec{x})\right)$.

   Condition statement: Function $\xi^{\left(\vec{d}\right)}$ *is quasi-groupic for all its relevant arguments of input.*

Let $\nabla \equiv F(\Delta^{\alpha}, \Delta)$ and let $\Xi_{(n,\diamondsuit)}$ and $\Psi_{(n,\diamondsuit)}$ stand for the set of functions ξ and ψ, respectively, satisfying the above conditions for a selected parameter $n \in N$ in a *PZK-CS* allowing *QHC* of $\diamondsuit \in \nabla$. Consider from this point forward that exists an implicitly defined family $q = \left\{q^{(n)} : n \in N\right\}$ of functions $q^{(n)} : \nabla \rightarrow F\left(X_{(n)}{}^{\alpha}, X_{(n)}\right)$ satisfying $(\forall \diamondsuit \in \nabla)\left(q^{(n)}(\diamondsuit) \in \Xi_{(n,\diamondsuit)}\right)$ and $q^{(n)}(\diamondsuit) = \xi \Rightarrow \phi^{(n,1)}{}_{\diamondsuit}(\xi) = 1$.

QHC sketch: $\langle d \in \Delta, \diamondsuit \rangle \underset{f^{(n,1)}}{\overset{\phi^{(n,1)}}{\rightleftarrows}} \left\langle x \in X_{(n)}, \xi \in \Xi_{(n,\diamondsuit)} \right\rangle \overset{f^{(n,2)}}{\rightarrow} \left\langle y \in Y_{(n)}, \psi \in \Psi_{(n,\diamondsuit)} \right\rangle$

Comment As a specific example, consider the *CS* used in [1], based on Jacobi Symbol and Blum integers, that satisfies the properties here required for *PZK-CS* allowing *QHC* of $+_2$ (sum *mod* 2). In that case $f^{(n)}$ is squaring modulo n, and $q^{(n)}(+_2) = \times_m$ (multiplication *mod* n). Note that *QHC* of a function $\diamondsuit$ doesn't imply that $\diamondsuit$ itself is a quasigroup. In fact, a major breakthrough would be exactly to find a *PZK-CS* allowing *QHC* of some function that is not a quasigroup, namely function Nand ($\overline{\wedge} \in F\left(\Delta^2, \Delta\right)$, with $\Delta = \{0, 1\}$.

3 Membership in Language with Committed Alphabet: Examples

Consider a language L and a relation $r \subseteq X \times Y$ satisfying $\langle x, y \rangle \in r \Rightarrow y \in L$. In a typical Interactive Proof System ([2]) for language L, given a public element element $y \in L$, P wants to prove (to V) the knowledge of a certificate of membership of y, i.e. of an element x such that $\langle x, y \rangle \in r$. For the *IPS* to be Zero-Knowledge (*ZK*), V must get convinced of the assertion ($y \in L$) without acquiring information that enables the determination of x.

Another perspective, however, is to consider a relation $r \subseteq X \times Y$ satisfying $\langle x, y \rangle \in r \Rightarrow x \in L$. In particular – and making now the connection with the structure *PZK-CS* defined in the previous section – consider a relation r satisfying $\langle x, y \rangle \in r \Rightarrow \left(f^{(n,2)}(x) = y \wedge f^{(n,1)}(x) \in L\right)$, for some language L. In this case, the membership proof (or argument) will be about a secret element, instead of a public one. In what follows, in the perspective of *arguing* knowledge of a secret membership certificate, a special type of interactive argument system (*IAS*) is defined – (see [4] for a distinction between the concepts of proof and argument).

Definition 3.1 (*PZK-IAS-MLCA*) *A PZK-IAS for membership in language* $L_{(\alpha)} \subseteq \Delta^{\alpha}$ *with committed alphabet (PZK-IAS-MLCA-*$L_{(\alpha)}$*) is a (complete and sound) PZK-IAS in which, for any* $\vec{d} \in L_{(\alpha)}$*, after agreement of a PZK-CS and respective parameter n, P is able to generate a codification* $\vec{x} \leftarrow \left[\phi^{(n,1)}\left(\vec{d}\right)\right] : X_{(n)}{}^{\alpha}$ *of* $\vec{d}$*, publish its commitment* $\vec{y} = f^{(n,2)}(\vec{x})$ *("blob") and then convince V that it knows a secret decommitment* $\vec{x}$ *whose decodification* $\vec{d} = f^{(n,1)}(\vec{x})$ *is indeed an element of* $L_{(\alpha)}$*, without V gaining any relevant additional information (in a PZK sense).*

Note: The framework constructed after the examples of this section will demonstrate the properties of soundness and *PZK* and show how all *PZK-IAS-MLCA*s can be parallelized.

3.1 "Generalized nand" – using QHC of $+_2$ (sum mod 2)

Consider function $\overline{\wedge} : \mathbb{Z}_2{}^* \to \mathbb{Z}_2$ ($nand$) satisfying $\overline{\wedge}\left(\vec{d}\right) =_2 \left(\prod_{j \in \{0,\dots,|\vec{d}|\}} d_j\right) +_2 1$. The set obtained by concatenating each of the $2^{\alpha-1}$ distinct possible inputs in $\mathbb{Z}_2{}^{\alpha-1}$ with the respective output of function $\overline{\wedge}$ gives rise to language $\left[\overline{\wedge_{(\alpha)}}\right] = \left\{\vec{d} \in \mathbb{Z}_2{}^{\alpha} : d_{\alpha-1} = \overline{\wedge}\left(d_0 \cdot \ldots \cdot d_{\alpha-2}\right)\right\}$ (generalized α-Nand). Note that $\left[\overline{\wedge_{(\alpha)}}\right]$ can also be characterized as the set of elements $\vec{d}$ in $\mathbb{Z}_2{}^{\alpha}$ that satisfy $\sum_{j \in \{0,\dots,\alpha-2\}} \left[d_j +_2 d_{\alpha-1} =_2 1\right]^? + (\alpha-2) \times \left[d_{\alpha-1} =_2 1\right]^? \geq \alpha - 1$.

Let function $\vec{\mu} : \mathbb{Z}_2{}^{\alpha} \to \mathbb{Z}_2{}^{2\alpha-3}$ be defined by $\mu_j\left(\vec{d}\right) = d_j +_2 d_{\alpha-1}$ if $0 \leq j \leq \alpha-2$ and $\mu_j\left(\vec{d}\right) = d_{\alpha-1}$ if $\alpha-1 \leq j \leq 2\alpha-3$. The equivalence $\left[\vec{d} \in \left[\overline{\wedge_{(\alpha)}}\right]\right]^? \equiv \left[1 \in^{\alpha-1} \mu\left(\vec{d}\right)\right]^?$ of predicates, with $i \in^j \vec{d}$ standing for $\#\left(\{d_k : d_k = i\}\right) \geq j$, imply that $\vec{d}$ in $\mathbb{Z}_2{}^{\alpha}$ is an element of $\left[\overline{\wedge_{(\alpha)}}\right]$ if and only if there are at least $\alpha - 1$ components of $\mu\left(\vec{d}\right)$ with value 1.

The table at the side sketchs this property for the case $\alpha = 3$, with $\vec{\mu}\left(\vec{d}\right) = \langle d_0 +_2 d_2, d_1 +_2 d_2, d_2 \rangle$. $\vec{d}$ in $\mathbb{Z}_2{}^3$ is in $\left[\overline{\wedge_{(3)}}\right]$ if and only if there are at least two components of $\mu\left(\vec{d}\right)$ with value 1. Considering again a generic integer α, assume that a *PZK-CS* allowing QHC of $+_2$ is already initialized with public key parameter n, and that P has published an initial commitment $\vec{y} \in Y_{(n)}{}^{\alpha}$, for which it knows a secret decommitment $\vec{x} \in X_{(n)}{}^{\alpha}$ satisfying $\vec{y} = f^{(n,2)}\left(\vec{x}\right)$ and $f^{(n,1)}\left(\vec{x}\right) \in \left[\overline{\wedge_{(\alpha)}}\right]$.

