DFA-Based Functional Encryption: Adaptive Security from Dual System Encryption

Somindu C. Ramanna
Indian Statistical Institute
203, B. T. Road, Kolkata - 700108
somindu_r@isical.ac.in

Abstract

We present an adaptively secure functional encryption (FE) scheme based on deterministic finite automata (DFA). The construction uses composite-order bilinear pairings and is built upon the selectively secure DFA-based FE scheme of Waters (Crypto 2012). The scheme is proven secure using the dual system methodology under static subgroup decision assumptions. A dual system proof requires generating of semi-functional components from the instance. In addition, these components must be shown to be properly distributed in an attacker's view. This can be ensured by imposing a restriction on the automata and strings over which the scheme is built i.e., every symbol can appear at most once in a string and in the set of transition tuples of an automata. First a basic construction with the restrictions is obtained and proved to be adaptively secure. We then show how to extend this basic scheme to a full scheme where the restrictions can be relaxed by placing a bound on the number of occurrences of any symbol in a string and in the set of transitions. With the relaxed restrictions, our system supports functionality defined by a larger class of regular languages.

1 Introduction

Functional encryption (FE) is a sophisticated form of public key encryption that provides access control on secret data based on certain policies. A more general form of FE also provides the ability to compute functions over encrypted data (formalised in [BSW12]). In a functional encryption (FE) systems that provide access control, a ciphertext encrypts a message m and an associated attribute or index Ψ that describes the user's credentials. In a *public index model*, the quantity Ψ is revealed in the ciphertext. A key encodes a predicate or an access policy Φ. Decryption succeeds and outputs m if relation $R(\Psi, \Phi)$ holds. User secret keys are issued by a trusted authority called the private key generator (PKG). The form of FE described above is called *key-policy* functional encryption since the policy is encoded in the key. A complementary form called *ciphertext-policy* FE is also studied where the policy is embedded in the ciphertext and index in the key.

Functional encryption schemes with different kinds of functionalities have been studied. These include attribute-based encryption (ABE) [SW05, GPSW06, OSW07, BSW07, Wat11, LW12], inner-product encryption [KSW08, OT09, OT10] and many others in both the ciphertext-policy and key-policy settings. All of these schemes have one property in common – the functions only deal with fixed-size inputs. Moreover, only a few ABE constructions [LOS$^+$10, OT10, LW12, OT12] are known to have adaptive security without random oracles. Waters [Wat12] went beyond fixed-size inputs and proposed a functional encryption scheme that operates over arbitrary-sized inputs. In this system, a secret key is associated with a deterministic finite automaton (DFA) $\mathcal{M}$ and the index Ψ is a string w over the input alphabet of the DFA.

Decryption succeeds if $\mathcal{M}$ accepts w. As a result, the system supports the class of regular languages. This construction was shown to be selectively secure without random oracles based on the eXpanded Decisional Bilinear Diffie-Hellman Exponent (XDBDHE) assumption parametrised by ℓ, the length of the challenge string. Over arbitrary sized inputs, there are no known schemes that achieve adaptive security.

Our Contribution. We construct a DFA-based FE scheme that achieves adaptive security without random oracles. The scheme is built upon composite order pairings that have natural structure (orthogonality and parameter hiding) suitable for dual system proofs. Using the dual system technique, the scheme is proved secure under three static subgroup decision assumptions over composite-order pairings.

First of all, let us see why a direct adaptation of dual system method fails. Consider a system with Σ as the alphabet. Since most DFAs used in practice have small alphabets, we can pick a group element H_σ corresponding to each symbol H_σ and include these elements in the public parameters. Let $w = w_1 \cdots w_\ell$ be a string over Σ to which a ciphertext $\mathcal{C}$ is encrypted and $\mathcal{SK}_\mathcal{M}$, a secret key for an automaton $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$. String w is encoded in $\mathcal{C}$ in such a way that the order of symbols is also maintained. Suppose that we attempt defining semi-functional components in the usual way. In the dual system method, semi-functional components for ciphertexts and keys usually mimic the structure of the normal ciphertexts and keys respectively. But these are generated using some secret elements so that their distribution is statistically hidden from the adversary. Since there is a single group element (hash H_σ) for each symbol σ, there will be a corresponding scalar in the semi-functional portion for each symbol during simulation. If symbols are repeated, then so are these scalars. But giving out too many copies of these values will reveal them information theoretically to the attacker which defeats the dual system proof. This holds for both strings and automata.

The solution to this problem is to restrict the number of occurrences of symbols in transitions and strings during system setup. We adapt a technique previously used by Lewko *et.al.* [LOS+10] in the context of attribute-based encryption over monotone access structures. A string w can contain at most one occurrence of each $\sigma \in \Sigma$. Similarly, at most one transition can contain a symbol σ. We call the resulting construction the *basic construction*, denoted $\mathcal{BFE}$. This scheme supports only an extremely small class of languages. For instance, consider the alphabet $\{0, 1\}$. With the single-use restriction, then the scheme works for only 4 strings - $0, 1, 01, 10$! Nevertheless, this restriction can be relaxed and we show this by via our next (full) construction, $\mathcal{FFE}$. This scheme is obtained by putting a bound on the number of occurrences of each symbol in strings as well as transitions at setup. Suppose a symbol can appear at most $\mathsf{s}_{\mathsf{max}}$ times in a string and at most $\mathsf{t}_{\mathsf{max}}$ times in the set of transitions. Then our public parameters will contain $\mathsf{s}_{\mathsf{max}} \times \mathsf{t}_{\mathsf{max}}$ group elements corresponding to each symbol. Essentially H_σ is replaced by a matrix $\mathbf{H}_\sigma$ of order $\mathsf{s}_{\mathsf{max}} \times \mathsf{t}_{\mathsf{max}}$. Ciphertext and key are defined for w and $\mathcal{M}$ (respectively) in such a way that only one acceptance path and hence decryption sequence exists if $\mathcal{M}$ accepts w. Also, if $\mathcal{M}$ rejects w, then there is no way to decrypt. Since each entry in $\mathbf{H}_\sigma$ is distinct, simulating semi-functional components will no longer be a problem. If we assume $\mathsf{s}_{\mathsf{max}}$ and $\mathsf{t}_{\mathsf{max}}$ to be linear in κ, the security parameter, then this scheme supports a significantly large class of functionalities. Although the selectively secure scheme of [Wat12] supports unbounded functionality, security is only limited to bounded functionality for otherwise the ℓ-XDBDHE assumption becomes meaningless. On the other hand, our system is limited to bounded functionality in the construction itself and in addition is adaptively security.

2 Preliminaries

This section provides basic notation, definitions and complexity assumptions in composite-order pairings. Definition and security model for DFA-based functional encryption can be found in Appendix A.

Definition 2.1 (Deterministic Finite Automaton)**.** A deterministic finite automaton (DFA) $\mathcal{M}$ is a 5-tuple $(Q, \Sigma, q_0, F, \delta)$ where $Q \neq \varnothing$ is a finite set of *states*, $\Sigma \neq \varnothing$ denotes the *input alphabet*, $q_0 \in Q$ is the *start state*, $\varnothing \neq F \subseteq Q$ is the set of *final states* and $\delta : Q \times \Sigma \to Q$ is called the *transition function*.

It is well-known [HMU00] that any DFA $\mathcal{M}$, one can construct $\mathcal{M}'$ such that $\mathcal{M}'$ has a unique final state and both $\mathcal{M}$ and $\mathcal{M}'$ accept the same set of languages.

2.1 Notation

A composite order pairing is represented as a tuple $(p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ where p_1, p_2, p_3 prime, $|\mathbb{G}| = |\mathbb{G}_T| = N = p_1p_2p_3$, $G = \langle G \rangle$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is the pairing function. Define $\mathcal{G}_{\text{pub}} = (N, \mathbb{G}, \mathbb{G}_T, e, G)$ where $N = p_1p_2p_3$. Also let $\mathbb{G}_B$ denote the subgroup of order B of $\mathbb{G}$. This representation is particular to those pairings where the group order is a product of three distinct primes. In general, the order could be any composite number that is hard to factor. We denote elements of groups $\mathbb{G}_{p_2}, \mathbb{G}_{p_3}$ with subscripts 2 and 3 respectively. Elements of $\mathbb{G}_{p_1}$ and $\mathbb{G}$ are written without a subscript. The meaning will be clear from the context.

Our construction is based on DFAs that have a unique final state. We thus use the notation $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$ with q_f being the final state. Transitions of an automaton $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$ are represented as 3-tuples of the form $t = (q_x, q_y, \sigma)$ where $\delta(q_x, \sigma) = \{q_y\}$. Let $\mathcal{T}$ denote the set of all transition tuples t.

The notation $[a, b]$ represents the set $\{a, a+1, a+2, \ldots, b\}$ for two integers $a < b$. For a set $\mathcal{X}$, the notation $x_1, \ldots, x_k \xleftarrow{\text{R}} \mathcal{X}$ symbolises $x_1, \ldots, x_k$ being sampled independently from $\mathcal{X}$ according to distribution R. We use this interchangeably with the notation $x_1, \ldots, x_k \in_{\text{R}} \mathcal{X}$. The uniform distribution is denoted by U. For a (probabilistic) algorithm $\mathcal{A}$, $x \longleftarrow \mathcal{A}(\cdot)$ means that x is chosen according to the output distribution of $\mathcal{A}$ (which of course may be determined by its input).

2.2 Complexity Assumptions

We state three Decisional SubGroup (DSG) assumptions in composite order groups equipped with a bilinear pairing. Each of the following problems is defined based on a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ generated according to some distribution.

