cryptography

MDPI

Article

Forward-Secure Linkable Ring Signatures from Bilinear Maps

Xavier Boyen [1] and Thomas Haines [2,*]

[1] Queensland University of Technology, Brisbane 4000, Australia; xb@boyen.org
[2] Polyas GmbH, 10179 Berlin, Germany
* Correspondence: t.haines@polyas.de

Received: 14 September 2018; Accepted: 1 November 2018; Published: 8 November 2018

Abstract: We present the first linkable ring signature scheme with both unconditional anonymity and forward-secure key update: a powerful tool which has direct applications in elegantly addressing a number of simultaneous constraints in remote electronic voting. We propose a comprehensive security model, and construct a scheme based on the hardness of finding discrete logarithms, and (for forward security) inverting bilinear or multilinear maps of moderate degree to match the time granularity of forward security. We prove efficient security reductions—which, of independent interest, apply to, and are much tighter than, linkable ring signatures without forward security, thereby vastly improving the provable security of these legacy schemes. If efficient multilinear maps should ever admit a secure realisation, our contribution would elegantly address a number of problems heretofore unsolved in the important application of (multi-election) practical Internet voting. Even if multilinear maps are never obtained, our minimal two-epoch construction instantiated from bilinear maps can be combinatorially boosted to synthesise a polynomial time granularity, which would be sufficient for Internet voting and more.

Keywords: linkable ring signature; bilinear map; multilinear map; electronic voting; forward security; unconditional anonymity

1. Introduction

Ring signatures, and especially linkable ring signatures, garner much interest in the applied cryptographic community for their promise to simplify certain aspects of the notoriously hard problem of remote electronic voting, which has conflicting and often frustrating security requirements. In particular, linkability [1] or the closely related notion of traceability [2], make it easy to detect when the same signer has signed twice on the same matter, thereby preventing double spending in an electronic cash system, double voting in the same election. This, when combined with the anonymity properties of ring signatures, provides a secure mechanism to validate votes without breaking privacy. This paper is an extended version of our work in [3].

However, thus far, these signatures have not assisted in simultaneously resolving two critical issues in electronic voting. These two issues are: (1) how to register voters; and (2) how to ensure the voters' long term privacy. Indeed, at present, most proposed electronic voting schemes use cryptography which is likely to allow adversaries to break the privacy of the voters at some point in the future.

To address these issues, an offline key update mechanism would allow the potentially costly registration of a voter's public key to happen once, whereafter the corresponding private key can be refreshed or updated multiple times, efficiently and *non-interactively*, for use in subsequent elections. In this context, forward security refers to the notion that the leakage or compromise of an updated private key will not compromise one's privacy in a past election—or let an attacker forge signatures

ostensibly in the past, which could be linked to real votes. For practical electoral systems in particular, it is important that the public-key update mechanism be efficient and non-interactive. The ideal public-key update is the identity function, or "no-op." The private-key update serves to provide forward security to protect old elections against future data exposure and compromises.

The related but different notion of unconditional anonymity refers to the inability, even by a computationally unbounded attacker, to identify a signer without knowledge of their private key. This notion is important to protect the voter against future increases in computational power (or cryptanalytic attacks, or quantum computers), once they have destroyed their private key after it is no longer needed. Together with linkability, these features make substantially easier the task of designing a secure and useable remote election protocol. Our forward-secure linkable ring signature scheme, when dropped into a number of existing election protocols, directly results in a straightforward and secure electronic voting solution without the cumbersome and procedurally risky steps that would normally be necessary to manage a dedicated key for each election. The additional improvements we detail in this extended version mean that, in many cases, the new scheme would also be more efficient.

Unfortunately—as often with the contradictory requirements of voting—it is easy to convince oneself that anonymity can only hold unconditionally if no *authentic* private key for the relevant signing ring is ever leaked, not even after having been updated. Indeed, if an adversary knows a voter's authentic private key, he can always trivially deanonymise their current and future votes using the linkability feature. The same is true for past votes if a past private key can be recovered, by brute force or by breaking a hardness assumption, from a current key. It is an interesting area of future research to see if a key update mechanism, perhaps context dependent, could be developed which would prevent the discovery of past private keys from future keys even for a computationally unbounded adversary. However, such a mechanism would seem to have too many interesting applications to not have been discovered already, unless it were highly non-trivial. In light of this, we deliberately choose to focus on the problem of achieving unconditional anonymity against outsiders, but only computational forward security against insiders in the sense of unforgeability and anonymity after key update.

1.1. Our Results

We present the first linkable ring signature with unconditional anonymity and forward-secure key update, which is a tool that enables significantly more simple and secure remote electronic voting, even within the framework of existing electronic voting protocols, and opens the door to a number of simplified general anonymous authentication protocols for online systems.

To achieve our result, we construct a linkable ring signature from unconditionally hiding commitments, and make sparing use of a bilinear pairing or multilinear map [4,5] to lift it to multiple time periods or "epochs". Without forward security or key update, our results are inspired by the linkable ring scheme from [1]—which we incidentally vastly improve via much tighter security reductions. (The original linkable ring signatures of Liu et al. [1,6] had proofs with losses exponential in the number of users, due to nested use of the forking lemma [7] on Pedersen commitments [8] in the random-oracle model. Our updated proofs and reductions are independent of the number of users, thanks to a single consolidated use of the forking lemma, and the same techniques directly apply to their construction.)

To get forward security, we build from an l-multilinear map an $\mathcal{O}(k^l)$-time one-way private-key update mechanism which requires no public-key update—for some choice of k. We prove the scheme information-theoretically anonymous, and its other security properties from Discrete Logarithm and two multilinear-map hardness assumptions—one of which amounts to a natural generalisation of the neo-classic Multilinear Decoding Problem [4] and the other is a natural generalisation of Decisional Diffie–Hellman. Notably, a mere 2-linear map (also known as *bilinear pairing*) already gives us forward security for $\mathcal{O}(k^2)$ time periods.

This extended paper expands on our original [3] in numerous ways. The main new technical contribution is a new combinatorial trick which allows us to gain a quadratic number of time

periods from bilinear maps. This is accompanied by updated and improved definitions and proofs. The updated definitions increase the adversary's powers, to better model insider attacks in particular. We have also taken advantage of the increased space available to provide more detailed discussion, including complexity analysis.

1.2. Related Work

Group signatures were introduced by Chaum et al. [9]. They allow the members of a group to generate signatures which can only be verified as emanating from one authorised signer within that group, with the additional property that the signature can be "opened" to reveal the true signer. The ability to open a signature is an important requirement in certain managed applications, but presents an unacceptable privacy loophole in the context of electronic voting. (In the UK, there is a requirement that a judge be able to order a voter's ballot revealed. Group signatures would be perfect for such subtle voter intimidation, although Continentals would of course disapprove.)

Ring signatures are a variation of group signatures which allow neither pre-authorisation of keys nor deanonymisation of signatures, and hence, do not have those privacy issues. Ring signatures were first presented by Rivest et al. [10] as a way to leak secrets anonymously. Rivest et al.'s initial suggested application was the leaking of information by a cabinet minister. Their scheme also had the nice property that the keys involved could belong to different public key schemes, which made setup very flexible. Since then, many variants have been proposed to suit a large number of applications. For elections, double voting is a major issue which vanilla ring signatures are not readily able to rectify. Linkable ring signatures [6] and traceable ring signatures [2] have been proposed as a way to address this issue. Nevertheless, neither of [2,6] or their variants provides forward security; hence, in a voting application they would require impractically frequent re-registration of new keys to ensure acceptable levels of privacy.

Subsequent notable results in that area include Liu et al. [1], who presented a linkable ring signature with unconditional anonymity, but still without forward security. Our scheme addresses this shortcoming, by providing an offline (non-interactive) private-key-update mechanism with forward security (as well as much improved security reduction tightness over the previous schemes). There are also emerging variants based on cryptographic accumulators as well as post-quantum variants.

1.2.1. Multilinear Maps

Following the blockbuster impact of bilinear maps on cryptography, the question of using multilinear maps for cryptographic applications was first studied at a theoretical level by Boneh and Silverberg [11]. Nearly a decade later, Garg et al. [4] proposed the first practical candidate construction, based on lattice problems. There have since been several additional candidates from lattice- and number-based assumptions, as well as attacks and repair attempts [5,12–15], with the side of the "offence" presently having the upper hand.

Generally speaking, multilinear maps are useful in allowing more complicated structures in cryptographic constructions. For instance, Diffie–Hellman key exchange in a group supports two parities, with a bilinear map three parties, but on an l-linear map would support l parties. The added structure also allows new techniques to be used which have no equivalence on bilinear maps or standard groups.

Our basic scheme relies on a multilinear generalisation of the Discrete Logarithm problem, which is a weaker assumption than the myriad of Diffie–Hellman variants and extensions typically found in cryptographic constructions based on bilinear or multilinear maps. Our combinatorially boosted version relies on natural generalisations of Computational and Decisional Diffie–Hellman. However, it should be noted that there are no currently unbroken candidates for multilinear maps, and hence the construction in this work is currently only realisable with bilinear pairings.

Our vastly improved security reductions for this class of unconditionally anonymous linkable ring signature scheme with or without forward security still apply, although, providing substantial

improvements to the concrete security of [1,6]. We discuss in Section 3.2 the major issues at hand regarding the known multilinear-map candidate constructions.