$\vec{d} \in \Delta^3$	$\mu\left(\vec{d}\right)$	$\left[\vec{d} \in \left[\overline{\wedge_{(3)}}\right]\right]^?$
000	000	
001	111	True
010	010	
011	101	True
100	100	
101	011	True
110	110	True
111	001	

Procedure In order to reduce the probability of sucessful cheating by P to a value negligibly superior to 2^{-s}, s rounds (sequentially or in parallel) of the following steps are performed:

Witness Let $\vec{\nu} : \mathbb{Z}_2{}^{\alpha} \times \mathbb{Z}_2{}^{2\alpha-3} \to \mathbb{Z}_2{}^{2\alpha-3}$ satisfy $\vec{\nu}\left(\vec{d}, \vec{d'}\right) = \vec{\mu}\left(\vec{d}\right) +_2 \left(\vec{d'}\right)$. Let $\Pi(m)$ stand for the set of permutations of $\langle 0, \dots, m-1 \rangle$, for any $m \in \mathbb{N}_1$. P selects permutation $\pi \leftarrow \left[U\left(\Pi\left(2\alpha-3\right)\right)\right]$, determines functions $\vec{\diamondsuit} = \pi\left(\vec{\nu}\right)$ and $\vec{\xi} = q^{(n)}\left(\vec{\diamondsuit}\right)$, selects codification $\overrightarrow{x'} \leftarrow \left[\phi^{(n,1)}\left(0^{2\alpha-3}\right)\right]$, calculates $\overrightarrow{x''} = \vec{\xi}\left(\vec{x}, \overrightarrow{x'}\right)$ and sends witness $\overrightarrow{y''} = f^{(n,2)}\left(\overrightarrow{x''}\right)$ to V.

Challenge V uniformly selects a challenge $e \in \{0,1\}$ and sends it to P.

Response If $e = 0$, P discloses $\vec{\xi}$ (univocally defined by permutation π) and codification $\overrightarrow{x'}$. If $e = 1$, P discloses $\alpha - 1$ distinct components of $\overrightarrow{x''}$ codifying value 1.

Verification If $e = 0$, V calculates $\overrightarrow{y'} = f^{(n,2)}\left(\overrightarrow{x'}\right)$ and $\vec{\psi} = f^{(n,2)}\left(\vec{\xi}\right)$ and verifies that $\vec{\psi}\left(\vec{y}, \overrightarrow{y'}\right) = \overrightarrow{y''}$, $f^{(n,1)}\left(\overrightarrow{x'}\right) = 0^{2\alpha-3}$ and $f^{(n,1)}\left(\overrightarrow{\xi'}\right) \in \Pi\left(2\alpha-3\right)$. If $e = 1$, V verifies, for the $\alpha - 1$ indexes j of disclosed components, that $f^{(n,2)}\left(x''_j\right) = y''_j$ and $f^{(n,1)}\left(x''_j\right) = 1$.

Comment In [3], using a specific CS that allows QHC of $+_2$, it's presented a method of "permuted truth tables" enabling a *PZK-IAS* for $\left[\overline{\wedge_{(3)}}\right]$ (nand). The direct generalization of that result for other languages has a complexity proportional to the size of the string of all elements of the language, giving for $\left[\overline{\wedge_{(\alpha)}}\right]$ an overall complexity in $O\left(\alpha \times 2^{\alpha}\right)$. The above protocol presents a complexity improvement, since it has complexity proportional to $2\alpha - 3$, which is in $O(\alpha)$.

3.2 "Multiplication modulo 3"

Consider function $\times_3 : \mathbb{Z}_3{}^* \to \mathbb{Z}_3$ satisfying $\times_3\left(\vec{d}\right) =_3 \Pi_{\left(j \in \left\{0,\dots,\left|\vec{d}\right|\right\}\right)} d_j\,(mod\ 3)$ (multiplication $mod\ 3$). Language $\left[\times_{(\alpha,3)}\right] = \left\{\vec{d} \in \mathbb{Z}_3{}^\alpha : d_{\alpha-1} = \times_3\left(d_0 \cdot \dots \cdot d_{\alpha-2}\right)\right\}$ is obtained by concatenating each of the $3^{\alpha-1}$ distinct possible inputs in $\mathbb{Z}_3{}^{\alpha-1}$ with the respective output of function $\times_3$. Consider, for $\alpha = 3$, auxiliary function $\vec{\mu} : \mathbb{Z}_3{}^3 \to \mathbb{Z}_3{}^5$ satisfying $\vec{\mu}\left(\vec{d}\right) = \langle 2d_1 +_3 2d_2, d_0 +_3 d_2, 2d_0 +_3 d_2, 1 +_3 2d_0, 2 +_3 2d_0 +_3 d_2\rangle$. The equivalence $\left[\vec{d} \in \left[\times_{(3,3)}\right]\right]^? \equiv \left[0 \in^2 \mu\left(\vec{d}\right) \wedge 1 \in^1 \mu\left(\vec{d}\right) \wedge 2 \in^1 \mu\left(\vec{d}\right)\right]^?$ of predicates (found after a computational search), immediately suggests the procedure of an *PZK-IAS-MLCA* , defined similarly to the one described above for $\left[\overline{\wedge_{(3)}}\right]$ – only differing in $\vec{\mu} : \mathbb{Z}_2{}^3 \to \mathbb{Z}_2{}^5$ (and the parameters dependent on $\vec{\mu}$). The associativity of $\times_3$ makes it easy to consider language $\left[\times_{(\alpha,3)}\right]$ with $\alpha > 3$.

3.3 "String Equality" (over $\mathbb{Z}_m$)

Consider language $\left[=_{(m,\alpha)}\right] = \left\{\overrightarrow{d^{(1)}} \cdot \overrightarrow{d^{(2)}} \in \mathbb{Z}_m{}^{2\alpha} : \overrightarrow{d^{(2)}} = \overrightarrow{d^{(2)}}\right\}$ of pairs of equal strings of length $\alpha \in \mathbb{N}_2$ over alphabet $\mathbb{Z}_m$. Let two strings $\overrightarrow{d^{(1)}}$ and $\overrightarrow{d^{(2)}}$, both in $\mathbb{Z}_m{}^\alpha$, be codified by $\overrightarrow{x^{(1)}}$ and $\overrightarrow{x^{(2)}}$ and committed by $\overrightarrow{y^{(1)}}$ and $\overrightarrow{y^{(2)}}$, respectively, using a *PZK-CS* that allows *QHC* of $+_m : \mathbb{Z}_m{}^* \to \mathbb{Z}_m$.

Aspect 1 Let $\overrightarrow{\mu^{(0)}} : \mathbb{Z}_2{}^{2\alpha} \to \mathbb{Z}_2{}^\alpha$ be defined by $\overrightarrow{\mu^{(0)}}\left(\overrightarrow{d\,(1)}, \overrightarrow{d\,(2)}\right) = \overrightarrow{d^{(1)}} +_m (m-1) \times_m \overrightarrow{d^{(2)}}$, with $(m-1)\times_m\overrightarrow{d^{(2)}}$ calculated as a sum of $m-1$ equal terms. Using *QHC* of $+_m$, it's possible to calculate functions $\overrightarrow{\xi^{(0)}} = q^{(n)}\left(\overrightarrow{\mu^{(0)}}\right)$ and $\overrightarrow{\psi^{(0)}} = f^{(n,2)}\left(\overrightarrow{\xi^{(0)}}\right)$, codification $\overrightarrow{x^{(0)}} = \overrightarrow{\xi^{(0)}}\left(\overrightarrow{x^{(1)}}, \overrightarrow{x^{(2)}}\right)$ and blob $\overrightarrow{y^{(0)}} = \overrightarrow{\psi^{(0)}}\left(\overrightarrow{y^{(1)}}, \overrightarrow{y^{(2)}}\right)$ satisfying $f^{(n,1)}\left(\overrightarrow{x^{(0)}}\right) = \overrightarrow{d^{(0)}}$ and $f^{(n,2)}\left(\overrightarrow{x^{(0)}}\right) = \overrightarrow{y^{(0)}}$. The equivalence $\left[\overrightarrow{d^{(1)}} = \overrightarrow{d^{(2)}}\right]^? \equiv \left[\overrightarrow{d^{(0)}} = 0^\alpha\right]$, suggests focusing simply on a *PZK-IAS-MLCA* for language $\{0^\alpha\}$.

Aspect 2 Consider function $\mu^{(\vec{c})} : \mathbb{Z}_2{}^\alpha \to \mathbb{Z}_2$, defined by $\mu^{(\vec{c})}\left(d^{(0)}\right) = \sum_{j \in \mathbb{Z}_\alpha} c_j \times_m d^{(0)}{}_j$, with $c_j \times_m d^{(0)}{}_j$ being calculated as a sum of c_j equal terms, for every $\vec{c} \in \mathbb{Z}_m{}^\alpha$. Let $d^{(\vec{c})} \equiv \mu^{(\vec{c})}\left(\overrightarrow{d^{(0)}}\right)$. Note that $\#\left\{\vec{c} \in \mathbb{Z}_m{}^\alpha : d^{(\vec{c})} \neq_m 0\right\}$ equals 0 if $\overrightarrow{d^{(0)}} = 0^\alpha$ and is no less than $m^\alpha/2$ if $\overrightarrow{d^{(0)}} \neq 0^\alpha$. So, for each selected $\vec{c} \leftarrow [U(\mathbb{Z}_m{}^\alpha)]$, if $d^{(0)} =_m 0^\alpha$ then $d^{(\vec{c})}$ is 0 with certainty, while if $d^{(0)} \neq_m 0^\alpha$ then $d^{(\vec{c})}$ differs from 0 with probability of at least 1/2. Again, *QHC* of $+_m$ allows the calculation of functions $\xi^{(\vec{c})} = q^{(n)}\left(\mu^{(\vec{c})}\right)$ and $\psi^{(\vec{c})} = f^{(n,2)}\left(\xi^{(\vec{c})}\right)$, codification $x^{(\vec{c})} = \xi^{(\vec{c})}\left(x^{(0)}\right)$ and blob $y^{(\vec{c})} = \psi^{(\vec{c})}\left(y^{(0)}\right)$, together satisfying $f^{(n,1)}\left(x^{(\vec{c})}\right) = d^{(\vec{c})}$ and $f^{(n,2)}\left(x^{(\vec{c})}\right) = y^{(\vec{c})}$.