Assumption DSG1

Define a distribution $\mathcal{D}$ as follows: $P \xleftarrow{\text{U}} \mathbb{G}_{p_1}$, $P_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$, $\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_3)$. For an algorithm $\mathscr{A}$ that returns a bit, define its advantage in solving the DSG1 problem as

$$\mathsf{Adv}^{\text{DSG1}}_{\mathcal{G}}(\mathscr{A}) = \left|\Pr[\mathscr{A}(\mathcal{D}, T_1) = 1] - \Pr[\mathscr{A}(\mathcal{D}, T_2) = 1]\right|,$$

where $T_1 \in_{\text{U}} \mathbb{G}_{p_1}$ and $T_2 \in_{\text{U}} \mathbb{G}_{p_1p_2}$. The (t, ε)-DSG1 assumption is said to hold if for every algorithm $\mathscr{A}$ running in time at most t,

$$\mathsf{Adv}^{\text{DSG1}}_{\mathcal{G}}(\mathscr{A}) \leq \varepsilon.$$

Assumption DSG2

Define a distribution $\mathcal{D}$ as follows:

$$P, X \xleftarrow{\text{U}} \mathbb{G}_{p_1},\ P_2, X_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2},\ P_3, X_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3},$$

$$\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_3, X + P_2, X_2 + X_3).$$

For an algorithm $\mathscr{A}$ that returns a bit, define its advantage in solving the DSG1 problem as

$$\mathsf{Adv}_{\mathcal{G}}^{\text{DSG1}}(\mathscr{A}) = \left|\Pr[\mathscr{A}(\mathcal{D}, T_1) = 1] - \Pr[\mathscr{A}(\mathcal{D}, T_2) = 1]\right|,$$

where $T_1 \in_{\text{U}} \mathbb{G}_{p_1 p_2}$ and $T_2 \in_{\text{U}} \mathbb{G}$. The (t, ε)-DSG2 assumption is said to hold if for every algorithm $\mathscr{A}$ running in time at most t,

$$\mathsf{Adv}_{\mathcal{G}}^{\text{DSG2}}(\mathscr{A}) \leq \varepsilon.$$

Assumption DSG3
Define a distribution $\mathcal{D}$ as follows:

$$\alpha, s \xleftarrow{\text{U}} \mathbb{Z}_N,\ P \xleftarrow{\text{U}} \mathbb{G}_{p_1},\ P_2, X_2, Y_2 \xleftarrow{\text{U}} \mathbb{G}_{p_2},\ P_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3},$$

$$\mathcal{D} = (\mathcal{G}_{\text{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2).$$

For an algorithm $\mathscr{A}$ that returns a bit, define its advantage in solving the DSG1 problem as

$$\mathsf{Adv}_{\mathcal{G}}^{\text{DSG1}}(\mathscr{A}) = \left|\Pr[\mathscr{A}(\mathcal{D}, e(P,P)^{\alpha s}) = 1] - \Pr[\mathscr{A}(\mathcal{D}, X_T) = 1]\right|,$$

where $T_1 \in_{\text{U}} \mathbb{G}_T$. The (t, ε)-DSG3 assumption is said to hold if for every algorithm $\mathscr{A}$ running in time at most t,

$$\mathsf{Adv}_{\mathcal{G}}^{\text{DSG3}}(\mathscr{A}) \leq \varepsilon.$$

3 Basic Construction

Described here is a basic construction of DFA-based functional encryption scheme $\mathcal{BFE} = (\mathcal{BFE}.\mathsf{Setup}, \mathcal{BFE}.\mathsf{KeyGen}, \mathcal{BFE}.\mathsf{Encrypt}, \mathcal{BFE}.\mathsf{Decrypt})$ in the composite order pairing setting. We impose the following restrictions on automata and strings over which the scheme is built.

Restriction 1: Keys are created only for automata with a unique final state and a single transition corresponding to each symbol

Restriction 2: Input string (part of the ciphertext) can contain only a single occurrence of each symbol

These restrictions are required for the proof to go through. In Section 5, we describe how to extend the basic scheme $\mathcal{BFE}$ to a full scheme $\mathcal{FFE}$ with relaxed restrictions and similar security guarantee.

The construction is similar to that of Waters [Wat12]. Encryption is done in the group $\mathbb{G}_{p_1}$ but the structure is different from that of [Wat12]. Components of the key are elements of $\mathbb{G}_{p_1 p_3}$ and have the same structure as the keys in [Wat12] except that they are additionally randomised by elements of $\mathbb{G}_{p_3}$. The group $\mathbb{G}_{p_2}$ forms the semi-functional space.

$\mathcal{BFE}.\mathsf{Setup}(\Sigma, \kappa)$: Generate a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ according to the security parameter κ. Choose elements $P, H_{\text{start}}, H_{\text{end}}, (H_\sigma, U_\sigma)_{\sigma \in \Sigma} \xleftarrow{\text{U}} \mathbb{G}_{p_1}$, $P_3 \xleftarrow{\text{U}} \mathbb{G}_{p_3}$ and $\alpha \xleftarrow{\text{U}} \mathbb{Z}_N$. The public parameters and master secret are given by

$\mathcal{PP}$: $(\mathcal{G}_{\text{pub}}, \Sigma, P, H_{\text{start}}, H_{\text{end}}, H_\lambda, (H_\sigma, U_\sigma)_{\sigma \in \Sigma}, e(P,P)^\alpha)$,
$\mathcal{MSK}$: $(-\alpha P, P_3)$.

In [Wat12], only a single element U was uses to maintain the link between consecutive symbols but here we require an separate group element U_σ corresponding to each symbol σ. This is helpful in the dual system proof.

$\mathcal{BFE}.\mathsf{Encrypt}(\mathcal{PP}, w = w_1 \cdots w_\ell, m)$: Choose randomisers $s_0, s_1, \ldots, s_\ell \xleftarrow{\text{U}} \mathbb{Z}_N$. Compute the ciphertext elements as follows.

$$C_m = m \cdot e(P,P)^{\alpha s_\ell},$$

$$C_{0,1} = C_{\text{start},1} = s_0 P, \quad C_{\text{start},2} = s_0 H_{\text{start}},$$

For $i = 1, \ldots, \ell$,
$$C_{i,1} = s_i P, \quad C_{i,2} = s_i H_{w_i} + s_{i-1} U_{w_i},$$

$$C_{\text{end},1} = C_{\ell,1} = s_\ell P, \quad C_{\text{end},2} = s_\ell H_{\text{end}}.$$

The ciphertext is given by $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,1}, C_{i,2})_{i \in [1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$.

$\mathcal{BFE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M} = (Q, \Sigma, q_0, q_f, \delta))$: For each $x \in \mathbb{Z}_{|Q|}$, pick $D_x \xleftarrow{\text{U}} \mathbb{G}_{p_1}$. Choose elements r_{start}, for all $t \in \mathcal{T}$, r_t and r_{end} uniformly and independently at random from $\mathbb{Z}_N$. Let $R_{\text{start},1}, R_{\text{start},2}$, $(R_{t,1}, R_{t,2}, R_{t,3})_{t \in \mathcal{T}}$ and $R_{\text{end},1}, R_{\text{end},2}$ be randomly chosen elements of $\mathbb{G}_{p_3}$. Compute the elements of the key as follows.

$$K_{\text{start},1} = D_0 + r_{\text{start}} H_{\text{start}} + R_{\text{start},1}, \quad K_{\text{start},2} = r_{\text{start}} P + R_{\text{start},2},$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$$K_{t,1} = -D_x + r_t U_\sigma + R_{t,1}, \quad K_{t,2} = r_t P + R_{t,2}, \quad K_{t,3} = D_y + r_t H_\sigma + R_{t,3},$$

$$K_{\text{end},1} = -\alpha P + D_f + r_{\text{end}} H_{\text{end}} + R_{\text{end},1}, \quad K_{\text{end},2} = r_{\text{end}} P + R_{\text{end},2}.$$

Here D_f corresponds to the final state q_f. The secret key for automaton $\mathcal{M}$ is given by $\mathcal{SK}_\mathcal{M} = (K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t \in \mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$.

$\mathcal{BFE}.\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_\mathcal{M})$: Suppose that $\mathsf{Accept}(\mathcal{M}, w) = 1$ and $w = w_1 \cdots w_\ell$. Then there exists a sequence of transitions $t_1, t_2, \ldots, t_\ell$ with $t_i = (q_{x_{i-1}}, q_{x_i}, w_i)$ where $x_0 = 0$ and $x_\ell = f$. Decryption consists of several stages of computation. First compute

$$\begin{aligned} A_0 &= e(C_{\text{start},1}, K_{\text{start},1}) e(C_{\text{start},2}, K_{\text{start},2})^{-1} \\ &= e(P, D_0)^{s_0} \end{aligned}$$

Then compute intermediate values A_i (for $i = 1, \ldots, \ell$) as follows.

$$\begin{aligned} A_i &= A_{i-1} \cdot e(C_{i-1,1}, K_{t_i,1}) e(C_{i,2}, K_{t_i,2})^{-1} e(C_{i,1}, K_{t_i,3}) \\ &= e(P, D_{x_i})^{s_i} \end{aligned}$$

The last intermediate $A_{\ell+1}$ is computed as

$$A_{\ell+1} = e(C_{\text{end},1}, K_{\text{end},1}) \cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} = e(P,P)^{\alpha s_\ell} e(D_f, P)^{s_\ell}.$$

Using A_ℓ and $A_{\ell+1}$ the message is unmasked as shown below.

$$m = C_m \cdot A_{\ell+1}^{-1} \cdot A_\ell.$$

Correctness. To show that decryption is correct, we need to show that the intermediate values $A_0, A_{\ell+1}$ and A_i for $i \in [1,\ell]$ have the claimed structure. It is enough to show that if A_{i-1} has the right structure, then so does A_i. By induction on i, it follows that $A_\ell = e(P, D_{x_\ell})^{s_\ell}$ for $i \in [1,\ell]$.

