Sufficient Conditions for Intractability over Black-Box Groups: Generic Lower Bounds for Generalized DL and DH Problems

Andy Rupp[1], Gregor Leander[1], Endre Bangerter[2], Alexander W. Dent[3], and Ahmad-Reza Sadeghi[1]

[1] Horst Görtz Institute for IT-Security (Germany)
{arupp, sadeghi}@crypto.rub.de, gregor.leander@rub.de
[2] Bern University of Applied Sciences (Switzerland)
endre.bangerter@bfh.ch
[3] Royal Holloway, University of London (United Kingdom)
a.dent@rhul.ac.uk

Abstract. The generic group model is a valuable methodology for analyzing the computational hardness of number-theoretic problems used in cryptography. Although generic hardness proofs exhibit many similarities, still the computational intractability of every newly introduced problem needs to be proven from scratch, a task that can easily become complicated and cumbersome when done rigorously. In this paper we make the first steps towards overcoming this problem by identifying criteria which guarantee the hardness of a problem in an extended generic model where algorithms are allowed to perform any operation representable by a polynomial function.

Keywords: Generic Group Model, Straight-Line Programs, Hardness Conditions, Lower Bounds

1 Introduction

The *generic group model* was introduced by Nechaev [1] and Shoup [2]. In this model one considers algorithms that given a group G as black box, may only perform a restricted set of operations on the elements of G such as applying the group law, inversion of group elements and equality testing. Since in this model the group is treated as black box, the algorithms cannot exploit any special properties of a concrete group representation.

[1] The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the authors' views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The names of the last three authors are in alphabetical order.

Many fundamental cryptographic problems were proven to be computationally intractable in the generic model, most notably the discrete logarithm problem (DLP), the computational and decisional Diffie-Hellman problem (DHP and DDHP) [2], and the root extraction problem (in groups of hidden order) [3]. These intractability results are considered to be evidence supporting cryptographic assumptions of number-theoretic nature which underly the security of a vast number of systems of applied cryptography. Moreover, loosely speaking, it has become considered good practice, when making new intractability assumptions, to prove the underlying problem to be hard in the generic model. Many novel assumptions rely on more complex algebraic settings than the standard assumptions. They involve multiple groups and operations on group elements additional to the basic operations. Examples include the numerous assumptions based on bilinear pairings (e.g., see [4, 5]). Since the properties ensuring generic hardness had not been well-studied and formalized before this work, for each novel problem an entire hardness proof had to be done from scratch.

A generic group algorithm can only perform a subset of the operations that can be performed by an algorithm that may exploit specific properties of the representation of group elements. This implies that proving a problem to be intractable in the generic group model is a *necessary, but not sufficient* condition for the problem to be intractable in any concrete group. A generically intractable problem that is easy in any concrete group has been considered in [6].

Our contributions. In a nutshell, we identify the core aspects making cryptographic problems hard in the generic model. We provide a set of conditions, which given the description of a cryptographic problem allow one to check whether the problem at hand is intractable with respect to generic algorithms performing certain operations. In this way we aim at (i) providing means to structure and analyze the rapidly growing set of cryptographic assumptions as motivated in [7] and (ii) making the first steps towards automatically checkable hardness conditions in the generic model.

Related Work. In [8] the author analyzes a generalization of the Diffie-Hellman problem, the P-Diffie-Hellman problem: given group elements (g, g^{x_1}, g^{x_2}) the challenge is to compute $g^{P(x_1,x_2)}$, where P is a (non-linear) polynomial and g is a generator of some group G. Among other results, it is shown there that the computational and decisional variant of this problem class is hard in the generic model. Another general problem class has been introduced in [9] to cover DH related problems over bilinear groups. The authors show that decisional problems belonging to this class are hard in the generic model.

Recent work by Bresson et al. [10] independently analyzes generalized decisional problems over a single prime order group in the plain model. They showed that under several restrictions a so-called (P, Q)-DDH problem is efficiently reducible to the standard DDH problem. However, one important requirement for

applying their results is that the P and Q polynomials describing the problem need to be *power-free*, i.e., variables are only allowed to occur with exponents being equal to zero or one.

2 Some Preliminaries

Let $\mathsf{poly}(x)$ denote the class of univariate polynomials in x with non-negative integer coefficients. We call a function f *negligible* if $\forall poly \in \mathsf{poly}(x)\, \exists \kappa_0\, \forall \kappa \geq \kappa_0 : f(\kappa) \leq \frac{1}{poly(\kappa)}$.

Throughout the paper we are concerned with multivariate *Laurent polynomials* over the ring $\mathbb{Z}_n$. Informally speaking, Laurent polynomials are polynomials whose variables may also have negative exponents. More precisely, a Laurent polynomial P over $\mathbb{Z}_n$ in indeterminates $X_1, \ldots, X_\ell$ is a finite sum $P = \sum a_{\alpha_1,\ldots,\alpha_\ell} X_1^{\alpha_1} \cdots X_\ell^{\alpha_\ell}$ where $a_{\alpha_1,\ldots,\alpha_l} \in \mathbb{Z}_n$ and $\alpha_i \in \mathbb{Z}$. The set of Laurent polynomials over $\mathbb{Z}_n$ forms a ring with the usual addition and multiplication. By $\deg(P) = \max\{\sum_i |\alpha_i| \mid a_{\alpha_1,\ldots,\alpha_l} \not\equiv 0 \bmod n\}$ we denote the *(absolute) total degree* of a Laurent polynomial $P \neq 0$. Furthermore, we denote by $\mathfrak{L}_n^{(\ell,c)}$ (where $0 \leq c \leq l$) the subring of Laurent polynomials over $\mathbb{Z}_n$ where only the variables $X_{c+1}, \ldots, X_\ell$ can appear with negative exponents. Note that for any $P \in \mathfrak{L}_n^{(\ell,c)}$ and $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}$ the evaluation $P(\mathbf{x})$ is well-defined.

If $\mathcal{A}$ is a probabilistic algorithm, then $y \xleftarrow{R} \mathcal{A}(x)$ denotes the assignment to y of the output of $\mathcal{A}$'s run on x with fresh random coins. Furthermore, by $[\mathcal{A}(x)]$ we denote the set of all possible outputs of a probabilistic algorithm $\mathcal{A}$ on input of a fixed value x. If S is a set, then $x \xleftarrow{R} S$ denotes the random generation of an element $x \in S$ using the uniform distribution.

3 Problem Classes

In this section we formally define the classes of computational problems under consideration. For our formalization we adapt and extend the framework in [11].