1.2.2. Voting Systems

In the world of election systems research, the recent Helios [16] protocol is, perhaps, the best known secure Internet voting scheme. It has seen a significant variety of expansions and applications [17,18], but one of its shortcomings is that the voters have to place (too) much trust on the election authority. Our linkable ring signature construction would fit nicely within the Helios protocol to enable powerful anonymous authentication and achieve privacy against the election authority, a property which is not achieved by most implementations of Helios. (In its standardised version [19], Helios relies on a mixnet technique to distribute the election authority's ability to deanonymise. Even for Helios implementations that use this technique, the ability to enforce anonymity in the authentication mechanism itself would provide stronger privacy guarantees.) More generally, and beyond election systems, our new signature scheme can be used as a general rate-limited (rate limitation in the context of authentication refers to an intentional bound on the number of uses, typically one, that can be made of a credential on a given target) anonymous authentication system with forward secrecy and information-theoretic privacy.

2. Definitions

A *forward secure linkable ring signature* (FS-LRS) scheme is a tuple of three probabilistic polynomial time algorithms (Setup, KeyGen, and Sign) and four deterministic polynomial time algorithms (Verify, Link, PubKeyUpd, and PriKeyUpd). (Our definitions are fairly direct forward-secure variants of Liu et al. [1].)

- **param** $\leftarrow$ **Setup**$(\lambda, \mathcal{T})$ on input security parameter λ and number of time periods $\mathcal{T}$, returns a public setup **param**.
- $(sk_i, pk_i) \leftarrow$**KeyGen**(**param**) given **param** returns a key pair (sk_i, pk_i).
- $\sigma \leftarrow$ **Sign**(**param**, $event, n, \vec{pk}_t, sk, M, t$) given an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a private key sk whose corresponding public key is in $\vec{pk}_t$, a message M and a time t, produces a signature σ.
- **accept**|**reject** $\leftarrow$ **Verify**(**param**, $event, n, \vec{pk}_t, M, \sigma, t$) given an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a message-signature pair (M, σ), and time t, returns **accept** or **reject**. We define a signature σ as valid for $(event, n, \vec{pk}_t, M, t)$ if **Verify** outputs **accept**.
- **linked**|**unlinked** $\leftarrow$ **Link**(**param**, $event, t, n_1, n_2, \vec{pk}_{t_1}, \vec{pk}_{t_2}, M_1, M_2, \sigma_1, \sigma_2$) given an event-id $event$, time t, two group sizes n_1, n_2, two sets $\vec{pk}_{t_1}, \vec{pk}_{t_2}$ of n_1, n_2 public keys respectively, and two valid signature and message pairs $(M_1, \sigma_1, M_2, \sigma_2)$, outputs **linked** or **unlinked**.
- $Z_{t+1} \leftarrow$ **PubKeyUpd**(**param**, Z_t) given a public key, Z at time t, produces a public key for time $t+1$.
- $sk_{t+1} \leftarrow$ **PriKeyUpd**(**param**, sk_t) given a private key sk at time t, produces the corresponding private key for time $t+1$.

We stress that in our current definitions **PubKeyUpd** and **PriKeyUpd** are deterministic functions. While it is interesting to consider systems where these functions might be randomised, this would need to be carefully considered. We are particularly concerned that randomised functions might severely limit the applications in which the signatures could be deployed.

2.1. Correctness Notions

To be functional, an **FS-LRS** scheme must satisfy the following:

- *Verification correctness:* Signatures signed correctly will verify.
- *Updating correctness:* For any time period of the system, the secret key derived from the private-key update function will create a valid signature on a ring, verifiable using the public key derived using the public-key update.

- *Linking correctness:* Two honestly created signatures on the same event and time period will link if and only if they have the same signer. (This is implied by the two security notions of linkability and non-slanderability; see below.)

2.2. Security Model

Security of FS-LRS has five aspects: unconditional anonymity, linkability, non-slanderability, forward-secure unforgeability, and forward-secure anonymity. (Ideally, we would be able to achieve forward-secure unconditional anonymity, and hence combine the first and last security properties. However, intuitively this property appears to be too strong to be achievable.)

2.2.1. Reflections

In our initial paper, we presented our definitions following fairly directly from Liu et al. in [1]. In this paper, we have made a few changes; primarily, we have strengthened the adversary's powers in the anonymity definitions. We have also modified the signing oracle to allow the adversary access to genuine signatures where the adversary does not know who the signer is. This is to better reflect the real adversary's powers in electronic voting. We note that our definitions and reductions are strictly stronger and tighter than Liu et al.

If a forward-secure linkable ring signature could be developed with much stronger privacy than any present solution, this would dramatically simply the security definitions. However, at present it is difficult to write definitions which closely match the expected level of security without being overly verbose. For instance, Liu et al.'s definition of privacy does not provide the adversary with access to genuine signatures, other than the challenge. In any deployed scheme, the adversary would having access to such signatures and hence the model clearly fails to capture reality. However, Liu et al.'s scheme appears to have a minimal loss of privacy under a differential privacy attack. This case highlights both the inadequacies of the model and the strength of the scheme. In summary, we believe our definitions capture well the intuitive definition of security; nevertheless, it is an interesting area of future research to present a scheme with stronger privacy that allows the simplification of the security model.

2.2.2. Oracles

The following oracles model the ability of the adversary to break the scheme:

- $pk_{i,t} \leftarrow \mathcal{JO}(t)$. The *Joining Oracle*, upon request, adds a new user to the system, and returns the public key pk of the new user at the time t.
- $sk_{i,t} \leftarrow \mathcal{CO}(pk_i, t)$. The *Corruption Oracle*, on input a previously joined public key pk_i, returns the matching secret key sk_i at the time t.
- $\sigma' \leftarrow \mathcal{SO}(event, n, \vec{pk}_t, \vec{pk}_\Pi, M, t)$. The *Signing Oracle*, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a set of public keys of possible signers $\vec{pk}_\Pi \subset \vec{pk}_t$, a message M, and a time t, returns a valid signature σ'.

We omit the time and user subscripts t, i when clear from context. In particular, our public key does not undergo updating, so pk_t will be independent of t.

- $h \leftarrow \mathcal{H}(x)$. The *Random Oracle*, on input x, returns h independently and uniformly at random. If an x is repeated, the same h will be returned again.

2.2.3. Unconditional Anonymity

It should not be possible for an adversary $\mathcal{A}$ to tell the public key of the signer with a probability larger than $1/n'$, where n' is the number of uncorrupted keys in the ring, even if the adversary has unlimited computing resources. Specifically, **FS-LRS** unconditional anonymity is defined in a game between a challenger $\mathcal{C}$ and an unbounded adversary $\mathcal{A}$ with access to $\mathcal{JO}, \mathcal{SO}, \mathcal{CO}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query $\mathcal{JO}$,$\mathcal{SO}$, and $\mathcal{CO}$ according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a time t, a group size n, a set of $\vec{pk}_t$ of n public keys such that all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$ and no key in $\vec{pk}_t$ has been input to $\mathcal{SO}$ in a set which was not a superset of $\vec{pk}_t$, a message M, and a time t. Parsing the set $\vec{pk}_t$ as $\{pk_1, ..., pk_n\}$, $\mathcal{C}$ picks $\pi \in \{1, ..., n\}$ uniformly from the uncorrupted subset and computes $\sigma_\pi = \textbf{Sign}(e, n, \vec{pk}_t, sk_\pi, M, t)$, where sk_π is a valid private key corresponding to pk_π at time t. The signature σ_π is given to $\mathcal{A}$.
4. $\mathcal{A}$ outputs a guess $\pi' \in \{1, ..., n\}$.

We denote the adversary's advantage by $\mathbf{Adv}^{Anon}_{\mathcal{A}}(\lambda) = |Pr[\pi = \pi'] - \frac{1}{n'}|$.

Definition 1. ***Unconditional Anonymity****. An* ***FS-LRS*** *scheme is unconditionally anonymous if, for all unbounded adversaries* $\mathcal{A}$, $\boldsymbol{Adv}^{Anon}_{\mathcal{A}}(\lambda)$ *is zero.*

2.2.4. Linkability

It should be infeasible for the same signer to generate two signatures for the same event and time, such that they are determined to be **unlinked**. Linkability for an **FS-LRS** scheme is defined in a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ with access to oracles $\mathcal{JO}$, $\mathcal{CO}$, $\mathcal{SO}$ and $\mathcal{H}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id *event*, a time t, two sets $\vec{pk}_{t_1}$, $\vec{pk}_{t_2}$ of public keys of sizes n_1, n_2, two messages M_1, M_2, and two signatures σ_1, σ_2.

$\mathcal{A}$ wins the game if:

- All of the public keys in $\vec{pk}_{t_i}$ are query outputs of $\mathcal{JO}$;
- **Verify**$(event, n_i, \vec{pk}_{t_i}, M_i, \sigma_i, \text{t}) =$ **accept** for σ_1, σ_2 not outputs of $\mathcal{SO}$;
- At most one query has been made to $\mathcal{CO}$; and
- **Link**(σ_1, σ_2) = **unlinked**.

We denote the adversary's advantage as $\mathbf{Adv}^{Link}_{\mathcal{A}}(\lambda) = Pr[\mathcal{A}$ wins the game$]$.

