Adaptive Pseudo-Free Groups and Applications*

Dario Catalano[1], Dario Fiore[2]** and Bogdan Warinschi[3]

[1] Dipartimento di Matematica e Informatica,
Università di Catania, Italy.
catalano@dmi.unict.it
[2] École Normale Supérieure, CNRS - INRIA, Paris, France.
dario.fiore@ens.fr
[3] Dept. Computer Science, University of Bristol, UK
bogdan@cs.bris.ac.uk

Abstract. A computational group is *pseudo-free* if an adversary cannot find solutions in this group for equations that are not trivially solvable in the free group. This notion was put forth by Rivest as a unifying abstraction of multiple group-related hardness assumptions commonly used in cryptography. Rivest's conjecture that the RSA group is pseudo-free had been settled by Micciancio for the case of RSA moduli that are the product of two safe primes. This result holds for a static setting where the adversary is only given the description of the group (together with a set of randomly chosen generators) and has to come up with the equation and the solution.
In this paper we explore a powerful extension of the notion of pseudo-freeness. We identify, motivate, and study pseudo-freeness in face of *adaptive* adversaries who may learn solutions to other non-trivial equations before having to solve a new non-trivial equation.
Our first contribution is a carefully crafted definition of *adaptive* pseudo-freeness that walks a fine line between being too weak and being unsatisfiable. We give generic constructions that show how any group that satisfies our definition can be used to construct digital signatures and network signature schemes.
Next, we prove that the RSA group meets our more stringent notion of pseudo-freeness and as a consequence we obtain different results. First, we obtain a new network (homomorphic) signature scheme in the standard model. Secondly, we demonstrate the generality of our framework for signatures by showing that *all* existing strong RSA-based signature schemes are instantiations of our generic construction in the RSA group.

1 Introduction

Background. The search for abstractions that capture the essential security properties of primitives and protocols is crucial in cryptography. Among other benefits, such abstractions allow for modular security analysis, reusable and scalable proofs. The random oracle model [5], the universal composability framework [9] and variants [1, 3, 19] of the Dolev-Yao models [11] are results of this research direction. Most of the existing results in this direction (the above examples included) tackle mostly primitives and protocols and are not concerned with the more basic mathematical structures that underlie current cryptographic constructions. One notable exception is the work on pseudo-free groups, a notion put forth by Hohenberger [16] and later refined by Rivest [20]. In this paper we continue the investigation of this abstraction.

Roughly speaking, a computational group $\mathbb{G}$ (a group where the group operations have efficient implementations) is pseudo-free if it behaves as a free group as far as a computationally bounded

* An extended abstract of this paper appears in the proceedings of Eurocrypt 2011. The work described in this paper has been supported in part by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

** Work partially done while student at University of Catania.

adversary is concerned. More specifically, a group is pseudo-free if an adversary who is given a description of the group cannot find solutions for non-trivial equations. Here, non-triviality means that the equation does not have a solution in the free group. For instance, in a pseudo-free group given a random element a it should be hard to find a solution for an equation of the form $x^e = a$, when $e \neq 1$, or for the equation $x_1^2 x_2^4 = a^5$, but not for the equation $x_1 x_2^3 = a^5$. This last equation is trivial since it can be solved over the free group (it has $x_1 = a^2, x_2 = a$ as solution in the free group) and a solution in the free group immediately translates to a solution over $\mathbb{G}$. The notion of pseudo-freeness generalizes the strong RSA assumption (when $\mathbb{G}$ is an RSA group) but also numerous other assumptions currently used in cryptography; see [20] for further details. Rivest's conjecture that the RSA group is pseudo-free was largely settled by Micciancio [18] who proved that this is indeed the case when the RSA modulus is the product of two safe primes.

In its most basic form that had been studied so far, the notion of pseudo-free groups did not lend itself easily to applications. The problem is that in most of the interesting uses of the RSA group the adversary is not only given a description of the group, but often he is allowed to see solutions to non-trivial equations before having to come up with his own new equation and solution. This is the case for example in RSA-based signature schemes where one can think of a signature as the solution to some non-trivial equation. A chosen-message attack allows the adversary access to an oracle that solves (non-trivial) equations over the group, and a forgery is a solution to a new equation.

This problem was recognized early on by Rivest [20] who also left as open problems the design of a notion of pseudo-freeness for adaptive adversaries and, of course, whether such groups exist. In this paper we put forth such a notion, prove that the RSA group is adaptive pseudo-free, and exhibit several applications for adaptive pseudo-free groups. We detail our results next.

Adaptive pseudo-free groups. We first extend the notion of pseudo-freeness to adaptive adversaries. Informally, we consider an adversary that can see solutions for some equations and has as goal solving a new non-trivial equation. As explained above, this scenario captures typical uses of groups in cryptography.

Our definition involves two design decisions. The first is to fix the type of equations for which the adversary is allowed to see solutions and how are these equations chosen: too much freedom in selecting these equations immediately leads to potentially unsatisfiable notions, whereas too severe restrictions may not model the expected intuition of what an adaptive adversary is and may not allow for applications. In the definition that we propose, equations are selected from a distribution over the set of equations. Importantly, the distribution depends on a parameter supplied by the adversary. This models the idea that in applications, the adversary may have some control over how the equations are selected. Different choices for this distribution lead to a variety of adversaries from very weak ones where no equation is provided (precisely the setting of pseudo-freeness proposed earlier), to a setting where the adversary has no influence on the choice of equations, and ending with the very strong notion where the adversary basically selects the equations on his own.

The second issue is to define what is a non-trivial equation in the adaptive setting. Indeed, previous definitions of triviality do not apply since in our new setting the adversary knows additional relations between the group elements which in turn may help him in solving additional equations. We define non-triviality in a way motivated by existing uses of groups in cryptography and an analysis of equations over quotients of free groups.

Our definition is for the case of univariate equations but can be easily extended to multivariate equations as well as systems of equations.

Generic constructions for signatures. Our definition of pseudo-freeness is parametrized by a distribution over equations. We show that for any distribution in a class of distributions that satisfy certain criteria, one can construct secure digital signatures and network coding signature schemes. The requirements on the distribution include the ability to efficiently check membership in the support of the distribution, and a property on the distribution of the exponents in the equation. Informally, these requirements are used to enforce that each equation freshly drawn from the distribution is most likely non-trivial with respect to previously sampled equations. We show that an adversary that breaks the signature scheme must also contradict the pseudo-freeness of the underlying group.

Our generic construction for network coding signatures is secure in the vanilla model based only on the adaptive pseudo-freeness of the underlying group. Any instantiation of such groups would thus yield network signature schemes secure in the standard model. Indeed, given the instantiation that we discuss below, our framework yields the first RSA-based network coding homomorphic signature scheme secure in the standard model.

The RSA group is adaptive pseudo-free. Next, we turn to proving that the RSA group is adaptive pseudo-free. We do so for a class of distributions closely related but slightly more general than the distributions that yield signatures schemes. We show that an adversary that contradicts pseudo-freeness of the RSA group with respect to the distribution can be used to contradict the strong RSA assumption. We also prove that the RSA group is pseudo-free for a weaker version of adaptive adversaries who output their inputs to the distribution non-adaptively, but in this case the proof is for a larger class of distributions.

We do not attempt to prove adaptive pseudo-freeness of the RSA group for multivariate equations. While this is potentially an interesting topic for further research, we are not aware of cryptographic applications where such equations are used.

Instantiations. An appealing interpretation of the proof of adaptive pseudo-freeness for the RSA group is that it distills the core argument that underlies the typical security proofs for signatures based on the strong RSA assumption. Each such proof explains how a signature forgery can be used to break strong RSA. In this sense our proof is a generalization to a broader (abstractly defined) set of equations rather than the particular equations that define an individual signature scheme.

Indeed, we show that virtually *all* strong RSA signature schemes are instances of our generic construction. We explain how to obtain the schemes by Cramer and Shoup [10], Fischlin [12], Camenisch and Lysyanskaya [8], Zhu [22], Hofheinz and Kiltz [15], and that by Gennaro, Halevi, and Rabin [13] by instantiating our generic distribution in appropriate ways. The security of all of these schemes follows as a corollary from the security of our generic construction.

2 Preliminaries

A number N is called a *RSA modulus* if it is the product of two distinct prime numbers p, q. $QR_N \subseteq \mathbb{Z}_N^*$ is called the set of *quadratic residues* modulo N, namely $QR_N = \{\tau \in \mathbb{Z}_N^* : \tau = z^2 \bmod N, z \in \mathbb{Z}_N^*\}$.

Definition 1 (Safe primes). *A prime p is called* safe prime *if $p = 2p' + 1$ where p' is also prime.*

The Strong RSA Assumption was introduced by Baric and Pfitzmann in [4]. Essentially it is a variant of RSA where the adversary is allowed to choose the exponent e for which it has to extract the root. It is formally defined as follows.

Definition 2 (Strong RSA). *Let N be a random RSA modulus of length k where $k \in \mathbb{N}$ is the security parameter and τ be a random element in $\mathbb{Z}_N^*$. Then we say that the Strong RSA assumption holds if for any PPT adversary $\mathcal{A}$ the probability*

$$Pr[(y, e) \leftarrow \mathcal{A}(N, \tau) : y^e = \tau \bmod N]$$

is negligible in k.

In this paper we use a variant of this assumption where τ is taken from the set QR_N. As shown in [10] such variant is implied by the standard Strong RSA.

2.1 Division Intractable Functions

In our work we use the notion of *division intractable functions*. Informally, a function H is division intractable if an adversary $\mathcal{A}$ cannot find $x_1, x_2, \ldots, x_t, y$ such that: $y \neq x_i$ and $H(y)$ divides the product of the $H(x_i)$'s. It is easy to see that this notion is satisfied by any function that maps inputs to (distinct) prime numbers. Such mappings can be instantiated without making any cryptographic assumptions (see [7] for a construction), but they are not very efficient in practice.

Gennaro *et al.* introduced in [13] the notion of division intractable hash functions and also showed how to get practical implementations of them. We recall below the formal definition.

Definition 3 (Division Intractable Hash Functions). *Let $\mathcal{H}$ be a family of hash functions with poly(k)-bit input and k bit output. We say that $\mathcal{H}$ is* division intractable *if for any PPT adversary it is hard to win the following game:*

1. *a function H is chosen at random from $\mathcal{H}$;*
2. *the adversary outputs $x_1, x_2, \ldots, x_t, y$ such that: (i) $y \neq x_i \; \forall i = 1, \ldots, t$ and (ii) $H(y) | \prod_{i=1}^{t} H(x_i)$.*

2.2 Signatures

A digital signature scheme Π is given by a triple of algorithms $(\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ for key generation, signing, and verifying respectively. Key generation takes as input a security parameter k and returns a pair of keys $(\mathsf{sk}, \mathsf{vk})$ for producing and verifying signatures, respectively. On input a signing key sk and a message m, the signature algorithm produces a signature σ. The verification algorithm takes as input a triple vk, m, σ and tests if signature σ is a valid signature on m with respect to verification key vk.

We recall two security notions for signature schemes.

Definition 4 (Security of signature schemes). *Consider the experiment $\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{A},\Pi}(k)$ where a signing, verification key-pair $(\mathsf{sk}, \mathsf{vk})$ is generated for security parameter k. Then, the adversary is given vk and is provided with a signing oracle that produces signatures on the messages that the adversary (adaptively) queries. Eventually, the adversary outputs a tentative forgery $(m*, \sigma*)$. The experiment returns 1 if σ^* is a valid signature on m^* and m^* had not been queried to the signature oracle. We call $\mathbf{Exp}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Pi}(k)$ the related experiment where $(m*, \sigma*)$ is considered a forgery if it is different from all the pairs (m_i, σ_i) obtained from the signature oracle. A signature scheme Π is* unforgeable under chosen message attack *if for any probabilistic, polynomial time adversary $\mathcal{A}$ the advantage of $\mathbf{Adv}^{\mathsf{uf\text{-}cma}}_{\mathcal{A},\Pi}(k) = \Pr[\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{A},\Pi}(k) = 1]$ is a negligible function. The signature scheme is* strongly-unforgeable *if $\mathbf{Adv}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Pi}(k) = \Pr[\mathbf{Exp}^{\mathsf{suf\text{-}cma}}_{\mathcal{A},\Pi}(k) = 1]$ is a negligible function.*

It is also possible to consider a relaxed experiment where the adversary is required to choose the messages for which it wants to see the signatures, before receiving the public key. Signature schemes that are proved with respect to such experiment are said to be *weakly-secure*.