$$
\begin{aligned}
A_0 &= e(C_{\text{start},1}, K_{\text{start},1})e(C_{\text{start},2}, K_{\text{start},2})^{-1} \\
&= e(s_0 P, D_0 + r_{\text{start}} H_{\text{start}} + R_{\text{start},1})e(s_o H_{\text{start}}, r_{\text{start}} P + R_{\text{start},2})^{-1} \\
&= e(P, D_0)^{s_0} e(P, H_{\text{start}})^{s_0 r_{\text{start}}} e(H_{\text{start}}, P)^{-s_0 r_{\text{start}}} \\
&= e(P, D_0)^{s_0} \\
A_i &= A_{i-1} \cdot e(C_{i-1,1}, K_{t_i,1})e(C_{i,2}, K_{t_i,2})^{-1} e(C_{i,1}, K_{t_i,3}) \\
&= e(P, D_{x_{i-1}})^{s_{i-1}} e(s_{i-1} P, -D_{x_{i-1}} + r_{t_i} U_{w_{i-1}} + R_{t_i,1})e(s_i H_{w_i} + s_{i-1} U_{w_i}, r_{t_i} P + R_{t_i,2})^{-1} \\
&\quad e(s_i P, D_{x_i} + r_{t_i} H_{w_i} + R_{t_i,3}) \\
&= e(P, D_{x_{i-1}})^{s_{i-1}} e(P, D_{x_{i-1}})^{-s_{i-1}} e(P, U_{w_{i-1}})^{s_{i-1} r_{t_i}} e(H_{w_i}, P)^{-s_i r_{t_i}} e(U_{w_i}, P)^{-s_{i-1} r_{t_i}} e(P, D_{x_i})^{s_i} e(P, H_{w_i})^{s_i r_{t_i}} \\
&= e(P, D_{x_i})^{s_i} \\
A_{\ell+1} &= e(C_{\text{end},1}, K_{\text{end},1}) \cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} \\
&= e(s_\ell P, -\alpha P + D_f + r_{\text{end}} H_{\text{end}} + R_{\text{end},1})e(s_\ell H_{\text{end}}, r_{\text{end}} P + R_{\text{end},2})^{-1} \\
&= e(P, P)^{-\alpha s_\ell} e(P, D_f)^{s_\ell} e(P, H_{\text{end}})^{s_\ell r_{\text{end}}} e(H_{\text{end}}, P)^{-s_\ell r_{\text{end}}} \\
&= e(P, P)^{\alpha s_\ell} e(D_f, P)^{s_\ell}
\end{aligned}
$$

Note that $\mathbb{G}_{p_3}$ components get cancelled due to the orthogonality property of composite order groups.

Ciphertext-Policy FE. It is possible to obtain a ciphertext-policy FE scheme by constructing a dual of the above scheme. The structure of the ciphertext and key get interchanged. A key will encode a string w and a ciphertext will encode an automaton $\mathcal{M}$. Also, randomisation in $\mathbb{G}_{p_3}$ is done only for the key (i.e., components corresponding to the input string w). The same assumptions can also be used for the proof of security.

4 Security Proof

We prove security of $\mathcal{BFE}$ using the method of dual system encryption [Wat09]. This requires defining semi-functional ciphertexts and keys.

4.1 Defining Semi-Functionality

Two types of semi-functional keys need to be defined for our proof of security – Type-1 and Type-2. Let P_2 be a random generator of the group $\mathbb{G}_{p_2}$ and

$$\pi_{\text{start}}, (\pi_{h,\sigma}, \pi_{u,\sigma})_{\sigma \in \Sigma} \xleftarrow{\text{U}} \mathbb{Z}_N.$$

These scalars are common to both semi-functional keys and ciphertexts.

Semi-functional Ciphertext

Pick $\gamma_0, \ldots, \gamma_\ell, \pi_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N$. Semi-functional ciphertext is obtained by modifying normally generated ciphertext $\mathcal{C} = (C_m, C_{\text{start},1}, C_{\text{start},2}, (C_{i,1}, C_{i,2})_{i \in [1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$ as:

$$C_{\text{start},1} \leftarrow C_{\text{start},1} + \gamma_0 P_2, \quad C_{\text{start},2} \leftarrow C_{\text{start},2} + \gamma_0 \pi_{\text{start}} P_2,$$

For $i = 1, \ldots, \ell$,
$C_{i,1} \leftarrow C_{i,1} + \gamma_i P_2, \quad C_{i,2} \leftarrow C_{i,2} + (\gamma_i \pi_{h,w_i} + \gamma_{i-1}\pi_{u,w_i})P_2,$

$C_{\text{end},1} \leftarrow C_{\text{end},1} + \gamma_\ell P_2, \quad C_{\text{end},2} \leftarrow C_{\text{end},1} + \pi_{\text{end}} P_2.$

C_m remains unchanged. Restriction 2 mentioned in Section 3 is required here to ensure that only one value of $\pi_{h,\sigma}$ or $\pi_{u,\sigma}$ is revealed for any $\sigma \in \Sigma$ in the challenge ciphertext. Keeping value of $\pi_{\cdot,\sigma}$ statistically hidden is very essential for the security argument. On the other hand, providing too many copies of $\pi_{\cdot,\sigma}$ would information theoretically reveal its value to the adversary.

Type-1 Semi-functional Key

Let $\mu_{\text{start}}, \mu_{\text{end}}, (\mu_t)_{t\in\mathcal{T}}, \tau_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N$, $(z_x)_{q_x \in Q} \xleftarrow{\text{U}} \mathbb{Z}_N$ and $\mathcal{SK}_{\mathcal{M}} = (K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$ be a normal key generated by the $\mathcal{BFE}.\mathsf{KeyGen}$ algorithm. Its components are modified as:

$K_{\text{start},1} \leftarrow K_{\text{start},1} + (z_0 + \mu_{\text{start}}\pi_{\text{start}})P_2, \quad K_{\text{start},2} \leftarrow K_{\text{start},2} + \mu_{\text{start}} P_2,$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$K_{t,1} \leftarrow K_{t,1} + (z_x + \mu_t \pi_{u,\sigma})P_2, \quad K_{t,2} \leftarrow K_{t,2} + \mu_t P_2, \quad K_{t,3} \leftarrow K_{t,3} + (z_y + \mu_t \pi_{h,\sigma})P_2 \ ,$

$K_{\text{end},1} \leftarrow K_{\text{end},1} + (z_f + \tau_{\text{end}})P_2, \quad K_{\text{end},2} \leftarrow K_{\text{end},2} + \mu_{\text{end}} P_2.$

The first restriction plays a crucial role here. It ensures that the π-values are statistically hidden from the adversary.

Type-2 Semi-functional Key

Type 2 semi-functional keys are similar to Type-1 except that the components $K_{\text{start},1}, K_{\text{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}$ will no longer have any semi-functional terms. Also, $K_{\text{end},1}$ does not contain the scalar z_f.

In the proof, it is ensured that at most one key can be Type-1 semi-functional at any point in the hybrid sequence of games. The rest of the semi-functional keys are Type-2. Otherwise, multiple copies of the π-values would have to be provided to the adversary and the whole purpose of imposing the two restrictions would be defeated.

Consider decryption of a ciphertext $\mathcal{C}$ for message m and string $w = w_1 \cdots w_\ell$ by a key $\mathcal{SK}_{\mathcal{M}}$ where $\mathsf{Accept}(\mathcal{M}, w) = 1$. Decryption succeeds unless both $\mathcal{C}$ and $\mathcal{SK}_{\mathcal{M}}$ semi-functional. This is because $\mathbb{G}_{p_2}$ (semi-functional) components get cancelled when paired with elements of $\mathbb{G}_{p_1}$ (by orthogonal property of composite order pairing groups). When both $\mathcal{C}$ and $\mathcal{SK}_{\mathcal{M}}$ are semi-functional, the message is masked by an extra factor - $e(P_2, P_2)^{(\mu_{\text{end}}\pi_{\text{end}} - \gamma_\ell \tau_{\text{end}})}$. To see this, note that all other semi-functional components get cancelled since they only mimic the structure of the ciphertext and key, in addition to having π-values common. Decryption will succeed only if $\mu_{\text{end}}\pi_{\text{end}} = \gamma_\ell \tau_{\text{end}}$. We will call such a pair of ciphertext and key as *nominally semi-functional*.

We require algorithms $\mathsf{ReRandCT}$ and $\mathsf{ReRandK}$ for randomising ciphertexts and keys respectively in the proof to ensure correct distribution of components. Essentially, these algorithms additively rerandomise ciphertexts and keys. These are defined in Appendix B.

4.2 Reductions

We prove $\mathsf{IND\text{-}STR\text{-}CPA}$-security of $\mathcal{BFE}$ under the three assumptions DSG1, DSG2 and DSG3.