Definition 2. ***Linkability****. An* ***FS-LRS*** *scheme is linkable if for all Probabilistic Polynomial-Time (PPT) adversaries* $\mathcal{A}$, $\boldsymbol{Adv}^{Link}_{\mathcal{A}}(\lambda)$ *is negligible.*

2.2.5. Non-Slanderability

Non-slanderability ensures that no signer can generate a signature which is determined to be **linked** with another signature not generated by the signer. **FS-LRS** non-slanderabilty is defined in a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ with access to the oracles $\mathcal{JO}$, $\mathcal{CO}$, $\mathcal{SO}$ and $\mathcal{H}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id *event*, a time t, a group size n, a message M, a set of n public keys $\vec{pk}_t$, and the public key of an insider $pk_\pi \in \vec{pk}_t$. $\mathcal{C}$ uses the private key sk_π correspoing to pk_π to run **Sign**$(event, n, \vec{pk}_t, sk_\pi, M, t)$ and to produce a signature σ' given to $\mathcal{A}$.
4. $\mathcal{A}$ queries oracles adaptively. In particular, $\mathcal{A}$ is allowed to query any public key which is not pk_π to $\mathcal{CO}$.
5. $\mathcal{A}$ outputs n^*, n^* public keys $\vec{pk}_t^*$, a message M^*, and a signature $\sigma^* \neq \sigma'$.

A wins the game if:

- **Verify**$(event, n^*, \vec{pk}_t^*, M^*, \sigma^*, \text{t}) =$ **accept** on σ^* not an output of $\mathcal{SO}$;
- all of the public keys in $\vec{pk}_t^*$, $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;
- pk_π has not been queried to $\mathcal{CO}$; and

- **Link**(σ^*, σ') = **linked**.

We denote the adversary's advantage by $\mathbf{Adv}_{\mathcal{A}}^{NS}(\lambda) = Pr[\mathcal{A}$ wins the game$]$.

Definition 3. ***Non-slanderabilty****. An* ***FS-LRS*** *scheme is non-slanderable if for any PPT adversaries* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{NS}(\lambda)$ *is negligible.*

2.2.6. Forward-Secure Unforgeability

Forward-secure unforgeability ensures that it is not feasible for an adversary with a private key for a time period strictly greater than t to create valid signatures for any period less than or equal to t. Forward-secure unforgeability is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ given access to the oracles $\mathcal{JO}$, $\mathcal{CO}$, $\mathcal{SO}$ and $\mathcal{H}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a group size n, a set $\vec{pk}_t$ of n public keys, a message M, a time t and a signature σ.

A wins the game if:

- **Verify**$(e, n, \vec{pk}_t, M, \sigma, t) =$ **accept**;
- all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;
- no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$ at time t or earlier; and
- σ is not a query output of $\mathcal{SO}$.

We denote the adversary's advantage by $\mathbf{Adv}_{\mathcal{A}}^{FS-Unf}(\lambda) = Pr[\mathcal{A}$ wins the game$]$.

Definition 4. ***Forward-Secure Unforgability****. An* ***FS-LRS*** *scheme is forward-secure against forgeries if for PPT adversaries* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{FS-Unf}(\lambda)$ *is negligible.*

2.2.7. Forward-Secure Anonymity

Forward-secure anonymity ensures that it is not feasible for an adversary with a private key for a time period strictly greater than t to de-anonymise signatures for any time period less than or equal to t. Forward-secure anonymity is defined in a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ given access to oracles $\mathcal{JO}$, $\mathcal{CO}$, and $\mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a time t, a group size n, a set of $\vec{pk}_t$ of n public keys such that all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$, and a message M, Parsing the set $\vec{pk}_t$ as $\{pk_1, ..., pk_n\}$. $\mathcal{C}$ randomly picks $\pi \in \{1, ..., n\}$, and computes $\sigma_\pi =$ **Sign**$(e, n, \vec{pk}_t, sk_\pi, M, t)$, where sk_π is a valid private key corresponding to pk_π at time t. The signature σ_π is given to $\mathcal{A}$.
4. $\mathcal{A}$ outputs a guess $\pi' \in \{1, ..., n\}$.

A wins the game if:

- $\pi = \pi'$;
- e and t have never been input together to $\mathcal{SO}$; and
- no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$ at time t or earlier.

We denote the adversary's advantage by $\mathbf{Adv}_{\mathcal{A}}^{FS-Anon}(\lambda) = Pr[\mathcal{A}$ wins the game$]$.

Definition 5. ***Forward-Secure Anonymity****. An* ***FS-LRS*** *scheme is forward-secure anonymous if for any PPT adversaries* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{FS-Anon}(\lambda)$ *is negligible.*

3. Multilinear Maps

Since multilinear maps can be naturally presented as a generalisation of bilinear maps, and conversely bilinear maps as a concrete case of multilinear maps, we present the rest of our work using the the following multilinear map notation.

Our notation is similar to that used by Zhandry in [20]. Let $\mathcal{E}$ be an $l-$linear map over additive cyclic groups $[\mathbb{G}]_1, .., [\mathbb{G}]_l$ of prime order q, where $[\mathbb{G}]_0 = \mathbb{Z}_q$ and all $[\mathbb{G}]_i$ for $i = 1, ..., l$ are homomorphic to $(\mathbb{Z}_q, +)$. Let $[\alpha]_i$ denote the element $\alpha \in \mathbb{Z}_q$ raised to the level-i group $[\mathbb{G}]_i$, for $i \in (0, .., l)$. Let $\alpha \in_R [\mathbb{G}]_i$ denote the random sampling of an element in $[\mathbb{G}]_i$. We have access to efficient functions:

Addition, **Add** or $+$: Given two elements $[\alpha]_i, [\beta]_i$ returns $[\alpha + \beta]_i$.
Negation, **Neg** or $-$: Given one element $[\alpha]_i$ returns $[-\alpha]_i$.
Cross-level multiplication or multilinear **Map**, denoted $\mathcal{E}$: Given two elements $[\alpha]_i, [\beta]_j$, returns $[\alpha * \beta]_{i+j}$.

The cryptographic security of multilinear maps requires, among other things, that multiplication *within* any $[\mathbb{G}]_i$ be hard for $i > 0$.

We generalise the notation above and denote by $[\vec{a}]_i \in [\mathbb{G}^n]_i$ a vector of n elements in $[\mathbb{G}]_i$. Such a vector $[\vec{a}]_i$ is then also denoted by $([a_1]_i, ..., [a_n]_i)$.

3.1. Multilinear Assumptions

The discrete log problem when stated using multilinear map notation is as follows:

Definition 6 (Multilinear Discrete-log Problem (MDLP) [4])**.** *For any PPT algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\alpha]_1) = [\alpha]_0]$ is negligible, where $\alpha \in_R \mathbb{Z}_q$.*

This definition generalises to i-Decoding Problem (i-DP) to which the unforgability of our initial results reduced.

Definition 7 (i-Decoding Problem (i-DP) [4])**.** *For any PPT algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\delta]_i) = [\delta]_j]$ is negligible, where $j < i$ and $\delta \in_R \mathbb{Z}_q$.*

A related assumption to i-Decoding Problem is the k-Sub-exponent Multilinear Decoding Problem (k-SMDP). We note in passing that a specific case of k-SMDP where $k = 3$ on a Bilinear pairing is the Bilinear Computational Diffie–Hellman Problem.

Definition 8 (k-Sub-exponent Multilinear Decoding Problem.)**.** *For any PPT algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\vec{\alpha}]_1) = ([\vec{e}]_0, [\prod a_i^{e_i}]_{\sum e_i - 1})]$ is negligible, where $[\vec{\alpha}]_1 \in_R [\mathbb{G}^k]_1$, and $\sum e_i \leq l$. The adversary $\mathcal{A}$ is given access to an oracle $\mathcal{HO}$ which given $[\vec{w}]_0$ returns $[a_i^{w_i}]_{\sum w_i - 1}$. $\mathcal{A}$ only wins if $[\vec{e}]$ cannot be trivially derived from any $[\vec{w}]$.*

It is to the k-Sub-exponent Multilinear Decoding Problem that our combinatorial boosted variant's unforgability reduces. All three of the definitions presented so far are specific cases of the (k, b)-Generalised Multilinear Decoding Problem ((k, b)-GMDP).

Definition 9 ((k, b)-Generalised Multilinear Decoding Problem.)**.** *For any PPT algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\vec{\alpha}]_b) = ([\vec{e}]_0, [\prod a_i^{e_i}]_{b * \sum e_i - 1})]$ is negligible, where $[\vec{\alpha}]_b \in_R [\mathbb{G}^k]_b$ and $b * \sum e_i \leq l$. The adversary $\mathcal{A}$ is given access to an oracle $\mathcal{HO}$ which given $[\vec{w}, q]_0$ returns $[a_i^{w_i}]_q$. $\mathcal{A}$ only wins if $[\prod a_i^{e_i}]_{b * \sum e_i - 1}$ cannot be trivially derived from any $[a_i^{w_i}]_q$.*

In our unforgeability-related reductions, we reduce to the (k, b)-Generalised Multilinear Decoding Problem for convenience and conciseness. Depending on the parameters with which our scheme was instantiated, this means the proof implies a reduction to one of the three specific cases:

1. If configured similar to Liu et al. [1], it reduces to $MDLP$.
2. If configured similar to our initial results, it reduces to i-DP.
3. If configured to exploit the combinatorial trick, it reduces to k-SMDP.