Definition 1 (DL-/DH-type problem). *A* DL-/DH-type problem $\mathcal{P}$ *is characterized by*

- *A tuple of parameters*
$$Param_{\mathcal{P}} = (k, \ell, c, z)$$
consisting of some constants $k, \ell \in \mathbb{N}, c \in \mathbb{N}_0$ *where* $c \leq \ell$ *and* $z \in \mathsf{poly}(x)$.
- *A structure instance generator* $\mathsf{SIGen}_{\mathcal{P}}(\kappa)$ *that on input of a security parameter* κ *outputs a tuple of the form*
$$((\mathbf{G}, \mathbf{g}, n), (\mathbf{I}, Q)),$$
where

- $(\mathbf{G}, \mathbf{g}, n)$ *denotes the* algebraic structure instance *consisting of descriptions of cyclic groups* $\mathbf{G} = (G_1, \ldots, G_k)$ *of order* n *and corresponding generators* $\mathbf{g} = (g_1, \ldots, g_k)$*,*
- $(\mathbf{I}, Q)$ *denotes the* relation structure instance *consisting of the* input polynomials $\mathbf{I} = (\mathbf{I_1}, \ldots, \mathbf{I_k})$*, with* $\mathbf{I_i} \subset \mathfrak{L}_n^{(\ell,c)}$*,* $|\mathbf{I_i}| \leq z(\kappa)$*, and the* challenge polynomial $Q \in \mathfrak{L}_n^{(\ell,c)}$*.*

Then a problem instance *of* $\mathcal{P}$ *consists of a structure instance* $((\mathbf{G}, \mathbf{g}, n), (\mathbf{I}, Q)) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}(\kappa)$ *and group elements* $(g_i^{P(\mathbf{x})} | P \in \mathbf{I_i}, 1 \leq i \leq k)$*, where* $\mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}$ *are secret values. Given such a problem instance, the challenge is to compute*

$$\begin{cases} Q(\mathbf{x}), & \textit{for a DL-type problem} \\ g_1^{Q(\mathbf{x})}, & \textit{for a DH-type problem} \end{cases}.$$

Numerous cryptographically relevant problems fall into the class of DL-type or DH-type problems. Examples are problems such as the DLP [2], DHP [2], a variant of the representation problem [11], generalized DHP [11], square and inverse exponent problem [11], bilinear DHP [4], w-bilinear DH inversion problem [5], w-bilinear DH exponent problem [9], co-bilinear DHP [4], and many more. In Appendix A we extend our definitions and conditions to also include problems like the w-strong DH and w-strong BDH problem where the challenge is specified by a *rational function*. As an illustration of the definition, we consider the w-BDHI problem in more detail.

Example 1 (w-BDHIP). For the w-BDHI problem we have parameters $Param_{w\text{-BDHI}} = (3, 1, 0, w + 1)$ and a structure instance generator $\mathsf{SIGen}_{w\text{-BDHI}}$ that on input κ returns

$$((\mathbf{G} = (G_1, G_2, G_3), \mathbf{g} = (g_1, g_2, g_3), p),$$
$$(\mathbf{I} = (\mathbf{I_1} = \{1\}, \mathbf{I_2} = \{1, X_1^1, \ldots, X_1^{w(\kappa)}\}, \mathbf{I_3} = \{1\}), Q = X_1^{-1}))$$

such that p is a prime, there exists a non-degenerate, efficiently computable bilinear mapping $e : G_2 \times G_3 \to G_1$ with $e(g_2, g_3) = g_1$, and an isomorphism $\psi : G_2 \to G_3$ with $\psi(g_2) = g_3$. A problem instance additionally comprises group elements $(g_i^{P(\mathbf{x})} | P \in \mathbf{I_i}, 1 \leq i \leq 3) = (g_1, g_2, g_2^{x_1}, \ldots, g_2^{x_1^{w(\kappa)}}, g_3)$, where $\mathbf{x} = x_1 \xleftarrow{\mathrm{R}} \mathbb{Z}_n^*$, and the task is to compute $g_1^{Q(\mathbf{x})} = g_1^{x_1^{-1}}$.

In the remainder of this paper, we are often only interested in individual parts of the output of $\mathsf{SIGen}_{\mathcal{P}}$. To this end, we introduce the following simplifying notation: By $\$ \xleftarrow{\mathrm{R}} \mathsf{SIGen}_{\mathcal{P}}^{\$}(\kappa)$, where $\$$ is a wildcard character, we denote the projection of $\mathsf{SIGen}_{\mathcal{P}}$'s output to the part $\$$. For instance, $(n, \mathbf{I}, Q) \xleftarrow{\mathrm{R}} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q)}(\kappa)$ denotes the projection of the output to the triple consisting of the group order, the input polynomials, and the challenge polynomial. Furthermore, by

$[\mathsf{SIGen}^{\$}_{\mathcal{P}}(\kappa)]$ we denote the set of all possible outputs \$ for a given fixed security parameter κ.

4 Extending Shoup's Generic Group Model

4.1 Generic Operations

For our framework we restrict to consider operations of the form $\circ : G_{s_1} \times \ldots \times G_{s_u} \to G_d$, where $u \geq 1$, $s_1, \ldots, s_u, d \in \{1, \ldots, k\}$ are some fixed constants (that do not depend on κ). Furthermore, we demand that the action of $\circ$ on the group elements can be represented by a fixed regular polynomial. That means, there exists a fixed $F \in \mathbb{Z}[Y_1, \ldots, Y_u]$ (also not depending on κ) such that for any generators $g_{s_1}, \ldots, g_{s_u}, g_d$ given as part of a problem instance we have that $\circ(a_1, \ldots, a_u) = g_d^{F(y_1,\ldots,y_u)}$ where $a_1 = g_{s_1}^{y_1}, \ldots, a_u = g_{s_u}^{y_u}$. For instance, the bilinear mapping $e : G_2 \times G_3 \to G_1$ which is part of the algebraic setting of the w-BDHIP is such an operation: for any g_2, g_3 and $g_1 = e(g_2, g_3)$ it holds that $e(a_1, a_2) = e(g_2^{y_1}, g_3^{y_2}) = g_1^{F(y_1,y_2)}$ where $F = Y_1 Y_2$. In fact, to the best of our knowledge, virtually any deterministic operation considered in the context of the generic group model in the literature so far belongs to this class of operations.

We represent an operation of the above form by a tuple $(\circ, s_1, \ldots, s_u, d, F)$, where the first component is a symbol serving as a unique identifier of the operation. The set of allowed operations can thus be specified by a set of such tuples. The full version of this paper [12] explains how to extend the operation set to include decision oracles.

Example 2 (Operations Set for w-BDHIP). The operations set $\Omega = \{(\circ_1, 1, 1, 1, Y_1 + Y_2), (\circ_2, 2, 2, 2, Y_1 + Y_2), (\circ_3, 3, 3, 3, Y_1 + Y_2), (inv_1, 1, 1, -Y_1), (inv_2, 2, 2, -Y_1), (inv_3, 3, 3, -Y_1), (\psi, 2, 3, Y_1), (e, 2, 3, 1, Y_1 \cdot Y_2)\}$ specifies operations for the group law $(\circ_i)$ and inversion (inv_i) over each group as well as the isomorphism $\psi : G_2 \to G_3$ and the bilinear map $e : G_2 \times G_3 \to G_1$.

4.2 Generic Group Algorithms and Intractability

In this section, we formally model the notion of generic group algorithms for DL-/DH-type problems. We adapt Shoup's generic group model [2] for this purpose.

Let $S_n \subset \{0,1\}^{\lceil \log_2(n) \rceil}$ denote a set of bit strings of cardinality n and Σ_n the set of all bijective functions from $\mathbb{Z}_n$ to S_n. Furthermore, let $\sigma = (\sigma_1, \ldots, \sigma_k) \in \Sigma_n^k$ be a k-tuple of randomly chosen encoding functions for the groups $G_1, \ldots, G_k \cong \mathbb{Z}_n$.