Procedure For each $\vec{c} \leftarrow [U(\mathbb{Z}_m{}^\alpha)]$, a single-round *PZK-IAS-MLCA* arguing that $d^{(\vec{c})}$ equals 0 gives some confidence about $\overrightarrow{d^{(1)}} \cdot \overrightarrow{d^{(2)}} \in \left[=_{(m,\alpha)}\right]$. The probability of sucess for a dishonest prover, knowing decommitments $x^{(1)}$ and $x^{(2)}$ of $y^{(1)}$ and $y^{(2)}$ but correponding to elements $d^{(1)} \neq d^{(2)}$, is utmost $3/4 = 1/2 + 1/2 \times 1/2$ (1/2 probability that $\vec{c}$ satisfies $d^{(\vec{c})} = 0$ plus, if $d^{(\vec{c})} \neq 0$ – with 1/2 probability – being able with 1/2 probability to respond to a dishonestly produced witness – see how in the simulator description in the next section). To reduce to 2^{-s} the overall probability, $s \times \log_{4/3} 2$ single-round *IAS*s are executed, each with a new selection $\vec{c} \leftarrow [U(\mathbb{Z}_m{}^\alpha)]$. In the same way that for language $\left[\overline{\wedge_{(\alpha)}}\right]$ was argued that in a set of $2\alpha - 3$ elements there were at least $\alpha - 1$ with value 1, in this case it's argued that in a set of just one element there's one with value 0.

Comment Since in each round, the witness and response steps demand the communication of a single blob and decommitment, respectively, the overall communication complexity of the *PZK-IAS-MLCA* is in $O(s)$, instead of $O(s \times \alpha)$ as in the example mentioned also in [3] for string equality, that runs every index $j \in \mathbb{Z}_\alpha$ and showing that $d_j^{(1)} \neq d_j^{(1)}$.

3.4 "String Inequality" (over $\mathbb{Z}_m$)

The language of pairs of different strings in ${\mathbb{Z}_m}^\alpha$ is easily defined as $\left[\neq_{(m,\alpha)}\right] = {\mathbb{Z}_m}^{2\alpha} \backslash \left[=_{(m,\alpha)}\right]$. A reasoning similar to the one made initially for "string equality" allows reducing the problem to arguing that vector $\overrightarrow{d^{(0)}} = \overrightarrow{d^{(1)}} +_m (m-1) \times_m \overrightarrow{d^{(2)}}$ is different from 0^α. This case may seem simpler than the one for "string equality", but caution be taken. Let index $l \in \mathbb{Z}_\alpha$ satisfy ${d^{(0)}}_l \neq 0$.

Caution 1 Index j (there may be several satisfying the above condition) shouldn't be disclosed. Let, as for "string equality", $d^{(\vec{c})} \equiv \sum_{j \in \mathbb{Z}_\alpha} c_j \times_m {d^{(0)}}_j$, for every $c_j \in {\mathbb{Z}_m}^\alpha$. Let also, for every $\overrightarrow{c(0)} \in {\mathbb{Z}_m}^\alpha$, vector $\overrightarrow{c(1)} \in {\mathbb{Z}_m}^\alpha$ be defined by $(\forall j \in \mathbb{Z}_\alpha : j \neq l) \left(c(1)_j = m - c(0)_j\right)$ and $c(1)_l = (m - c(0)_l + 1) \, (mod \, m)$. Note that at least one of $d^{\left(\overrightarrow{c(0)}\right)}$ or $d^{\left(\overrightarrow{c(1)}\right)}$ differs from 0. The problem thus reduces to arguing that at least one of two components is different from 0.

Caution 2 Let index $i \in \mathbb{Z}_2$ satisfy $d^{\left(\overrightarrow{c(i)}\right)} \neq 0$. The case with $m = 2$ is simple because it resumes to show that a component with value 1 exists. For $m > 2$, however, the value of $d^{\left(\overrightarrow{c(i)}\right)}$ must not be disclosed (or it would give information besides $\overrightarrow{d^{(1)}} \cdot \overrightarrow{d^{(2)}} \in \left[\neq_{(\alpha,m)}\right]$). Let $\mathbb{Z}_m^*$ be the set of elements of $\mathbb{Z}_m$ coprime with m. When m is prime[5], the equivalence $\left[d^{\left(\overrightarrow{c(i)}\right)} \neq_m 0\right]^? \Leftrightarrow \left[\mathbb{Z}_m^* = \left\{k \times_m d^{\left(\overrightarrow{c(i)}\right)} : k \in \mathbb{Z}_m^*\right\}\right]^?$ suggests a procedure.

Procedure s rounds are executed as follows:

Witness P selects $\overrightarrow{c(0)} \leftarrow \left[U\left({\mathbb{Z}_m}^\alpha\right)\right]$ and $k \leftarrow \left[U\left(\mathbb{Z}_m^*\right)\right]$, determines $\overrightarrow{c(1)}$ (from $\overrightarrow{c(0)}$ as defined above) and $\overrightarrow{\diamondsuit} : {\mathbb{Z}_m}^{2\alpha} \to {\mathbb{Z}_m}^2$ satisfying $\diamondsuit_i \left(\overrightarrow{d^{(1)}} \cdot \overrightarrow{d^{(2)}}, \overrightarrow{d'}\right) = {d'}_i +_m k \times_m d^{\left(\overrightarrow{c(i)}\right)}$ with $i \in \mathbb{Z}_2$ (value $k \times_m d^{\left(\overrightarrow{c(i)}\right)}$ is in fact calculated as the sum of k equal terms $d^{\left(\overrightarrow{c(i)}\right)}$), calculates $\vec{\xi} = q^{(n)}\left(\overrightarrow{\diamondsuit}\right)$, determines $\overrightarrow{d'} = 0^2$, computes $\overrightarrow{x'} \leftarrow \left[\phi^{(n,1)}\left(\overrightarrow{d'}\right)\right]$, calculates $\overrightarrow{x''} = \xi\left(\vec{x}, \overrightarrow{x'}\right)$ and sends witness $\overrightarrow{y''} = f^{(n,2)}\left(\overrightarrow{x''}\right)$ to V.

Challenge V selects a challenge $e \leftarrow [U(\{0,1,2\})]$ and sends it to P.

Response If $e \in \{0,1\}$, P discloses ξ_e (univocally defined by $\overrightarrow{c(e)}$ and k) and discloses ${x'}_e$. If $e = 2$, P discloses ${x''}_i$ for some i such that $d^{\left(\overrightarrow{c(i)}\right)} \neq 0$.

Verification If $e \in \{0,1\}$, V verifies that $f^{(n,1)}(x') = 0$, calculates $\diamondsuit_e = f^{(n,1)}(\xi_e)$ (univocally determining $\overrightarrow{c(e)}$ and k) and verifies that $\overrightarrow{c(e)} \in {\mathbb{Z}_m}^\alpha$ and $k \in \mathbb{Z}_m^*$, calculates ${y'}_e = f^{(n,2)}({x'}_e)$ and $\psi_e = f^{(n,2)}(\xi_e)$ and verifies that $\psi_e\left(\vec{y}, \overrightarrow{y'}\right) = {y''}_e$. If $e = 2$, V verifies that $f^{(n,2)}({x''}_i) = {y''}_i$ and $f^{(n,1)}({x''}_i) \in \mathbb{Z}_m^*$.

Comment Responding simultaneously to $e = 0$ and $e = 1$ would disclose l, since it's the only index $j \in \mathbb{Z}_\alpha$ for which $c(0)_j + c(1)_j = 1$. Responding simultaneously to a challenge $e = i \in \{0,1\}$ and $e = 2$ would disclose $d^{\left(\overrightarrow{c(i)}\right)}$, because it's univocally defined by k, ${d'}_i$ and ${d''}_i = {d'}_i + k \times d^{\left(\overrightarrow{c(i)}\right)}$.

[5]The more complex case with m not prime easily derives from $\left[(\exists l \in \mathbb{Z}_\alpha)\,(\exists d' \in \{1, ..., m-1\})\,(d' +_m d_l = 0)\right]^?$.