In the same way that the Discrete Log Problem is generalised to Multilinear Discrete Log Problem (MDLP), the Decisional Diffie–Hellman problem generalises to Multilinear Decisional Diffie–Hellman (MDDH) problem. Intuitively, given three group elements, it is infeasible to tell if one is the product of the others, provided that the sum of any two levels is greater than maximum allowed.

Definition 10 (Multilinear Decisional Diffie–Hellman Problem i-(MDDH))**.** *For any PPT $\mathcal{A}$, the distinguishing probability $Pr[\mathcal{A}([\alpha]_i, [\beta]_{l-i+1}, [\gamma]_l) = \text{“true”} - \mathcal{A}([\alpha]_i, [\beta]_{l-i+1}, [\alpha\beta]_l) = \text{“true”}]$ is negligible, where $\alpha, \beta, \gamma \in_R \mathbb{Z}_q$ and $i \geq 1$.*

The parallel to k-SMDP for anonymity is k-Sub-exponent Multilinear Decisional Diffie–Hellman. We, again, note in passing that a specific case of k-SMDDH where $k = 3$ on a Bilinear pairing is the Bilinear Decisional Diffie–Hellman Problem.

Definition 11 (Sub-exponent Multilinear Decisional Diffie–Hellman Problem k-(SMDDH))**.** *For any PPT $\mathcal{A}$, the distinguishing probability $Pr[\mathcal{A}([\vec{e}]_0, [\vec{\alpha}]_1, [\beta]_{l-\sum e_i+1}, [\gamma]_l) = \text{“true”} - \mathcal{A}([\vec{e}]_0, [\vec{\alpha}]_1, [\beta]_{l-\sum e_i+1}, [\alpha_i^{e_i}\beta]_l) = \text{“true”}]$ is negligible, where $[\vec{\alpha}]_1 \in_r [\mathbb{G}^k]_1$ and $\beta, \gamma \in_R \mathbb{Z}_q$ and $\sum e_i \geq 1$. The adversary $\mathcal{A}$ is given access to an oracle $\mathcal{HO}$ which given $[\vec{w}]_0$ returns $[a_i^{w_i}]_{\sum w_i - 1}$. $\mathcal{A}$ only wins if $[\vec{e}]$ cannot be trivially derived from any $[\vec{w}]$.*

As with the previous set of assumptions we present a generalised assumption which includes both MDDH and SMDDH as specific cases. We call this assumption Generalised Sub-exponent Multilinear Decisional Diffie–Hellman Problem (k, b).

Definition 12 (Generalised Sub-exponent Multilinear Decisional Diffie–Hellman Problem (k, b)-(GMDDH))**.** *For any PPT $\mathcal{A}$, the distinguishing probability $Pr[\mathcal{A}([\vec{e}]_0, [\vec{\alpha}]_b, [\beta]_{l-b*\sum e_i+1}, [\gamma]_l) = \text{“true”} - \mathcal{A}([\vec{e}]_0, [\vec{\alpha}]_b, [\beta]_{l-b*\sum e_i+1}, [\alpha_i^{e_i}\beta]_l) = \text{“true”}]$ is negligible, where $[\vec{\alpha}]_b \in_r [\mathbb{G}^k]_b$ and $\beta, \gamma \in_R \mathbb{Z}_q$ and $b * \sum e_i \geq 1$. The adversary $\mathcal{A}$ is given access to an oracle $\mathcal{HO}$ which given $[\vec{w}, q]_0$ returns $[a_i^{w_i}]_q$. $\mathcal{A}$ only wins if $[\prod a_i^{e_i}]_{b*\sum e_i - 1}$ cannot be trivially derived from any $[a_i^{w_i}]_q$.*

In our forward-anonymity reduction, we reduce to the (k, b)-Generalised Sub-exponent Multilinear Decisional Diffie–Hellman Problem for convenience and conciseness. Depending on the parameters with which our scheme was instantiated, this means the proof implies a reduction to one of the three specific cases:

1. If configured similar to Liu et al. [1], it reduces to DDH.
2. If configured similar to our initial results, it reduces to i-MDDH.
3. If configured to exploit the combinatorial trick, it reduces to k-SMDDH.

3.2. Is Multilinearity Achievable?

Three major multilinear map candidates have been proposed in [4,14,15]. Since their introduction, they have been the targets of many attacks, patches, and more attacks that remain unpatched.

One powerful class of attacks on multilinear maps are the so-called “zeroising” attacks; they run in polynomial time but require the availability of an encoding of zero in the lower levels of the multilinear ladder [4,21]. There are also sub-exponential and quantum attacks [22–24]. Further to this, recently Miles et al. introduced a class of “annihilation” attacks on multilinear maps [25].

There are reasons to believe that multilinear maps may be unrealisable. In particular, their near-equivalence to indistinguishability obfuscation [26]—an extremely powerful tool which in an even stronger variant is known not to exist [27]—is worrying. Furthermore, Boneh and Silverberg [11],

in their original paper on applications of hypothetical multilinear maps, presented several results which cast doubt on the likeliness of multilinear maps' existence, and soberingly conclude that "such maps might have to either come from outside the realm of algebraic geometry, or occur as 'unnatural' computable maps arising from geometry."

If multilinear maps fail to be repaired, bilinear maps still give us an efficient two-period FS-LRS scheme that can be combinatorially boosted to multiple periods.

4. Construction

4.1. Notation

For conciseness, we use a number of specific notations. We use a mod b to denote the remainder of a divided by b. Given two vectors of group elements of equal length $\vec{a}^1$ and $\vec{a}^2$ we will use $\vec{a}^\delta$ to refer to $(a_1^1 - a_1^2, ..., a_n^1 - a_n^2)$, we also let $\vec{a}^{\delta'}$ refer to $(a_1^2 - a_1^1, ..., a_n^2 - a_n^1)$.

4.2. Intuition

We now give the basic intuition of the scheme. We note that the first two paragraphs apply to almost all schemes following from Liu et al. [1]. The formal details follow in Section 4.6.

To ensure unconditional anonymity in spite of linkability, a Pedersen commitment can provide unconditional hiding with computational binding of the private key in the public key. The Pedersen commitment uses a common reference string consisting of two elements $[g]$ and $[h]$ of unknown relation. The private key is a pair $[x]$ and $[y]$ whose commitment, $[gx + hy]$ is the public key. A multilinear map can then raise and ratchet the private key at each time period, which provides forward security.

In the signature, we use the Fiat–Shamir heuristic on two knowledge-of-discrete-logarithm proofs, rolled into one. Using a topic specific value d, the signer proves firstly that they know x behind $f = dx$, and secondly that they know x and y such that $gx + hy$ is one of the public keys. Random challenges c_i serve as decoys for the other public keys. Since both the real challenge c and the decoy challenges c_i are uniformly random, an adversary is unable to discern which party signed.

This basic system, described above, is adapted with a combinatorial trick which, for a choice of k, leverages an l-linear map for $\sum_{i=1}^{l} \binom{k+i-1}{i}$ time periods. (See our preliminary results in [3] for a clean presentation of the basic system with full details. We stress that the general system, presented here, implies the preliminary results but is more complicated in its presentation.) The value $k \geq 1$ may be freely chosen but the public key and private key sizes are linear in k.

4.3. Explanation of Parameters

We wish to avoid presenting, and proving, the generalised version—based on the combinatorial trick—separately from our initial result. For this reason, the presentation that follows is deliberately abstract and parameterised so that both the general version and the original result are special cases. This is necessary since neither version is a special cases of the other. The abstract presentation also means that the proofs are the same for both. We parameterise on four inputs:

1. The multilinear map size l
2. The combinatorial constant k
3. The initial public key level b
4. The number of time periods $\mathcal{T}$

Our initial result in [3] is a special case of the following construction where $k = 1$ and $b = l$. The variant including the combinatorial trick is a special case where $b = 1$. To make the paper easier to read, we present a slightly simplified construction which requires that b is either 1 or l. We believe that the more general construction for an arbitrary choice of $b \leq l$ works but since it significantly complicates the presentation we omit the details.

4.4. Time t to Key Indices

The public key for a time period t is a particular multiplicative combination of public key vector. For instance, for the case of $l = 2$, $k = 3$, $b = 1$, the initial public key vector is $[Z_1]_1, [Z_2]_1, [Z_3]_1$ and private key vector is $([x_1]_0, [y_1]_0), ([x_2]_0, [y_2]_0), ([x_3]_0, [y_3]_0)$. After the signer generates the initial private key vector, they store $[x_1]_1, [x_2]_1, [x_3]_1, [x_1h]_1, [x_2h]_1, [x_3h]_1, [y_1]_0, [y_2]_0, [y_3]_0$ for the length of the election, and temporally store $[x_1]_0, [x_2]_0, [x_3]_0$. We call the vector stored for the length of the election the semi-private key vector since it need only be kept secret for unconditional anonymity but even if it where public, it would not allow an adversary to break privacy without breaking (k,b)-GMDDH.

To sign at time t, the signer needs the private key for that time and the semi-private key vector. The nine time periods have the following states, shown in Table 1.

Table 1. Overview of state changes.