A *generic algorithm* $\mathcal{A}$ is a probabilistic algorithm that is given access to a *generic (multi-) group oracle* $\mathcal{O}_\Omega$ allowing $\mathcal{A}$ to perform operations from Ω on encoded group elements. Since any cyclic group of order n is isomorphic to

$(\mathbb{Z}_n, +)$, we will always use $\mathbb{Z}_n$ with generator 1 for the internal representation of a group G_i.

As internal state $\mathcal{O}_\Omega$ maintains two types of lists, namely *element lists* $L_1, \ldots, L_k$, where a $L_i \subset \mathfrak{L}_n^{(\ell,c)}$, and *encoding lists* $E_1, \ldots, E_k$, where $E_i \subset S_n$. For an index j let $L_{i,j}$ and $E_{i,j}$ denote the j-th entry of L_i and E_i, respectively. Each list L_i is initially populated with the corresponding input polynomials given as part of a problem instance of a DL-/DH-type problem $\mathcal{P}$, i.e., $L_i = (P|P \in \mathbf{I_i})$. A list E_i contains the encodings of the group elements corresponding to the entries of L_i, i.e., $E_{i,j} = \sigma_i(L_{i,j}(\mathbf{x}))$. E_i is initialized with $E_i = (\sigma_i(P(\mathbf{x}))|P \in \mathbf{I_i})$. $\mathcal{A}$ is given (read) access to all encodings lists. In order to be able to perform operations on the randomly encoded elements, the algorithm may query $\mathcal{O}_\Omega$. Let $(\circ, s_1, \ldots, s_u, d, F)$ be an operation from Ω. Upon receiving a query $(\circ, j_1, \ldots, j_u)$, the oracle computes $P := F(L_{s_1,j_1}, \ldots, L_{s_u,j_u})$, appends P to L_d and $\sigma_d(P(\mathbf{x}))$ to the encoding list E_d. After having issued a number of queries, $\mathcal{A}$ eventually provides its final output. In the case that $\mathcal{P}$ is a DL-type problem, we say that $\mathcal{A}$ has solved the problem instance of $\mathcal{P}$ if its output a satisfies $Q(\mathbf{x}) - a \equiv 0 \bmod n$. In the case that $\mathcal{P}$ is a DH-type problem, $\mathcal{A}$ has solved the problem instance if its output $\sigma_1(a)$ satisfies $Q(\mathbf{x}) - a \equiv 0 \bmod n$.

Let a DL-/DH-type problem over cyclic groups $G_1, \ldots, G_k$ of order n be given. We can write the group order as $n = p^e \cdot s$ with $\gcd(p, s) = 1$ where p be the largest prime factor of n. Then for each i it holds that $G_i \cong G_i^{(p^e)} \times G_i^{(s)}$ where $G_i^{(p^e)}$ and $G_i^{(s)}$ are cyclic groups of order p^e and s, respectively. It is easy to see that solving an instance of a DL-/DH-type over groups G_i of order n is equivalent for a generic algorithm to solving it *separately* over the subgroups $G_i^{(p^e)}$ *and* the subgroups $G_i^{(s)}$. Thus, computing a solution over the groups G_i is a least as hard for generic algorithms as computing a solution over the groups $G_i^{(p^e)}$. In the following we always assume that $\mathsf{SIGen}_\mathcal{P}$ on input κ generates groups of prime power order $n = p^e$ with $p > 2^\kappa$ and $e > 0$.

Definition 2 (q-GGA). *A q-GGA is a generic group algorithm that for any $\kappa \in \mathbb{N}$, it receives as part of its input, makes at most $q(\kappa)$ queries to the generic group oracle.*

Definition 3 (GGA-intractability of DL-type Problems). *A DL-type problem $\mathcal{P}$ is (Ω, q, ν)-GGA-intractable if for all q-GGA $\mathcal{A}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[Q(\mathbf{x}) \equiv a \bmod n \;\middle|\; \begin{array}{l} (n, \mathbf{I}, Q) \xleftarrow{R} \mathsf{SIGen}_\mathcal{P}^{(n,\mathbf{I},Q)}(\kappa); \sigma \xleftarrow{R} \Sigma_n^k; \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}; \\ a \xleftarrow{R} \mathcal{A}^{\mathcal{O}_\Omega}(\kappa, n, \mathbf{I}, Q, (\sigma_i(P(\mathbf{x}))|P \in \mathbf{I_i})_{1 \le i \le k}) \end{array} \right] \le \nu(\kappa)$$

Definition 4 (GGA-intractability of DH-type Problems). *A DH-type problem $\mathcal{P}$ is (Ω, q, ν)-GGA-intractable if for all q-GGA $\mathcal{A}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[Q(\mathbf{x}) \equiv a \bmod n \;\middle|\; \begin{array}{l} (n, \mathbf{I}, Q) \xleftarrow{R} \mathsf{SIGen}_\mathcal{P}^{(n,\mathbf{I},Q)}(\kappa); \sigma \xleftarrow{R} \Sigma_n^k; \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}; \\ \sigma_1(a) \xleftarrow{R} \mathcal{A}^{\mathcal{O}_\Omega}(\kappa, n, \mathbf{I}, Q, (\sigma_i(P(\mathbf{x}))|P \in \mathbf{I_i})_{1 \le i \le k}) \end{array} \right] \le \nu(\kappa)$$

5 Abstract Hardness Conditions: Linking GGA and SLP Intractability

Informally speaking, the grade of intractability of a DL-/DH-type problem with respect to generic algorithms can be "measured" by means of two "quantities":

1. The probability of gaining information about the secret choices $\mathbf{x}$ in the course of a computation by means of non-trivial equalities between group elements. This quantity is called *leak-resistance*.
2. The probability to solve problem instances using a trivial strategy, i.e., by taking actions independently of (in)equalities of computed group elements and thus independent of the specific problem instance. This quantity is called *SLP-intractability*.

For formalizing both quantities, we make use of so-called straight-line program (SLP) generators. Note that SLPs are a very common concept in the field of computational algebra and has also proved its usefulness in the area of cryptography. However, the SLP model and the GGA model have not been explicitly related in the literature so far.

Definition 5 ((Ω, q)-SLP-generator). *A (Ω, q)-SLP-generator $\mathcal{S}$ is a probabilistic algorithm that on input $(\kappa, n, \mathbf{I}, Q)$, outputs lists $(L_1, \ldots, L_k)$ where $L_i \subset \mathfrak{L}_n^{(\ell,c)}$. Each list L_i is initially populated with $L_i = (P|P \in \mathbf{I_i})$. The algorithm can append a polynomial to a list by applying an operation from Ω to polynomials already contained in the lists, i.e., for an operation $(\circ, s_1, \ldots, s_u, d, F) \in \Omega$ and existing polynomials $P_1 \in L_{s_1}, \ldots, P_u \in L_{s_u}$ the algorithm can append $F(P_1, \ldots, P_u)$ to L_d. In this way, the algorithm may add up to $q(\kappa)$ polynomials in total to the lists. The algorithm additionally outputs an element $a \in \mathbb{Z}_n$ in the case of a DL-type problem and a polynomial $P \in L_1$ in the case of DH-type problem, respectively.*

Let us first formalize the leak-resistance of a problem. When do group elements actually leak information due to equality relations? To see this, reconsider the definition of the generic oracle in Section 4.2 and observe that two encodings $E_{i,j}$ and $E_{i,j'}$ are equal if and only if the evaluation $(L_{i,j} - L_{i,j'})(\mathbf{x})$ yields zero. However, it is clear that such an equality relation yields no information about particular choices $\mathbf{x}$ if it holds for *all* elements from $\mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}$. Thus, denoting the ideal of $\mathfrak{L}_n^{(\ell,c)}$ containing all Laurent polynomials that are effectively zero over $\mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}$ by

$$\mathcal{I}_n = \{P \in \mathfrak{L}_n^{(\ell,c)} \mid \forall \mathbf{x} \in \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c} : \ P(\mathbf{x}) \equiv 0 \bmod n\} \tag{1}$$

an equality yields no information at all if $(L_{i,j} - L_{i,j'}) \in \mathcal{I}_n$. Otherwise, a *non-trivial collision* occurred and $\mathcal{A}$ learns that $\mathbf{x}$ is a modular root of $L_{i,j} - L_{i,j'}$.