3.5 "Graph 3-colorations"

In [5] was presented a first example of a *PZK-IPS* for *G3C* (3-colorable graphs) – a *NP*-Complete language – based on the sole assumption that secure encryption functions exist. For a soundness error probability of about $e^{-s} \approx 2.72^{-s}$ and a graph with nv vertices and ne edges, that protocol requires $ne \times s$ rounds, each requiring encryption of the color of all nv vertices, thus making an overall complexity of about $O(nv \times ne \times s)$. However, assuming *QHC* of $+_2$, the communication complexity can be reduced to $O((nv + ne) \times s)$ as explained ahead. Although this improvement could be obtained as a trivial consequence of a *PZK-IAS-MLCA* for *Nand*, by constructing a circuit of nand gates verifying the validity of 3-colorations, the following form seems to be worth mentioning for its elegant simplicity. Consider a graph $G = \langle \mathbb{Z}_n v, E \rangle$ with nv vertices and ne edges in ${\mathbb{Z}_{nv}}^2$ and a valid 3-coloration mapping $col : \mathbb{Z}_{nv} \to {\mathbb{Z}_2}^2 \setminus \{00\}$. Consider also that a *PZK-CS* allowing *QHC* of $+_2$ has been initialized with public-key parameter n and that P has published, for every vertex $i \in \mathbb{Z}_{nv}$, the commitment $\overrightarrow{y^{(i)}} \in {Y_{(n)}}^2$ of a secret codification $\overrightarrow{x^{(i)}} \in {X_{(n)}}^2$ of colour $\overrightarrow{d^{(i)}} = \overrightarrow{col}(i) \in {\mathbb{Z}_2}^2$. The equivalences of predicates $\left[\overrightarrow{d^{(i)}} \in \{01, 10, 11\}\right]^? \equiv \left[1 \in^1 \overrightarrow{d^{(i)}}\right]^?$ (for $i \in \mathbb{Z}_{nv}$) and $\left[\overrightarrow{d^{(i)}} \neq \overrightarrow{d^{(j)}}\right]^? \equiv \left[1 \in^1 \left(\overrightarrow{d^{(i)}} +_2 \overrightarrow{d^{(j)}}\right)\right]^?$ (for $\langle i, j \rangle \in E$), immediately suggest a very simple *PZK-IAS-MLCA* for *G3C*. Basically the protocol consists of $nv + ne$ protocols of arguing that a given pair in ${\mathbb{Z}_2}^2$ includes element 1.

4 A generalized framework for *PZK-IAS-MLCA*

Motivation and guidelines All the examples above present obvious similarities and differences among each other. It seems worthwhile the definition of a framework in which all the examples can be framed and each can be defined as a specific parametrization. In particular, this will allow that a formal description of each procedure and a demonstration of properties of soundness and *PZK* occur simply as a consequence of the general structure, instead of having to make exaustive description and proofs for every example. Note that the *PZK-CS* structure already differentiates three spaces, associated with sets Δ (alphabet), $X_{(n)}$ (codifications) and $Y_{(n)}$ (commitments), related by means of homomorphisms $f^{(n,1)}$ and $f^{(n,2)}$. Note also that in each *PZK-IAS-MLCA* three types of elements may be disclosed in responses, namely ξ, x' and x'' related to space $X_{(n)}$, each corresponding, respectively, to $\diamondsuit$, d' and d'' related to space Δ and, respectively, to ψ, y' and y'' related to space $Y_{(n)}$. The framework will among other things generalize the functions for which *QHC* may be available and will consider integrated responses disclosing components of every type. General requirements will be stated in order that properties of soundness and *PZK* (in sequential and parallel versions of execution) are a consequence of the structural definition. For each case it will then only be necessary to make an approriate parametrization and select an adequate *PZK-CS*.

4.1 Parameters introduction

Abbreviation Let $\nabla \equiv F\left(\Delta^\alpha \times \Delta^{\alpha'}, \Delta^{\alpha''}\right)$ and $\nabla' \equiv F\left(\Delta^\alpha \times \Delta^{\alpha'} \times \Delta^{\alpha''}, \Delta^{\alpha''}\right)$, with α, α' and α'' in $\mathbb{N}_1$.

Witness construction Consider a secret $\overrightarrow{d} \in \Delta^\alpha$. Generators $\overrightarrow{\diamondsuit} \in \nabla^{\alpha''}$ and $\overrightarrow{d'} \in \Delta^{\alpha'}$ are selected using *PF* $\phi^{(W)} \in \Re\left(\Delta^\alpha, \nabla^{\alpha''} \times \Delta^{\alpha'}\right)$, so that witness $\overrightarrow{d''} = \overrightarrow{\diamondsuit}\left(\overrightarrow{d}, \overrightarrow{d'}\right)$ is calculated. By definition let $W_{\left(\overrightarrow{d}\right)} \equiv \left\{\left\langle \overrightarrow{\diamondsuit}, \overrightarrow{d'} \right\rangle : {\phi^{(W)}}_{\overrightarrow{d}}\left(\overrightarrow{\diamondsuit}, \overrightarrow{d'}\right) > 0\right\}$ and $W \equiv \bigcup_{\overrightarrow{d} \in L_{(\alpha)}} W\left(\overrightarrow{d}\right)$.

Challenges Let $\mathbb{Z}_\varepsilon$ be the set of possible challenges, Tr the set of possible (partial or complete) transcripts and s the number of rounds of an execution of a *PZK-IAS-MLCA* . $ch = U(\mathbb{Z}_\varepsilon)$ is the *PD* used for honest challenge selection, while $ch \in \Re(Tr, \mathbb{Z}_\varepsilon)$ or $ch \in \Re\left(Tr, {\mathbb{Z}_\varepsilon}^s\right)$ are the *PD*s used for dishonest selection in the sequential or parallel versions, respectively.

Equation solving Let function $\chi : \nabla \to P(\nabla')$ be such that $\overrightarrow{\Diamond'} \in \chi\left(\overrightarrow{\Diamond}\right)$ implies $\overrightarrow{d''} = \overrightarrow{\Diamond}\left(\overrightarrow{d}, \overrightarrow{d'}\right) \Leftrightarrow \overrightarrow{d''} = \overrightarrow{\Diamond'}\left(\overrightarrow{d}, \overrightarrow{d'}, \overrightarrow{d''}\right)$ (Note that $\overrightarrow{\Diamond'}$ must enables explicity evaluation of $\overrightarrow{d''}$.

Responses Responses may disclose components of three types: $\vec{\Diamond}$ and $\overrightarrow{d'}$ (witness generators) and $\overrightarrow{d''}$ (witness). Consider function ρ defined by $\rho\left(\left\langle\overrightarrow{\Diamond'}, \overrightarrow{d'}, \overrightarrow{d''}\right\rangle, \langle J_0, J_1, J_2\rangle\right) = \langle r_0, r_1, r_2\rangle$, with $r_0 \equiv \left\{\left\langle j_0, \Diamond'_{j_0}\right\rangle : j_0 \in J_{(0)}\right\}$, $r_1 \equiv \left\{\left\langle j_1, d'_{j_1}\right\rangle : j_1 \in J_{(1)}\right\}$ and $r_2 \equiv \left\{\left\langle j_2, d''_{j_2}\right\rangle : j_2 \in J_{(2)}\right\}$. R is defined as the set of triplets $\langle r_0, r_1, r_2\rangle$, with $\overrightarrow{\Diamond'}, \overrightarrow{d'}, \overrightarrow{d''}$ running over $\nabla'^{\alpha''}, \Delta^{\alpha'}, \Delta^{\alpha''}$ and with $J_{(0)}, J_{(1)}, J_{(2)}$ running over all subsets of $\mathbb{Z}_{\alpha''}, \mathbb{Z}_{\alpha'}, \mathbb{Z}_{\alpha''}$, respectively. After determination of a witness $d'' = \vec{\Diamond}\left(\vec{d}, \overrightarrow{d'}, \overrightarrow{d''}\right)$ and a challenge $e \in \mathbb{Z}_\varepsilon$, the response (in high-level) is selected as $\overrightarrow{r} \leftarrow \left[\phi^{(R)}\left(\overrightarrow{d}, \vec{\Diamond}, \overrightarrow{d'}, e\right)\right]$ using some PF $\phi^{(R)} \in \Re\left(\Delta^\alpha \times \nabla^{\alpha''} \times \Delta^{\alpha'} \times \mathbb{Z}_\varepsilon, R\right)$. By definition, let $R_{(e)} \equiv \bigcup_{\vec{d}, \vec{\Diamond}, \overrightarrow{d'} \in \Delta^\alpha, \nabla^{\alpha''}, \Delta^{\alpha'}} \left\{\overrightarrow{r} \in R : \phi^{(R)}{}_{\overrightarrow{d}, \vec{\Diamond}, \overrightarrow{d'}, e}\left(\overrightarrow{r}\right) > 0\right\}$.