3 Static pseudo-free groups

As warm up, we recall the notion of pseudo-free groups as introduced by Rivest [20]. To distinguish it from the notions that we develop in this paper we refer to the older notion as *static* pseudo-free groups.

FREE ABELIAN GROUPS. For any set of symbols $A = \{a_1, a_2, \ldots, a_m\}$ we write A^{-1} for the set of symbols $A^{-1} = \{a_1^{-1}, a_2^{-1}, \ldots, a_m^{-1}\}$. Let $X = \{x_1, \ldots, x_n\}$ and $A = \{a_1, \ldots, a_m\}$ be two disjoint sets of variables and constant symbols. An equation over X with constants in A is a pair $\lambda = (w_1, w_2) \in (X^* \times A^*)$. We usually write an equation $\lambda = (w_1, w_2)$ as $w_1 = w_2$ and looking ahead (we will only consider these equations over abelian groups), we may also write it as $x_1^{e_1} x_2^{e_2} \cdots x_n^{e_n} = a_1^{s_1} a_2^{s_2} \cdots a_m^{s_m}$ where $\{e_1, \ldots, e_n\}$ and $\{s_1, \ldots, s_m\}$ are integers.

Let $(G, \cdot)$ be an arbitrary abelian group and $\alpha : A \to G$ be an interpretation of the constants in A as group elements. We write λ_α for the equation λ interpreted over G via α. An evaluation $\psi : X \to G$ is a solution for λ_α if

$$\psi(x_1)^{e_1} \cdots \psi(x_n)^{e_n} = \alpha(a_1)^{s_1} \cdots \alpha(a_m)^{s_m}.$$

Any equation λ over X and A can be viewed as an equation over the free group $\mathcal{F}(A)$ via the interpretation $1_A : A \to \mathcal{F}(A)$ that maps a to a. It can be easily shown [20, 18] that the equation λ_{1_A} has a solution in $\mathcal{F}(A)$ if and only if $\forall i = 1, \ldots, m$, it holds $gcd(e_1, \ldots, e_n) \mid s_i$. We call such equations *trivial*, in the sense that these equations have solutions over the free group. All of the other equations are deemed *non-trivial*.

STATIC PSEUDO-FREE GROUPS. A computational group consists of a (finite) set of representations for the group elements together with efficient implementations for the two group operations. Informally, a computational group is pseudo-free if it is hard to find an equation which is unsatisfiable over the free group, together with a solution in the computational group. It is worth noting that if the order of the group is known then finding solutions for non-trivial equations may be easy. Therefore, the notion of pseudo-free groups holds for families $\mathcal{G} = \{\mathbb{G}_N\}_{N \in \mathcal{N}_k}$ of computational groups where N is chosen at random from the set of indexes $\mathcal{N}_k$ (typically these are the strings of length k) and the corresponding order $ord(\mathbb{G}_N)$ is hidden to the adversary.

In the following we recall the formal definition given by Micciancio in [18] (which is similar to that of Rivest [20]). The adversary that is considered in the following definition is static (in that it is only allowed to see a description of the group, but obtains no further information). To distinguish this class of groups from others that we define in this paper we call them *static pseudo-free groups*.

Definition 5 (Static Pseudo-Free Groups [18]). *A family of computational groups $\mathcal{G} = \{\mathbb{G}_N\}_N$ is* static pseudo-free *if for any set A of polynomial size $|A| = p(k)$ (where k is a security parameter), and PPT algorithm $\mathcal{A}$, the following holds. Let $N \in \mathcal{N}_k$ be a randomly chosen group index, and define $\alpha : A \to \mathbb{G}_N$ by choosing $\alpha(a)$ uniformly at random in $\mathbb{G}_N$, for each $a \in A$. Then, the probability (over the selection of α) that on input (N, α) adversary $\mathcal{A}$ outputs an equation λ and a solution ψ for λ_α is negligible in k.*

4 Adaptive pseudo-free groups

A ROUGH DEFINITION. The notion described above requires an adversary to produce a solution for some non-trivial equation only given some randomly chosen generators to be used in the equation,

but no additional information. In contrast, the notion that we develop attempts to capture the idea that an adversary against the computational group gets to see several equations with solutions, and then attempts to solve a new *non-trivial* equation. A typical cryptographic game that captures this situation involves an adversary $\mathcal{A}$ who works against a Challenger as follows.

Setup The Challenger chooses a random instance of the computational group $\mathbb{G}_N$ (by picking a random index $N \xleftarrow{\$} \mathcal{N}_k$) from a family $\mathcal{G} = \{\mathbb{G}_N\}_{N \in \mathcal{N}_k}$. Then he fixes an assignment $\alpha : A \to \mathbb{G}_N$ for the set of constants and gives $(\alpha, \mathbb{G}_N)$ to the adversary.

Equations queries In this phase the adversary is allowed to see non-trivial equations together with their solutions.

Challenge At some point the adversary is supposed to output a new "non-trivial" equation λ^* (defined by $(e^*, \boldsymbol{s^*})$) together with a solution ψ^*.

Notice that the above description incorporates an assumption that we make for simplicity, namely that all equations are univariate. In general, any univariate equation over A is of the form: $x^e = a_1^{s_1} a_2^{s_2} \cdots a_m^{s_m}$. For the case of static pseudo-free groups, this restriction is justified by the following lemma that was proved by Micciancio in [18]. Informally the lemma says that any (multivariate) equation and solution (λ, ψ) can be efficiently transformed into a univariate equation and solution (λ', ψ'). Whilst we extend the definition of trivial equations to the multivariate case in Appendix A, it would be interesting to see if a similar lemma is possible in the context of adaptive pseudo-freeness.

Lemma 1 ([18]). *For any computational group family $\mathcal{G}$, there is a PPT algorithm that on input an equation λ over constants A and variables X,a group $\mathbb{G}$ from $\mathcal{G}$, and a variable assignment $\psi : X \to \mathbb{G}$, outputs a univariate equation λ' and value $\psi' \in \mathbb{G}$ such that: (1) if λ is unsatisfiable over the free group $\mathcal{F}(A)$, then λ' is also unsatisfiable over $\mathcal{F}(A)$ and (2) for any assignment $\alpha : A \to \mathbb{G}$, if ψ is a solution to λ_α, then ψ' is a solution to λ'_α.*

The general definition of pseudo-freeness that we sketched above leaves open two important points: 1) How are the equations for which the adversary sees solutions produced? and 2) What does "non-trivial equation" mean when other equations and solutions are given? We discuss and give answers to these two problems in Sections 4.1 and 4.2 respectively.

4.1 A spectrum of adaptive adversaries

The second phase of the above generic game requires that adversaries be given non-trivial equations together with their solutions, so we need to clarify how are these equations produced. Here we identify a whole spectrum of possible choices. The weakest definition one might consider is one where the adversary does not have any control over these equations. For instance, this means that, whenever the Challenger is queried in the second phase, the Challenger chooses an equation λ_i (more precisely it chooses its exponents $(e_i, \boldsymbol{s_i})$) and gives λ_i and its solution in $\mathbb{G}$, ψ_i, to the adversary. Unfortunately, in such a game the adversary is not really adaptive: it may receive all the equations and solutions at once.

The strongest possible notion, and perhaps the most natural one, would be to consider an adversary that is allowed to choose equations λ_i (namely their respective exponents $(e_i, \boldsymbol{s^i})$) in any way it wants. In particular the choice of the equations can be done in an adaptive way, namely $\mathcal{A}$ asks for an equation, sees its solutions, then chooses another equation and so on. We call this

definition "Strong Adaptive Pseudo-freeness". Unfortunately this choice seems to lead to an unrealizable notion.[4] We therefore settle on an intermediary variant where the adversary is allowed to be adaptive, but still cannot choose the equations in a completely arbitrary way. Instead, we consider a setting where the equations are selected from the set of all equations according to some distribution over which the adversary has some *limited* control. We formulate this limitation via a *parametric distribution* φ over the set of all possible equations. Sampling from such a distribution requires some parameter M of some appropriate length which is provided by the adversary. The distribution then produces a tuple of $m+1$ integers which for expressivity we write $(e, \boldsymbol{s})$. Here e is an integer (the exponent for the variable) and $\boldsymbol{s}$ is a vector of m integers (the exponents for the generators). The idea is that once the parameter M is fixed, $\varphi(M)$ is some fixed distribution from which $(e, \boldsymbol{s})$ are drawn. Notice that the two ends of the spectrum can be modeled via appropriate choices of φ.

4.2 Non-trivial equation w.r.t. other equations

Our definition of adaptive pseudo-freeness requires an adversary to find a solution to a non-trivial equation. In the original setting of Rivest, non-triviality of an equation simply meant that the equation has no solution in the free group. In our setting, non-triviality is less clear: the adversary is already given solutions for some equations which may lead to solutions for other equations that are difficult to solve otherwise. In this section we develop a notion of triviality for equations given solutions to other equations. Our ultimate goal is to characterize, using the world and vocabulary afferent to free groups those equations that cannot be solved in the computational group.

General deducibility modulo equations. We frame the discussion in slightly more general terms to obtain a framework suitable for talking about non-triviality of both univariate and multivariate equations.

Let $\mathcal{F}$ be the free abelian group generated by the set $\{a_1, a_2, \ldots, a_m\}$ and let $\Lambda \subseteq \mathcal{F} \times \mathcal{F}$ be an arbitrary binary relation on $\mathcal{F}$ that models equalities between words in $\mathcal{F}$ (equations with solutions can be thought of as such relations). We therefore aim to characterize the set of all equalities that can be derived from Λ. Recall that eventually these equalities are interpreted over computational groups, hence there are two ways for an adversary to derive new equalities. The first is to use the group operations and their properties. For example, if $\Lambda = \{a_1 a_2 = a_1^2 a_4\}$, then it can also be derived that $a_1 a_2^2 = a_1^2 a_4 a_2 = a_1^3 a_4^2$, where the first equality is obtained by simply multiplying a_2 to the known equation, and the second equality follows using the commutativity of $\mathcal{F}$ and the known equality. The second possibility reflects an ability that computational adversaries have (when working against computational groups). Specifically, if an equality of the form $w_1^q = w_2^q$ can be derived in a computational group, then the equality $w_1 = w_2$ can also be derived (provided that q is relatively prime with the order of the group). Furthermore, since we search for an abstraction independent of the order of the group, we have to consider the above possibility for any q. The following definition is motivated by the above discussion.

Definition 6. *Let $\mathcal{F}$ be a freely generated abelian group and let $\Lambda \subseteq \mathcal{F} \times \mathcal{F}$ be an arbitrary binary relation on $\mathcal{F}$. Let $\equiv_\Lambda$ be the smallest congruence on $\mathcal{F}$ that:*

- $\Lambda \subseteq \equiv_\Lambda$

[4] For example, it is not clear at all if a group like $\mathbb{Z}_N^*$ can be proved strongly-adaptive pseudo-free under any reasonable assumption (e.g. Strong RSA).

– $\forall q \in \mathbb{N}, \forall w_1, w_2 \in \mathcal{F},\ w_1^q \equiv_\Lambda w_2^q \implies w_1 \equiv_\Lambda w_2$.

Then, w_1 and w_2 are trivially equal with respect to Λ if $w_1 \equiv_\Lambda w_2$.