By Definition 6 we capture the chance that information about the secret choices $\mathbf{x}$ is leaked in the course of a computation due to non-trivial equalities between group elements. For this purpose we can make use of (Ω, q)-SLP-generators since they generate all possible sequences of polynomials that may occur in an execution of a q-GGA.

Definition 6 (Leak-resistance). *A DL-/DH-type problem $\mathcal{P}$ is (Ω, q, ν)-leak-resistant if for all (Ω, q)-SLP-generators $\mathcal{S}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[\begin{array}{c} \exists i \text{ and } P, P' \in L_i \text{ such that} \\ (P-P')(\mathbf{x}) \equiv 0 \bmod n \ \wedge \ P - P' \notin \mathcal{I}_n \end{array} \middle| \begin{array}{l} (n, \mathbf{I}, Q) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q)}(\kappa); \\ (L_1, \ldots, L_k) \xleftarrow{R} \mathcal{S}(\kappa, n, \mathbf{I}, Q); \\ \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell - c} \end{array}\right] \leq \nu(\kappa)$$

Now assume that no information about $\mathbf{x}$ can be gained. In this case, we can restrict to consider algorithms applying trivial solution strategies to solve instances of a problem. That means, we can restrict our considerations to the subclass of generic algorithms that, when fixing all inputs except for the choice of $\mathbf{x}$, always apply the *same fixed sequence of operations* from Ω and provide the same output in order to solve an arbitrary problem instance. Thus, the algorithm actually acts as a straight-line program in this case.

Definition 7 (SLP-intractability of DL-Type Problems). *A DL-type problem $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable if for all (Ω, q)-SLP-generators $\mathcal{S}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[Q(\mathbf{x}) \equiv a \bmod n \middle| \begin{array}{l} (n, \mathbf{I}, Q) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q)}(\kappa); \\ (a, L_1, \ldots, L_k) \xleftarrow{R} \mathcal{S}(\kappa, n, \mathbf{I}, Q); \\ \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell - c} \end{array}\right] \leq \nu(\kappa)$$

Definition 8 (SLP-intractability of DH-type Problems). *A DH-type problem $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable if for all (Ω, q)-SLP-generators $\mathcal{S}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[(P-Q)(\mathbf{x}) \equiv 0 \bmod n \middle| \begin{array}{l} (n, \mathbf{I}, Q) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q)}(\kappa); \\ (P, L_1, \ldots, L_k) \xleftarrow{R} \mathcal{S}(\kappa, n, \mathbf{I}, Q); \\ \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell - c} \end{array}\right] \leq \nu(\kappa)$$

Theorem 1 (GGA-intractability of DL-/DH-type Problems). *If a DL-type problem is (Ω, q, ν_1)-leak-resistant and (Ω, q, ν_2)-SLP-intractable then it is $(\Omega, q, \nu_1 + \nu_2)$-GGA-intractable. If a DH-type problem is (Ω, q, ν_1)-leak-resistant and (Ω, q, ν_2)-SLP-intractable then it is $(\Omega, q, \frac{1}{2^{\kappa} - (q(\kappa) + z(\kappa))} + \nu_1 + \nu_2)$-GGA-intractable.*

The proof of this theorem is given in the full version of the paper [12].

6 Practical Conditions

In this section, we present easily checkable conditions ensuring that a DL-/DH-type problem is (Ω, q, ν_1)-leak-resistant and (Ω, q, ν_2)-SLP-intractable with q being polynomial and ν_1 and ν_2 being negligible functions in the security parameter. Reviewing the corresponding definitions, we see that the probabilities ν_1 and ν_2 are closely related to the probability of randomly picking roots of certain multivariate Laurent polynomials. Lemma 1 shows in turn that the probability of finding such a root is small for non-zero polynomials in $\mathfrak{L}_n^{(\ell,c)}$ having low total degrees.

Lemma 1. *Let p be a prime, $e \in \mathbb{N}$, $n = p^e$, and let $P \in \mathfrak{L}_n^{(\ell,c)}$ be a non-zero Laurent polynomial of total degree d. Then for $\mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}$ we have*

$$\Pr[P(\mathbf{x}) \equiv 0 \bmod n] \leq \frac{(\ell - c + 1)d}{p - 1}\,.$$

6.1 Operations Sets as Graphs: Bounding Polynomial Degrees

We aim at formalizing the class of operations sets that only allow for a small rise in the degrees of polynomials that can be generated by any (Ω, q)-SLP-generator $\mathcal{S}$. Remember, these are the polynomials that can be generated from the input polynomials by applying operations from Ω at most $q(\kappa)$ times. To this end, we introduce a special type of graph, called *operations set graph* (Definition 9), modeling an operations set and reflecting the corresponding rise of degrees.

Definition 9 (Operations Set Graph). *An operations set graph $\mathcal{G} = (V, E)$ is a directed multi-edge multi-vertex graph. There are two types of vertices, namely group and product vertices. The vertex set V contains at least one group vertex. Each group vertex in V is labeled with a unique integer. All product vertices are labeled by Π. Any edge in E may connect two group vertices or a group and a product vertex.*

Let Ω be an operations set involving k groups. Then the operations set graph $\mathcal{G}_\Omega = (V, E)$ corresponding to Ω is constructed as follows: V is initialized with k group vertices representing the k different groups, where these vertices are labeled with the numbers that are used in the specification of Ω, say the numbers 1 to k. For each operation $(\circ, s_1, \ldots, s_u, d, F) \in \Omega$ we add additional product vertices to V and edges to E. Let $F = \sum_i M_i$ be represented as the sum of non-zero monomials. Then for each M_i we do the following:

1. We add a product vertex and an edge from this vertex to the group vertex with label d.

2. For each variable Y_j $(1 \leq j \leq u)$ occurring with non-zero exponent ℓ in M_i we add ℓ edges from the group vertex labeled with the integer s_j to the product vertex just added before.

In order to embed the notion of increasing polynomial degrees by applying operations into the graph model we introduce the following graph terminology: We associate each group vertex in a graph with a number, called *weight*. The weight may change by doing walks through the graph. Taking a *walk* through the graph means to take an arbitrary path that contains exactly two group vertices (that are not necessarily different) where one of these vertices is the start point and the other is the end point of the path. A walk modifies the weight of the end vertex in the following way:

- If the path contains only the two group vertices, the new weight is set to be the maximum of the weights of the start and end vertex.
- If the path contains a product vertex, the new weight is set to be the maximum of the old weight and $\sum_{j=1}^{u} w_j$, where u is the indegree and w_j is the weight of the j-th predecessor of this product vertex.