Theorem 4.1. *If the* (ε_1, t')-DSG1, (ε_2, t')-DSG2, (ε_3, t')-DSG3 *assumptions hold, then* $\mathcal{BFE}$ *is* (ε, t, ν)-IND-STR-CPA *secure where*

$$\varepsilon \leq \varepsilon_1 + 2\nu\varepsilon_2 + \varepsilon_3$$

and $t = t' - O(\nu\Sigma\rho)$*, where* ρ *is an upper bound on the time required for one scalar multiplication in* $\mathbb{G}$*.*

Proof. The proof is organised as a hybrid argument over a sequence of $2\nu + 3$ games – $\mathrm{Game}_{real}, \mathrm{Game}_{0,1}, (\mathrm{Game}_{k,0}, \mathrm{Game}_{k,1})_{k=1}^{\nu}, \mathrm{Game}_{final}$. Game_{real} denotes the actual CPA-security game for DFA-based FE ind-cpa. $\mathrm{Game}_{0,1}$ is just like Game_{real} except that the challenge ciphertext is semi-functional. In $\mathrm{Game}_{k,0}$ (for $1 \leq k \leq \nu$), challenge ciphertext is semi-functional, the first $k-1$ keys returned to the adversary are Type-2 semi-functional, k-th key Type-1 semi-functional and the rest are normal. $\mathrm{Game}_{k,1}$ $(1 \leq k \leq \nu)$ is such that first k keys are Type-2 semi-functional and rest are normal. Game_{final} is similar to $\mathrm{Game}_{\nu,1}$ except that now the challenge ciphertext is a semi-functional encryption of a random message. Let $\mathcal{E}_{\square}$ denote the events that the adversary wins in $\mathrm{Game}_{\square}$. Note that, in Game_{final}, the challenge ciphertext is an encryption of a random message and hence bit β is statistically hidden from the adversary's view implying that $\Pr[\mathcal{E}_{final}] = 1/2$.

The advantage of an t-time adversary $\mathscr{A}$ in winning the ind-cpa against the FE scheme in the ind-cpa, is given by

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathrm{FE}}(\mathscr{A}) = \left|\Pr[\mathcal{E}_{actual}] - \frac{1}{2}\right|.$$

We have

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{BFE}}(\mathscr{A}) &= \left|\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{final}]\right| \\
&\leq \left|\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]\right| + \sum_{k=1}^{\nu} \left(\left|\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]\right| + \left|\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]\right|\right) \\
&\quad + \left|\Pr[\mathcal{E}_{\nu}] - \Pr[\mathcal{E}_{final}]\right| \\
&\leq \varepsilon_{\mathrm{DSG1}} + 2\nu\varepsilon_{\mathrm{DSG2}} + \varepsilon_{\mathrm{DSG3}}
\end{aligned}$$

The last inequality follows from the lemmas 4.2, 4.3, 4.4 and 4.5. □

In all the lemmas, $\mathscr{A}$ is a t-time adversary against the FE scheme and $\mathscr{B}$ is an algorithm running in time t' that interacts with $\mathscr{A}$ and solves one of the three problems DSG1, DSG2 or DSG3.

Lemma 4.2. $\left|\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]\right| \leq \varepsilon_1$.

Proof. $\mathscr{B}$ receives an instance of problem DSG1, $(\mathcal{G}_{\mathrm{pub}}, P, P_3, T)$, where $T = \theta P + \theta_2 P_2$ and its task is to decide whether $\theta_2 = 0$ or $\theta_2 \in_{\mathrm{U}} \mathbb{Z}_{p_2}$. The different phases of the game are simulated as described below.

Setup: $\mathscr{B}$ picks $\alpha, v_{\mathrm{start}}, v_{\mathrm{end}}, \{v_{h,\sigma}, v_{u,\sigma}\}_{\sigma\in\Sigma} \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, sets $H_{\mathrm{start}} = v_{\mathrm{start}}P$, $H_{\mathrm{end}} = v_{\mathrm{end}}P$, $H_\sigma = v_{h,\sigma}P$ and $U_\sigma = v_{u,\sigma}P$. It provides $\mathcal{PP}$ to $\mathscr{A}$ and computes $\mathcal{MSK}$.

Key extraction queries: For a query on automaton $\mathcal{M}$, $\mathscr{B}$ runs the $\mathcal{BFE}$.KeyGen algorithm with input $\mathcal{M}$ and returns the output to $\mathscr{A}$. No generator of $\mathbb{G}_{p_2}$ is provided to $\mathscr{B}$ and hence semi-functional keys cannot be generated.

Challenge: $\mathscr{A}$ provides two messages m_0, m_1, challenge string $w_1^* \cdots w_{\ell^*}^*$. $\mathscr{B}$ chooses $\beta \xleftarrow{\mathrm{U}} \{0,1\}$, $s'_0, \ldots, s'_{\ell^*} \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and encrypts m_β to w^* as follows.

$C_m = m_\beta \cdot e(P,T)^{\alpha s'_{\ell^*}}$,

$C_{0,1} = s'_0 T, \quad C_{\text{start},2} = s'_0 v_{\text{start}} T,$

For $i = 1, \ldots, \ell^*$,
$C_{i,1} = s'_i T, \quad C_{i,2} = (s'_i v_{h,w_i} + s'_{i-1} v_{u,w_i}) T,$

$C_{\text{end},1} = C_{\ell,1}, \quad C_{\text{end},2} = s'_\ell v_{\text{end}} T.$

Randomiser s_i is inherently set to $s'_i \theta$ for $i = 0, \ldots, \ell^*$. Let $\mathcal{C}^* = (C_m, C_{\text{start},1}, C_{\text{start},2}, \{C_{i,1}, C_{i,2}\}_{i \in [1,\ell]}, C_{\text{end},1}, C_{\text{end},2}, w)$. $\mathscr{B}$ returns $\mathsf{ReRandCT}(\mathcal{C}^*)$ to $\mathscr{A}$.

Guess: $\mathscr{A}$ returns its guess β'.

If $\theta_2 = 0$, then $\mathcal{C}^*$ is a normal encryption of m_β. Otherwise $\theta_2 \in_{\text{U}} \mathbb{Z}_{p_2}$ making $\mathcal{C}^*$ a semi-functional ciphertext for m_β with $\gamma_i = s'_i \theta_2$ for $i = 1, \ldots, \ell^*$, $\pi_{\text{start}} = v_{\text{start}}$, $\pi_{\text{end}} = s_{\ell'} v_{\text{end}}$, $\pi_{u,\sigma} = v_{u,\sigma}$ and $\pi_{h,\sigma} = v_{h,\sigma}$ for all $\sigma \in \Sigma$. The ciphertext is well-formed. For instance,

$$\begin{aligned} C_{i,2} &= (s'_i v_{h,w_i} + s'_{i-1} v_{u,w_i}) T \\ &= s'_i v_{h,w_i} \theta P + s'_{i-1} v_{u,w_i} \theta P + s'_i v_{h,w_i} \theta_2 P_2 + s'_{i-1} v_{u,w_i} \theta_2 P_2 \\ &= s_i H_{w_i} + s_{i-1} U_{w_i} + (\gamma_i \pi_{h,w_i} + \gamma_{i-1} \pi_{u,w_i}) P_2 \end{aligned}$$

The rest of the components can be shown to be well-formed in a similar way. The v's are embedded in the public parameters and hence their values modulo p_1 are revealed to the adversary in an information theoretic sense. However their values modulo p_2 remain hidden (by Chinese remainder theorem) thus resulting in the proper distribution of the π's. The s_i's are merely scaled by θ_2 to obtain γ_i's and hence the γ_i's are uniformly and independently distributed. The randomisers for the ciphertext's normal components are also properly distributed since it is rerandomised.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have

$$\begin{aligned} \varepsilon_1 \geq \mathsf{Adv}^{\text{DSG1}}_{\mathcal{G}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T \in_{\text{U}} \mathbb{G}_{p_1}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \in_{\text{U}} \mathbb{G}_{p_1 p_2}]| \\ &= |\Pr[\mathscr{A} \text{ wins} \mid T \in_{\text{U}} \mathbb{G}_{p_1}] - \Pr[\mathscr{A} \text{ wins} \mid T \in_{\text{U}} \mathbb{G}_{p_1 p_2}]| \\ &= |\Pr[\mathscr{A} \text{ wins in } \text{Game}_{actual}] - \Pr[\mathscr{A} \text{ wins in } \text{Game}_{0,1}]| \\ &= |\Pr[\mathcal{E}_{actual}] - \Pr[\mathcal{E}_{0,1}]| \end{aligned}$$

as required. □

Lemma 4.3. $|\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]| \leq \varepsilon_2$ *for* $1 \leq k \leq \nu$.

Proof. An $(\mathcal{G}_{\text{pub}}, P, P_3, X + P_2, X_2 + X_3, T)$ of DSG2 is given to $\mathscr{B}$ and the goal is to decide whether $T \in_{\text{U}} \mathbb{G}_{p_1 p_3}$ or $T \in_{\text{U}} \mathbb{G}$. In other words, if $T = \theta P + \theta_2 P_2 + \theta_3 P_3$ then $\mathscr{B}$ has to determine whether $\theta_3 = 0$ or $\theta_3 \in_{\text{U}} \mathbb{Z}_{p_3}$.

Setup: Scalars $\alpha, v_{\text{start}}, v_{\text{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma \in \Sigma}$ are chosen from $\mathbb{Z}_N$ independently according to the uniform distribution. Parameters are set as follows: $H_{\text{start}} = v_{\text{start}} P$, $H_{\text{end}} = v_{\text{end}} P$, $H_\sigma = v_{h,\sigma} P$ and $U_\sigma = v_{u,\sigma} P$. $\mathcal{PP}$ is given to $\mathscr{A}$ and $\mathscr{B}$ keeps $\mathcal{MSK}$.

Key extraction queries: Suppose $\mathscr{A}$ makes key extraction queries on $\mathcal{M}_1, \ldots, \mathcal{M}_\nu$. $\mathscr{B}$ generates key for $\mathcal{M}_i$ depending on i as follows.

Case $i > k$: $\mathscr{B}$ runs the $\mathcal{BFE}.\mathsf{KeyGen}$ algorithm and returns the resulting (normal) key to $\mathscr{A}$.

Case $i < k$: $\mathscr{B}$ first obtains $\mathcal{SK}_{\mathcal{M}_i} \longleftarrow \mathcal{BFE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M}_i)$ and then modifies its components to obtain a Type-2 semi-functional key for $\mathcal{M}_i$ as follows. Since a generator of $\mathbb{G}_{p_2}$ is not available, $\mathscr{B}$ uses element $X_2 + X_3$ to construct the semi-functional components.