Next, we derive an explicit description for $\equiv_\Lambda$. Let $\Lambda = \{(w_{1,1}, w_{2,1}), (w_{1,2}, w_{2,2}), \ldots, (w_{1,t}, w_{2,t})\}$. Consider the binary relation R_Λ on $\mathcal{F}$ defined by: $(w_1, w_2) \in R_\Lambda$ if and only if there exist $l_1, l_2, \ldots, l_t \in \mathbb{Q}$ such that

$$w_1 = w_2 \cdot \Pi_{i=1}^{t} (w_{1,i}^{-1} \cdot w_{2,i})^{l_i}$$

Here, exponentiation of a word $w = a_1^{s_1} a_2^{s_2} \ldots a_n^{s_n}$ with a rational number $l = p/q$ is defined (in the obvious way) if and only if q divides $\gcd_{1 \le i \le n} p \cdot s_i$

The following proposition states that $\equiv_\Lambda$ and R_Λ are one and the same relation.

Proposition 1. *Let R_Λ and $\equiv_\Lambda$ defined as above. Then $(w_1, w_2) \in R_\Lambda$ if and only if $(w_1, w_2) \in \equiv_\Lambda$.*

The proposition follows by the next two lemmas:

Lemma 2. $\equiv_\Lambda \subseteq R_\Lambda$

Proof. We prove that R_Λ is a congruence and has all of the closure properties required from $\equiv_\Lambda$ (so the desired inclusion follows since $\equiv_\Lambda$ is the smallest congruence with these properties).

- R_Λ is reflexive. Let $w \in \mathcal{F}$ arbitrary. Then we derive that $(w, w) \in R_\Lambda$ by setting $l_1 = l_2 = \ldots = l_t = 0$
- R_Λ is symmetric. for w_1 and w_2 such that $(w_1, w_2) \in R_\Lambda$, so there exists $l_1, l_2, \ldots, l_t \in \mathbb{Q}$ such that $w_1 = w_2 \cdot \Pi_{k=1}^{t} \left(w_{1,k}^{-1} w_{2,k}\right)^{l_k}$. Then $(w_2, w_1) \in R_\Lambda$ by fixing the coefficients for the linear combination to $-l_1, -l_2, \ldots, -l_t$.
- R_Λ is transitive. If $l_1, l_2, \ldots, l_t$ show that $R_\Lambda(w_1, w_2)$ and $m_1, m_2, \ldots, m_t$ show that $R_\Lambda(w_2, w_3)$ then $l_1 + m_1, l_2 + m_2, \ldots, l_t + m_t$ show that $R_\Lambda(w_1, w_3)$.
- R_Λ commutes with the operations. Let w_1, w_2, w_1', w_2' such that $(w_1, w_2), (w_1', w_2') \in R_\Lambda$, so there exists $l_1, l_2, \ldots, l_t, m_1, m_2, \ldots, m_t$ such that $w_1 = w_2 \cdot \Pi_{k=1}^{t} \left(w_{1,k}^{-1} w_{2,k}\right)^{l_k}$ and $w_1' = w_2' \cdot \Pi_{k=1}^{t} \left(w_{1,k}^{-1} w_{2,k}\right)^{m_k}$. Then $(w_1 w_1', w_2 w_2') \in R_\Lambda$ (take the coefficients for the required linear combination to be $l_k + m_k$ for any $1 \le k \le t$). Also, we have that $(w_1^{-1}, w_1'^{-1}) \in R_\Lambda$: take the required coefficients to be $-l_1, -l_2, \ldots, -l_t$.
- $\Lambda \subseteq R_\Lambda$. To show that $(w_{1,k}, w_{2,k}) \in R_\Lambda$ for any $1 \le k \le t$, set all of $l_1, l_2, \ldots, l_t$ be equal 0 with the exception of l_k for an arbitrary $1 \le k \le t$ which is set to 1.
- Let w_1, w_2 be such that $(w_1^q, w_2^q) \in R_\Lambda$. By the definition of R_Λ there exists $l_1, l_2, \ldots, l_t$ such that $w_1^q = w_2^q \cdot \Pi_{k=1}^{t} \left(w_{1,k}^{-1} w_{2,k}\right)^{l_k}$. It follows that $(w_1, w_2) \in R_\Lambda$ by setting the coefficients of the linear combination to $l_1/q, l_2/q, \ldots, l_t/q$.

Since R_Λ satisfies all of the properties that $\equiv_\Lambda$ satisfies, and the latter is the smallest congruence with these properties, it follows that $\equiv_\Lambda \subseteq R_\Lambda$.

Lemma 3. $R_\Lambda \subseteq \equiv_\Lambda$

Proof. Define the operations $R, S, T, I, Q : \mathcal{P}(\mathcal{F} \times \mathcal{F}) \to \mathcal{P}(\mathcal{F} \times \mathcal{F})$ as follows.

- $S(\mathcal{S}) = \{(x, y) \mid (y, x) \in \mathcal{S}\}$

– $T(\mathcal{S}) = \{(x,y) \mid \exists z \in \mathcal{F} : (x,z),(z,y) \in \mathcal{S}\}$
– $Q(\mathcal{S}) = \{(x,y) \mid \exists q \in \mathbb{Z}, (x^q, y^q) \in \mathcal{S}\}$
– $I(\mathcal{S}) = \{(x,y) \mid (x^{-1}, y^{-1}) \in \mathcal{S}\}$
– $M(\mathcal{S}) = \{(x_1x_2, y_1y_2) \mid (x_1,y_1),(x_2,y_2) \in \mathcal{S}\}$

Since all of the operations above commute with each other, the congruence$\equiv_\Lambda$ is the closure of the set $(\mathcal{F} \times \mathcal{F} \cup \Lambda)$ under the above operations. It is easy to see that $\mathcal{F} \times \mathcal{F} \cup \Lambda \subseteq R_\Lambda$ and that for any set $\mathcal{S}$ if $\mathcal{S} \subseteq R_\Lambda$ then $O(\mathcal{S}) \subseteq R_\Lambda$ for any operation $O \in \{S, T, Q, I, M\}$. The desired inclusion then follows.

Trivial equations. Using the notion of deducibility modulo equations developed above we can now specify the class of equations that we consider trivial (given solutions for the equations in some set Λ). For simplicity, we focus on the case of univariate equations which is more relevant for the cryptographic applications of this paper. The definition easily extends to the case of multivariate equations (for completeness we give this variation in Appendix A). Assume that we are given a set of equations

$$\Lambda = \left\{ x^{e_k} = a_1^{s_1^k} \cdots a_m^{s_m^k} \right\}_{k=1}^{t}$$

together with $\{\phi_k\}_{k=1}^t$, their corresponding solutions. (Notice that these are equations in a computational group; solutions for these equations may simply not exist in a free group). Let $\mathcal{F}$ be the the free abelian group generated by $\{\phi_1, \phi_2, \ldots, \phi_t, a_1, a_2, \ldots, a_m\}$ (interpreted as symbols). The equations in Λ induce a binary relation on $\mathcal{F}$ which (by a slight abuse of notation) we also call Λ. So $\Lambda = \{(\phi_k^{e_k}, a_1^{s_1^k} \cdots a_m^{s_m^k}) \mid 1 \leq k \leq t\}$. The following definition simply is a particular instance of Definition 6 to the case of univariate equations.

Definition 7. *Equation $x^{e^*} = a_1^{s_1^*} \cdots a_m^{s_m^*}$ is trivial with respect to Λ if the equation has a solution over $\mathcal{F}/\equiv_\Lambda$.*

We use the characterization of $\equiv_\Lambda$ that we gave earlier to explicitly determine the class of trivial equations. Let

$$x^{e^*} = a_1^{s_1^*} \cdots a_m^{s_m^*} \tag{1}$$

be an equation that has a solution over $\mathcal{F}/\Lambda$. Let $\phi = \phi_1^{k_1} \cdots \phi_t^{k_t} a_1^{v_1} \cdots a_m^{v_m}$ be such a solution. From the explicit characterization of $\equiv_\Lambda$ there exists $l_1, \ldots, l_t$ in $\mathbb{Q}$ such that

$$(\phi_1^{k_1} \cdots \phi_t^{k_t} a_1^{v_1} \cdots a_m^{v_m})^{e^*} = a_1^{s_1^*} a_2^{s_2^*} \cdots a_m^{s_m^*} \cdot \Pi_{i=1}^t \left(\phi_i^{e_i} \cdot \Pi_{k=1}^m a_k^{-s_k^i} \right)^{l_i} \tag{2}$$

Since equality is standard equality over $\mathcal{F}$, the relation above translates (via symbol by symbol matching of exponents) into the following requirement. Equation (1) has a solution if there exist $v_1 \cdots v_m, k_1 \cdots k_t$ in $\mathbb{Z}$ and $l_1, \ldots, l_t \in \mathbb{Q}$ such that:

1. $k_i e^* = e_i \cdot l_i$ (for all $1 \leq i \leq t$)
2. $v_i e^* = s_i^* - \sum_{j=1}^t l_j s_i^{(j)}$ (for all $1 \leq i \leq m$)

The converse of the above statement is also true: if integers $v_1, \cdots v_m, k_1, \ldots, k_t$ and rationals $l_1, \cdots, l_t$ exist such that Equation 2 holds then $\phi = \phi_1^{k_1} \cdots \phi_t^{k_t} a_1^{v_1} \cdots a_m^{v_m}$ is a solution for Equation (1) over $\mathcal{F}/\equiv_\Lambda$.

Finally, we express these two conditions in a more compact matrix form which will be simpler to use in our proofs. Given the set of equations

$$\Lambda = \left\{ x^{e_k} = a_1^{s_1^k} \cdots a_m^{s_m^k} \right\}_{k=1}^{t}$$

we define the following quantities:

$$\Sigma = \begin{bmatrix} s_1^1 & \cdots & s_1^t \\ \vdots & & \vdots \\ s_m^1 & \cdots & s_m^t \end{bmatrix} \text{ and } E = \begin{bmatrix} 1/e_1 & & 0 \\ & 1/e_2 & \\ 0 & & \ddots \\ & & & 1/e_t \end{bmatrix}$$

These quantities are dependent on Λ but we do not show the dependency explicitly to avoid heavy notation.

Proposition 2 (Trivial equation w.r.t. a set of equations). *Equation* $\lambda^* : x^{e^*} = a_1^{s_1^*} \cdots a_m^{s_m^*}$ *is* trivial *w.r.t* Λ *if and only if:*

$$\exists k \in \mathbb{Z}^t, V \in \mathbb{Z}^m : e^*(\Sigma E k + V) = \boldsymbol{s^*}$$

where $\boldsymbol{s^*} = [s_1^* \cdots s_m^*]^T$.

Proof. The proposition follows by simply setting $l_i = k_i \frac{e^*}{e_i}$ for all $1 \leq i \leq t$.

4.3 A definition of adaptive pseudo-free groups

The definition of adaptive pseudo-freeness that we give below is for a set A of m generators, a computational group $\{\mathbb{G}_N\}_N$ and is parameterized by a distribution $\varphi(\cdot)$ as discussed in Section 4.1.

Setup The Challenger chooses a random instance of the computational group $\mathbb{G}_N$ (by picking a random index $N \xleftarrow{\$} \mathcal{N}_k$) from a family $\mathcal{G} = \{\mathbb{G}_N\}_{N \in \mathcal{N}_k}$. Then he fixes an assignment $\alpha : A \to \mathbb{G}_N$ for the set A of generators and a specific parametric distribution φ for the exponents. The adversary is given in input the assignment $\alpha : A \to \mathbb{G}_N$ and the descriptions of the computational group and the parametric distribution φ.

Equations queries In this phase the adversary is allowed to adaptively query the Challenger on equations and see their solutions. More precisely, $\mathcal{A}$ controls the queried equations via the parametric distribution φ. Namely, for each query it chooses a parameter M_i and hands it to the Challenger. The Challenger runs $(e_i, \boldsymbol{s^i}) \leftarrow \varphi(M_i)$, computes the solution ψ_i for the equation λ_i, which is $x^{e_i} = a_1^{s_1^i} \cdots a_m^{s_m^i}$ and gives $(\psi_i, e_i, \boldsymbol{s^i})$ to $\mathcal{A}$.