We define a *free walk* to be a walk through a path that only consists of the two group vertices and no other vertex. A *non-free walk* is a walk through a path containing a product vertex. It is important to observe that

- a non-free walk can actually increase the maximum vertex weight of a graph in contrast to a free-walk.
- after each non-free walk the weight of any vertex can be changed at most finitely many times by doing free walks.

Hence, the following definition of the *q-weight* makes sense: Let q be a fixed positive number. We consider finite sequences of walks through a graph, where each sequence consists of exactly q non-free walks and an arbitrary finite number of free walks. We define the q-weight of a (group) vertex to be the maximum weight of this vertex over all such sequences. Similarly, we define the q-weight of an operations set graph to be the maximum of the q-weights of all its vertices.

Obviously, the q-weights of the vertices $1, \ldots, k$ of an operations set graph $\mathcal{G}_\Omega$ can be used to upper bound the degrees of the output polynomials $L_1, \ldots, L_k$ of any (Ω, q)-SLP-generator $\mathcal{S}$ when setting the initial weight of each group vertex i to the maximal degree of the polynomials in $\mathbf{I_i}$. Similarly, we can bound the maximum positive or negative exponent of a single variable X_j by setting the initial weight of the group vertex i to be the maximum degree of X_j in any polynomial in $\mathbf{I_i}$.

With regard to the definition of the q-weight, we can immediately simplify the structure of operations set graphs: Clearly, we do not change the q-weight

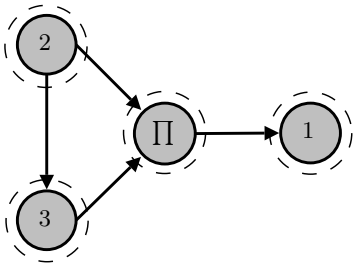

Fig. 1. $\mathcal{G}_\Omega$ for $\Omega = \{(\circ_1, 1, 1, 1, Y_1 + Y_2), (\circ_2, 2, 2, 2, Y_1 + Y_2), (\circ_3, 3, 3, 3, Y_1 + Y_2), (inv_1, 1, 1, -Y_1), (inv_2, 2, 2, -Y_1), (inv_3, 3, 3, -Y_1), (\psi, 2, 3, Y_1), (e, 2, 3, 1, Y_1 \cdot Y_2)\}$ of w-BDHIP. Strongly connected components are marked by dashed borders.

of a graph if we remove self-loops and product vertices with indegree 1, where in the latter case the two edges entering and leaving the vertex are replaced by a single edge going from the predecessor vertex to the successor vertex. We call such a graph a *reduced* operations set graph. As an illustrating example, consider the reduced operations set graph depicted in Figure 1, which belongs to the operations set for the w-BDHI problem (cf. Example 2).

The following condition characterizes graphs that do not allow for a super-polynomial grow of vertex weights. Intuitively, it prohibits any kind of repeated doubling. For the q-weight of operations set graphs satisfying Condition 1, it is possible to derive non-trivial upper bounds as given in Theorem 2. The proof is given in the full version of the paper [12].

Condition 1 *Let $\mathcal{G}_\Omega$ be a reduced operations set graph. Then for every strongly connected component*[2] *$\mathcal{C}$ of $\mathcal{G}_\Omega$ it holds that every product vertex contained in $\mathcal{C}$ has at most one incoming edge from a vertex that is also contained in $\mathcal{C}$.*

Theorem 2. *Let $\mathcal{G}_\Omega$ be a reduced operations set graph satisfying Condition 1. Let n_1 denote the number of product vertices contained in $\mathcal{G}_\Omega$, $u_{\max}$ the maximal indegree of these product vertices, $d_{\max}$ the maximal initial weight of any group vertex, and n_2 the number of SCCs containing at least one product and one group vertex. Then the q-weight of $\mathcal{G}_\Omega$ is upper bounded by*

$$D(n_1, n_2, u_{\max}, d_{\max}, q) = \begin{cases} d_{\max}(u_{\max})^{n_1}, & n_2 = 0 \\ d_{\max}\mathsf{e}^{n_1}, & n_2 > 0 \text{ and } q < \frac{\mathsf{e}}{u_{\max}} n_1 \\ d_{\max}\left(\frac{u_{\max} q}{n_1}\right)^{n_1}, & n_2 > 0 \text{ and } q \geq \frac{\mathsf{e}}{u_{\max}} n_1 \end{cases},$$

where e *denotes Euler's number.*

[2] A strongly connected component of a directed graph $\mathcal{G}_\Omega = (V, E)$ is a maximal set of vertices $U \subset V$ s.t. every two vertices in U are reachable from each other. The strongly connected components of a graph can be computed in time $O(|V| + |E|)$.

Example 3. Condition 1 is satisfied for $\mathcal{G}_\Omega$ depicted in Figure 1 since the strongly connected component containing the product vertex contains no other vertices. We have $n_1 = 1$, $n_2 = 0$, and $u_{\max} = 2$. Since the problem instance implies $d_{\max} = w$ we have that the q-weight of the graph is bounded by $2w$.

Note that the factor by which the (maximal) initial weight of the vertices can be increased only depends on the particular operations set graph. Hence, once we have shown that an operations set only allows to increase degrees by a low (i.e., polynomial) factor, this certainly holds for all problems involving this operations set and does not need to be reproven (as it is currently done in the literature).

It is possible to devise a graph algorithm (Algorithm 1) that finds individual bounds on the q-weights of the group vertices which are often tighter than the generic bound from Theorem 2. The principle of the algorithm is simple. We consider the directed acyclic graph that is composed of the SCCs of the operations set graph. We move from the sources to the sinks of the DAG and recursively bound the q-weights of the vertices within each SCC. In the end when all SCCs are labeled with such a bound, the q-weight of a group vertex is simply set to be the q-weight bound of the (unique) SCC in which it is contained.

6.2 Practical Conditions: Leak-Resistance

To provide leak-resistance, we ensure that any difference of two distinct polynomials computable by a (Ω, q)-SLP-generator is of low degree. We do so by demanding that the input polynomials $\mathbf{I}$ of a problem $\mathcal{P}$ have low degrees (Condition 2) and restrict to operations sets Ω only allowing for small increase of degrees (Condition 1). If these conditions are satisfied, we can derive a concrete leak-resistance bound ν for any runtime bound q (Theorem 3).