4.2 Execution procedure

***PZK-CS* initialization** Fixed a security parameter k, V selects a pair $\langle n, t\rangle \leftarrow \left[\phi^{(0)}(k)\right]$ of public and private key parameters and, by means of some *IPS*, proves to P that n belongs to N (i.e. that the *CS* with value n is indeed a *PZK-CS*). For **parallel versions** of the *PZK-IAS-MLCA*, V also proves to have the ability to invert the *PZK-CS*.[6] Remember that n univocally defines $\phi^{(n,1)}$, $f^{(n,1)}$, $f^{(n,2)}$ and $q^{(n)}$.

Initial commitment P computes codification $\vec{x} \leftarrow \left[\phi^{(n,1)}\left(\vec{d}\right)\right]$ and sends $\vec{y} = f^{(n,2)}(\vec{x})$ to V.

Iterations To minimize the probability of successful cheating by P, to a value negligibly superior to $\#(\Delta)^{-s}$, s iterations of type $\langle\vec{w}, e, \vec{r}\rangle$ ("witness $\to$ challenge $\to$ response") are executed, sequentially or in parallel, as follows.

Witness P computes $\left\langle\vec{\Diamond}, \overrightarrow{d'}\right\rangle \leftarrow \left[\phi^{(W)}\left(\vec{d}\right)\right]$, $\vec{\xi} = q^{(n)}\left(\vec{\Diamond}\right)$, $\overrightarrow{x'} \leftarrow \left[\phi^{(n,1)}\left(\overrightarrow{d'}\right)\right]$ and $\overrightarrow{x''} = \vec{\xi}\left(\vec{x}, \overrightarrow{x'}\right)$ and sends witness $\overrightarrow{y''} = f^{(n,2)}\left(\overrightarrow{x''}\right)$ to V. Let $\overrightarrow{d''} \equiv \vec{\Diamond}\left(\vec{d}, \overrightarrow{d'}\right)$.

Challenge V Computes challenge $e \leftarrow [U(\mathbb{Z}_\varepsilon)]$ (uniform selection) and sends it to P.

Response P computes $\overrightarrow{r} \leftarrow \left[\phi^{(R)}\left(\overrightarrow{d}, \vec{\Diamond}, \overrightarrow{d'}, e\right)\right]$, with $r_0 \equiv \left\{\left\langle j_0, \Diamond'_{j_0}\right\rangle : j_0 \in J_{(0)}\right\}$, $r_1 \equiv \left\{\langle j_1, d'_{j_1}\rangle : j_1 \in J_{(1)}\right\}$ and $r_2 \equiv \left\{\langle j_2, d''_{j_2}\rangle : j_2 \in J_{(2)}\right\}$, with $\vec{J} \in \times_{i \in \mathbb{Z}_3} P\left(\mathbb{Z}_{\alpha(i)}\right)$ and $\overrightarrow{\Diamond'} \in \chi\left(\vec{\Diamond}\right)$. Computes $\overrightarrow{\xi'} = q^{(n)}\left(\overrightarrow{\Diamond'}\right)$ and sends response $\overrightarrow{r'} = \langle r'_0, r'_1, r'_2\rangle$ to V, with $r'_0 = \left\{\left\langle j_0, \Diamond'_{j_0}, \xi'_{j_0}\right\rangle : j_0 \in J_0\right\}$, $r'_1 = \left\{\langle j_1, d'_{j_1}, x'_{j_1}\rangle : j_1 \in J_1\right\}$ and $r'_2 = \left\{\langle j_2, d''_{j_2}, x''_{j_2}\rangle : j_2 \in J_2\right\}$.

Verification V verifies that $\vec{r} \in^? R_{(e)}$ (with $R_{(e)}$ defined specifically for each *PZK-IAS-MLCA*). For each $j_1 \in J_{(1)}$ verifies that $f^{(n,1)}(x'_{j_1}) =^? d'_{j_1}$ and determines $y'_{j_1} = f^{(n,2)}(x'_{j_1})$. For each $j_2 \in J_{(2)}$ verifies that $f^{(n,1)}(x''_{j_2}) =^? d''_{j_2}$ and $f^{(n,2)}(x''_{j_2}) =^? y''_{j_2}$. For each $j_0 \in J_{(0)}$ verifies that $f^{(n,1)}\left(\xi'_{j_0}\right) =^? \Diamond'_{j_0}$, determines $\psi'_{j_0} = f^{(n,2)}\left(\xi'_{j_0}\right)$ and verifies that $\psi'_{j_0}\left(\overrightarrow{y}, \overrightarrow{y'}, \overrightarrow{y''}\right) =^? y''_{j_0}$ (possible because $(\forall j_0 \in J_0)\left(\gamma_1\left(\Diamond'_{j_0}\right) \subseteq J_1\right)$ will be required).

4.3 Conditions on parameters

Some requirements will be more easily understood after a familiarization with the simulators (sequential and parallel) defined in the two following subsections.

[6]See in appendix B how this can be easily accomplished.

Completeness The protocol must be accepted if P and V act honestly. Consider a response $\vec{r} = \rho\left(\left\langle \overrightarrow{\diamondsuit'}, \overrightarrow{d'}, \overrightarrow{d''} \right\rangle, \vec{J}\right) \in R_{(e)}$ given after a challenge $e \in \mathbb{Z}_\varepsilon$ to a witness $\overrightarrow{d''}$ generated using pair $\left\langle \overrightarrow{\diamondsuit}, \overrightarrow{d'} \right\rangle \in W$ of generators. Requirement: $\overrightarrow{\diamondsuit'}$ is in $\chi\left(\overrightarrow{\diamondsuit}\right)$; the *PZK-CS* allows *QHC* of $\overrightarrow{\diamondsuit}$ and $\overrightarrow{\diamondsuit'}$; and $J_{(1)} = \bigcup_{j_0 \in J_{(0)}} \gamma_1\left(\diamondsuit'_{j_0}\right)$ (component d''_{j_0} can be verified using $\diamondsuit'_{j_0}$).

***PZK* in witnesses** It's required that every component y''_{j_2} is probabilistically independent of blob $\vec{y}$ and of every other blob component $y''_{j'_2}$ with $j'_2 \neq j_2$. General requirement: For every function $\overrightarrow{\xi} = q^{(n)}\left(\overrightarrow{\diamondsuit}\right)$ for which $\vec{\diamondsuit}$ is susceptible of being a witness generator. The conditional probability of obtaining a witness $\overrightarrow{y''} = f^{(n,2)}\left(\vec{\xi}\left(\vec{x}, \overrightarrow{x'}\right)\right)$, knowing that $\overrightarrow{x'} \leftarrow \left[\phi^{(n,1)}\left(\overrightarrow{d'}\right)\right]$, is equal to $\phi^{(PZK)}\left(\overrightarrow{y''}\right)$ (remember *PF* $\phi^{(PZK)}$ from the definition of *PZK-CS*). A sufficient requirement: The set of α'' equations $\overrightarrow{x''} =^? \overrightarrow{\xi}\left(\overrightarrow{x}, \overrightarrow{x'}\right)$ (with all components of $\vec{x}$, $\overrightarrow{x'}$ and $\overrightarrow{x''}$ as incognits) can't be solved to give a relation relating only components of $\overrightarrow{x}$ and $\overrightarrow{x''}$. Notes: The quasigroupicness of $\overrightarrow{\xi}$ makes it possible to solve the system of linear equations in order of components x'_{j_1}, with $j_1 \in \gamma_1\left(\overrightarrow{\xi}\right)$. In this case linear independence implies probabilistic independence.

Soundness in responses In this framework, soundness is based on the assumption of infeasibility to invert functions $f^{(n,2)}$ (as defined along with *PZK-CS*). Consider initial secret $\overrightarrow{x}$ and generators $\overrightarrow{x'}$ and $\overrightarrow{\xi}$ of witness decommitment $\overrightarrow{x''} = \overrightarrow{\xi}\left(\overrightarrow{x}, \overrightarrow{x'}\right)$. A response to some challenge $e \in \mathbb{Z}_\varepsilon$ discloses some components ξ'_{j_0}, x'_{j_1} and x''_{j_2}, for some $\overrightarrow{\xi'} \in \chi\left(\overrightarrow{\xi}\right)$. This response can be summarized in a set of equations $\left\{x''_{j_0} =^? \xi_{j_0}\left(\overrightarrow{x}, \overrightarrow{x'}, \overrightarrow{x''}\right) : j_0 \in J_{(0)}\right\}$, where each x'_{j_1} for $j_1 \in J_{(1)}$ and x''_{j_2} for $j_2 \in J_{(2)}$ is substituted by the disclosed value (and is thus no longer an incognit). Requirement: Whatever two valid responses to two different challenges but the same witness, the set of equations obtained by joining the two responses can be solved in order of some function $\xi'(\vec{x})$ relating only components of $\overrightarrow{x}$ (note that this is the inverse of some $\psi'(\vec{y})$).