Challenge Once the adversary has seen the solutions, then it is supposed to output an equation λ^* (defined by $(e^*, \boldsymbol{s^*})$) together with a solution ψ^*. We say that $\mathcal{A}$ wins this game if λ^* is a non-trivial equation.

Definition 8 (Adaptive pseudo-free groups). *$\mathcal{G}$ is a family of adaptive pseudo-free groups w.r.t. distribution φ, if for any set A of polynomial size, any PPT adversary $\mathcal{A}$ wins in the game above with at most negligible probability.*

We restate several of the reasons that justify the above definition. Although the definition is parametrized by a distribution, we feel this is the right way of modeling an adversary who is adaptive but not all-powerful. As explained, by varying the distribution one obtains a large spectrum of potentially interesting instantiations, starting with static pseudo-freeness all the way to strong adaptive pseudo-freeness. Finally, we show that for some fixed distributions adaptive pseudo-freeness implies immediately secure signature schemes.

5 Applications of adaptive pseudo-free groups

As an application of adaptive pseudo-free groups we show how to obtain signature and network coding signature schemes out of pseudo-free groups. For our signature construction we exhibit a class of parametric distributions φ_ℓ and show that any family of groups that is adaptive pseudo-free w.r.t. $\varphi \in \varphi_\ell$ immediately yields a signature scheme that is strongly-unforgeable under chosen-message attack. We also explain how to adapt the distribution and the proof to obtain the analogous result for (non-strongly) unforgeable schemes.

5.1 Signatures from adaptive pseudo-free groups

The class of parametric distributions φ_ℓ. In this section we introduce a specific class of parametric distributions $\varphi_\ell : \{0,1\}^\ell \to \mathbb{Z}^{1+m} \times \{0,1\}^{a(\ell)}$.

For any input $M \in \{0,1\}^\ell$ and an integer ℓ, $\varphi_\ell(M)$ outputs a tuple $(e, \boldsymbol{s}, r)$ such that:

- r is a binary string taken according to some arbitrary distribution D_r;
- $e = H(r)$ where $H : \{0,1\}^{a(\ell)} \to \{0,1\}^{b(\ell)}$ is a division intractable function (see Section 2) and $a(\cdot)$ and $b(\cdot)$ are polynomials;
- $s_1 = 1$;
- $s_i \in \mathbb{Z}_e$ (i.e. $s_i < e$) $\forall i = 2, \ldots, m$ for some efficiently samplable distribution D_{s_i}.

Also we require that $\varphi_\ell(M)$ produces an output $(e, \boldsymbol{s}, r)$ for which one can efficiently tell that it belongs to the support of $\varphi_\ell(M)$. Formally, we require that φ_ℓ is equipped with an efficient algorithm $Ver_{\varphi_\ell}(\cdot,\cdot,\cdot,\cdot)$ that, on input $(e, \boldsymbol{s}, r, M)$, outputs 1 if $(e, \boldsymbol{s}, r)$ is in the support of $\varphi_\ell(M)$ and 0 otherwise. Moreover we require $Ver_{\varphi_\ell}(e, \boldsymbol{s}, r, M)$ to be such that, for all PPT adversaries $\mathcal{A}$ the probability

$$\Pr\left[(e, \boldsymbol{s}, r, M_1, M_2) \leftarrow \mathcal{A}(\varphi_\ell) : M_1 \neq M_2 \wedge Ver_{\varphi_\ell}(e, \boldsymbol{s}, r, M_1) = 1 \wedge Ver_{\varphi_\ell}(e, \boldsymbol{s}, r, M_2) = 1\right]$$

is at most negligible.

Signature scheme construction. We now show how to build a signature scheme from any family of groups $\mathcal{G}$ that is adaptive pseudo-free w.r.t. $\hat{\varphi} \in \varphi_\ell$.

Let $\hat{\varphi}$ be a parametric distribution taken from the class φ_ℓ and let $\mathcal{G}$ be a family of groups that is adaptive pseudo-free w.r.t. $\hat{\varphi}$. Then we have the following signature scheme $\mathsf{PFSig} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$:

$\mathsf{KG}(1^k)$ Let $A = \{a_1, \ldots, a_m\}$ and $X = \{x\}$ be the sets of constants variable symbols. The key generation algorithm selects a random group $\mathbb{G}$ from $\mathcal{G}$, fixes an assignment $\alpha : A \to \mathbb{G}$ for the symbols in A and finally it sets $\mathsf{vk} = (X, A, \alpha, \mathbb{G}, \hat{\varphi})$ as the public verification key and $\mathsf{sk} = ord(\mathbb{G})$ as the secret signing key. The input space of $\hat{\varphi}$, $\mathcal{M}$, is taken as the message space of the signature scheme.

$\mathsf{Sign}(\mathsf{sk}, M)$ The signing algorithm proceeds as follows:
- $(e, \boldsymbol{s}, r) \leftarrow \hat{\varphi}(M)$
- Use $ord(\mathbb{G})$ to solve the equation $x^e = a_1^{s_1} \cdots a_m^{s_m}$. Let $\psi : X \to \mathbb{G}$ be the satisfying assignment for x. The algorithm outputs $\sigma = (e, \boldsymbol{s}, r, \psi)$ as the signature for M.

$\mathsf{Ver}(\mathsf{vk}, M, \sigma)$ To verify a signature σ for a message M, the verification algorithm proceeds as follows:
- Check if $Ver_{\hat{\varphi}}(e, \boldsymbol{s}, r, M) = 1$ and if the equation $x^e = a_1^{s_1} \cdots a_m^{s_m}$ is satisfied in $\mathbb{G}$ by $\psi(x)$.
- If both the checks are true, output 1, otherwise 0.

Security of the signature scheme. In this section we prove the security of the proposed signature scheme under the assumption that $\mathcal{G}$ is adaptive pseudo-free w.r.t. $\hat{\varphi}$. In particular we can state the following theorem:

Theorem 1. *If $\mathcal{G}$ is a family of adaptive pseudo-free groups w.r.t. distribution $\hat{\varphi} \in \varphi_\ell$, then the signature scheme* PFSig *is strongly-unforgeable under chosen-message attack.*

Proof. For sake of contradiction, assume there exists an adversary $\mathcal{A}$ that is able to break the security of PFSig with non-negligible probability. Then we can build a simulator algorithm $\mathcal{B}$ that is able to break adaptive pseudo-freeness of $\mathcal{G}$ w.r.t. $\hat{\varphi}$.

Let X and A be the sets of variable and constant symbols. At the beginning of the game $\mathcal{B}$ receives $(\alpha, \mathbb{G})$ and the description of $\hat{\varphi}$ from its challenger. It sets $vk = (X, A, \alpha, \mathbb{G})$ and runs $\mathcal{A}$ on input vk.

Whenever $\mathcal{A}$ asks for a signature on a message $M_i \in \mathcal{M}$, $\mathcal{B}$ hands M_i to its challenger and gets back $(e_i, \boldsymbol{s}^i, r_i, \psi_i)$ where $(e_i, \boldsymbol{s}^i, r_i)$ is taken from $\hat{\varphi}(M_i)$ (i.e. $Ver_{\hat{\varphi}}(e_i, \boldsymbol{s}^i, r_i, M_i) = 1$) and ψ_i is a valid solution for the equation λ_i defined by the exponents $(e_i, \boldsymbol{s}^i)$. $\mathcal{B}$ gives $\sigma_i = (e_i, \boldsymbol{s}^i, r_i, \psi_i)$ as a signature for the message M_i. It is easy to see that σ_i are valid signatures and that they are distributed as in the real case.

In the end $\mathcal{A}$ is supposed to output a valid forgery (M^*, σ^*) (i.e. it holds that $(M^*, \sigma^*) \neq (M_i, \sigma_i)\ \forall i = 1, \ldots, t$ where t is the number of queries made by the adversary). Finally $\mathcal{B}$ outputs $\sigma^* = (e^*, \boldsymbol{s}^*, r^*, \psi^*)$ to its challenger.

Since (M^*, σ^*) is a valid forgery, we have that ψ^* is a solution for the equation $x^{e^*} = a_1^{s_1^*} \cdots a_m^{s_m^*}$ and that $Ver_{\hat{\varphi}}(e^*, \boldsymbol{s}^*, r^*, M^*) = 1$. To conclude the proof of security it remains to show that the equation $(e^*, \boldsymbol{s}^*)$ is non-trivial.

More precisely, we will prove the following lemma.

Lemma 4. *Let $(M^*, \sigma^*) = (M^*, (e^*, \boldsymbol{s}^*, r^*, \psi^*))$ be a valid forgery for the scheme* PFSig *w.r.t. to the set $\{(M_i, \sigma_i)\}_{i=1}^t$ of previously issued signatures, then the equation defined by $(e^*, \boldsymbol{s}^*)$ is* non-trivial *w.r.t. to the set of equations $\Lambda = \{(e_i, \boldsymbol{s}^i)\}_{i=1}^t$.*

Proof (Lemma 4). According to Proposition 2 (and for properly defined Σ, E) we want to show that

$$\forall k \in \mathbb{Z}^t, V \in \mathbb{Z}^m : e^*(\Sigma E k + V) \neq \boldsymbol{s}^*.$$

For sake of contradiction, assume there exist $\hat{k} \in \mathbb{Z}^t$ and $\hat{V} \in \mathbb{Z}^m$ such that $e^*(\Sigma E \hat{k} + \hat{V}) = \boldsymbol{s}^*$. Then we show that this contradicts at least one of our assumptions.

Let $P = \prod_{i=1}^t e_i$ and ρ_j be the j-th row of $(\Sigma E \hat{k})$:

$$\rho_j = \frac{s_j^1 \hat{k}_1}{e_1} + \ldots + \frac{s_j^t \hat{k}_t}{e_t} = \frac{\sum_{l=1}^t (s_j^l \hat{k}_l \prod_{i \neq l} e_i)}{P}.$$

$\forall j = 1, \ldots, m$ it holds $e^* \rho_j = s_j^* - e^* \hat{V}_j$ or equivalently

$$\frac{s_j^* P}{e^*} = \sum_{l=1}^{t} (s_j^l \hat{k}_l \prod_{i \neq l} e_i) + \hat{V}_j P. \tag{3}$$

Since both $\left(\sum_{l=1}^{t} s_j^l \hat{k}_l \prod_{i \neq l} e_i\right)$ and $(\hat{V}_j P)$ are integers, then $\frac{s_j^* P}{e^*}$ must be an integer too. In particular this must hold even for $j = 1$ and thus it must be that $e^* \mid P$ (as $s_1^* = 1$).

Then we can have different cases that contradict our assumptions:

- $e^* \mid P$ and $r^* \neq r_j$. This contradicts that H (in φ_ℓ) is division intractable.
- $e^* \mid P$ and $r^* = r_j$ (i.e. $e^* = e_j$). In this case, $\forall i = 1, \ldots, m$ we have

$$s_i^* = s_i^j \hat{k}_j + e^* \left(\frac{\sum_{l=1, l \neq j}^{t} s_i^l \hat{k}_l (\prod_{i \neq l} e_i)}{P} + \hat{V}_i \right)$$

  from which $\boldsymbol{s}^* = \boldsymbol{s}^j \hat{k}_j \bmod e^*$. For any choices of $\hat{k}_i$, $i \neq j$, the last equation is satisfied for $\hat{k}_j = 1 \bmod e^*$ (as $s_1^* = s_1^j = 1$) and thus $\boldsymbol{s}^* = \boldsymbol{s}^j$ (since $\boldsymbol{s}^*, \boldsymbol{s}^j \in \mathbb{Z}_{e^*}^m$). This means that in this case we have $(e^*, \boldsymbol{s}^*, r^*) = (e_j, \boldsymbol{s}^j, r_j)$.
  Then we can have two different subcases:
  - $M^* \neq M_j$. This contradicts the security property on the verification algorithm of $\hat{\varphi}$.
  - $M^* = M_j$. This contradicts that (M^*, σ^*) is a forgery.