Condition 2 *There exists $r_1 \in \mathsf{poly}(x)$ such that for all $\kappa \in \mathbb{N}$, $\mathbf{I} \in [\mathsf{SIGen}^{\mathbf{I}}_{\mathcal{P}}(\kappa)]$ we have* $\max_{1 \le i \le k, P \in \mathbf{I_i}} (\deg(P)) \le r_1(\kappa)$

Theorem 3. *Let Ω be an operations set such that Condition 1 is satisfied. Furthermore, let $\mathcal{P}$ be a DL-type or DH-type problem satisfying Condition 2. Then for any $q \in \mathsf{poly}(x)$, the problem $\mathcal{P}$ is (Ω, q, ν)-leak-resistant, where*

$$\nu(\kappa) = 2^{-\kappa} k (q(\kappa) + z(\kappa))^2 (\ell - c + 1) D(n_1, n_2, u_{\max}, r_1(\kappa), q(\kappa)) \,.$$

Example 4 (Leak-resistance for w-BDHIP). The degrees of the input polynomials of the w-BDHI problem are polynomially upper bounded through w by definition. Example 3 showed that Ω satisfies Condition 1 yielding $D(1, 0, 2, w(\kappa), q(\kappa)) = 2w(\kappa)$. Furthermore, for w-BDHIP we have parameters $k = 3$, $\ell = 1$, and $c = 0$. Thus, by Theorem 3 the problem $\mathcal{P}$ is (Ω, q, ν)-leak-resistant, where $\nu(\kappa) = 2^{-\kappa} 12 (q(\kappa) + w(\kappa) + 1)^2 w(\kappa)$.

Algorithm 1 Computation of the q-weigths of group vertices.

Input: q, reduced operations set graph $\mathcal{G}$ satisfying Condition 1, initial weights for the k group vertices in G

Output: q-weights $w_1, \ldots, w_k$ of vertices $1, \ldots, k$

Perform a topological sort on the DAG of $\mathcal{G}$, i.e., arrange the SCCs of $\mathcal{G}$ in layers 0 to ℓ such that SCCs in layer j can only receive edges from SCCs contained in layers $i < j$.
for each layer $j = 0 : \ell$ **do**
  for each SCC $\mathcal{C}$ in layer j **do**
    if $\mathcal{C}$ consists only of group vertices **then**
      set weight of $\mathcal{C}$ to maximum of weights of vertices contained in $\mathcal{C}$ and weights of SCCs in layers $i < j$ having edges to $\mathcal{C}$
    end if
    if $\mathcal{C}$ consists only of a single product vertex **then**
      set weight of $\mathcal{C}$ to sum of weights of SCCs in layers $i < j$ having edges to $\mathcal{C}$
    end if
    if $\mathcal{C}$ consists of at least one product vertex and one group vertex **then**
      let w be the maximum of the weights of group vertices contained in $\mathcal{C}$ and the weights of SCCs in layers $i < j$ having edges to these group vertices
      for each product vertex Π in $\mathcal{C}$, compute sum of weights of SCCs in layers $i < j$ having edges to Π, and let v be the maximum of these sums
      set weight of $\mathcal{C}$ to $w + qv$
    end if
  end for
end for
for $i = 1 : k$ **do**
  set w_i to weight of SCC containing the group vertex i
end for

6.3 Practical Conditions: SLP-intractability of DL-type Problems

In view of Lemma 1, in order to ensure SLP-intractability for a DL-type problem it suffices to require the challenge polynomial being non-constant (Condition 3) and of low degree (Condition 4).

Condition 3 *There exists* $\kappa_0 \in \mathbb{N}$ *such that for all* $\kappa \geq \kappa_0$, $(n, Q) \in [\mathsf{SIGen}_{\mathcal{P}}^{(n,Q)}(\kappa)]$ *the polynomial* Q *is not a constant in* $\mathfrak{L}_n^{(\ell,c)}$.

Condition 4 *There exists* $r_2 \in \mathsf{poly}(x)$ *such that for all* $\kappa \in \mathbb{N}$, $Q \in [\mathsf{SIGen}_{\mathcal{P}}^{Q}(\kappa)]$ *we have* $\deg(Q) \leq r_2(\kappa)$.

Assuming the above conditions are satisfied for a DL-type problem, Theorem 4 implies that the problem is (Ω, q, ν)-SLP-intractable, where q is an arbitrary polynomial and ν is a negligible function in the security parameter.

Theorem 4. *Let $\mathcal{P}$ be a DL-type problem satisfying Condition 4 and Condition 3. Then for any $q \in \mathsf{poly}(x)$ and any operations set Ω, $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable, where*

$$\nu(\kappa) = \begin{cases} 1, & \kappa < \kappa_0 \\ \frac{(\ell - c + 1) r_2(\kappa)}{2^\kappa}, & \kappa \geq \kappa_0 \end{cases}.$$

6.4 Practical Conditions: SLP-intractability of DH-type Problems

To ensure SLP-intractability of DH-type problems we formulate similar conditions as in the case of DL-type problems. More precisely, we ensure that the difference polynomials considered in the definition of SLP-intractability (Definition 8) are never zero and of low degree.

The non-triviality condition (Condition 5) states that an efficient SLP-generator can hardly ever compute the challenge polynomial, and thus solve the problem with probability 1.

Condition 5 *For every $q \in \mathsf{poly}(x)$ there exists $\kappa_0 \in \mathbb{N}$ such that for all $\kappa \geq \kappa_0$, (Ω, q)-SLP-generators $\mathcal{S}$, $(n, \mathbf{I}, Q) \in [\mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q)}(\kappa)]$, and $(P, L_1, \ldots, L_k) \in [\mathcal{S}(\kappa, n, \mathbf{I}, Q)]$ we have $P \neq Q$ in $\mathfrak{L}_n^{(\ell,c)}$.*

We note that Condition 5 appears to be more complex compared to the practical conditions seen so far and it is not clear to us how to verify it in its full generality. However, it is usually easy to check in the case of a problem of practical relevance. Usually, one of the following properties is satisfied implying the validity of Condition 5:

- The total degree of $P \in L_1$ is bounded by a value which is smaller than the total degree of Q.
- The positive/negative degree of $P \in L_1$ is bounded by a value which is smaller than the positive/negative degree of Q.
- The positive/negative degree of some variable X_j of $P \in L_1$ is bounded by a value which is smaller than the positive/negative degree of that variable in Q.

Remember, that we can make use of the results from Section 6.1 for proving that a problem satisfies one of these properties.

Moreover, we have to prevent that an (Ω, q)-SLP-generator outputs a polynomial $P \neq Q$ which frequently "collides" with Q and thus constitutes a good interpolation for Q. If P is low degree (Conditions 1 and 2), then it is sufficient to demand that Q is of low degree as well (Condition 4).

Hence, we need the practical conditions for leak-resistance in addition to the ones stated in this section for showing that a DH-type problem is (Ω, q, ν)-SLP-intractable, where ν is a negligible function in the security parameter.

Theorem 5. *Let Ω be an operations set such that Condition 1 is satisfied. Furthermore, let $\mathcal{P}$ be DH-type problem satisfying Condition 2, Condition 4 and Condition 5. Then for any $q \in \mathsf{poly}(x)$, the problem $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable, where*

$$\nu(\kappa) = \begin{cases} 1, & \kappa < \kappa_0 \\ \frac{(\ell - c + 1)(r_2(\kappa) + D(n_1, n_2, u_{\max}, r_1(\kappa), q(\kappa)))}{2^\kappa}, & \kappa \geq \kappa_0 \end{cases}$$

is a negligible function.