Time Period	Public Key	Private Key	Private Store
0	$[Z_1]_1$	$[x_1]_0$	$[x_1^2]_1, [x_2]_0, [x_3]_0$
1	$[Z_1^2]_2$	$[x_1^2]_1$	$[x_2]_0, [x_3]_0$
2	$[Z_2]_1$	$[x_2]$	$[x_2x_1]_1, [x_2^2]_1[x_3]_0$
3	$[Z_2Z_1]_2$	$[x_2x_1]$	$[x_2^2]_1[x_3]_0$
4	$[Z_2^2]_2$	$[x_2^2]$	$[x_3]_0$
5	$[Z_3]_1$	$[x_3]_0$	$[x_3x_1]_1, [x_3x_2]_1, [x_3^2]_1$
6	$[Z_3Z_1]_2$	$[x_3x_1]_1$	$[x_3x_2]_1, [x_3^2]_1$
7	$[Z_3Z_2]_2$	$[x_3x_2]_1$	$[x_3^2]_1$
8	$[Z_3^2]_2$	$[x_3^2]_1$	

We denote $PK(t \in \mathcal{T})$ the function which when given a time period t returns the set of indices needed to generate public key for that time period. We denote the output of $PK(t)$ as τ and the cardinality of τ as κ. This function is very easy to evaluate if the value for $t-1$ is known, so in high-use situations it would be practical to remember the previous value and increment. However, if a more direct method is desirable, one possible implementation follows in Algorithm 1.

Algorithm 1: Index generation for time period t.

Data: t
Result: indices
indices = $\varnothing$;
temp = t
while $temp \geq 0$ **do**
  find the smallest $i \in (1,...,k)$ where $t < bi + \sum_{s=1}^{i} \sum_{j=b}^{l} \binom{i+j-1}{j}$
  indices = indices $\cup \{i\}$
  $temp = temp - \sum_{s=1}^{i-1} \sum_{j=b}^{l-1} \binom{i+j-1}{j} - b$
end

4.5. Public Key Structure

When $b = l$, as in our original results, the public keys have a very simple structure. However, the public key in the extended version, for a time t and $\tau = PK(t)$, is of the form $[\prod_{i \in \tau} Z_i]_\kappa$. For example, if $\tau = (1,2)$, then the public key is $[g^2x_1x_2 + gx_1hy_2 + gx_2hy_1 + h^2y_1y_2]$. To simply the presentation, we let $KP(i, \tau, \vec{x}, \vec{y}) = \sum_{j \in \binom{\tau}{i} j' \in \binom{\tau/j}{k-i}} (\prod_{\iota \in j} x_\iota \prod_{\iota \in j'} y_\iota)$. In the example above, $KP(0, (1,2), \vec{x}, \vec{y}) = y_1y_2$, $KP(1, (1,2), \vec{x}, \vec{y}) = x_1y_2 + x_2y_1$ and $KP(2, (1,2), \vec{x}, \vec{y}) = x_1x_2$. The public key can be generated as $\sum_{i=0}^{\kappa} g^i h^{\kappa-i} KP(i, \tau, \vec{x}, \vec{y})$. We sometimes just write $KP(i)$, when $\tau, \vec{x}, \vec{y}$ are clear from context.

4.6. Formal Description

4.6.1. Setup($\lambda, \mathcal{T}$)(k,l,b)

Take as input: The multilinear map size l, the combinatorial constant k, the initial key level b, and the number of time periods $\mathcal{T} \geq 1$. It is required that $bk + \sum_{i=b+1}^{l} \binom{k+i-1}{i} \geq \mathcal{T}$. Denote by $t \in (0, 1, 2, ..., \mathcal{T}-1)$ the current time period. Run a multilinear map setup algorithm to construct a bounded-level l-multilinear map and obtain its public parameters **mmpp**. Let H_i denote the ith element in a family of hash functions H such that H_i: $\{0,1\}^* \rightarrow [\mathbb{G}]_i$. Construct $[g]_0 = H_0$("Generator-g") and $[h]_b = H_b$("Generator-h"). The public **param** are (**mmpp**, $k, b, [g]_0, [h]_b, H$,"Generator-g","Generator-h").

4.6.2. KeyGen(**param**)

Sample $[\vec{x}]_0, [\vec{y}]_0 \in_R [\mathbb{G}^k]_0$ and let $[\vec{Z}]_b = ([Z_1]_b, ..., [Z_k]_b)$, where $[Z_i]_b = \mathcal{E}(\mathcal{E}([g]_0, [x_i]_0), [1]_b) + \mathcal{E}([h]_b, [y_i]_0) = [g * x_i + h * y_i]_b$. The public key is $pk = [\vec{Z}]_b$ and initial secret key $sk = ([\vec{x}]_0, [\vec{y}]_0)$. Store $([\vec{x}]_b, ([x_0 h]_b, ..., [x_k h]_b), [\vec{y}]_0)$ for the duration of the election and $[\vec{x}]_0$ temporarily.

4.6.3. Sign(**param**, *event*, n, $\vec{pk}_t$, sk_π, M, t)

On input $(event, n, \vec{pk}_t, sk_\pi, M, t)$, with: *event* some description, n the ring size, $\vec{pk} = \{pk_1, ..., pk_n\} = \{[\vec{Z}_1]_b, ..., [\vec{Z}_n]_b\}$ the ring public keys, sk_π the signer's secret key with public key $pk_\pi \in \vec{pk}_t$ (without loss of generality (w.l.o.g.) $\pi \in [1, n]$), M the message, and t the time period. The signer runs $PK(t)$ and gets τ whose cardinality we denote κ, and generates the specific public keys for time t as $[\vec{pk}']_{\kappa * b}$ where $[pk'_i]_{\kappa * b} = [\prod_{j \in \tau} Z_{i,j}]_{\kappa * b}$. Let $\nu = \kappa + (\mathcal{T} \bmod b) - 1$. The signer (holder of $sk_\pi = ([\prod_{j \in \tau} x_j]_\nu, ..., [\prod_{j \in \tau} y_j]_0)$ does the following:

1. Hash $[d]_{l-\nu} = H_{l-\nu}(t||event)$, and multilinearly map $[f]_l = \mathcal{E}([d]_{l-\nu}, [\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)]_\nu)$.
2. For $i \in (0, ..., \kappa)$ sample $[r_i]_0 \in_R [\mathbb{G}]_0$ and $[c_1]_0, ..., [c_{\pi-1}]_0, [c_{\pi+1}]_0, ..., [c_n]_0 \in_R [\mathbb{G}]_0$.
3. Compute $[K]_l = \sum_{i=0}^{\kappa} \mathcal{E}([r_i]_i, [g^i h^{\kappa-i}]_{l-i}) + \sum_{i=1, i \neq \pi}^{n} \mathcal{E}([pk'_i]_l, [c_i]_0)$, and $[K']_l = \mathcal{E}([d]_{l-\nu}, [\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} r_i]_\nu) + \mathcal{E}([f]_l, \sum_{i=1, i \neq \pi}^{n} [c_i]_0)$.
4. Find $[c_\pi]_0$ s.t. $[c_\pi]_0 = H_0(\vec{pk}_t||event||[f]_l||M||[K]_l||[K']_l||t) - \sum_{i=1, i \neq \pi}^{n} [c_i]_0$.
5. Compute $[\tilde{r}_0]_0 = [r_0 - c_\pi KP(0)]_0$ and $[\tilde{r}_1]_\nu = \sum_{i=1}^{\kappa} \mathcal{E}([g^{i-\kappa} h^{\kappa-i}], [r_i - c_\pi KP(i)])$.
6. Output the signature $\sigma = ([f]_l, [\tilde{r}_0]_0, [\tilde{r}_1]_\nu, [c_1]_0, ..., [c_n]_0)$.

4.6.4. Verify(**param**, *event*, n, $\vec{pk}_t$, M, σ, t)

On input $(event, n, \vec{pk}_t, M, \sigma, t)$, first run $PK(t)$ and get τ, and generate the specific public keys for time t as $[\vec{pk}']_{\kappa * b}$ where $[pk'_i]_{\kappa * b} = [\prod_{j \in \tau} Z_{i,j}]_{\kappa * b}$. Let $\nu = \kappa + (\mathcal{T} \bmod b) - 1$ and $[d]_{l-\nu} = H_{l-\nu}(t||event)$ and, using the components of $\sigma = ([f]_l, [\tilde{r}_0]_0, [\tilde{r}_1]_\nu, [c_1]_0, ..., [c_n]_0)$, compute

$$[K]_l = \mathcal{E}([\tilde{r}_0]_0, [h^\kappa]_l) + \mathcal{E}([\tilde{r}_1]_\nu, [g^\kappa]_{l-\nu+1}) + \sum_{i=1}^{n} \mathcal{E}([pk'_i]_l, [c_i]_0)$$

$$[K']_l = \mathcal{E}([d]_{l-\nu}, [\tilde{r}_1]_\nu) + \mathcal{E}([f]_l, \sum_{i=1}^{n} [c_i]_0)$$

$$\text{and} \quad [c_0]_0 = H_0(\vec{pk}_t||event||[f]_l||M||[K]_l||[K']_l||t)$$

then check and output whether $\sum_{i=1}^{n} [c_i]_0 = [c_0]_0$.