$\mu'_{\text{end}}, \tau'_{\text{end}} \xleftarrow{\text{U}} \mathbb{Z}_N$,
$K_{\text{end},1} \leftarrow K_{\text{end},1} + \tau'_{\text{end}}(X_2 + X_3), \quad K_{\text{end},2} \leftarrow K_{\text{end},2} + \mu'_{\text{end}}(X_2 + X_3)$.

The term $\mu_{\text{end}} P_2$ is set to $\mu'_{\text{end}} X_2$. Similarly, $\tau_{\text{end}} P_2 = \tau'_{\text{end}} X_2$. The components $K_{\text{end},1}, K_{\text{end},2}$ already have uniform random elements of $\mathbb{G}_{p_3}$ embedded in them. Hence adding multiples of X_3 will not change the distribution of the $\mathbb{G}_{p_3}$ components.

Case $i = k$: $\mathscr{B}$ computes $\mathcal{SK}_{\mathcal{M}_k}$ embedding the challenge T from the instance.

For each $x \in \mathbb{Z}_{|Q|}$, $d_x \xleftarrow{\text{U}} \mathbb{Z}_N$
$r'_{\text{start}}, r'_{\text{end}}, \{r'_t\}_{t \in \mathcal{T}} \xleftarrow{\text{U}} \mathbb{Z}_N$

$K_{\text{start},1} = (d_0 + r'_{\text{start}} v_{\text{start}})T, \quad K_{\text{start},2} = r'_{\text{start}} T,$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$K_{t,1} = (-d_x + r'_t v_{u,\sigma})T, \quad K_{t,2} = r'_t T, \quad K_{t,3} = (d_y + r'_t v_{h,\sigma})T,$

$K_{\text{end},1} = -\alpha P + (d_f + r'_{\text{end}} v_{\text{end}})T, \quad K_{\text{end},2} = r'_{\text{end}} T.$

Let $\mathcal{SK}_{\mathcal{M}} = (K_{\text{start},1}, K_{\text{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t \in \mathcal{T}}, K_{\text{end},1}, K_{\text{end},2})$. $\mathscr{B}$ returns $\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}_k})$ to $\mathscr{A}$. We have $T = \theta P + \theta_2 P_2 + \theta_3 P_3$ where θ_2 could be zero. Hence every component is made up of elements of $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_3}$ and possibly elements of $\mathbb{G}_{p_2}$. The $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ elements are properly distributed due to the invocation of $\mathsf{ReRandK}$ algorithm. If $\theta_2 = 0$, $\mathcal{SK}_{\mathcal{M}_k}$ is normal. Otherwise, $\theta_2 \in_{\text{U}} \mathbb{G}_{p_2}$ making $\mathcal{SK}_{\mathcal{M}_k}$ Type-1 semi-functional. The randomisers for the semi-functional components are set as: $z_x = d_x \theta_2$ for all $q_x \in Q$, $\mu_{\text{start}} = r'_{\text{start}} \theta_2$, $\mu_{\text{end}} = r'_{\text{end}} \theta_2$, $\mu_t = r'_t \theta_2$ for all $t \in \mathcal{T}$; $\pi_{\text{start}} = v_{\text{start}}$, $\pi_{u,\sigma} = v_{u,\sigma}$, $\pi_{h,\sigma} = v_{h,\sigma}$ for each $\sigma \in \Sigma$ and $\tau_{\text{end}} = r'_{\text{end}} v_{\text{end}} \theta_2$. Although v's are provided to the adversary via the public parameters, their values modulo p_2 remain hidden from the adversary (by Chinese remainder theorem). The μ's are uniformly distributed by the choice of r''s. Hence the π's and τ_{end} are uniformly distributed in $\mathscr{A}$'s view.

Challenge: $\mathscr{B}$ receives messages m_0, m_1 and challenge string $w^* = w_1^* \cdots w_{\ell^*}^*$ from $\mathscr{A}$. It chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and constructs ciphertext $\mathcal{C}^*$ as follows.

$\gamma_0, \ldots, \gamma_{\ell^*} \xleftarrow{\text{U}} \mathbb{Z}_N$

$C_m = m_\beta \cdot e(P, X + P_2)^{\alpha \gamma_{\ell^*}},$

$C_{0,1} = \gamma_0 (X + P_2), \quad C_{\text{start},2} = \gamma_0 v_{\text{start}} (X + P_2),$

For $i = 1, \ldots, \ell^*$,
$C_{i,1} = \gamma_i (X + P_2), \quad C_{i,2} = (\gamma_i v_{h,w_i} + \gamma_{i-1} v_{u,w_i})(X + P_2),$

$C_{\text{end},1} = C_{\ell,1}, \quad C_{\text{end},2} = \gamma_\ell v_{\text{end}} (X + P_2),$

setting $s_i = \theta \gamma_i$ for $i \in [0, \ell^*]$. The output of $\mathsf{ReRandCT}(\mathcal{C}^*)$ is returned to $\mathscr{A}$. The π values (except π_{end}) are set to the corresponding v's modulo p_2. These are equal to the π-values of the k-th key thus satisfying the requirements for Type-1 semi-functionality. Also π_{end} is set to $\gamma_\ell v_{\text{end}}$ which is uniformly distributed (by the choice of γ_ℓ) and also independent of τ_{end}. This is because v_{end} is randomised by r'_{end} in τ_{end}. Note that after calling $\mathsf{ReRandCT}$ the randomisers for the $\mathbb{G}_{p_1}$ components will have the proper distribution.

Guess: $\mathscr{A}$ sends $\mathscr{B}$ its guess β'.

We now argue that the challenge ciphertext and k-th key combined, do not reveal any information about the π-values to the adversary. For this consider a transition $t = (q_x, q_y, \sigma)$ and suppose the i-th set of components in $\mathcal{C}^*$ are for the symbol σ (i.e., $w_i^* = \sigma$). Then $C_{i,\cdot}$ and $K_{t,\cdot}$ components will share the same π-values. Assume that the μ_t and γ_i, γ_{i-1} values are statistically revealed to the adversary. It essentially gets hold of 3 equations (corresponding to semi-functional components of $K_{t,1}, K_{t,3}, C_{w,2}$) in 4 unknowns $(\pi_{h,\sigma}, \pi_{u,\sigma}, z_x, z_y)$. Using these the adversary cannot gain any information about these quantities. They remain information theoretically hidden and thus appear uniformly distributed in the attackers' view.

$\mathscr{B}$ could attempt to create a semi-functional ciphertext for a string w' accepted by $\mathcal{M}_k$ and check whether $\mathcal{SK}_{\mathcal{M}_k}$ is semi-functional or not. But any such attempt will end up setting

$$\mu_{\text{end}}\pi_{\text{end}} - \tau_{\text{end}}\gamma_\ell = (r'_{\text{end}}\theta_2)(\gamma_\ell v_{\text{end}}) - (r'_{\text{end}} v_{\text{end}}\theta_2)\gamma_\ell = 0 \pmod{p_2}.$$

This implies that the ciphertext-key pair is nominally semi-functional. Decryption succeeds and provides no information to $\mathscr{B}$ about the distribution of T.

If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have

$$\begin{aligned}
\varepsilon_2 \geq \mathsf{Adv}_{\mathcal{G}}^{\text{DSG2}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T \in_{\text{U}} \mathbb{G}_{p_1 p_3}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \in_{\text{U}} \mathbb{G}]| \\
&= |\Pr[\mathscr{A} \text{ wins} \mid T \in_{\text{U}} \mathbb{G}_{p_1 P_3}] - \Pr[\mathscr{A} \text{ wins} \mid T \in_{\text{U}} \mathbb{G}]| \\
&= |\Pr[\mathscr{A} \text{ wins in } \text{Game}_{k-1,1}] - \Pr[\mathscr{A} \text{ wins in } \text{Game}_{k,0}]| \\
&= |\Pr[\mathcal{E}_{k-1,1}] - \Pr[\mathcal{E}_{k,0}]|
\end{aligned}$$

as required. □

Lemma 4.4. $|\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]| \leq \varepsilon_2$ *for* $1 \leq k \leq \nu$.

The proof is similar to that of Lemma 4.3 except for the simulation of the k-key. The end components of this key are additionally rerandomised in $\mathbb{G}_{p_2}$ to ensure that it remains semi-functional with its type depending on whether the instance is real or random. The proof is provided in Appendix C.

Lemma 4.5. $|\Pr[\mathcal{E}_{\nu,1}] - \Pr[\mathcal{E}_{final}]| \leq \varepsilon_3$.

The idea of the proof is as follows. Let $(\mathcal{G}_{\text{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2, T)$ be the instance of DSG3 using which the game needs to be simulated. α from the instance is the α of the system master secret. The scalar s from the instance will be mapped to the randomiser that is used to mask the message i.e., s_{ℓ^*}, where ℓ^* is the length of the challenge string. Since generators of subgroups corresponding to all three primes, (semi-functional) keys and ciphertexts can be generated. The main trick lies in generating the $K_{\text{end},1}$ components of the keys since they have α embedded in them and also in computing the ciphertext terms corresponding to the randomiser s_{ℓ^*}. Due to lack of space, the proof details are given in Appendix D.

5 Full Construction

The restrictions on $\mathcal{BFE}$ scheme confines the functionality support to a small subclass of regular languages. It is possible to expand the supported class of languages via an extension of $\mathcal{BFE}$. The extension provides the ability to deal with multiple occurrences of symbols both in the input string and transitions of the automata. The number of occurrences is however bounded at setup time. As a result, the sizes of public parameters, keys and ciphertexts increase by a factor proportional to these bounds.