Example 5 (SLP-intractability of w-BDHIP). Remember that for this problem the challenge polynomial is fixed to $Q = X_1^{-1}$. Moreover, observe that all variables occurring in the input polynomials only have positive exponents. Thus, any polynomial $P \in L_1$ has only positive exponents in any variable. Hence, Condition 5 is trivially satisfied (independently of the considered operations set Ω).[3] Condition 4 is satisfied since we always have $\deg(Q) = 1 =: r_2(\kappa)$. As we have already seen in the previous sections, Conditions 1 and 2 hold yielding the upper bound $D(1, 0, 2, w(\kappa), q(\kappa)) = 2w(\kappa)$ on the degrees of the polynomials $P \in L_1$. Thus, by Theorem 5 the problem is (Ω, q, ν)-SLP-intractable, where $\nu(\kappa) = 2^{-\kappa}(2 + 4w(\kappa))$.

References

1. Nechaev, V.I.: Complexity of a determinate algorithm for the discrete logarithm. Mathematical Notes **55**(2) (1994) 165–172
2. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Advances in Cryptology: Proceedings of EUROCRYPT 1997. Volume 1233 of Lecture Notes in Computer Science., Springer-Verlag (1997) 256–266
3. Damgård, I., Koprowski, M.: Generic lower bounds for root extraction and signature schemes in general groups. In: Advances in Cryptology: Proceedings of EUROCRYPT 2002. Volume 2332 of Lecture Notes in Computer Science., Springer-Verlag (2002) 256–271
4. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Advances in Cryptology: Proceedings of CRYPTO 2001. Volume 2139 of Lecture Notes in Computer Science., Springer-Verlag (2001) 213–229
5. Boneh, D., Boyen, X.: Efficient selective-ID secure identity based encryption without random oracles. In: Advances in Cryptology: Proceedings of EUROCRYPT 2004. Volume 3027 of Lecture Notes in Computer Science., Springer-Verlag (2004) 223–238
6. Dent, A.W.: Adapting the weaknesses of the random oracle model to the generic group model. In: Advances in Cryptology: Proceedings of ASIACRYPT 2002. Volume 2501 of Lecture Notes in Computer Science., Springer-Verlag (2002) 100–109

[3] Note that $X_1^{-1} \neq X_1^{\phi(n)-1}$ in $\mathfrak{L}_n^{(1,0)}$ but these polynomials evaluate to the same value for all $x_1 \in \mathbb{Z}_n^*$. However, Conditions 1 and 2 ensure that for any efficient SLP-generator there exists κ_0 s.t. for all $\kappa \geq \kappa_0$ the polynomial $X_1^{\phi(n)-1}$ cannot be computed.

7. Boneh, D.: Number-theoretic assumptions. Invited Talk at TCC's Special Session on Assumptions for Cryptography (2007)
8. Kiltz, E.: A tool box of cryptographic functions related to the Diffie-Hellman function. In: INDOCRYPT '01: Proceedings of the Second International Conference on Cryptology in India. Volume 2247 of Lecture Notes in Computer Science., Springer-Verlag (2001) 339–350
9. Boneh, D., Boyen, X., Goh, E.: Hierarchical identity based encryption with constant size ciphertext (full paper). Cryptology ePrint Archive, Report 2005/015 (2005) `http://eprint.iacr.org/2005/015`.
10. Bresson, E., Lakhnech, Y., Mazaré, L., Warinschi, B.: A generalization of DDH with applications to protocol analysis and computational soundness. In Menezes, A.J., ed.: Advances in Cryptology: Proceedings of CRYPTO 2007. Volume 4622 of Lecture Notes in Computer Science., Springer-Verlag (August 2007) 482–499
11. Sadeghi, A., Steiner, M.: Assumptions related to discrete logarithms: Why subtleties make a real difference. In: Advances in Cryptology: Proceedings of EUROCRYPT 2001. Volume 2045 of Lecture Notes in Computer Science., Springer-Verlag (2001) 243–260
12. Rupp, A., Leander, G., Bangerter, E., Dent, A.W., Sadeghi, A.R.: Sufficient conditions for intractability over black-box groups: Generic lower bounds for generalized DL and DH problems (full version). Cryptology ePrint Archive, Report 2007/360 (2007) `http://eprint.iacr.org/2007/360`.

A Rational Functions Specifying Problem Challenges

Our framework so far only covers problems where the solution of a problem instance can be represented as a Laurent polynomial. This restriction excludes important problems like the w-strong Diffie-Hellman problem or the w-strong Bilinear Diffie-Hellman problem. Informally speaking, the w-SDH problem can be described as follows: Given group elements $g, g^{x^1}, g^{x^2}, \ldots, g^{x^w}$, where $x \xleftarrow{R} \mathbb{Z}_p^*$, the task is to find an integer $v \in \mathbb{Z}_p^*$ and a group element a such that $a = g^{\frac{1}{x+v}}$. Observe that here the solution is defined by a *rational function* of the secret choices *and* the value v that can be chosen freely. If $\frac{1}{x+v}$ is not defined over $\mathbb{Z}_p$ for particular x and v, then the problem instance is deemed to be not solved.

To let the class of DL-/DH-type problems (Definition 1) cover this problem type we do the following: We first need introduce two additional parameters ℓ' and c' defining the range $\mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}$ from which the algorithm is allowed to choose the value $\mathbf{v}$. Furthermore, we consider structure instance generators $\mathsf{SIGen}_{\mathcal{P}}$ which output two Laurent polynomials Q_1 and Q_2 over $\mathbb{Z}_n$ in the variables $X_1, \ldots, X_\ell, V_1, \ldots V_{\ell'}$, where only the variables $X_{c+1}, \ldots, X_\ell$ and $V_{c'+1}, \ldots, V_{\ell'}$ may appear with negative exponents. These polynomials represent a rational function

$$R : (\mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c}) \times (\mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}) \to \mathbb{Z}_n\,,$$
$$(\mathbf{x}, \mathbf{v}) \mapsto \frac{Q_1(\mathbf{x}, \mathbf{v})}{Q_2(\mathbf{x}, \mathbf{v})}\,.$$

A problem instance of such an extended DL-/DH-type problem is defined as before. Given a problem instance, the challenge is to output some $\mathbf{v} \in \mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}$ with $Q_2(\mathbf{x},\mathbf{v}) \in \mathbb{Z}_n^*$ and the element

$$\begin{cases} \frac{Q_1(\mathbf{x},\mathbf{v})}{(Q_2(\mathbf{x},\mathbf{v}))}, & \text{for a DL-type problem} \\ g_1^{\frac{Q_1(\mathbf{x},\mathbf{v})}{(Q_2(\mathbf{x},\mathbf{v}))}}, & \text{for a DH-type problem} \end{cases}. \tag{2}$$

Adapting most of the framework to the new definition is quite straightforward. In fact, the definition of leak-resistance, the corresponding conditions and theorems stay the same since the definition is completely independent of the challenge polynomial. In the following, we only sketch important differences to the previous version of the conditions.