Notice that if one relaxes a bit the requirements on the parametric distribution $\hat{\varphi}$, Theorems 1 leads to different flavors of digital signature schemes. For instance, one might consider the distribution $\hat{\varphi}'$, which slightly generalizes the parametric distribution $\hat{\varphi}$ as follows. $\hat{\varphi}'$ is exactly as $\hat{\varphi}$ with the only difference that s_2 is chosen unformly in $\mathbb{Z}_B$ for some value $B > e$. It is easy to rewrite the proof of Theorem 1 in order to show the following

Corollary 1. *If $\mathcal{G}$ is a family of adaptive pseudo-free groups w.r.t. distribution $\hat{\varphi}'$, then the signature scheme* PFSig *is unforgeable under chosen-message attack.*

Informally, what this corollary is saying is that by (slightly) generalizing the parametric distribution one gets a signature scheme where unforgeability is guaranteed only for previously unsigned messages (i.e. the scheme is not strongly unforgeable).

5.2 Network coding signatures from adaptive pseudo-free groups

In this section we show that our framework allows to encompass network coding signature schemes as defined and constructed by [6, 14]. In particular, by combining previous theorems with ideas from [14] we construct the first RSA-based network coding homomorphic signature scheme provably secure without random oracle. In the following we will represent files V to be signed as collections $(v^{(1)}, \ldots, v^{(m)})$ where each $v^{(i)}$ is a n-dimensional vector of the form $(v_1, \ldots, v_n)$. To sign V the signer signs every single vector $v^{(i)}$ separately. Informally this is done using a signature scheme that allows some form of (controlled) malleability. In this way, if we interpret signatures as solutions of non trivial equations, one can easily compute solutions for any linear combination of the given equations. This simple observation, when combined with ideas from [14], can be used to construct a secure signature scheme for network coding without random oracles.

BACKGROUND ON LINEAR CODING SCHEMES. In linear network coding [2, 21], a file to be transmitted is viewed as an ordered sequence of n-dimensional vectors $v_1, ...v_m$ (defined over the integers or over some finite field). Before transmission, the source node creates the m augmented vectors $w_1, \ldots, w_m$ obtained by prepending to v_i a vector u_i, of length m. Each u_i contains a 1 in ith position and 0 in all the remaining positions (m is typically much smaller than n). These augmented vectors are then sent by the source as packets in the network. Each node in the network processes packets as follows. When receiving $w_1 \ldots w_m$, a node computes some linear combination of the received packets (e.g., using coefficients randomly chosen from a suitable domain) and transmits the resulting vector on its outgoing edges. In other words, each node transmits a linear combination of the vectors it receives. To recover the original file a node must receive m (valid) vectors w_i of the form described above, for which the corresponding u_i's are linearly independent. Thus, denoting with U the matrix whose rows are $u_1, \ldots, u_m$ and V the matrix whose rows are $v_1, \ldots, v_m$ the original message can be retrieved as

$$M = U^{-1}V$$

The idea sketched above is susceptible to pollution attacks where malicious nodes inject invalid vectors in the network so that to make reconstruction of the original file impossible. To overcome this problem a viable solution is to use *network coding signatures*. The basic requirement of such schemes is that they allow to efficiently check if a given vector is valid, i.e. if it has been obtained as linear combination of valid vectors $w_1, \ldots w_k$. More details about network signatures can be found in [6, 14]. We recall the formal definitions in Appendix B.

Our Network Coding Signature Scheme Here we describe our network coding signature scheme. First, however, we discuss some additional details required to properly present the scheme. As already mentioned, a file to be signed is expressed as a set of vectors $(v^{(1)}, \ldots, v^{(m)})$ of n components each. Such vectors will be prepended with m unitary vectors $u^{(i)}$ (of m components each). Let us denote with $w^{(i)}$ the resulting vectors.

Using a similar notation as [14] we denote with $Q = \{0, \ldots, q-1\}$ (for some prime q) the set from which coefficients are (randomly) sampled. We denote with L an upper bound on the path length from the source to any target. By these positions $B = mq^L$ denotes the largest possible value of u-coordinates in (honestly-generated) vectors. Moreover denoting with M an upper bound on the magnitude of the coordinates of initial vectors $v^{(1)}, \ldots, v^{(m)}$, we set $B^* = MB$.

Let φ_N be the following parametric distribution. It takes as input some random identifier fid, a vector space V and a bound B^*. Let ℓ_s be a security parameter and ℓ be an integer such that $2^\ell > B^*$, compute $e = H(\mathsf{fid})$ where $H : \{0,1\}^* \to \{0,1\}^\ell$ is a division intractable function. Next, for each $v^{(i)} = (v_1^{(i)}, \ldots, v_n^{(i)}) \in V$ it proceeds as follows. First, it samples (uniformly and at random) a $\ell + \ell_s$-bit random integer s_i and outputs $(s_i, u^{(i)}, v^{(i)})$. The global output of φ_N is then

$$(e, \{(s_i, u^{(i)}, v^{(i)})\}_{i=1}^m)$$

Notice that φ_N is a simple extension of distribution $\hat{\varphi}'$ described above. It is straightforward to show that it fits the requirements of corollary 1 as well.

Let $\mathcal{G}$ be a family of groups that is adaptive pseudo-free w.r.t. φ_N. Then we have the following signature scheme $\mathsf{NetPFSig} = (\mathsf{NetKG}, \mathsf{NetSign}, \mathsf{NetVer})$:

$\mathsf{NetKG}(1^k, n)$ Let $A = \{g, g_1, \ldots, g_n, h_1, \ldots, h_m\}$ and $X = \{x\}$ be the sets of constants variable symbols. The key generation algorithm selects a random group $\mathbb{G}$ from $\mathcal{G}$, fixes an assignment

$\alpha : A \to \mathbb{G}$ for the symbols in A and finally it sets $\mathsf{vk} = (X, A, \alpha, \mathbb{G}, \varphi_N)$ as the public verification key and $\mathsf{sk} = ord(\mathbb{G})$ as the secret signing key. The input space of φ_N, $\mathcal{M}$, is taken as the set of m-dimensional vectors whose components are positive integers of magnitude at most M.

$\mathsf{Sign}(\mathsf{sk}, V)$ The signing algorithm proceeds as follows. A random identifier fid for the vector space V is chosen. Next, it runs $\varphi_N(V, B^*, \mathsf{fid})$ to get back $(e, \{(s_i, u^{(i)}, v^{(i)})\}_{i=1}^m)$. Finally, for $i = 1$ to m, it uses $ord(\mathbb{G})$ to solve the equation

$$x_i^e = g^{s_i} \prod_{j=1}^{m} h_j^{u_j^{(i)}} \prod_{j=1}^{n} g_j^{v_j^{(i)}}$$

Let $\psi : X \to \mathbb{G}$ be the satisfying assignment for x_i and $\sigma_i = (e, s_i, u^{(i)}, v^{(i)}, \mathsf{fid}, \psi)$ the signature for $w^{(i)}$ The algorithm outputs $\sigma = (\sigma_1, \ldots \sigma_m)$ as the signature for V.

$\mathsf{Ver}(\mathsf{vk}, V, \sigma)$ To verify a signature σ for a vector space V, the verification algorithm proceeds as follows

- Check if $Ver_{\varphi_N}(e, V, B^*, \mathsf{fid}, \{(s_i, u^{(i)}, v^{(i)})\}_{i=1}^m) = 1$,[5] and if the equations $x_i^e = g^{s_i} g_1^{v_1^{(i)}} \cdots g_n^{v_n^{(i)}} h_1^{u^{(i)}} \cdots h_m^{u_m^{(i)}}$ are all satisfied in $\mathbb{G}$ by $\psi(x_i)$.
- If all the checks are true, output 1, otherwise 0.

$\mathsf{Combine}(\mathsf{vk}, \mathsf{fid}, w_1, \ldots, w_\ell, \sigma_1, \ldots, \sigma_\ell)$ To combine signatures σ_i, corresponding to vectors w_i sharing the same fid, a node proceeds as follows.

- It discards any w_i having u coordinates negative or larger than $B/(mq)$, or having v coordinates negative or larger than $B^*/(mq)$. Without loss of generality we keep calling $w_1, \ldots w_\ell$ the remaining vectors.
- It chooses random $\alpha_1, \ldots \alpha_\ell \in Q$, set $w = \sum_{i=1}^{\ell} \alpha_i w_i$ and it outputs the signature $\sigma = (e, s, w, \mathsf{fid}, \psi)$ on w which is obtained by computing

$$\psi = \prod_{i=1}^{\ell} \psi_i^{\alpha_i}, \qquad s = \sum_{i=1}^{\ell} \alpha_i s_i$$

One can easily rewrite the proof of corollary 1 to prove the following.

Theorem 2. *If $\mathcal{G}$ is a family of adaptive pseudo-free groups w.r.t. distribution φ_N, then the* NetPFSig *signature scheme described above is a secure (homomorphic) network coding signature.*

6 The RSA group is adaptive pseudo-free

In Section 4 we have defined the notion of adaptive pseudo-free groups and in Section 5 have showed a class of parametric distributions (called φ_ℓ) that allows to build signatures from the sole assumption that a family of groups is adaptive pseudo-free w.r.t. $\hat{\varphi} \in \varphi_\ell$. At this stage, it is therefore interesting to find a computational group candidate to be proved adaptive pseudo-free. As proved by Micciancio in [18], the only group that we know to be pseudo-free is the RSA group $\mathbb{Z}_N^*$ of integers modulo N, where N is the product of two "safe" primes and the sampling procedure takes elements from QR_N. Therefore we aim to prove adaptive pseudo-freeness for the same group.

[5] We implicitly assume that the Ver_{φ_N} verification algorithm rejects immediately if any of the u coordinates is negative or larger than B, or if any of the v coordinates is negative or larger than B^*

A PARAMETRIC DISTRIBUTION $\hat{\varphi}$. First of all we need to define the specific parametric distribution for which we will prove adaptive pseudo-freeness of the RSA group.

Let us consider the following $\hat{\varphi} : \mathcal{M} \to \mathbb{Z} \times \mathbb{Z}^m \times \{0,1\}^*$, where $\mathcal{M} = \{0,1\}^\ell$. For any input $M \in \mathcal{M}$, $\hat{\varphi}(M)$ outputs a tuple $(e, \boldsymbol{s}, r)$ that is defined as follows:

- r is a random binary string
- $e = H(r)$ where $H : \{0,1\}^* \to \{0,1\}^\ell$ is a division intractable function (see definition in Section 2)
- $s_1 = 1$
- s_2 is uniformly distributed in $\mathbb{Z}_e$
- For $3 \leq i \leq m$, each s_i is taken with an arbitrary (but efficiently samplable) distribution D_{s_i} in $\mathbb{Z}_e$ such that the tuple $s_3, \ldots, s_m$ is binding to M[6].

The verification algorithm $Ver_{\hat{\varphi}}(e, \boldsymbol{s}, r, M)$ checks that $e = H(r)$ and that $s_3, \ldots, s_m$ are binding w.r.t. M. It is straightforward to verify that $\hat{\varphi}$ is contained in the class φ_ℓ defined in section 5.1.

We state the following theorem.

Theorem 3. *If the Strong-RSA Assumption holds, then $\mathbb{Z}_N^*$ is adaptive pseudo-free w.r.t. $\hat{\varphi}$.*

Proof. For sake of contradiction, we assume that $\mathbb{Z}_N^*$ is *not* adaptive pseudo-free w.r.t. $\hat{\varphi}$. According to Definition 8, this means that there exists an efficient PPT adversary $\mathcal{A}$ that with non-negligible probability is able to output an equation λ^* (defined by $(e^*, \boldsymbol{s^*})$) together with a solution ψ^* such that λ^* is non-trivial w.r.t. to the set Λ of previously queried equations. In order to prove the theorem we will show that we can build an algorithm $\mathcal{B}$ out of $\mathcal{A}$ that breaks the Strong-RSA Assumption (more precisely its variant where $\tau \in QR_N$).