Comment Although descriptions about ξ require a specific *PZK-CS* to be defined, its quasigroupicness will often enable the verifications related with equation solving to be made just by looking at the indexes $\gamma(\xi)$ of relevant components of imput. In specific cases where $\overrightarrow{\diamondsuit}$ has the same components of relevant input than $\overrightarrow{\xi}$, then all the requirements can be checked in space Δ.

Sequential simulator – *PZK* in responses Requirement: Exists a computable *PF* $\phi^{(SS-R)} \in \Re(\mathbb{Z}_\varepsilon, R)$ satisfying, for every $\vec{d} \in L_{(\alpha)}$, $(\forall e, \vec{r} \in \mathbb{Z}_\varepsilon, R)$ $\left(\phi^{(SS-R)}{}_e(\vec{r}) = \sum_{\left\langle \vec{\diamondsuit}, \overrightarrow{d'} \right\rangle \in W_{(\vec{d})}} \phi^{(W)}{}_{\vec{d}}\left(\overrightarrow{\diamondsuit}, \overrightarrow{d'}\right) \times \phi^{(R)}{}_{\vec{d}, \vec{\diamondsuit}, \overrightarrow{d'}, e}(\vec{r})\right)$. Note: This guarantees the existence of a sequential simulator without need of an element $\vec{d}$.

Parallel simulator – *PZK* in witnesses and responses Requirement: Exists computable *PD* $g^{(PS-W)} \in PD(W)$ and a computable *PF* $\phi^{(PS-R)} \in \Re(W \times \mathbb{Z}_e, R)$ satisfying $(\forall e, \vec{r} \in \mathbb{Z}_\varepsilon, R)$ $\left(\sum_{\left\langle \vec{\diamondsuit}, \overrightarrow{d'} \right\rangle \in W} g^{(PS-W)}\left(\left\langle \vec{\diamondsuit}, \overrightarrow{d'} \right\rangle\right) \times \phi^{(PS-R)}{}_{\left\langle \vec{\diamondsuit}, \overrightarrow{d'} \right\rangle, e}(\vec{r})\right) = \phi^{(SS-R)}{}_e(\vec{r})$. Also, it's required that $\left(\forall e, \vec{r} \in \mathbb{Z}_\varepsilon, R_{(e)}\right)\left(J_0 \bigcap J_2 = \emptyset\right)$ Note: After a selection of witnesses (using $g^{(PS-W)}$), responses can be selected with the same probability as in real executions. The first conditions guarantee the existence of a parallel simmulator without the need of an element $\vec{d}$. The last condition, relating sets J_0 and J_2 of disclosed components, eliminates the necessity of inverting functions $\overrightarrow{\xi}$.

Comment After these parameter's conditioning, the parameterization of specific *PZK-IAS-MLCA*s depend on the definition of *PF*s $\phi^{(W)}$ and $\phi^{(R)}$, in conjunction with all parameters that relate them with the language $L_{(\alpha)}$ in question.

4.4 Sequential simulator

Sequential versions are specially adequated when P doens't trust in V's ability to invert $f^{(n,2)}$. Consider the simulation of an execution where the possibly dishonest behaviour of V in the selection of challenges is defined by $ch \in \Re\left(Tr, \mathbb{Z}_\varepsilon\right)$, with Tr being the set of possible transcripts.

Initialization The *PZK-CS* initialization is trivial because V is the prover in that stage. The initial *blob* is selected as $\vec{y} \leftarrow \left[\left({\phi^{(PZK)}}_n\right)^{\alpha''}\right]$. The transcript initializes as $tr = \langle\langle k, n, s, ..., \vec{y}\rangle\rangle$, with "..." being information about the validation of n.

Iterations s iterations of the following steps are executed **sequentially**:

Challenge anticipation Challenge is guessed as $e \leftarrow \left[U\left(\mathbb{Z}_\varepsilon\right)\right]$ (uniform selection).

Simultaneous computation of witness and response Selects $\vec{r} \leftarrow \left[\phi^{(SS-R)}\left(e\right)\right]$. Let $\vec{r} \equiv \rho\left(\left\langle\overrightarrow{\Diamond'}, \overrightarrow{d'}, \overrightarrow{d''}\right\rangle, \vec{J}\right)$. For $j_1 \in J_{(1)}$: computes ${x'}_{j_1} \leftarrow \left[\phi^{(n,1)}\left({d'}_{j_1}\right)\right]$ and ${y'}_{j_1} = f^{(n,2)}\left({x'}_{j_1}\right)$. For $j_1 \in \mathbb{Z}_{\alpha'} \backslash J_{(1)}$: computes ${y'}_{j_1} \leftarrow \left[{\phi^{(PZK)}}_n\right]$. For $j_2 \in J_{(2)}$: computes ${x''}_{j_2} \leftarrow \left[\phi^{(n,1)}\left({d''}_{j_2}\right)\right]$ and ${y''}_{j_2} = f^{(n,2)}\left({x''}_{j_2}\right)$. For $j_0 \in J_{(0)}$: determines ${\xi'}_{j_0} = q^{(n)}\left(\Diamond'_{j_0}\right)$, $\psi'_{j_0} = f^{(n,2)}\left({\xi'}_{j_0}\right)$ and[7] $y''_{j_0} = {\psi'}_{j_0}\left(\vec{y}, \vec{y'}, \vec{y''}\right)$. Sets $\vec{r'} = \langle {r'}_0, {r'}_1, {r'}_2\rangle$, with ${r'}_0 = \left\{\left\langle j_0, {\Diamond'}_{j_0}, {\xi'}_{j_0}\right\rangle : j_0 \in J_{(0)}\right\}$, ${r'}_1 = \left\{\langle j_1, {d'}_{j_1}, {x'}_{j_1}\rangle : j_1 \in J_{(1)}\right\}$ and ${r'}_2 = \left\{\langle j_2, {d'}_{j_2}, {x'}_{j_2}\rangle : j_2 \in J_{(2)}\right\}$.

Challenge verification and transcript update Determines $e' \leftarrow \left[ch\left(\langle tr, \vec{y}\rangle\right)\right]$. If $e = e'$ the transcript is updated as $tr \rightarrow \left\langle tr, \left\langle \vec{y}, e, \vec{r'}\right\rangle\right\rangle$ and the number of rounds incremented by one. If $e \neq e'$ the all round repeats from the beginning. Note that the expected number of rounds is about $\varepsilon \times s$, because ε^{-1} is the probability of guessing a challenge.

4.5 Parallel simulator

Consider the simulation of a parallel version in which V is able to invert the $f^{(n,2)}$ and has a possibly dishonest behaviour, in the selection of challenges, defined by $ch \in \Re\left(Tr, {\mathbb{Z}_\varepsilon}^s\right)$.

Initialization Besides an initialization of the *PZK-CS* similar to the sequential simulator case, it's also simulated the proof that V makes about the ability to invert the $f^{(n,2)}$. The initial *blob* is also selected as $\vec{y} \leftarrow \left[\left({\phi^{(PZK)}}_n\right)^{\alpha''}\right]$.

Witnesses The anticipated guessing of challenges isn't feasible now because of the exponential cardinality (in s) of the challenge space ${\mathbb{Z}_\varepsilon}^s$. For $l \in s$: selects generators $\left\langle\overrightarrow{\Diamond^{(l)}}, \overrightarrow{d'^{(l)}}\right\rangle \leftarrow \left[g^{(PS-W)}\right]$, determines function $\overrightarrow{\psi^{(l)}} = f^{(n,2)}\left(\overrightarrow{\xi^{(l)}} \leftarrow \left[\phi^{(n,1)}\left(\overrightarrow{\Diamond^{(l)}}\right)\right]\right)$, selects $\overrightarrow{y'^{(l)}} \leftarrow \left[\left({\phi^{(PZK)}}_n\right)^{\alpha'}\right]$ and determines witness $\overrightarrow{y''^{(l)}} = \overrightarrow{\psi^{(l)}}\left(\vec{y}, \overrightarrow{y'^{(l)}}, \overrightarrow{y''^{(l)}}\right)$.

Challenge Transcript is initiallized as $tr = \left\langle\left\langle k, n, ..., s, \vec{y}, \left\langle\overrightarrow{y''^{(l)}} : l \in \mathbb{Z}_s\right\rangle\right\rangle\right\rangle$. Then, selects challenges as $\left\langle e^{(l)} : l \in \mathbb{Z}_s\right\rangle \leftarrow \left[ch\left(tr\right)\right]$, with $ch \in \Re\left(Tr, {\mathbb{Z}_\varepsilon}^s\right)$.