For this purpose, we need to introduce some new notation: By

$$\mathfrak{F}(\mathfrak{L}_n^{(\ell,c)}) := \left\{ \frac{Q_1}{Q_2} \mid Q_1, Q_2 \in \mathfrak{L}_n^{(\ell,c)},\ Q_2 \text{ is not a zero-divisor} \right\}$$

we denote the ring of fractions of $\mathfrak{L}_n^{(\ell,c)}$. An element $\frac{a}{b} \in \mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$ with $a, b \in \mathbb{Z}_n$ is called a *constant fraction*. The ring $\mathfrak{L}_n^{(\ell,c)}$ can be seen as a subring of this ring by identifying $Q \in \mathfrak{L}_n^{(\ell,c)}$ with $\frac{Q}{1} \in \mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$. Note that if we evaluate the fraction $\frac{Q_1}{Q_2}$ with some $\mathbf{v} \in \mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}$ we obtain a fraction $\frac{Q_1(\mathbf{X},\mathbf{v})}{Q_2(\mathbf{X},\mathbf{v})}$ that is not necessarily a well-defined element of $\mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$. This is because $Q_2(\mathbf{X},\mathbf{v})$ might be a zero-divisor in $\mathfrak{L}_n^{(\ell,c)}$. However, we can exclude this case, because by choosing such a fraction (i.e., by selecting this particular $\mathbf{v}$) an algorithm can never solve a problem instance.

We stipulate the following definitions for the SLP-intractability of a (extended) DL-type and a DH-type problem, respectively. Note that the SLP-generators now additionally output $\mathbf{v}$ in order to select a specific fraction.

Definition 10 (SLP-intractability of DL-Type Problems). *A DL-type problem $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable if for all (Ω, q)-SLP-generators $\mathcal{S}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[\begin{array}{c} Q_2(\mathbf{x},\mathbf{v}) \in \mathbb{Z}_n^* \text{ and} \\ R(\mathbf{x},\mathbf{v}) \equiv a \bmod n \end{array} \middle| \begin{array}{l} (n, \mathbf{I}, Q_1, Q_2) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q_1,Q_2)}(\kappa); \\ (\mathbf{v}, a, L_1, \ldots, L_k) \xleftarrow{R} \mathcal{S}(\kappa, n, \mathbf{I}, Q_1, Q_2); \\ R \leftarrow \frac{Q_1}{Q_2}; \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c} \end{array} \right] \leq \nu(\kappa)$$

Definition 11 (SLP-intractability of DH-type Problems). *A DH-type problem $\mathcal{P}$ is (Ω, q, ν)-SLP-intractable if for all (Ω, q)-SLP-generators $\mathcal{S}$ and $\kappa \in \mathbb{N}$ we have*

$$\Pr\left[\begin{array}{c} Q_2(\mathbf{x},\mathbf{v}) \in \mathbb{Z}_n^* \text{ and} \\ (P - R(\mathbf{X},\mathbf{v}))(\mathbf{x}) \equiv 0 \bmod n \end{array} \middle| \begin{array}{l} (n, \mathbf{I}, Q_1, Q_2) \xleftarrow{R} \mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q_1,Q_2)}(\kappa); \\ (\mathbf{v}, P, L_1, \ldots, L_k) \xleftarrow{R} \mathcal{S}(\kappa, n, \mathbf{I}, Q_1, Q_2); \\ R \leftarrow \frac{Q_1}{Q_2}; \mathbf{x} \xleftarrow{R} \mathbb{Z}_n^c \times (\mathbb{Z}_n^*)^{\ell-c} \end{array} \right] \leq \nu(\kappa)$$

The GGA-intractability of a DL-/DH-type problem is still related in the same way to the leak-resistance property and the SLP-intractability of the problem. That means, Theorem 1 holds unchanged for our extension.

To ensure SLP-intractability, we have Condition 6 and 7 for DL-type problems and Condition 6 and 8 for DH-type problems. These conditions imply (Ω, q, ν)-SLP-intractability for the same negligible functions ν as stated in Theorems 4 and 5.

Condition 6 *There exists* $r_2 \in \mathsf{poly}(x)$ *such that for all* $\kappa \in \mathbb{N}$, $(n, Q_1, Q_2) \in [\mathsf{SIGen}_{\mathcal{P}}^{(n,Q_1,Q_2)}(\kappa)]$, *and* $\mathbf{v} \in \mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}$ *we have* $\max\{\deg(Q_1(\mathbf{X}, \mathbf{v})), \deg(Q_2(\mathbf{X}, \mathbf{v}))\} \leq r_2(\kappa)$.

Condition 7 *There exists* $\kappa_0 \in \mathbb{N}$ *such that for all* $\kappa \geq \kappa_0$, $(n, Q_1, Q_2) \in [\mathsf{SIGen}_{\mathcal{P}}^{(n,Q_1,Q_2)}(\kappa)]$, *and* $\mathbf{v} \in \mathbb{Z}_n^{c'} \times (\mathbb{Z}_n^*)^{\ell'-c'}$ *we have that* $\frac{Q_1(\mathbf{X},\mathbf{v})}{Q_2(\mathbf{X},\mathbf{v})}$ *is not a constant fraction in* $\mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$.

Condition 8 *For every* $q \in \mathsf{poly}(x)$ *there exists* $\kappa_0 \in \mathbb{N}$ *such that for all* $\kappa \geq \kappa_0$, (Ω, q)*-SLP-generators* $\mathcal{S}$, $(n, \mathbf{I}, Q_1, Q_2) \in [\mathsf{SIGen}_{\mathcal{P}}^{(n,\mathbf{I},Q_1,Q_2)}(\kappa)]$, *and* $(\mathbf{v}, P, L_1, \ldots, L_k) \in [\mathcal{S}(\kappa, n, \mathbf{I}, Q_1, Q_2)]$ *we have that* $\frac{Q_1(\mathbf{X},\mathbf{v})}{Q_2(\mathbf{X},\mathbf{v})} \neq P$ *in* $\mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$.

Example 6 (SLP-intractability of w-SDHP). For the w-SDH problem we have parameters $Param_{w\text{-SDH}} = (k = 1, \ell = 1, c = 0, z = w + 1, \ell' = 1, c' = 0)$ and a structure instance generator $\mathsf{SIGen}_{w\text{-SDH}}$ that on input κ returns

$$((\mathbf{G} = G_1, \mathbf{g} = g_1, n = p), (\mathbf{I} = \mathbf{I_1} = \{1, X_1^1, \ldots, X_1^{w(\kappa)}\}, Q_1 = 1, Q_2 = X_1 + V_1)).$$

Note that for any $v_1 \in \mathbb{Z}_p^*$, the fraction $\frac{Q_1(\mathbf{X},\mathbf{v})}{Q_2(\mathbf{X},\mathbf{v})} = \frac{1}{X_1+v_1}$ is an element of $\mathfrak{F}(\mathfrak{L}_n^{(\ell,c)})$ but not an element of the subring $\mathfrak{L}_n^{(\ell,c)}$. Hence, Condition 8 is trivially satisfied, since P is always a Laurent polynomial (independently of the considered operations set Ω). Condition 6 is satisfied since we always have $\max\{\deg(Q_1(\mathbf{X}, \mathbf{v}), \deg(Q_2(\mathbf{X}, \mathbf{v}))\} = 1 =: r_2(\kappa)$. As we can easily see, Conditions 1 and 2 hold assuming an operations set containing operations for performing the group law and inversion of elements in G_1, this yields an upper bound $D(0, 0, 0, w(\kappa), q(\kappa)) = w(\kappa)$ on the degrees of the polynomials $P \in L_1$. Thus, the problem is (Ω, q, ν)-SLP-intractable, where $\nu(\kappa) = 2^{-\kappa}(w(\kappa) + 1)$.