For $i = 1$ to t (where t is the number of queries made by $\mathcal{A}$), let $(e_i, \boldsymbol{s}^i, r_i) \leftarrow \hat{\varphi}(M_i)$.

If we consider e^* and the set $\{e_1, \ldots, e_t\}$ we can distinguish two types of adversaries:

Type I the adversary outputs e^* such that $e* \nmid \prod_{i=1}^t e_i$,
Type II the adversary outputs e^* such that $e* \mid \prod_{i=1}^t e_i$.

At the beginning of the game we guess on the type of adversary we have and will set up the proper simulation according to such guess. Notice that the guess will be right with probability at least $1/2$.

Type I. In the case of a Type I adversary we show how to build a simulator $\mathcal{B}$ that breaks Strong-RSA with non-negligible probability. $\mathcal{B}$ takes as input (N, τ) where N is the product of two safe primes p, q (where $p = 2p' + 1$ and $q = 2q' + 1$) and $\tau \in QR_N$. Its goal is to find an e-th root y of τ for e of its choice.

In the following we describe the simulator $\mathcal{B}$ during the three phases of the game.

Setup $\mathcal{B}$ chooses in advance t random strings $r_1, \ldots, r_t$ and computes $e_i = H(r_i)$ $\forall 1 = 1, \ldots, t$.

Then it fixes the assignment α for the constant symbols as follows:

- pick random $z_1, z_2, \ldots, z_m \xleftarrow{\$} \{1, \ldots, N^2\}$
- let $E = \prod_{i=1}^t e_i$ and set $\alpha(a_1) = \tau^{Ez_1}$ and $\alpha(a_i) = \alpha(a_1)^{z_i}$ for all $i = 2$ to m.

[6] This means that there exists an efficient algorithm that on input $(M, s_3, \ldots, s_m)$ outputs 1 if $s_3, \ldots, s_m$ are created w.r.t. M

Finally $\mathcal{B}$ gives α (and the description of $\mathbb{Z}_N^*$) to the adversary $\mathcal{A}$.

For ease of exposition we will use a_i instead of $\alpha(a_i)$ to refer group elements. For all $2 \leq i \leq m$ let $z_i = b_i p'q' + c_i$ where $0 \leq c_i < p'q'$. Since each z_i is chosen from a suitably large interval, the distributions of each $(z_i \bmod p'q')$ is statistically indistinguishable from the uniform distribution over $\mathbb{Z}_{p'q'}$. So $a_1, a_2, \ldots, a_m$ are distributed like random quadratic residues of $\mathbb{Z}_N^*$. Moreover the conditional distribution of b_i given c_i is statistically indistinguishable from the uniform distribution over $\{0, \ldots, \lfloor N^2/p'q' \rfloor\}$.

Equations queries At this stage $\mathcal{A}$ is allowed to adaptively query equations by submitting parameters $M^1, \ldots, M^t$ for $\hat{\varphi}$. Therefore $\mathcal{B}$ has to solve such equations and give the corresponding solutions to $\mathcal{A}$. For all $i \in \{1, \ldots, t\}$, each query M^i is managed as follows. $\mathcal{B}$ chooses the exponents $s_2^i, \ldots, s_m^i \in \mathbb{Z}_{e_i}$ according to $\hat{\varphi}(M^i)$. Then $\mathcal{B}$ computes the solution of $\lambda_i \equiv x^{e_i} = a_1 \cdot a_2^{s_2^i} \cdots a_m^{s_m^i}$ as follows:

- let $E_i = \prod_{j=1, j \neq i}^{t} e_j$
- $\psi_i(x) = (\tau^{E_i})^{z_1 + \sum_{j=2}^{m} s_j^i z_j}$

Finally $\mathcal{B}$ gives $(e_i, \boldsymbol{s}^i, r_i, \psi_i)$ to $\mathcal{A}$. It is easy to see that ψ_i is a valid solution for λ_i and that the equations are distributed as in the real case.

Challenge Once the previous phase is over, $\mathcal{A}$ is supposed to output an equation λ^*, for M^* (together with a solution ψ^*) which is non-trivial w.r.t. $\Lambda = \{\lambda_i\}_{i=1}^t$. Since $(e^*, \boldsymbol{s^*}, r^*)$ are distributed according to $\hat{\varphi}(M^*)$ we have:

$$\psi^*(x)^{e^*} = a_1 a_2^{s_2^*} \cdots a_m^{s_m^*} = \tau^{E(z_1 + \sum_{j=2}^{m} z_j s_j^*)}.$$

Let $E' = E(z_1 + \sum_{j=2}^{m} z_j s_j^*)$ and $d = gcd(e^*, E')$. Provided that $e^* \nmid E'$ $\mathcal{B}$ can use standard techniques (i.e. Shamir's trick) to extract an (e^*/d)-th root y of τ and thus it can output $(e^*/d, y)$ to break Strong-RSA.

Therefore we are left with the task of showing that $e^* \nmid E'$ with non-negligible probability. Let r be a prime dividing e^*. Since we are assuming a Type I adversary it holds $r \nmid E$. Thus the point is to show that $r \nmid (z_1 + \sum_{j=2}^{m} z_j s_j^*)$ with non-negligible probability.

As pointed out before, let $z_i = b_i p'q' + c_i$. Since each b_i is essentially hidden to the view of any adversary, r may depend only on the c_i's. Since $r \nmid p'q'$ the probability that $r \mid (z_1 + z_2 s_2^* + \ldots + z_m s_m^*)$, or equivalently $(z_1 + z_2 s_2^* + \ldots + z_m s_m^*) = 0 \bmod r$, is close to $1/r$. This means that $e^* \nmid E'$ with probability close to $1 - 1/r$, for the smallest prime factor r of e^*.

Type II. The case of a Type II adversary is a bit more complicated. Since $e* \mid \prod_{i=1}^{t} e_i$ we can have two cases:

1. $r^* \neq r_i \; \forall i = 1, \ldots, t$. In this case it is easy to see that our assumption on $\hat{\varphi}$ is not satisfied as we would be able to break the division intractability of the function H. Indeed we have $(r_1, \ldots, r_t)$ and $r^* \neq r_i, \forall i = 1, \ldots, t$ such that $H(r^*) = e^* \mid \prod_{i=1}^{t} e_i$ (where $e_i = H(r_i)$).
2. $r^* = r_j$ for some $j \in \{1, \ldots, t\}$ (i.e. $e^* = e_j$). The simulation for this case is described below. Precisely we will show how to build an algorithm $\mathcal{B}$ that breaks Strong-RSA with non-negligible probability.

Before giving the details of the simulation we first give some intuitions that will be useful to understand our approach.

Let $\{(e_i, \boldsymbol{s}^i)\}_{i=1}^t$ be the exponents of the t queried equations and $(e^*, \boldsymbol{s}^*)$ be the ones of λ^*. Since λ^* is non-trivial we have that $\forall k \in \mathbb{Z}^t$ and $\forall V \in \mathbb{Z}^m$:

$$e^* \left(\begin{bmatrix} 1 & 1 & \cdots & 1 \\ s_2^1 & s_2^2 & \cdots & s_2^t \\ \vdots & & & \vdots \\ s_m^1 & s_m^2 & \cdots & s_m^t \end{bmatrix} \begin{bmatrix} k_1/e_1 \\ k_2/e_2 \\ \vdots \\ k_t/e_t \end{bmatrix} + \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_m \end{bmatrix} \right) \neq \begin{bmatrix} 1 \\ s_2^* \\ \vdots \\ s_m^* \end{bmatrix}.$$

Namely, at least one of the following m inequalities must hold:

1. $e^*(k_1 e_2 \cdots e_t + \ldots + e_1 e_2 \cdots e_{t-1} k_t) \neq (1 - V_1 e^*)(e_1 \cdots e_t)$
2. $e^*(s_2^1 k_1 e_2 \cdots e_t + \ldots + s_2^t k_t e_1 e_2 \cdots e_{t-1}) \neq (s_2^* - V_2 e^*)(e_1 \cdots e_t)$

   $\vdots$

m. $e^*(s_m^1 k_1 e_2 \cdots e_t + \ldots + e_1 e_2 \cdots e_{t-1} s_m^t k_t) \neq (s_m^* - V_m e^*)(e_1 \cdots e_t)$

Since the fact above holds for *all* integer vectors $k \in \mathbb{Z}^t$ and $V \in \mathbb{Z}^m$, then it must hold even for $\hat{k}$ and $\hat{V}$ such that: $\hat{k}_j = 1$, $\hat{k}_i = 0\ \forall i \neq j$ and $\hat{V} = 0^m$.

In particular, for such choices of k and V, wlog we assume that the ν-th inequality holds. Since we are in the case that $e^* = e_j$, observe that the first equation is always satisfied for such $\hat{k}$ and $\hat{V}$. Thus it must hold $s_\nu^j \neq s_\nu^*$ for some $\nu \in \{2, \ldots, m\}$.

$\mathcal{B}$ can guess j and ν with non-negligible probability $1/(t(m-1))$ by picking them at random in $\{1, \ldots, t\}$ and $\{2, \ldots, m\}$ respectively. Then it performs the following simulation.

Setup $\mathcal{B}$ chooses $r_1, \ldots, r_t$ and computes $e_i = H(r_i)\ \forall i = 1, \ldots, t$. Then $\mathcal{B}$ picks random $u_1, \ldots, u_m \xleftarrow{\$} QR_N$, $z_\nu, \beta \xleftarrow{\$} \{1, \ldots, N^2\}$. and fixes the assignment for the constant symbols as follows: $\alpha(a_2) = \tau^{\prod_{i=1, i\neq j}^t e_i}$, $\alpha(a_\nu) = \alpha(a_2)^{z_\nu}$, $\alpha(a_1) = \alpha(a_2)^{-\beta} u_1^{\prod_{i=1}^t e_i}$ and $\alpha(a_i) = u_i^{\prod_{l=1}^t e_l}$ for $i = 3$ to m and $i \neq \nu$. Finally it gives α and the description of the group $\mathbb{Z}_N^*$ to $\mathcal{A}$.

For ease of exposition, in the following we will use a_i instead of $\alpha(a_i)$ to refer group elements.

Solving equations In this phase $\mathcal{B}$ is adaptively asked by $\mathcal{A}$ to solve at most t equations with parameters $M^1, \ldots, M^t$ respectively. For each parameter M^i, $\mathcal{B}$ chooses $s_2^i, \ldots, s_m^i$ according to $\hat{\varphi}(M^i)$. For all $i \in \{1, \ldots, t\} \backslash \{j\}$ $\mathcal{B}$ solves $\lambda_i \equiv x^{e_i} = a_1 a_2^{s_2^i} \cdots a_m^{s_m^i}$ by computing

$$\psi_i(x) = (\tau^{\prod_{l\neq i,j} e_l})^{1 + z_\nu s_\nu^i - \beta} \left(\prod_{j=1, j\neq 2, \nu}^m u_j^{s_j^i} \right)^{\prod_{l\neq i,j} e_l}.$$

It is easy to observe that ψ_i is a valid solution for λ_i.

In order to solve the j-th equation $\mathcal{B}$ uses a different approach. Let M^j be the queried parameter and $s_3^j, \ldots, s_m^j$ be chosen according to M^j. $\mathcal{B}$ sets $s_2^j = \beta - z_\nu s_\nu^j \bmod e_j$ and find ω such that $\beta - z_\nu s_\nu^j = s_2^j + \omega e_j$. It then computes:

$$\psi_j(x) = \left(\tau^{-\omega} \prod_{i=1, i\neq 2, \nu}^m u_i^{s_i^j} \right)^{\prod_{l\neq j} e_l} = \sqrt[e_j]{a_1 a_2^{s_2^j} \cdots a_m^{s_m^j}}.$$

After having solved each equation, the simulator hands $(e_i, \boldsymbol{s}^i, r_i, \psi_i)$ to $\mathcal{A}$.