[7] It's trivial to calculate a permutation $\pi \in \Pi_{(\alpha'')}$, whose order enables explicit calculation of components of $\overrightarrow{y''}$.

Response Consider superscript (l) implicit from this point forward. For each $l \in \mathbb{Z}_s$: determines $\vec{r} \leftarrow \left[\phi^{(PS-R)}\left(\left\langle \vec{\Diamond}, \vec{d'} \right\rangle, e\right)\right]$. Let $\vec{r} \equiv \rho\left(\left\langle \vec{\Diamond'}, \vec{d'}, \vec{d''} \right\rangle, \vec{J}\right)$ with $\vec{\Diamond'} \in \chi\left(\vec{\Diamond}\right)$. For $j_0 \in J_{(0)}$: selects $\xi'_{j_0} \leftarrow \left[\phi^{(n,1)}\left(\Diamond'_{j_0}\right)\right]$ and determines $\psi'_{j_0} = f^{(n,2)}\left(\xi'_{j_0}\right)$. Using the ability to invert $f^{(n,2)}$, computes[8] for $j_1 \in J_{(1)} \equiv \bigcup_{j_0 \in J_{(0)}} \gamma_1\left(\Diamond'_{j_0}\right)$ components x'_{j_1} satisfying $f^{(n,2)}\left(x'_{j_1}\right) = y'_{j_1}$ and $f^{(n,1)}\left(x'_{j_1}\right) = d'_{j_1}$ and computes for $j_2 \in J_{(2)}$ components x''_{j_2} satisfying $f^{(n,2)}\left(x''_{j_2}\right) = y''_{j_2}$ and $f^{(n,1)}\left(x''_{j_2}\right) = d''_{j_2}$. Defines $\vec{r'}$ as in the sequential version.

Transcript update The transcript is finalized as $tr = \left\langle tr, \left\langle e^{(l)} : l \in \mathbb{Z}_s \right\rangle, \left\langle \overrightarrow{r'^{(l)}} : l \in \mathbb{Z}_s \right\rangle \right\rangle$.

5 Some considerations

Some accomplishments 1) Communication complexity improvements were obtained for *PZK-IAS-MLCA*s for 4 different languages ($\left[\overline{\wedge}_{(\alpha)}\right]$, $\left[=_{(\alpha,m)}\right]$, $\left[\neq_{(\alpha,m)}\right]$ and *G3C*), assuming *QHC* of $+_2$ or $+_m$; 2) The defined framework allows conceptualization of functionalities considering just high-level properties, namely alternative forms to recognize languages, that "intuitively" suggest the definition of *PF*s $q^{(W)}$ and $q^{(R)}$ when a *PZK-CS* allowing *QHC* of some function $\Diamond$ is available; 3) Simulators were constructed for both sequential and parallel versions of the *PZK-IAS-MLCA*.

An interesting problem The example in section 3 for language $\left[\times_{(\alpha=3,3)}\right]$ was solved after equivaling membership with a condition of type $0 \in^2 \mu\left(\vec{d}\right) \wedge 1 \in^1 \mu\left(\vec{d}\right) \wedge 2 \in^1 \mu\left(\vec{d}\right)$, for some "strange" function $\vec{\mu} : {\mathbb{Z}_3}^3 \rightarrow {\mathbb{Z}_3}^5$. This apparent non-triviality suggests the following problem: Is there any straightforward way, for general $m \in \mathbb{N}$, to define function $\vec{\mu} : \mathbb{Z}_m^3 \rightarrow {\mathbb{Z}_m}^{\alpha''}$ and respective properties of type $\wedge_{j \in \mathbb{Z}_m} j \in^k (j)\, \vec{\mu}\left(\vec{d}\right)$ that enable an equivalence with membership in $\left[\times_{(\alpha=3,m)}\right]$?

Clarifying an open problem It remains an open problem nowadays to find a *CS* allowing homomorphic encryption of a complete function in a Boolean sense, as is the case of $Nand$ ($\overline{\wedge} : \{0,1\}^* \rightarrow \{0,1\}$). By using an eventual *PZK-CS* allowing *QHC* of a complete function it would be possible to homomorphically produce a witness y'' whose decommitment x'' codified an element d'' equal to the value of the membership predicate $\left[\vec{d} \in L_{(\alpha)}\right]^?$, for any verifiable language $L_{(\alpha)}$. This would be a breakthrough in terms of communication complexity improvement, since arguments for membership in any verifiable language would reduce to disclosure of a single decommitment. Consider the following reasoning: A complete function is not quasigroupic (because composition of functions is closed under quasigroupicness). Also, the composition of a complete function may produce any other function. Thus, without loss of generality, consider a function $\vec{\Diamond} : \Delta^2 \rightarrow \Delta$, satisfying $d^{(0)} = \Diamond\left(d^{(1)} \cdot d^{(2)}\right) = \Diamond\left(d^{(1)} \cdot d^{(3)}\right)$, for some specific values $d^{(0)}, d^{(1)}, d^{(2)}, d^{(3)}$ all in Δ and satisfying $d^{(2)} \neq d^{(3)}$. Consider now function $\vec{\xi} = q^{(n)}\left(\vec{\Diamond}\right)$ and, for $i \in \{0,1,2,3\}$, codifications $x^{(i)}$ and commitments $y^{(i)}$ satisfying $f^{(n,1)}\left(x^{(i)}\right) = d^{(i)}$ and $f^{(n,2)}\left(x^{(i)}\right) = y^{(i)}$ and also $x^{(0)} = \xi\left(x^{(1)}, x^{(2)}\right) = \xi\left(x^{(1)}, x^{(3)}\right)$ and $y^{(0)} = \psi\left(y^{(1)}, y^{(2)}\right) = \psi\left(y^{(1)}, y^{(3)}\right)$. Because ψ is quasigroupic, $y^{(2)}$ and $y^{(3)}$ must be equal. Assuming now that inversion of function ξ is computationally feasible, given $x^{(0)}$ and $x^{(1)}$ it's possible to solve $x^{(0)} = \xi\left(x^{(1)}, x^{(2)}\right)$ in order of $x^{(2)}$ and solve $x^{(0)} = \xi\left(x^{(1)}, x^{(3)}\right)$ in order of $x^{(3)}$. But since $d^{(2)} \neq d^{(3)}$, the pair of values $\left\langle x^{(2)}, x^{(3)} \right\rangle$ constitute, according to the definition of *PZK-CS* a trapdoor for $f^{(n,2)}$. Thus, it must be concluded that an eventual *PZK-CS* allowing *QHC* of a complete Boolean must be such that inversion of functions ξ is not computationally feasible, i.e. functions ξ are themselves one-way.

[8] Let $t \in {X_{(n)}}^2$ be a trapdoor for n, as described in the definition of *PZK-CS*, enabling inversion of $f^{(n,2)}$ using *PF* $\phi^{(SND)}$. Components x'_{j_1} are computed as $x'_{j_1} \leftarrow \left[\phi^{(SND)}\left(n, t, y'_{j_1}, d'_{j_1}\right)\right]$

6 Acknowledgments

I thank professor Paulo Mateus, from Math Department in Instituto Superior Técnico of Universidade Técnica de Lisboa, for having supervised my final graduation work in 2003/2004 (from which this article follows), encouraging me to generalize some early stage results and concretize and improve notation in a final stage. I thank my parents for all the support given throughout the realization of this work.

7

References

[1] Manuel Blum; Coin Flipping by Telephone; Crypto, 1981, pages 11-15

[2] S. Goldwasser, S. Micali and C. Rackoff. The knowledge Complexity of Interactive Proof Systems. placecountry-regionSIAM J. Comp. Vol. 18 (1989), pp. 186-208. Preliminary version in 17th STOC, 1985.

[3] Gilles BRASSARD and Claude CREPEAU, Non Transitive Transfer of Confidence: A perfect Zero Knowledge Interactive Protocol for SAT and Beyond. In 27th Symp. of Found. of Computer Sci., pages 188-195. IEEE, 1986.

[4] G. Brassard, D. Chaum and C. Crépeau. Minimum Disclosure Proofs of Knowledge. *Journal of Computer and System Science*, Vol. 37, No 2, pages 156-189, 1988. Preliminary version by Brassard and Crépeau in 27th FOCS, 1986

[5] Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems; O. Goldreich, S. Micali, A. Wigderson; Journal of the ACM, Vol. 38, No. 1, p691-729, 1991. Preliminary version in 32nd FOCS, 1991.