We shall first define some notation. For a matrix $\mathbf{A} \in {\mathbb{Z}_N}^{m \times n}$, $\mathbf{A}[i, j]$ denotes the entry in i-th row and j-column of $\mathbf{A}$. Let $w = w_1 \ldots w_\ell$ be a string over the alphabet Σ and $\mathcal{T}$ be the (ordered) set of transitions of an automaton $\mathcal{M}$.

- $\mathsf{s}_{\mathsf{max}}$: bound on the number of occurrences of each symbol in a string
- $\mathsf{t}_{\mathsf{max}}$: the maximum number of transitions on any particular symbol
- $\mathsf{n}^{\mathsf{c}}[w,i]$: contains k if position i is the k-occurrence of the symbol w_i in w
- $\mathsf{n}^{\mathsf{k}}[\sigma,t]$: contains k if t is the k-transition on σ

The extended construction $\mathcal{FFE} = (\mathcal{FFE}.\mathsf{Setup}, \mathcal{FFE}.\mathsf{Encrypt}, \mathcal{FFE}.\mathsf{KeyGen}, \mathcal{FFE}.\mathsf{Decrypt})$ is described below.

$\mathcal{FFE}.\mathsf{Setup}(\Sigma, \kappa)$: Generate a composite order pairing $\mathcal{G} = (p_1, p_2, p_3, \mathbb{G}, \mathbb{G}_T, e, G)$ according to the security parameter κ. Choose elements $P, H_{\mathrm{start}}, H_{\mathrm{end}} \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1}$, $P_3 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}$, $\alpha \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and

$$\mathbf{H}_\sigma, \mathbf{U}_\sigma \xleftarrow{\mathrm{U}} (\mathbb{Z}_N)^{\mathsf{s}_{\mathsf{max}} \times \mathsf{t}_{\mathsf{max}}} \text{ for all } \sigma \in \Sigma.$$

The public parameters and master secret are given by

$$\mathcal{PP} : (\mathcal{G}_{\mathrm{pub}}, \Sigma, P, H_{\mathrm{start}}, H_{\mathrm{end}}, H_\lambda, (\mathbf{H}_\sigma, \mathbf{U}_\sigma)_{\sigma\in\Sigma}, e(P,P)^\alpha),$$
$$\mathcal{MSK}: (-\alpha P, P_3).$$

$\mathcal{FFE}.\mathsf{Encrypt}(\mathcal{PP}, w = w_1 \cdots w_\ell, m)$: Choose randomisers $s_0, s_1, \ldots, s_\ell \xleftarrow{\mathrm{U}} \mathbb{Z}_N$. Compute the ciphertext elements as follows.

$$C_m = m \cdot e(P,P)^{\alpha s_\ell},$$

$$C_{0,0} = C_{\mathrm{start},1} = s_0 P, \quad C_{\mathrm{start},2} = s_0 H_{\mathrm{start}},$$

For $i = 1, \ldots, \ell$,
$$C_{i,0} = s_i P, \quad (C_{i,j} = s_i \mathbf{H}_{w_i}[\mathsf{n}^{\mathsf{c}}[w,i], j] + s_{i-1} \mathbf{U}_{w_i}[\mathsf{n}^{\mathsf{c}}[w,i], j])_{j\in[1,\mathsf{t}_{\mathsf{max}}]},$$

$$C_{\mathrm{end},1} = C_{\ell,0} = s_\ell P, \quad C_{\mathrm{end},2} = s_\ell H_{\mathrm{end}}.$$

The ciphertext is given by $\mathcal{C} = (C_m, C_{\mathrm{start},1}, C_{\mathrm{start},2}, (C_{i,0}, C_{i,j})_{i\in[1,\ell], j\in[1,\mathsf{t}_{\mathsf{max}}]}, C_{\mathrm{end},1}, C_{\mathrm{end},2}, w)$.

$\mathcal{FFE}.\mathsf{KeyGen}(\mathcal{MSK}, \mathcal{M} = (Q, \Sigma, q_0, q_f, \delta))$: For each $x \in \mathbb{Z}_{|Q|}$, pick $D_x \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1}$. Choose elements r_{start}, for all $t \in \mathcal{T}$, r_t and r_{end} uniformly and independently at random from $\mathbb{Z}_N$. Let $R_{\mathrm{start},1}, R_{\mathrm{start},2}$, $(R_{t,1}, R_{t,2}, R_{t,3})_{t\in\mathcal{T}}$ and $R_{\mathrm{end},1}, R_{\mathrm{end},2}$ be randomly chosen elements of $\mathbb{G}_{p_3}$. Compute the elements of the key as follows.

$$K_{\mathrm{start},1} = D_0 + r_{\mathrm{start}} H_{\mathrm{start}} + R_{\mathrm{start},1}, \quad K_{\mathrm{start},2} = r_{\mathrm{start}} P + R_{\mathrm{start},2},$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$$K_{t,2} = r_t P + R_{t,2},$$
$$(K_{t,1,i} = -D_x + r_t \mathbf{U}_\sigma[i, \mathsf{n}^{\mathsf{k}}[\sigma,t]] + R_{t,1}, \quad K_{t,3,i} = D_y + r_t \mathbf{H}_\sigma[i, \mathsf{n}^{\mathsf{k}}[\sigma,t]] + R_{t,3})_{i\in[1,\mathsf{s}_{\mathsf{max}}]},$$

$$K_{\mathrm{end},1} = -\alpha P + D_f + r_{\mathrm{end}} H_{\mathrm{end}} + R_{\mathrm{end},1}, \quad K_{\mathrm{end},2} = r_{\mathrm{end}} P + R_{\mathrm{end},2}.$$

Here D_f corresponds to the final state q_f. The secret key for automaton $\mathcal{M}$ is given by $\mathcal{SK}_{\mathcal{M}} = (K_{\mathrm{start},1}, K_{\mathrm{start},2}, (K_{t,1}, K_{t,2}, K_{t,3})_{t\in\mathcal{T}}, K_{\mathrm{end},1}, K_{\mathrm{end},2})$.

$\mathcal{FFE}.\mathsf{Decrypt}(\mathcal{C}, \mathcal{SK}_{\mathcal{M}})$: Suppose that $\mathsf{Accept}(\mathcal{M}, w) = 1$ and $w = w_1 \cdots w_\ell$. Then there exists a sequence of transitions $t_1, t_2, \ldots, t_\ell$ with $t_i = (q_{x_{i-1}}, q_{x_i}, w_i)$ where $x_0 = 0$ and $x_\ell = f$. Decryption consists of several stages of computation. First compute

$$\begin{aligned} A_0 &= e(C_{\mathrm{start},1}, K_{\mathrm{start},1}) e(C_{\mathrm{start},1}, K_{\mathrm{start},2})^{-1} \\ &= e(P, D_0)^{s_0} \end{aligned}$$

Then compute intermediate values A_i (for $i = 1, \ldots, \ell$) as follows. Pick $C_{i,\mathsf{n}^{\mathsf{k}}[w_i,t_i]}$ and $K_{t_i,1,\mathsf{n}^{\mathsf{c}}[w_i,i]}, K_{t_i,3,\mathsf{n}^{\mathsf{c}}[w_i,i]}$. Such components exist and are unique.

$$\begin{aligned} A_i &= A_{i-1} \cdot e(C_{i-1,0}, K_{t_i,1,\mathsf{n}^{\mathsf{c}}[w_i,i]}) e(C_{i,\mathsf{n}^{\mathsf{k}}[w_i,t_i]}, K_{t_i,2})^{-1} e(C_{i,0}, K_{t_i,3,\mathsf{n}^{\mathsf{c}}[w_i,i]}) \\ &= e(P, D_{x_i})^{s_i} \end{aligned}$$

With any other pair of $C_{i,j}$ and $K_{t_i,1,k}, K_{t_i,3,k}$ it is not possible to cancel out $e(P, D_{x_{i-1}})^{s_{i-1}}$. The last intermediate $A_{\ell+1}$ is computed as

$$A_{\ell+1} = e(C_{\text{end},1}, K_{\text{end},1}) \cdot e(C_{\text{end},2}, K_{\text{end},2})^{-1} = e(P, P)^{\alpha s_\ell} e(D_f, P)^{s_\ell}.$$

Using A_ℓ and $A_{\ell+1}$ the message is unmasked as shown below.

$$m = C_m \cdot A_{\ell+1}^{-1} \cdot A_\ell.$$

Discussion. The construction essentially converts a DFA and string to a basic form by mapping each occurrence of a symbol σ to a different representation in the group. Consider a ciphertext for string w and automaton $\mathcal{M}$. In the full FE scheme, w and $\mathcal{M}$ are encoded so that there exists a unique sequence of decryption operations that result in the correct message if $\mathcal{M}$ accepts w. Given this, correctness of decryption follows. While arguing about security, the existence of $\mathsf{s}_{\mathsf{max}} \times \mathsf{t}_{\mathsf{max}}$ distinct representations for a symbol σ ensures that the semi-functional components for all occurrences of σ are independent of each other. Furthermore, the same rerandomisation technique can be employed to ensure proper distribution of keys and ciphertexts in the proof.

6 Conclusion

Using the dual system technique, we have obtained a DFA-based functional encryption scheme that has adaptive security under static assumptions in composite order pairings. The cost of achieving this is an increase in the sizes of the ciphertext and keys along with bounded functionality. It would be interesting to obtain adaptive security without restricting the number of occurrences of symbols in either the strings or transitions of automata. Another natural question is whether selective security can be achieved using static assumptions but without imposing the restrictions on the system.

Acknowledgement

We would like to thank Tapas Pandit and Prof. Palash Sarkar for helpful discussions and suggestions.

References

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334. IEEE Computer Society, 2007.