4.6.5. Link(**param**, *event*, t, n_1, n_2, $\vec{pk}_{t_1}$, $\vec{pk}_{t_2}$, M_1, M_2, σ_1, σ_2)

On input two signatures $\sigma_1 = ([f_1]_l, *)$ and $\sigma_2 = ([f_2]_l, *)$, two messages M_1 and M_2, an event description *event*, and a time t, first check whether the two signatures are valid. If yes, output **linked** if $[f_1]_l = [f_2]_l$; else output **unlinked**.

4.6.6. Private-Key Update(**param**, sk_t)

In a given time period t, to update the private key store for time period $t+1$, run $PK(t)$ and get τ. Let $\nu = \kappa + (\mathcal{T} \bmod b) - 1$. Remove $([\prod_{j\in\tau} x_j]_\nu$ from the private key store and, if $\nu < l$.

If $b = 1$, for all $\alpha \in (1, ..., \tau_\kappa)$ add to the store $[x_\alpha \prod_{j\in\tau} x_j]_{\nu+1}$, else add to the store $[\prod_{j\in\tau} x_j]_{\nu+1}$. (We note again that this can be done very efficiently incrementally but we present a more general approach).

If $\nu = l$ no update is required.

4.6.7. Public-Key Update(**param**, Z_t)

The public key does not need to be updated in our scheme.

4.7. Space and Time Complexity

We briefly detail the complexity of the scheme when instantiated on bilinear maps. The complexity is parameterised over the size of the ring n and the number of time periods $\mathcal{T}$. The complexity is shown in Table 2.

Table 2. Space and time complexity analysis.

Complexity Analysis	
Property	**Complexity**
Signature Size (Group Elements)	$n+3$
Public Key Size (Group Elements)	$\sqrt{\mathcal{T}}$
Secret Key Size (Group Elements)	$3 * \sqrt{\mathcal{T}}$
Signing Complexity (Pairings and Multiplications)	$\mathcal{O}(n)$
Verifying Complexity (Pairings and Multiplications)	$\mathcal{O}(n)$
Link Complexity (on Verified Signatures)	1
Private-Key Update Complexity (Pairings and Multiplications)	$\sqrt{\mathcal{T}}$
Public-Key Update Complexity (Pairings and Multiplications)	0

The exact complexity of the signing and verifying algorithms depends on what optimisations are used, which in turn depends on the context of deployment, in all cases the constant is small. The table makes clear the main advantage of our construction; namely, that the key size is sublinear in the number of time periods which preserving the asymptotic complexity of singing and verifying.

5. Correctness

5.1. Verification Correctness

For verification correctness, it suffices to show that the verification values K and K' calculated by each party are the same. We denote by $[K_s]$ and $[K_v]$, K as calculated by the signer and verifier respectively; we adopt the same notation for K'. The equations below show that $[K_s] = [K_v]$ and $[K'_s] = [K'_v]$ by applying the definitions of $[\tilde{r}_0]$ and $[\tilde{r}_1]$.

For K:

$$\begin{aligned}
[K_s] &= \sum_{i=0}^{\kappa} [r_i g^i h^{\kappa-i}] + \sum_{i=1, i\neq\pi}^{n} [pk'_i c_i] \\
&= [\tilde{r}_0 h^\kappa] + [h^\kappa c_\pi KP(0)] + [\tilde{r}_1 g^\kappa] + [\sum_{i=1}^{\kappa} c_\pi KP(i) g^i h^{\kappa-i}] + \sum_{i=1, i\neq\pi}^{n} [pk'_i c_i] \\
&= [\tilde{r}_0 h^\kappa] + [\tilde{r}_1 g^\kappa] + \sum_{i=1}^{n} [pk'_i c_i] \\
&= [K_v]
\end{aligned}$$

For K':

$$\begin{aligned}[K'_s] &= [d\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}r_i] + [f\sum_{i=1,i\neq\pi}^{n} c_i])\\ &= [d\tilde{r}_1] + [c_\pi d\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)] + [f\sum_{i=1,i\neq\pi}^{n} c_i]\\ &= [d\tilde{r}_1] + [f\sum_{i=1}^{n} c_i]\\ &= [K'_v]\end{aligned}$$

5.2. Linking Correctness

For a given event *event*, time t, and private key, the linking component $[f]_l$, computed from $[d]_{l-\nu} = H_{l-\nu}(t||event)$, is completely deterministic. Since the linking component is deterministic, under the above conditions, given any two signatures a simple equality check on the linking component suffices. Conversely, for a given event *event*, time t, and two different private keys, the linking element will be different. (While it is possible for two different private keys to have the same public key, violating the assertion above, this would also break the Pedersen commitments and reveal the relationship between g and h. It is also possible for the hash function to collide. These events are assumed of negligible probability.)

5.3. Update Correctness

To simplify the presentation we present the argument for b= 1 and $b = l$ separately.

If $b = l$, the "effective" public key for the first l time periods will be $[x_1 g + y_1 h]_l$ and then $[x_2 g + y_2 h]_l$ for the next l and so on. At every time period, the signer must be able to calculate the private key $([x_{t/l}]_{t \bmod l}, [y_{t/l}]_0)$. The private key update ensures the signer will always be able to generate private keys for the remaining "effective" public keys. First, notice that the signer always has $[\vec{y}]_0$ and that they start with $[\vec{x}]_0$. Secondly, the private key update for a time t removes $[x_{t/l}]_{t \bmod l}$ from the private key store and, and adds to the store $[x_{t/l}]_{(t \bmod l)+1}$.

If $b = 1$, then the public keys are defined as a multiplicative combination of the initial public keys $([Z_1]_1, ..., [Z_k]_1)$. Although the order in which the keys are used is optimised to reduce the private key size, the set of public keys for all time periods is composed of all the ways to choose k initial keys, with replacement, at level k for all $k \in (1, ..., l)$; we denote these combinations by τ. For each public key denoted $[\prod_{j\in\tau} Z_j]_\kappa$, the corresponding private key is $([\prod_{j\in\tau} x_j]_\nu, ..., [\prod_{j\in\tau} y_j]_0)$. To see that private key update ensures the signer will always be able to generate private key for the remaining public keys: First, notice that the signer always has $[\vec{y}]_0$ and that they start with $[\vec{x}]_0$. Secondly, the private key update removes the current private key $[\prod_{j\in\tau} x_j]_\nu$ from the private key store, and adds to the store $[x_\alpha \prod_{j\in\tau} x_j]_{\nu+1}$ for $\alpha \in (1, ..., \tau_\kappa)$.

6. Security

The proofs (other than the forward security ones) are similar to those of Liu et al. [1]. Despite the structural similarities, all of our reductions are exponentially more efficient.

Theorem 1. *The FS-LRS scheme is forward-secure against forgeries in the random-oracle model, if (k,b)-GMDP is hard.*

Proof. We show that the ability of the adversary to make corruption queries at times later than t does not allow it to calculate the private key or forge signatures at time t, without breaking (k, b)-GMDP.

Given an (k, b)-GMDP instance $([\vec{\alpha}]_b)$, $\mathcal{B}$ is asked to output some $([\vec{e}]_0, [\alpha_i^{e_i}]_{b*\sum e_i - 1})$ where $\sum e_i \leq l)$. $\mathcal{B}$ picks $[g]_0, [h']_0 \in_R [\mathbb{G}]_0$ and sets $[h]_b = \mathcal{E}([h'g]_0, [1]_b)$. $\mathcal{B}$ simulates the oracles thus:

- *Random Oracles* H_i: For query input H_0("GENERATOR-g"), $\mathcal{B}$ returns $[g]_0$. For query input H_b("GENERATOR-h"), $\mathcal{B}$ returns $[h]_b$. For other queries, $\mathcal{B}$ randomly picks $[\lambda]_0 \in_R [\mathbb{G}]_0$, sets $[a]_i = \mathcal{E}([\lambda]_0, [1]_i)$ and returns $[a]_i$.
- *Joining Oracle* $\mathcal{JO}$: Assume $\mathcal{A}$ can only query $\mathcal{JO}$ for a maximum n' times, where $n' \geq n$. W.l.o.g., $(1, ..., n)$ will be the indices for which $\mathcal{B}$ does not know the private keys and embeds the challenge, and $(n+1, ..., n')$ be the indices for which a private key is known. For the first n indices, $\mathcal{B}$ chooses $[\vec{x}]_0, [\vec{y}]_0 \in_R [\mathbb{G}^k]_0$ and sets $[\vec{Z}]_b = ([Z_1]_b, ..., [Z_k]_b)$ where $[Z_i]_b = [g\alpha_i x_i]_b + [hy_i]_b$. For the remaining indices it generates the public/private key pair as in the scheme. Upon the jth query, $\mathcal{B}$ returns the matching public key.
- *Corruption Oracle* $\mathcal{CO}$: On input a public key pk_i obtained from $\mathcal{JO}$, and a time t, $\mathcal{B}$ checks whether it is corresponding to $[n+1, n']$, if yes, then $\mathcal{B}$ returns the private key. Otherwise, $\mathcal{B}$ calls $\mathcal{O}(\tau, \nu)$ returns $sk_i = ([\prod_{j\in\tau} x_{i,j}\alpha_j]_\nu, ..., [\prod_{j\in\tau} y_{i,j}]_0)$.
- *Signing Oracle* $\mathcal{SO}$: On input a signing query for event *event*, a set of public key $\vec{pk}_t = \{[\vec{Z}_1]_b, ..., [\vec{Z}_n]_b\}$, the public key for the signer $[\vec{Z}_\pi]_b$, where $\pi \in [1, n]$, and a message M, and time t, $\mathcal{B}$ simulates as follows:

  1. If the query of $H_{l-\nu}(t||event)$ has not been made, carry out the H-query of $t||event$ as described above. Set $[d]_{l-\nu}$ to $H_{l-\nu}(t||event)$. Note that $\mathcal{B}$ knows the $[\lambda]_0$ that corresponds to $[d]_{l-\nu}$. $\mathcal{B}$ sets $[f]_l = [d * \sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)]_l$, which it can compute from the challenge $[\vec{\alpha}]_b$.
  2. If $\pi \in (n+1, ..., n')$, $\mathcal{B}$ knows the private key and computes the signature according to the algorithm.
  3. Otherwise, $\mathcal{B}$ randomly chooses $[\tilde{r}_0]_0 \in_R [\mathbb{G}]_0$ and $[\tilde{r}_1]_\nu \in_R [\mathbb{G}]_\nu$ and $[c_i]_0 \in_R [\mathbb{G}]_0$ for all $i \in [1, n]$ and sets the H_0 oracle output of

$$H_0\Big(\vec{pk}_t||event||[f]_l||M||\mathcal{E}([\tilde{r}_0]_0, [h^\kappa]_1) + \mathcal{E}([\tilde{r}_1]_\nu, [g^\kappa]_{l-\nu}) + \sum_{i=1}^{n} \mathcal{E}([pk'_i]_l, [c_i]_0)||\mathcal{E}([d]_{l-\nu}, [\tilde{r}_1]_\nu) + \mathcal{E}([f]_l, \sum_{i=1}^{n}[c_i]_0)||t\Big)$$

  4. $\mathcal{B}$ returns the signature $\sigma = ([f]_l, [\tilde{r}_0]_0, [\tilde{r}_1]_\nu, [c_1]_0, ..., [c_n]_0)$. $\mathcal{A}$ cannot distinguish between $\mathcal{B}$'s simulation and real life.

For one successful simulation, suppose the forgery returned by $\mathcal{A}$, on an event *event*, time t and a set of public keys $\vec{pk}''_t \in [1, ..., n]$, is $\sigma^1 = ([f]_l, [\tilde{r}_0^1]_0, [\tilde{r}_1^1]_\nu, [c_1^1]_0, ..., [c_n^1]_0)$. In the random-oracle model, $\mathcal{A}$ must have queried $H_{l-\nu}(t||event)$, denoted by $[d]_{l-\nu}$, and queried $H_0(\vec{pk}''||event||[f]_l||M||[K]_l||[K']_l||t)$ where

$$[K]_l = \mathcal{E}([\tilde{r}_0^1]_0, [h^\kappa]_l) + \mathcal{E}([\tilde{r}_1^1]_\nu, [g^\kappa]_{l-\nu}) + \sum_{i=1}^{n} \mathcal{E}([pk'_i]_l, [c_i^1]_0) \text{ and}$$
$$[K']_l = \mathcal{E}([d]_{l-\nu}, [\tilde{r}_1^1]_\nu) + \mathcal{E}([f]_l, \sum_{i=1}^{n}[c_i^1]_0)$$

After a successful rewind, we get another $\sigma^2 = ([f]_l, [\tilde{r}_0^2]_0, [\tilde{r}_1^2]_\nu, [c_1^2]_0, ..., [c_n^2]_0)$. Note that $[f]_l$, $[K]_l$, and $[K']_l$ must be the same in both signatures since we rewind only to the point of random oracle query. In the rewound execution, we force a change in the H_0 oracle output to the query which determines $\sum_{i=1}^{n}[c_i]$. Let λ denote those points $\in (1, ..., n)$ where $c_i^1 \neq c_i^2$. We can the extract $[\prod_{j\in\tau} \alpha_j]_\nu$ as $[\frac{\tilde{r}_0^\delta h^\kappa + \tilde{r}_1^\delta g^\kappa - \sum_{i\in\lambda} c_i^{\delta'} \sum_{j=0}^{\kappa-1} g^j h^{\kappa-j} KP(j)}{g^\kappa \sum_{i\in\lambda} c_i^{\delta'} \prod_{j\in\tau} x_{i,j}}]_\nu$. We demonstrate the correctness of the extraction, below, by showing that since $[K]$ simultaneously satisfies two equations the correctness follows by simple algebraic manipulation and the format of the keys.

$$[K] = [K]$$

We begin by substituting $[K]$ for the two equations it satisfies,

$$[\tilde{r}_0^1 h^\kappa] + [\tilde{r}_1^1 g^\kappa] + \sum_{i=1}^{n} [pk'_i c_i^1] = [\tilde{r}_0^2 h^\kappa] + [\tilde{r}_1^2 g^\kappa] + \sum_{i=1}^{n} [pk'_i c_i^2]$$

By subtraction, we have

$$[\tilde{r}_0^{\delta} h^{\kappa}] + [\tilde{r}_1^{\delta} g^{\kappa}] = [\sum_{i \in \lambda} pk'_i c_i^{\delta'}]$$

By definition of pk'_i,

$$[\tilde{r}_0^{\delta} h^{\kappa}] + [\tilde{r}_1^{\delta} g^{\kappa}] = [\sum_{i \in \lambda} c_i^{\delta'} (g^{\kappa} \prod_{j \in \tau} x_{i,j} a_j + \sum_{j=0}^{\kappa-1} g^j h^{\kappa-j} KP(j))]_l$$

By subtraction, we have

$$[\tilde{r}_0^{\delta} h^{\kappa} + \tilde{r}_1^{\delta} g^{\kappa} - \sum_{i \in \lambda} c_i^{\delta'} \sum_{j=0}^{\kappa-1} g^j h^{\kappa-j} KP(j)] = [\sum_{i \in \lambda} c_i^{\delta'} g^{\kappa} \prod_{j \in \tau} x_{i,j} \alpha_j]$$

By division,

$$[\frac{\tilde{r}_0^{\delta} h^{\kappa} + \tilde{r}_1^{\delta} g^{\kappa} - \sum_{i \in \lambda} c_i^{\delta'} \sum_{j=0}^{\kappa-1} g^j h^{\kappa-j} KP(j)}{g^{\kappa} \sum_{i \in \lambda} c_i^{\delta'} \prod_{j \in \tau} x_{i,j}}]_{\nu} = [\prod_{j \in \tau} \alpha_j]_{\nu}$$

By the forking lemma [7], the chance of each successful rewind simulation is at least $\xi/4$, where ξ is the probability that $\mathcal{A}$ successfully forges a signature. Hence, the probability that for a given adversary $\mathcal{A}$ we can extract $[\prod_{j \in \tau} \alpha_j]_{\nu}$ is $\frac{\xi}{4}$ times the probability that $\vec{pk}''_t \subseteq (1, ..., n)$. □

Theorem 2. *The FS-LRS scheme is unconditionally anonymous.*

Proof. The proof of unconditional anonymity is largely unchanged from [1], since both schemes rely on Pederson commitments. For each $\mathcal{JO}$ query, a value $[\vec{Z}] = ([Z_1], ..., [Z_k])$ where $[Z_i] = (\mathcal{E}([g], [x_i]) + \mathcal{E}([h], [y_i])$ is returned for some random pair $([\vec{x}], [\vec{y}])$. The challenge signature is created from the key of a random user in the ring. In contrast to Liu et al., we do allow the adversary access to the signing oracle. The access we grant gives the adversary the ability to learn, and therefore we assume knowledge of, the set of all genuine $[\vec{x}_i]$ in the challenge ring. Crucially, it does not grant the ability to learn which key part belongs to which public key.

In what follows, we show that the advantage of the adversary is information- theoretically zero. The proof is divided into three parts. First, we show that given a signature $\sigma = ([f]_l, [\tilde{r}_0]_0, [\tilde{r}_1]_{\nu}, [c_1]_0, ..., [c_n]_0)$ for a ring $([pk'_1], ..., [pk'_n])$ on message M, event *event* and time t, there exists a matching private key $([KP(\kappa)], ..., [KP(0)])$ for each possible public key $[\prod_{i \in \tau} Z_{\pi,i}]$, for any $\pi \in \{1, ..., n\}$, that can construct the linking tag $[f]$. That is, $[f] = \mathcal{E}(H_{l-\nu}(t||event), [\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)]_{\nu}) = \mathcal{E}([d], [\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)])$, where $[d] = H_{l-\nu}(t||event)$. Second, given a private key $([\prod_{i \in \tau} x_{\pi,i}], ..., [\prod_{i \in \tau} y_{\pi,i}])$, there exists a tuple $([r_{0_\pi}], ..., [r_{\kappa_\pi}])$ so that σ matches $([\prod_{i \in \tau} x_{\pi,i}], ..., [\prod_{i \in \tau} y_{\pi,i}])$ using randomness $([r_{0_\pi}], ..., [r_{\kappa_\pi}])$. Finally, for any $\pi \in \{1, ..., n\}$, the distribution of the tuple $([\prod_{i \in \tau} x_{\pi,i}], ..., [\prod_{i \in \tau} y_{\pi,i}], [r_{0_\pi}], ..., [r_{\kappa_\pi}])$ defined in parts one and two is identical.