Challenge In this phase $\mathcal{A}$ is supposed to output a non-trivial equation λ^* (defined by $(e^*, \boldsymbol{s^*})$), together with a solution ψ^*. If it is the case we show that $\mathcal{B}$ can extract a root of τ as follows. Let

$$\left(\frac{\psi^*(x)}{\psi_j(x)}\right)^{e^*} = a_2^{(s_2^*-s_2^j)+z_\nu(s_\nu^*-s_\nu^j)} \prod_{i=3,i\neq\nu}^{m} a_i^{(s_i^*-s_i^j)} = (\tau^{\prod_{l\neq j} e_l})^{(s_2^*-s_2^j)+z_\nu(s_\nu^*-s_\nu^j)} \left(\prod_{i=3,i\neq\nu}^{m} u_i^{(s_i^*-s_i^j)}\right)^{(\prod_{l\neq j} e_l)e_j}.$$

Since $e^* = e_j$ we obtain:

$$\left[\left(\frac{\psi^*(x)}{\psi_j(x)}\right)\left(\prod_{i=3,i\neq\nu}^{m} u_i^{(s_i^j-s_i^*)}\right)^{(\prod_{l\neq j} e_l)}\right]^{e^*} = (\tau^{\prod_{l\neq j} e_l})^{(s_2^*-s_2^j)+z_\nu(s_\nu^*-s_\nu^j)}.$$

Let $E' = (\prod_{l\neq j} e_l)(s_2^* - s_2^j + z_\nu(s_\nu^* - s_\nu^j))$. In order to extract a root of τ we have to show that $e^* \nmid E'$ with non-negligible probability. Observe that $e^* \nmid \prod_{l\neq j} e_l$ and that $z_\nu = bp'q' + c$ where b is information theoretically hidden to any adversary. Since $s_\nu^* - s_\nu^j \neq 0$ (by our guess) and $s_2^*, s_2^j \in \mathbb{Z}_{e^*}$, we have that $e^* \mid (s_2^* - s_2^j) + z_\nu(s_\nu^* - s_\nu^j)$ only with negligible probability. Thus $\mathcal{B}$ can use standard techniques (i.e. Shamir's trick) to extract an (e^*/d)-th root y of τ where $d = gcd(e^*, E')$.

□

As a corollary of the above theorem we can prove adaptive pseudo-freeness of the RSA group w.r.t. two new parametric distributions $\hat{\varphi}_s, \hat{\varphi_{ch}} \neq \hat{\varphi}$ which still are within the class φ_ℓ defined in section 5.1. In particular $\hat{\varphi}_s$ is a variant of $\hat{\varphi}$ where: $s_2 = 0$ and for all $i = 3$ to m, $s_i \in \{0, \ldots, p\}$ such that p is at most polynomial in the security parameter (and of course $p < e$).

Corollary 2. *If the Strong-RSA Assumption holds, then $\mathbb{Z}_N^*$ is adaptive pseudo-free w.r.t. $\hat{\varphi}_s$.*

The proofs follows from that of theorem 3. The intuition here is that when the s_i's are small they can be guessed in advance with non-negligible probability.

Instead $\hat{\varphi_{ch}}$ is a variant of $\hat{\varphi}$ where: $s_2 = 0$ and $s_3, \ldots, s_m \in \mathbb{Z}_e$ are obtained as output of a chameleon hash function $CH(M; R)$ computed on the parameter M and with randomness R.

Corollary 3. *If the Strong-RSA Assumption holds, and CH is a chameleon hash function, then $\mathbb{Z}_N^*$ is adaptive pseudo-free w.r.t. $\hat{\varphi_{ch}}$.*

The proof is the same as in Corollary 2. The intuition here is that one can use the chameleon property of CH in the simulation to "prepare" the s_i's in advance.

Weak adaptive pseudo-freeness of the RSA group. One may also consider a weaker notion of adaptive pseudo-freeness where the adversary is forced to choose the parameters $M^1, \ldots, M^t$ of its queries at the beginning of the game, i.e. before receiving the description of the group from the challenger.

If we consider such a notion, then we notice that our proof of theorem 3 still holds even w.r.t. a slightly more general distribution than $\hat{\varphi}$ where the entire tuple $(e, s_2, \ldots, s_m)$ needs to be bound to M. To see this, observe that all r_i's can be still computed at the beginning of the game as the simulator now knows $M_1, \ldots, M_t$ in advance.

It is trivial to see that starting from a weak-adaptive pseudo-free group our results of section 5.1 lead to the construction of signature schemes that are weakly-secure (see Definition 4).

7 A framework for Strong RSA-based Signatures

In this section we show that, in light of the results of theorems 1 and 3, and by appropriately instantiating the parametric distribution $\hat{\varphi}$, we get *all* the known constructions of Strong RSA-based digital signatures in the standard model (to the best of our knowledge).

Cramer Shoup Signatures. Cramer-Shoup's [10] signature scheme works as follows:

Key Generation Generate N as the product of two safe primes p and q. Also randomly choose two quadratic residues $a_1, a_3 \in QR_N$ and an $(\ell+1)$-bit prime e'. The public key is (N, a_1, a_3, e) and the private key is (p, q).

Sign To sign m, compute ℓ-bit hash value $H(m)$ with a collision-resistant hash function H and then compute $c = y^{e'} a_3^{H(m)}$ for a random $y \in QR_N$. Next pick a random $(\ell+1)$-bit prime $e \neq e'$ and solve (for x) the following equation $x^e = a_1 a_3^{H(c)} \bmod N$. The signature is (y, e, x)

Verification Check that the two equations above hold and that e is an $\ell+1$-bit (odd) integer different from e'.

While the signature above may look like based on a system of two equations, we observe that only for the second equation the signing process is required to find a solution (using the secret key) while the first equation (i.e. $c = y^{e'} a_3^{H(m)}$) is, *de facto*, a chameleon hash function computed on the message m and randomness y. In particular it is a chameleon hash based on the RSA assumption which, for efficiency, is implemented by sharing some parameters with the signature scheme. Therefore we can see Cramer-Shoup's scheme as a special case of our general framework when considering the following distribution.

φ^{CS} Choose r at random and set $e = H'(r)$ (where $H' : \{0,1\}^* \to \{0,1\}^{\ell+1}$ is a function that maps into primes of length $\ell+1$)
Let $c = CH(m; y)$ (CH is a chameleon hash function) and set $s_1 = 1$ and $s_3 = H(c)$ (H is a collision resistant hash function) All the remaining s_i's are set to 0.

It is easy to check that φ^{CS} is a special instantiation of $\hat{\varphi_{ch}}$, and so the security of the scheme is implied by Corollary 3.

Fischlin Signatures. Fischlin's [12] signature scheme can be seen as a simplification of Cramer-Shoup signature. The scheme works as follows:

Key Generation Generate N as the product of two safe primes p and q. Also randomly choose three quadratic residues $a_1, a_2, a_3 \in QR_N$. The public key is (N, a_1, a_2, a_3) and the private key is (p, q).

Sign To sign m compute the ℓ-bit hash value $H(m)$ with a collision-resistant hash function H. Next output a random $(\ell+1)$-bit prime e, a random ℓ-bit integer α and solve (for x) the following equation $x^e = a_1 a_2^{\alpha} a_3^{\alpha \oplus H(m)} \bmod N$. The signature is (e, x, α)

Verification Check that the equation above holds, that e is an $\ell+1$-bit (odd) integer and that α is an ℓ bit value.

The signature above can be seen as a special case of our general framework when considering the following distribution.

φ^{Fis} Choose r at random and set $e = H'(r)$ (where $H' : \{0,1\}^* \to \{0,1\}^{\ell+1}$ is a function that maps into primes of length $\ell + 1$)
Let $\alpha \in_R \{0,1\}^\ell$ and set $s_1 = 1$, $s_2 = \alpha$ and $s_3 = \alpha \oplus H(m)$ (H is a collision resistant hash function) All the remaining s_i's are set to 0.

It is easy to check that φ^{Fis} is a special instantiation of $\hat{\varphi}$.

Camenisch-Lysyanskaya Signatures. The scheme by Camenisch and Lysyanskaya [8] scheme works as follows

Key Generation Generate N as the product of two safe primes p and q. Also randomly choose three quadratic residues $a_1, a_2, a_3 \in QR_N$. The public key is (N, a_1, a_2, a_3) and the private key is (p, q).

Sign To sign m of length ℓ_m output a random $(\ell_m + 2)$-bit prime e, a random ℓ-bit integer s of length $\ell_s = |N| + \ell_m + \ell$ where ℓ is a security parameter and solve (for x) the following equation $x^e = a_1 a_2^s a_3^m \bmod N$. The signature is (e, x, s)

Verification Check that the the equation above holds and that e and s are of appropriate length.

The signature above can be seen as a special case of our general framework when considering the following distribution φ^{CL} (which is a special instantiation of $\hat{\varphi}'$) and Corollary 1.

φ^{CL} Choose r at random and set $e = H'(r)$ (where $H' : \{0,1\}^* \to \{0,1\}^{\ell+1}$ is a function that maps into primes of length $\ell + 1$)
Let $s \in_R \mathbb{Z}_B$ where $B > e$ is some bound of size at most ℓ_s and set $s_1 = 1$, $s_2 = s$ and $s_3 = m$ (H is a collision resistant hash function) All the remaining s_i's are set to 0.

Zhu's Signatures. Zhu proposed in [22] a variation of Cramer-Shoup's signature scheme. The proof of security was found incorrect and later fixed in [23]. This signature scheme is basically the same as the one by Camenisch and Lysyanskaya described above except that s is a random string of ℓ bits.

We can show that the Zhu's scheme is a special case of our general framework when considering the following distribution.

φ^{Zhu} Choose r at random and set $e = H(r)$ (where $H : \{0,1\}^* \to \{0,1\}^{\ell+1}$ is a function that maps into primes of length $\ell + 1$)
Let $s \xleftarrow{\$} \mathbb{Z}_e$ and set $s_1 = 1$, $s_2 = s$ and $s_3 = m$. All the remaining s_i's are set to 0.

Again, it is easy to check that φ^{Zhu} is a special instantiation of $\hat{\varphi}$.

Hofheinz-Kiltz Signatures. Hofheinz and Kiltz show in [15] how to use programmable hash functions to get a new efficient signature scheme based on Strong RSA. The description follows.

Key Generation Generate N as the product of two safe primes p and q. Also randomly choose $\ell + 1$ quadratic residues $a_0, a_1, \ldots, a_\ell \in QR_N$. The message space is $\{0,1\}^\ell$. The public key is $(N, a_0, a_1, \ldots a_\ell)$ and the private key is (p, q).

Sign To sign M compute the ℓ-bit integer $m = m_1 \cdots m_\ell$ as the output of some appropriate collision resistant hash function H. Next choose a random (ℓ)-bit prime e and solve (for x) the following equation

$$x^e = a_0 \prod_{i=1}^{\ell} a_i^{m_i} \bmod N$$

The signature is (e, x)

Verification Check that the the equation above holds and that e is an ℓ-bit (odd) integer.

It is easy to notice that its security emerges from corollary 2.

Gennaro-Halevi-Rabin Signatures. In [13] it is presented an efficient signature scheme that comes in two flavors. A basic (weakly secure) signature scheme and a fully secure (slightly less efficient) one that requires chameleon hash functions [17]. Here we discuss only the first version of the scheme.

Key Generation Generate N as the product of two safe primes p and q[7]. Also randomly choose a quadratic residues $a_1 \in QR_N$. The public key is (N, a_1) and the private key is (p, q).

Sign To sign m (of arbitrary length) compute the ℓ-bit hash value $H(m)$ with a division intractable hash function H and solve (for x) the following equation $x^e = a_1 \bmod N$. The signature is (e, x)

Verification Check that the equation above holds and that $e = H(m)$.