[6] placeI. Niven, H. Zuckerman, H. Montgomery; An Introduction to The Theory of Number; John Wiley & Sons, Inc.; 1991

[7] Lecture Notes on Cryptography; Shafi Goldwasser, Mihir Bellare; August 2001

[8] Oded Goldreich, Foundations of Cryptography, Vol.1, 2001 http://www.wisdom.weizmann.ac.il/~oded/frag.html

[9] C. Dwork and Larry Stockmeyer, 2-Round Zero Knowledge and Proof Auditors, extended abstract in Proceedings of 34th ACM Symposium on Theory of Computing (2002)

[10] *Zero-knowledge twenty years after its invention*, Oded Goldreich; Technical report, Department of Computer Science, Weizmann Institute of Science, 2002.

[11] Chunming Tang, Zhuojun Liu, Jinwang Liu; The Statistical Zero-knowledge Proof for Blum Integer Based on Discrete Logarithm; 2003

[12] Probabilistic Automata: system types, parallel composition and comparison, A. Sokolova and E. P. de Vink. Validation of Stochastic Systems, C. Baier, B. Haverkort, H. Hermanns, J.-P. Katoen and M. Siegle, editors. LNCS 2925, pp.1-43, 2004

[13] "Wikipedia – The Free Encyclopedia", http://en.wikipedia.org/: homomorphism, quasigroup, homomorphic encryption.

[14] Eric W. Weisstein. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/... : "Homomorphism", "Quasigroup", "Hyperreal Number.", "Group", "simple graph", "metric".

A Notation

Frequent Acronyms

- *CS*: Commitment Scheme
- *QHC*: Quasigroupic Homomorphic Commitment
- *IAS*: Interactive Argument System
- *MLCA*: Membership in language with Committed Alphabet
- *PD*: Probability Distribution
- *PF*: Probabilistic Function
- *PZK*: Perfect Zero Knowledge

Frequent general symbols

- $x, y, ... \in X, Y, ... : (x \in X) \wedge (y \in Y) \wedge ...$
- $[n = m]^? : 1$ if $n = m$, 0 otherwise.
- X , Y : set of input, set of output.
- $F(X, Y)$: Set of functions from X to Y.
- $PD(X, Y)$: Set of *PD* from X to Y.
- $\Re(X, Y)$: Set of *PF* from X to Y.
- $P(S)$: Power-set of set S.
- f , g , ϕ : Function, *PD*, *PF*.
- $\phi^{\langle -1, g \rangle}$: g -inverse of PF ϕ.
- $\phi_x(y)$: probability that *PF* ϕ returns output y when it has x as input.
- $y \leftarrow [\phi(x)] : Y$: element y is selected from Y using a *PD* $\phi(x)$.
- $U(X)$: Uniform *PD* over set X.
- $A \backslash B : \{x \in A : x \notin B\}$.
- $\#(S)$: cardinality of set S.
- Δ : Alphabet
- $\equiv$: Equality by definition.
- $\mathbb{Z}_n$: Set of non-negative integers inferior to n.
- $=_n, \times_n, +_n$: equality, multiplication and sum *mod* n.
- $\mathbb{N}_n$: Set of integers not inferior to n.

Notation in vectors and strings

- d_j : $(j+1)^{th}$ component of vector $\vec{d}$.
- $\langle a_0, a_1, ... \rangle$, $\langle a_i : i \in I \rangle$ or $a_0 \cdot a_1 \cdot ...$: Sequence, vector or string (order matters).
- $\vec{a} || \vec{b}$: concatenation of vectors $\vec{a}$ and $\vec{b}$.
- $|\vec{v}|$: length of vector $\vec{v}$.
- $\mathbb{Z}_n^*$: Set of finite length strings with components in $\mathbb{Z}_n$ (don't confuse with $\mathbb{Z}_n^*$).
- $\exists^i \delta \in \vec{d}$: exist exactly i components of $\vec{d}$ with value δ.
- $\delta \in^i \vec{d}$: exist at least i components of $\vec{d}$ with value δ.
- d^α : sequence $\langle d : j \in \mathbb{Z}_\alpha \rangle$.
- Subscript or superscript inside parenthesis (e.g. $X_{(n)}$ or $\xi^{(n)}$): enumeration of a set or element, respectively, from within a family.
- Subscript or superscript without parenthesis (e.g. d_j or Δ^α): component of a vector or power (Cartesian product), respectively.
- $\Pi_{(n)}$, $\Pi(\langle a_0, ... \rangle)$: Set of permutations of $\langle j : j \in \mathbb{Z}_n \rangle$ and of $\langle a_0, ... \rangle$, respectively.

B Notes

Function γ (relevant arguments of input)

Consider an arbitrary set S and arity $\alpha \in \mathbb{N}_1$. The indices of components that are relevant input to functions in $F(S^\alpha, S)$ may be determined by function $\gamma : F(S^\alpha, S) \to P(\mathbb{Z}_\alpha)$ defined by $(\forall f, j \in F(S^\alpha, S), \mathbb{Z}_\alpha)$
$\left[j \in (\mathbb{Z}_\alpha \backslash \gamma(f)) \Leftrightarrow \left[(\forall \vec{s} \in S^\alpha)(\exists \sigma \in S)\left(\forall \overrightarrow{s'} \in S^\alpha\right)\left((\wedge_{k \in \mathbb{Z}_\alpha : k \neq j} s_k = s'_k) \Rightarrow f\left(\overrightarrow{s'}\right) = \sigma\right)\right]\right]$.
Informally, index j belongs to $\gamma(f)$ if and only if the $(j+1)^{th}$ component of f 's input is relevant to its output.

Consider now, more generally, integer $n \in \mathbb{N}_2$ and arities $\alpha(i) \in \mathbb{N}_1$ for each $i \in \mathbb{Z}_n$. Indices of components of relevant input can be determined by vector function $\vec{\gamma} \in F\left(F\left(\times_{i \in \mathbb{Z}_n} S^{\alpha(i)}, S\right), \times_{i \in \mathbb{Z}_n} P\left(\mathbb{Z}_{\alpha(i)}\right)\right)$, defined by condition $\left(\forall f, i, j \in F(S^\alpha, S), \mathbb{Z}_n, \mathbb{Z}_{\alpha(i)}\right)$
$\left[j \in \left(\mathbb{Z}_{\alpha(i)} \backslash \gamma_i(f)\right) \Leftrightarrow \left[\left(\forall \vec{\vec{s}} \in \times_{i' \in \mathbb{Z}_n} S^\alpha\right)(\exists \sigma \in S)\left(\forall \overrightarrow{\overrightarrow{s'}} \in \times_{i' \in \mathbb{Z}_n} S^\alpha\right)\right.\right.$
$\left.\left.\left(\left[\left(\wedge_{i' \in \mathbb{Z}_n : i' \neq i} \vec{s}_i = \vec{s}_{i'}\right) \wedge \left(\wedge_{k \in \mathbb{Z}_\alpha : k \neq j} s_{i,k} = s'_{i,k}\right)\right] \Rightarrow f\left(\overrightarrow{\overrightarrow{s'}}\right) = \sigma\right)\right]\right]$. Informally, index j belongs to $\gamma_i(f)$ if and only if the $(j+1)^{th}$ component of the $(i+1)^{th}$ argument of input of f is relevant to its output.

Proving the ability to invert the *PZK-CS*

To enable a parallelization possibility of a *PZK-IAS-MLCA* (whose framework is defined in section 4), it's necessary that in the initialization process of the *PZK-CS*, agent V proves to have the ability to invert function $f^{(n,2)}$. For a sufficiently large integer s, the desired proof runs as follows:

1. V computes $\overrightarrow{y} = \left\langle y_j \leftarrow \left[\phi^{(PZK)}{}_n\right] : j \in \mathbb{Z}_s \right\rangle$ and sends it to P.
2. P selects $\overrightarrow{d} \leftarrow [U(\Delta^s)]$ and sends it to V.
3. V computes $\overrightarrow{x} = \left\langle \left(x_j \leftarrow \left[\phi^{(SND)}(n, t, y_j, d_j)\right] : X_{(n)}\right) : j \in \mathbb{Z}_s \right\rangle$ and sends it to P.
4. P verifies that $f^{(n,1)}(\overrightarrow{x}) =^? \overrightarrow{d}$ and $f^{(n,2)}(\overrightarrow{x}) =^? \overrightarrow{y}$.

The length s of vectors makes the probability of success for a dishonest P utmost negligibly superior to m^{-s}. A dishonest P (i.e. not able to invert $f^{(n,2)}$) would have a probability of m^{-s} of guessing challenge $\vec{d}$ and thus initially generating $x_j \leftarrow \left[\phi^{(n,1)}(d_j)\right]$ and only then $y_j = f^{(n,2)}(x_j)$, for each $j \in \mathbb{Z}_s$. If P didn't guess the challenge but still was capable of responding correctly with value x_j, then it would have two decommitments corresponding to different values in Δ. However, according to function $\phi^{(SND)}$ in the definition of *PZK-CS*, that pair of decommitments is indeed a trapdoor enabling inversion of $f^{(n,2)}$.