[BSW12] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: a new vision for public-key cryptography. *Commun. ACM*, 55(11):56–64, 2012.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[HMU00] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison Wesley, 2 edition, 2000.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.

[LOS+10] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010.

[LW12] Allison Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In Safavi-Naini and Canetti [SNC12], pages 180–198.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM Conference on Computer and Communications Security*, pages 195–203. ACM, 2007.

[OT09] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 214–231. Springer, 2009.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 191–208. Springer, 2010.

[OT11] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner-product encryption. Cryptology ePrint Archive, Report 2011/543, 2011. `http://eprint.iacr.org/`.

[OT12] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 349–366. Springer, 2012.

[SNC12] Reihaneh Safavi-Naini and Ran Canetti, editors. *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*. Springer, 2012.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.

[SW08] Elaine Shi and Brent Waters. Delegating capabilities in predicate encryption systems. In *Automata, Languages and Programming*, pages 560–578, 2008.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.

[Wat11] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011.

[Wat12] Brent Waters. Functional encryption for regular languages. In Safavi-Naini and Canetti [SNC12], pages 218–235.

A DFA-based Functional Encryption

The definition of DFA-based functional encryption described in [Wat12] is provided here.

A.1 Definition

A functional encryption (FE) scheme over DFA's consists of four probabilistic algorithms - Setup, KeyGen, Encrypt and Decrypt.

- Setup: takes as input a security parameter κ, generates the public parameters $\mathcal{PP}$ and the master secret $\mathcal{MSK}$ based on λ and the input alphabet Σ. Σ is part of $\mathcal{PP}$.
- KeyGen: receives the description of a DFA $\mathcal{M}$ and master secret $\mathcal{MSK}$ and outputs a secret key $\mathcal{SK}_{\mathcal{M}}$ corresponding to $\mathcal{M}$.
- Encrypt: inputs a message m, a string $w = w_1 w_2 \cdots w_\ell$ over Σ and returns a ciphertext $\mathcal{C}$ (which also contains w).
- Decrypt: inputs a ciphertext $\mathcal{C}$ and secret key $\mathcal{SK}_{\mathcal{M}}$. If $\mathsf{Accept}(\mathcal{M}, w) = 1$, the algorithm returns m; otherwise, returns $\perp$ indicating failure.

This is a key-policy functional encryption scheme. One can also define a ciphertext-policy scheme but we do not consider it since the techniques will be more or less similar.

A.2 Security

Security is modelled based on the notion of indistinguishability of ciphertexts under a chosen plaintext attack (CPA). It is defined via a game ind-cpa between an adversary $\mathscr{A}$ and a challenger consisting of several stages.

Setup: The challenger runs the Setup algorithm of the FE scheme and gives the public parameters to $\mathscr{A}$.

Phase 1: $\mathscr{A}$ makes a number of key extraction queries adaptively. For a query on automaton $\mathcal{M}$, the challenger runs the KeyGen algorithm of the FE scheme and returns its output $\mathcal{SK}_{\mathcal{M}}$ to $\mathscr{A}$.

Challenge: $\mathscr{A}$ provides two messages pairs m_0, m_1 and a challenge string $w^* = w_1^* w_2^* \cdots w_\ell^*$ subject to the condition that $\mathscr{A}$ does not request keys for any automaton that accepts w^* in **Phase 1** or **Phase 2**. The challenger then picks $\beta \xleftarrow{\text{U}} \{0,1\}$ and returns an encryption $\mathcal{C}^*$ of m_β under the string w^* to $\mathscr{A}$.

Phase 2: $\mathscr{A}$ issues more key extraction queries as in **Phase 1** with the restriction that none of the automata that are queried accept w^*.

Guess: $\mathscr{A}$ outputs a bit β'.

In the selective model, there is a stage **Initialise** before **Setup** in which the adversary commits to the input alphabet Σ and the challenge string w^*. Call this game ind-s-cpa.

If $\beta = \beta'$, then $\mathscr{A}$ wins the game. The advantage of $\mathscr{A}$ in breaking the security of the FE scheme in the ind-cpa game is given by

$$\mathsf{Adv}_{\mathrm{FE}}^{\mathsf{ind\text{-}cpa}}(\mathscr{A}) = \left|\Pr[\beta = \beta'] - \frac{1}{2}\right|.$$

The FE scheme is said to be (ε, t, ν)-IND-STR-CPA secure[1] (secure under chosen plaintext attack) if for every adversary $\mathscr{A}$ making at most ν queries and whose running time is t, it holds that $\mathsf{Adv}_{\mathrm{FE}}^{\mathsf{IND\text{-}STR\text{-}CPA}}(\mathscr{A}) \leq \varepsilon$.

B Algorithms for Rerandomisation

We describe the rerandomisation algorithms here. Except for the $\mathbb{G}_{p_3}$ components of the keys the algorithms are identical to those in [Wat12].

$\mathsf{ReRandCT}(\mathcal{C})$: This algorithm picks $s'_0, s'_1, \ldots, s'_\ell \xleftarrow{\mathrm{U}} \mathbb{Z}_N$ and modifies the ciphertext elements as shown below.

$C_m \leftarrow C_m \cdot e(P,P)^{\alpha s'_\ell}$,

$C_{\mathrm{start},1} \leftarrow C_{\mathrm{start},1} + s'_0 P, \quad C_{\mathrm{start},2} \leftarrow C_{\mathrm{start},2} + s'_0 H_{\mathrm{start}}$,

For $i = 1, \ldots, \ell$,
$C_{i,1} \leftarrow C_{i,2} + s'_i P, \quad C_{i,2} \leftarrow C_{i,2} + s'_i H_{w_i} + s'_{i-1} P_1$,

$C_{\mathrm{end},1} \leftarrow C_{\mathrm{end},1} + s'_\ell P, \quad C_{\mathrm{end},2} \leftarrow C_{\mathrm{end},2} + s'_\ell H_{\mathrm{end}}$.

The new randomisers for the ciphertext will be $s_i + s'_i$ $(i = 0, \ldots, \ell)$. The string w remains the same.

$\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}})$: Choose uniform and independent random scalars r'_{start}, for all $t \in \mathcal{T}$, r'_t and r'_{end} from $\mathbb{Z}_N$. Also choose $D'_x \xleftarrow{\mathrm{U}} \mathbb{G}_{p_1}$ for every $q_x \in Q$ and $R'_{\mathrm{start},1}, R'_{\mathrm{start},2}, \{R'_{t,1}, R'_{t,2}, R'_{t,3}\}_{t \in \mathcal{T}}, R'_{\mathrm{end},1}, R'_{\mathrm{end},2} \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}$. Reconstruct components of the key as follows.

$K_{\mathrm{start},1} \leftarrow K_{\mathrm{start},1} + D'_0 + r'_{\mathrm{start}} H_{\mathrm{start}} + R'_{\mathrm{start},1}, \quad K_{\mathrm{start},2} \leftarrow K_{\mathrm{start},2} + r'_{\mathrm{start}} P + R'_{\mathrm{start},2}$

For $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$K_{t,1} \leftarrow K_{t,1} - D'_x + r'_t P_1 + R'_{t,1}, \quad K_{t,2} \leftarrow K_{t,2} + r'_t P + R'_{t,2}, \quad K_{t,3} \leftarrow K_{t,3} + D'_y + r'_t H_\sigma + R'_{t,3}$,

$K_{\mathrm{end},1} \leftarrow K_{\mathrm{end},1} + D'_f + r'_{\mathrm{end}} H_{\mathrm{end}} + R'_{\mathrm{end},1}$,
$K_{\mathrm{end},2} \leftarrow K_{\mathrm{end},2} + r'_{\mathrm{end}} P + R'_{\mathrm{end},2}$.

[1] The abbreviation "STR" stands for string. "sSTR" denotes that the challenge string is chosen selectively.

C Proof of Lemma 4.4

Let $(\mathcal{G}_{\mathrm{pub}}, P, P_3, X + P_2, X_2 + X_3, T)$ be the instance of DSG2 that $\mathscr{B}$ has to solve i.e., decide whether $\theta_2 = 0$ or $\theta_2 \in_{\mathrm{U}} \mathbb{Z}_{p_3}$ where $T = \theta P + \theta_2 P_2 + \theta_3 P_3$.

Setup: Scalars $\alpha, v_{\mathrm{start}}, v_{\mathrm{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma\in\Sigma}$ are chosen from $\mathbb{Z}_N$ independently according to the uniform distribution. Parameters are set as follows: $H_{\mathrm{start}} = v_{\mathrm{start}}P$, $H_{\mathrm{end}} = v_{\mathrm{end}}P$, $H_\sigma = v_{h,\sigma}P$ and $U_\sigma = v_{u,\sigma}P$. $\mathcal{PP}$ is given to $\mathscr{A}$ and $\mathscr{B}$ keeps $\mathcal{MSK}$.

Key extraction queries: For key extraction queries on $\mathcal{M}_i$ for $i \neq k$, $\mathscr{B}$ answers the query as in proof of Lemma 4.3. The secret key for $\mathcal{M}_k$ is generated as follows.