The scheme above fits our framework for weakly-secure signature scheme (see section 6) when using the following distribution :

φ^{GHR} Choose $r = m$ and set $e = H(m)$ (where $H : \{0,1\}^* \rightarrow \{0,1\}^{\ell+1}$ is a division intractable hash function that maps into integers of length $\ell + 1$)
Set $s_1 = 1$. All the remaining s_i's are set to 0.

7.1 A new network signature from Strong RSA

It is easy to see that combining the results of Theorem 3 and Theorem 2 we obtain a concrete instantiation of the network coding signature scheme given in Section 5.2 whose security is thus based on Strong RSA in the standard model. We notice that our scheme is not as efficient as the one proposed by Gennaro *et al.* in [14], but it is secure in the standard model.

8 Conclusion

In this paper we have introduced a formal definition of adaptive pseudo-freeness. We have shown that under reasonable conditions the RSA group is adaptive pseudo-free for moduli that are products of safe primes, and exhibited the first direct cryptographic applications of adaptive pseudo-free groups: under some mild conditions, pseudo-free groups yield secure digital signature schemes. We have shown that all the RSA based signatures in the literature (to the best of our knowledge) can be seen as instantiations of our framework and furthermore we showed that our methodology yields a new network coding signature scheme in the standard model.

There are several interesting problems that we have not addressed. Here we enumerate some of them. The first obvious one, originally posed by Rivest, is what other groups used in cryptography are pseudo-free. A new construction would lead via our example to new signature schemes for example. Our results for RSA are only for univariate equations. It should be interesting to either justify this restriction through an analogue of Lemma 1 or if this is not possible, extend our study to multi-variate equations. A one-more RSA inversion problem where the adversary needs to compute the e'th root of $n + 1$ random group elements with access to only n RSA inversion queries has a

[7] In [13] this assumption is relaxed to consider safe primes or quasi-safe primes.

strong flavor of adaptive pseudo-freeness. The lack of a relation between the strong RSA problem and the one-more-RSA-inversion problem thus shows that proving general adaptive pseudo-freeness of the RSA group is difficult. Nevertheless, studying the relation between these two problems within our framework seems to be an interesting direction. Finally, we manage to prove pseudo-freeness for a large class of parametric distributions sufficient for cryptographic applications. It should be interesting to understands how far one can go with the limitations that we impose on the adversary by trying to enlarge this class.

References

1. Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 20(3):395, July 2007.
2. R. Ahlswede, Ning-Cai, S. Li, and R.W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, 2000.
3. Michael Backes, Birgit Pfitzmann, and Michael Waidner. A composable cryptographic library with nested operations. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 03*, pages 220–230, Washington D.C., USA, October 27–30, 2003. ACM Press.
4. Niko Bari and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 480–494, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Germany.
5. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.
6. Dan Boneh, David Freeman, Jonathan Katz, and Brent Waters. Signing a linear subspace: Signature schemes for network coding. In Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009*, volume 5443 of *LNCS*, pages 68–87, Irvine, CA, USA, March 18–20, 2009. Springer, Berlin, Germany.
7. Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 402–414, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.
8. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289, Amalfi, Italy, September 12–13, 2002. Springer, Berlin, Germany.
9. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145, Las Vegas, Nevada, USA, October 14–17, 2001. IEEE Computer Society Press.
10. Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. In *ACM CCS 99*, pages 46–51, Kent Ridge Digital Labs, Singapore, November 1–4, 1999. ACM Press.
11. D. Dolev and A.C. Yao. On the security of public key protocols. In *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*, pages 350–357, 1981.
12. Marc Fischlin. The Cramer-Shoup strong-RSA signature scheme revisited. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 116–129, Miami, USA, January 6–8, 2003. Springer, Berlin, Germany.
13. Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 123–139, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.
14. Rosario Gennaro, Jonathan Katz, Hugo Krawczyk, and Tal Rabin. Secure network coding over the integers. In *PKC 2010*, LNCS, pages 142–160. Springer, Berlin, Germany, 2010.
15. Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 21–38, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Berlin, Germany.
16. Susan Hohenberger. The cryptographic impact of groups with infeasible inversion. Master's thesis, Massachusetts Institute of Technology, EECS Dept., 2003.
17. Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS 2000*, San Diego, California, USA, February 2–4, 2000. The Internet Society.
18. Daniele Micciancio. The RSA group is pseudo-free. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 387–403, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

19. Daniele Micciancio and Bogdan Warinschi. Soundness of formal encryption in the presence of active adversaries. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 133–151, Cambridge, MA, USA, February 19–21, 2004. Springer, Berlin, Germany.
20. Ronald L. Rivest. On the notion of pseudo-free groups. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 505–521, Cambridge, MA, USA, February 19–21, 2004. Springer, Berlin, Germany.
21. Shuo-Yen Robert-Li, Raymond Y. Yeung, and Ning Cai. Linear network coding. *IEEE Transactions on Information Theory*, 49(2):371–381, 2003.
22. Huafei Zhu. New digital signature scheme attaining immunity to adaptive chosen-message attack. *Chinese Journal of Electronics*, 10(4):484–486, October 2001.
23. Huafei Zhu. A formal proof of zhu's signature scheme. Cryptology ePrint Archive, Report 2003/155, 2003. http://eprint.iacr.org/.

A Non-trivial multivariate equations

Here we obtain an explicit description of trivial multi-variate equations. Let

$$\Lambda = \{x_1^{e_1^k} x_2^{e_2^k} \dots x_n^{e_n^k} = a_1^{s_1^k} a_2^{s_2^k} \dots a_m^{s_m^k}\}_{k=1\dots t}$$

be a set of multivariate equations over $\mathcal{F}$, and let $\{\phi_1^k, \phi_2^k, \dots, \phi_n^k \mid k = 1 \dots t\}$ solutions for these equations.

As for the case of univariate equations we interpret these equations together with their solutions, as relations between words in the free group generated by

$$\{\phi_{1,k}, \phi_{2,k}, \dots, \phi_{n,k} \mid k = 1 \dots t\} \cup \{a_1, a_2, \dots, a_m\}.$$

Then, an equation $x_1^{e_1^*} x_2^{e_2^*} \dots x_n^{e_n^*} = a_1^{s_1^*} a_2^{s_2^*} \dots a_m^{s_m^*}$ is trivial if it has a solution over $\mathcal{F}/\equiv_\Lambda$. Assume that

$$\phi_i^* = \prod_{j=1}^{m} a_j^{v_j^i} \cdot \prod_{j=1}^{n} \prod_{l=1}^{t} \phi_{j,l}^{k_{l,j}^i}$$

is a solution for the equation (for some $v_j^i, k_{l,j}^i$ (with $1 \le j \le m, 1 \le l \le n, 1 \le i \le n$). Using the explicit characterization of $\equiv_\Lambda$ we obtain that there exist $l_1, l_2, \dots, l_t \in \mathbb{Q}$ such that:

$$\prod_{i=1}^{n} \phi_i^* = a_1^{s_1^*} a_2^{s_2^*} \dots a_m^{s_m^*} \prod_{i=1}^{t} (\prod_{j=1}^{n} \phi_{i,j}^{-e_i^j} \prod_{j=1}^{m} a_j^{s_j^i})^{l_i}$$

By replacing the expressions for ϕ_i^* in the above relation and matching the exponents of the different symbols we obtain that equation: $x_1^{e_1^*} x_2^{e_2^*} \dots x_n^{e_n^*} = a_1^{s_1^*} a_2^{s_2^*} \dots a_m^{s_m^*}$ is trivial with respect to Λ if there exist integers $v_j^i, k_{l,j}^i$ with $1 \le j \le m, 1 \le l \le n, 1 \le i \le n$ and rationals $l_1, l_2, \dots, l_t$ such that:

- For all $1 \le u \le t, 1 \le j \dots n$
$$\sum_{i=1}^{n} k_{u,j}^i e_i^* = e_j^u l_u$$
- For $1 \le j \le m$
$$\sum_{i=1}^{m} v_j^i e_i^* = s_j^* - \sum_{u=1}^{t} s_j^u l_u$$

B Network Coding Signatures

We recall the definitions of network coding signatures and network coding homomorphic signatures.

Definition 9. *A* network coding signature *is defined by a triple of algorithms (*$\mathsf{NetKG}, \mathsf{Sign}, \mathsf{Ver}$*) such that:*

$\mathsf{NetKG}(1^k, N)$ *On input the security parameter k and a parameter N, this algorithm outputs* $(\mathsf{vk}, \mathsf{sk})$ *where* sk *is the secret signing key and* vk *is the public verification key. N defines the size of the signed vectors.*

$\mathsf{Sign}(\mathsf{sk}, V, \mathsf{fid})$ *The signing algorithm takes as input the secret key* sk*, a random file identifier* fid *and an m-dimensional subspace $V \subset \mathbb{F}^N$ and outputs a signature σ.*

$\mathsf{Ver}(\mathsf{vk}, \mathsf{fid}, v, \sigma)$ *Given the public key* vk*, a file identifier* fid*, a vector $v \in \mathbb{F}^N$ and a signature σ, the algorithm outputs 0 (reject) or 1 (accept).*

For correctness, we require that for all honestly generated key pairs $(\mathsf{vk}, \mathsf{sk})$, all identifiers fid and all $V \subset \mathbb{F}^N$, if $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{fid}, V)$ then $\mathsf{Ver}(\mathsf{vk}, \mathsf{fid}, v, \sigma) = 1 \ \forall v \in V$.

A network coding signature is secure if it satisfies the following definition.

Definition 10. *Consider the following experiment between an adversary $\mathcal{A}$ and a challenger. At the beginning the adversary chooses a positive integer N and gives it to the Challenger, who runs* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{NetKG}(1^k, N)$ *and gives* vk *to $\mathcal{A}$. Then the adversary can adaptively ask for signatures on vector spaces $V_i \subset \mathbb{F}^N$ of its choice and finally $\mathcal{A}$ outputs a tuple* $(\mathsf{fid}^*, v^*, \sigma^*)$*. We say that the adversary wins if* $\mathsf{Ver}(\mathsf{vk}, \mathsf{fid}^*, v^*, \sigma^*) = 1$ *and either one of the following cases holds: (1)* $\mathsf{fid}^* \neq \mathsf{fid}_i$ *for all i; (2)* $\mathsf{fid}^* = \mathsf{fid}_i$ *for some i but $v^* \notin V_i$.*

Finally we give the formal definition of *homomorphic network coding signature*. As noticed by Boneh et al. [6] homomorphic network coding signatures are a special case of network coding signatures.

Definition 11. *A* homomorphic network coding signature scheme *is defined by a 4-tuple of algorithms (*$\mathsf{NetKG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Combine}$*) such that:*

$\mathsf{NetKG}(1^k, N)$ *On input the security parameter k and a parameter N, this algorithm outputs* $(\mathsf{vk}, \mathsf{sk})$ *where* sk *is the secret signing key and* vk *is the public verification key. N defines the size of the signed vectors.*

$\mathsf{Sign}(\mathsf{sk}, v, \mathsf{fid})$ *The signing algorithm takes as input the secret key* sk*, a random file identifier* fid *and a vector $v \in \mathbb{F}^N$ and outputs a signature σ.*

$\mathsf{Combine}(\mathsf{vk}, \mathsf{fid}, \{(w_i, \sigma_i)\}_{i=1}^{\ell})$ *This algorithm takes as input the public key* vk*, a file identifier* fid*, and a set of tuples (w_i, σ_i) where σ_i is a signature and $w_i \in \mathbb{F}$ is a coefficient. This algorithm outputs a new signature σ such that: if each σ_i is a valid signature on vector v_i, then σ is a valid signature for v obtained from linear combination $\sum_{i=1}^{\ell} w_i v_i$.*

$\mathsf{Ver}(\mathsf{vk}, \mathsf{fid}, v, \sigma)$ *Given the public key* vk*, a file identifier* fid*, a vector $v \in \mathbb{F}^N$ and a signature σ, the algorithm outputs 0 (reject) or 1 (accept).*