For each $x \in \mathbb{Z}_{|Q|}$, $d_x \xleftarrow{\mathrm{U}} \mathbb{Z}_N$
$r'_{\mathrm{start}}, r'_{\mathrm{end}}, \{r'_t\}_{t\in\mathcal{T}}, \mu_1, \mu_2 \xleftarrow{\mathrm{U}} \mathbb{Z}_N$

$$K_{\mathrm{start},1} = (d_0 + r'_{\mathrm{start}} v_{\mathrm{start}})T, \quad K_{\mathrm{start},2} = r'_{\mathrm{start}} T,$$

For all $t \in \mathcal{T}$ with $t = (q_x, q_y, \sigma)$ and $\sigma \in \Sigma$,
$$K_{t,1} = (-d_x + r'_t v_{u,\sigma})T, \quad K_{t,2} = r'_t T, \quad K_{t,3} = (d_y + r'_t v_\sigma)T,$$

$$K_{\mathrm{end},1} = -\alpha P + (d_f + r'_{\mathrm{end}} v_{\mathrm{end}})T + \mu_1(X_2 + X_3), \quad K_{\mathrm{end},2} = r'_{\mathrm{end}} T + \mu_2(X_2 + X_3).$$

Let $\mathcal{SK}_{\mathcal{M}} = (K_{\mathrm{start},1}, K_{\mathrm{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t\in\mathcal{T}}, K_{\mathrm{end},1}, K_{\mathrm{end},2})$. $\mathscr{B}$ returns $\mathsf{ReRandK}(\mathcal{SK}_{\mathcal{M}_k})$ to $\mathscr{A}$. If $\theta_2 \in_{\mathrm{U}} \mathbb{G}_{p_2}$, then $\mathcal{SK}_{\mathcal{M}_k}$ is Type-1 semi-functional; otherwise it is a Type-2 semi-functional key. Both τ_{end} and μ_{end} are set to random quantities in either cases to prevent $\mathscr{B}$ from generating a nominally semi-functional ciphertext to test $\mathcal{SK}_{\mathcal{M}_k}$'s type of semi-functionality. The randomisers for the Type-1 semi-functional components are set as: $\mu_{\mathrm{start}} = r'_{\mathrm{start}}\theta_2$, $\mu_t = r'_t\theta_2$ for all $t \in \mathcal{T}$; $\pi_{\mathrm{start}} = v_{\mathrm{start}}$, $\pi_{u,\sigma} = v_{u,\sigma}$ and $\pi_{h,\sigma} = v_\sigma$ for each $\sigma \in \Sigma$. Furthermore, since the key is rerandomised, its $\mathbb{G}_{p_1}$ and $\mathbb{G}_{p_3}$ components are properly distributed.

The **Challenge** and **Guess** phases are identical to Lemma 4.3. If the adversary wins ($\beta \neq \beta'$), then $\mathscr{B}$ returns 1; otherwise it returns 0. Therefore, we have $\varepsilon_2 \geq |\Pr[\mathcal{E}_{k,0}] - \Pr[\mathcal{E}_{k,1}]|$.

D Proof of Lemma 4.5

Given an instance $(\mathcal{G}_{\mathrm{pub}}, P, P_2, P_3, \alpha P + X_2, sP + Y_2, T)$ of DSG3, $\mathscr{B}$ has to decide whether $T = e(P,P)^{\alpha s}$ or $T \in_{\mathrm{U}} \mathbb{G}_T$. The game is simulated as follows.

Setup: Randomisers $v_{\mathrm{start}}, v_{\mathrm{end}}, \{v_{u,\sigma}, v_{h,\sigma}\}_{\sigma\in\Sigma}$ are sampled uniformly and independently from $\mathbb{Z}_N$. Then set $H_{\mathrm{start}} = v_{\mathrm{start}}P$, $H_{\mathrm{end}} = v_{\mathrm{end}}P$, for all $\sigma \in \Sigma$, $H_\sigma = v_{h,\sigma}P$, $U_\sigma = v_{u,\sigma}P$ and $e(P,P)^\alpha = e(\alpha P + X_2, P)$. The public parameters $\mathcal{PP}$ are provided to $\mathscr{A}$. Note that the simulator does not know the master secret key.

Key extraction queries: Since αP is masked with an element of $\mathbb{G}_{p_2}$, $\mathscr{B}$ can generate only Type-2 semi-functional keys. For a query on an automaton $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$, a key is constructed as follows. Sample $D_x \in_{\mathrm{U}} \mathbb{G}_{p_1}$ for all $q_x \in Q$. Construct the components $K_{\mathrm{start},1}, K_{\mathrm{start},2}, \{K_{t,1}, K_{t,2}, K_{t,3}\}_{t\in\mathcal{T}}$ just as in the $\mathcal{BFE}.\mathsf{KeyGen}$ algorithm. The master secret α is embedded only the term $K_{\mathrm{end},1}$ and the main trick lies in generating this component. The encoding of α in $\mathbb{G}_{p_1}$ is masked by $\mathbb{G}_{p_2}$-component and hence $\mathscr{B}$ cannot prevent $K_{\mathrm{end},1}$ from having semi-functional components. $\mathscr{B}$ chooses $\mu_{\mathrm{end}}, r_{\mathrm{end}} \xleftarrow{\mathrm{U}} \mathbb{Z}_N$, $R_{\mathrm{end},1}, R_{\mathrm{end},2} \xleftarrow{\mathrm{U}} \mathbb{G}_{p_3}$, $Z_2 \xleftarrow{\mathrm{U}} \mathbb{G}_{p_2}$ and computes

$$K_{\mathrm{end},1} = -(\alpha P + X_2) + D_f + r_{\mathrm{end}} H_{\mathrm{end}} + R_{\mathrm{end},1} + Z_2, \quad K_{\mathrm{end},2} = r_{\mathrm{end}} P + R_{\mathrm{end},2} + \mu_{\mathrm{end}} P_2$$

implicitly setting $\tau_{\text{end}} P_2 = X_2 + Z_2$. Scalars μ_{end} and Z_2 are freshly chosen for each key. Therefore, the values of τ_{end} for the keys remain properly distributed.

Challenge: $\mathscr{B}$ receives two messages m_0, m_1 along with a string $w^* = w_1^* \cdots w_{\ell^*}^*$ from $\mathscr{A}$; chooses $\beta \xleftarrow{\text{U}} \{0,1\}$ and constructs a ciphertext for m_β and w^* as described below.

$s_0, \ldots, s_{\ell^*-1}, \gamma_0, \ldots, \gamma_{\ell^*-1} \xleftarrow{\text{U}} \mathbb{Z}_N$;
$\pi_{\text{start}} \xleftarrow{\text{U}} \mathbb{Z}_N$, $\pi_{u,\sigma} \xleftarrow{\text{U}} \mathbb{Z}_N$ for all $\sigma \in \Sigma$;
for all $\sigma \in \Sigma \setminus \{w_{\ell^*}^*\}$, $\pi_{h,\sigma} \xleftarrow{\text{U}} \mathbb{Z}_N$, set $\pi_{h,w_{\ell^*}^*} = v_{h,w_{\ell^*}^*}$,

$C_m = m_\beta \cdot T$,
$C_{0,1} = s_0 P + \gamma_0 P_2, \quad C_{\text{start},2} = s_0 H_{\text{start}} + \gamma_0 \pi_{\text{start}} P_2,$

For $i = 1, \ldots, \ell^* - 1$,
$\quad C_{i,1} = s_i P + \gamma_i P_2, \quad C_{i,2} = s_i H_{w_i} + s_{i-1} U_\sigma + (\gamma_i \pi_{w_i} + \gamma_{i-1} \pi_{u,w_i}) P_2,$
$C_{\ell^*,1} = sP + Y_2, \quad C_{\ell^*,2} = v_{w_{\ell^*}^*}(sP + Y_2) + s_{i-1} U_\sigma + \gamma_{i-1} \pi_{u,w_i} P_2,$

$C_{\text{end},1} = C_{\ell^*,1}, \quad C_{\text{end},2} = v_{\text{end}}(sP + Y_2).$

implicitly setting $s_{\ell^*} = s$, $\gamma_{\ell^*} P_2 = Y_2$ and $\pi_{\text{end}} P_2 = v_{\text{end}} Y_2$. The values of $v_{h,w_{\ell^*}^*}$ and v_{end} modulo p_2 are hidden from the adversary and hence $\pi_{h,w_{\ell^*}^*}, \pi_{\text{end}}$ are uniformly and independently distributed in $\mathscr{A}$'s view. $\mathscr{B}$ returns $\mathcal{C}^*$ consisting of the above components to $\mathscr{A}$.

Guess: $\mathscr{A}$ makes its guess β' of β.

If $T = e(P,P)^{\alpha s}$ then we have $C_m = m_\beta \cdot T = m_\beta \cdot e(P,P)^{\alpha s_{\ell^*}}$ making $\mathcal{C}^*$ a semi-functional encryption of m_β and thus playing $\text{Game}_{\nu,1}$. Otherwise $T \in_{\text{U}} \mathbb{G}_T$ and $(C_m = m_\beta \cdot T) \in_{\text{U}} \mathbb{G}_T$. In this case, $\mathcal{C}^*$ will be a semi-functional encryption of a random message and $\mathscr{B}$ simulates Game_{final}. If the adversary wins the game then $\mathscr{B}$ returns 1; otherwise it returns 0. We therefore have,

$$\begin{aligned}
\varepsilon_3 \geq \mathsf{Adv}_{\mathcal{G}}^{\text{DSG3}}(\mathscr{B}) &= |\Pr[\mathscr{B} \text{ returns } 1 \mid T = e(P,P)^{\alpha s}] - \Pr[\mathscr{B} \text{ returns } 1 \mid T \in_{\text{U}} \mathbb{G}_T]| \\
&= |\Pr[\mathscr{A} \text{ wins} \mid T = e(P,P)^{\alpha s}] - \Pr[\mathscr{A} \text{ wins} \mid T \in_{\text{U}} \mathbb{G}_T]| \\
&= |\Pr[\mathscr{A} \text{ wins in } \text{Game}_{\nu,1}] - \Pr[\mathscr{A} \text{ wins in } \text{Game}_{final}]| \\
&= |\Pr[\mathcal{E}_{\nu,1}] - \Pr[\mathcal{E}_{final}]|
\end{aligned}$$

as required.