Therefore, in the view of the adversary, the signature σ is independent to the value π, the index of the actual signer. We conclude that even an unbounded adversary cannot guess the value of π better than at random. In details:

1. *Part I.* Let x, y be so that $[f] = \mathcal{E}([d], [x])$ and $[g] = \mathcal{E}([h], [y])$. Let $[z_i]$ be so that $[pk'_i] = \mathcal{E}([g^{\kappa}], [z_i])$ for $i = 1$ to n. For each $\pi \in \{1, ..., n\}$, consider the values

$$[\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)] = [x], \qquad \text{and} \qquad [g^{-\kappa} KP(0)] = \mathcal{E}([z_\pi] - [\sum_{i=1}^{\kappa} g^{i-\kappa} h^{\kappa-i} KP(i)], [y^{\kappa}])$$

Obviously, $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)])$ corresponds to the private key related to the public key $[pk'_\pi]$ (since $[pk'_\pi] = \mathcal{E}([g^\kappa], [z_\pi]) = \mathcal{E}([g^\kappa], [\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)] + [h^\kappa g^{-\kappa}KP(0)]) = [\sum_{i=0}^{\kappa} g^i h^{k-i}KP(i)])$ and $[f] = \mathcal{E}([d], [x]]) = \mathcal{E}([d], [\sum_{i=1}^{\kappa} g^{i-\kappa}h^{k-i}KP(i)])$.

2. *Part II.* For each possible $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)])$ defined in Part I, consider the values

$$[r_{0_\pi}] := [\tilde{r}_0] + \mathcal{E}([c_\pi], [g^{-\kappa}KP(0)]), \quad \text{and}$$
$$[\sum_{i=1}^{\kappa} g^{\kappa-i}h^{\kappa-i}r_{i_\pi}] := [\tilde{r}_1] + \mathcal{E}([c_\pi], [\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)]),$$

It can be seen that σ can be created by the private key $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)])$ using the randomness $([r_{0_\pi}], [\sum_{i=1}^{\kappa} g^{\kappa-i}h^{\kappa-i}r_{i_\pi}])$, for any $\pi \in \{1, ..., n\}$.

3. *Part III.* The distribution of $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)], [r_{0_\pi}], [\sum_{i=1}^{\kappa} g^{\kappa-i}h^{\kappa-i}r_{i_\pi}])$ for each possible π is identical to that of a signature created by a signer with public key $[pk'_\pi]$.

In other words, the signatures σ can be created by any signer equipped with private key $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)])$ for any $\pi \in \{1, ..., n\}$ using randomness $([r_{0_\pi}], [\sum_{i=1}^{\kappa} g^{\kappa-i}h^{\kappa-i}r_{i_\pi}])$. Even if the unbounded adversary can compute $([\sum_{i=1}^{\kappa} g^{i-\kappa}h^{\kappa-i}KP(i)], [g^{-\kappa}KP(0)], [r_{0_\pi}], [\sum_{i=1}^{\kappa} g^{\kappa-i}h^{\kappa-i}r_{i_\pi}])$ for all $\pi \in [n]$, it cannot discern, amongst the n' possible unknown choices, who the signer is.

We use the fact that a public key in our construction corresponds to multiple secret keys. For each public key in the ring of possible signers, there exists a unique corresponding private key, possibly unknown, that fits the given linking tag. □

Theorem 3. *The FS-LRS scheme is linkable in the Random Oracle Model (ROM), if the (k,b)-GMDP is hard.*

Proof. If $\mathcal{A}$ can produce two valid and unlinked signatures from just one private key, we can use this successfully to break GMDP.

Given an $(1, b)$-GMDP instance $([\alpha]_b)$, $\mathcal{B}$ is asked to output some $([e]_0, [\alpha^e]_{b*e-1})$ where $e \leq l$). $\mathcal{B}$ picks $[g]_0, \in_R [\mathbb{G}]_0$ and sets $[h]_b = \mathcal{E}([\alpha]_b)$. $\mathcal{B}$ simulates the oracles thus:

- *Random Oracles* H_i: For query input H_0("GENERATOR-g"), $\mathcal{B}$ returns $[g]_0$. For query input H_b("GENERATOR-h"), $\mathcal{B}$ returns $[h]_b$. For other queries, $\mathcal{B}$ randomly picks $[\lambda]_0 \in_R [\mathbb{G}]_0$, sets $[a]_i = \mathcal{E}([\lambda]_0, [1]_i)$ and returns $[a]_i$.
- *Joining Oracle* $\mathcal{JO}$: $\mathcal{B}$ generates the public/private key pair as in the scheme. Upon the jth query, $\mathcal{B}$ returns the matching public key.
- *Corruption Oracle* $\mathcal{CO}$: On input a public key pk_i obtained from $\mathcal{JO}$, and a time t, $\mathcal{B}$ returns the private key.
- *Signing Oracle* $\mathcal{SO}$: On input a signing query for event *event*, a set of public key $\vec{pk}_t = \{[\vec{Z}_1]_b, ..., [\vec{Z}_n]_b\}$, the public key for the signer $[\vec{Z}_\pi]_b$, where $\pi \in [1, n]$, and a message M, and time t, $\mathcal{B}$ simulates as follows:

  1. If the query of $H_{l-\nu}(t||event)$ has not been made, carry out the H-query of $t||event$ as described above. Set $[d]_{l-\nu}$ to $H_{l-\nu}(t||event)$. Note that $\mathcal{B}$ knows the $[\lambda]_0$ that corresponds to $[d]_{l-\nu}$. $\mathcal{B}$ sets $[f]_l = [d * \prod_{j\in\tau} \alpha_j x_{\pi,j}]_l$, which it can compute from the challenge $[\vec{\alpha}]_b$.
  2. $\mathcal{B}$ computes the signature according to the algorithm.

If given a pair of $\sigma^i = ([f^i]_l, [\tilde{r}_0^i]_0, [\tilde{r}_1^i]_\nu, [c_1^i]_0, ..., [c_n^i]_0)$ on an event *event*, time t, two sets of public keys $\vec{pk}_{t_i}$, and two messages M_i, then, in the random-oracle model, $\mathcal{A}$ must have queried $H_{l-\nu}(t||event)$ which are denoted by $[d]_{l-\nu}$, and two queries $H_0(\vec{pk}_{t_i}||event||[f]_l||M_i||[K^i]_l||[K'^i]_l||t)$ where

$$[K^i]_l = \mathcal{E}([\tilde{r}_0^i]_0, [h^\kappa]_l) + \mathcal{E}([\tilde{r}_1^i]_\nu, [g^\kappa]_{l-\nu}) + \sum_{i=1}^{n} \mathcal{E}([pk'_{t_j}]_l, [c_j^i]_0)$$
$$[K'^i]_l = \mathcal{E}([d]_{l-\nu}, [\tilde{r}_1^i]_\nu) + \mathcal{E}([f]_l, \sum_{j=1}^{n} [c_j^i]_0)$$

Since $\sigma^1 \neq \sigma^2$ and they are unlinked, by definition of linkability, we have $[f^1]_l \neq [f^2]_l$. Since, by definition of the game, the σ^i are both valid for the same time and event, $[d^1]_{l-\nu} = H_{l-\nu}(t||event) = [d^2]_{l-\nu}$. Write $[f^i]_l$ as $[d^i x_i]_l$, where we have shown $[d^1]_{l-\nu} = [d^2]_{l-\nu}$. Hence, $[x_1]_\nu \neq [x_2]_\nu$. Therefore, at most one $[f^i]_l$, and hence σ^i, encodes the pair $(\prod_{i\in\tau} x_{\pi,i}, ..., \prod_{i\in\tau} y_{\pi,i})$ which we gave to the adversary. We extract on the unrelated signature. Therefore, we have $[\tilde{r}_0^1]_{\kappa-1} \neq [\tilde{r}_0^2]_{\kappa-1}$ and find a response $[\alpha]_{\kappa-1}$ to the GMDP challenge as:

$$[\alpha^\kappa]_\nu = [\frac{\sum_{i\in\lambda} c_i^{\delta'}(\sum_{j=1}^{\kappa} g^j h^{\kappa-j} KP(j)) - \tilde{r}_1^\delta g^\kappa}{\tilde{r}_0^\delta - \sum_{i\in\lambda} c_i^{\delta'} KP(0)}]$$

We demonstrate the correctness of the extraction, below, by showing that since $[K]$ simultaneously satisfies two equations the correctness follows by simple algebraic manipulation and the format of the keys.

$$[K] = [K]$$

We begin by substituting $[K]$ for the two equations it satisfies,

$$[\tilde{r}_0^1\alpha^\kappa] + [\tilde{r}_1^1 g^\kappa] + \sum_{i=1}^{n}[pk_i' c_i^1] = [\tilde{r}_0^2\alpha^\kappa] + [\tilde{r}_1^2 g^\kappa] + \sum_{i=1}^{n}[pk_i' c_i^2]$$

By subtraction, we have

$$[\tilde{r}_0^\delta\alpha^\kappa] + [\tilde{r}_1^\delta g^\kappa] = [\sum_{i\in\lambda} pk_i' c_i^{\delta'}]$$

By definition of pk_i',

$$[\tilde{r}_0^\delta\alpha^\kappa] + [\tilde{r}_1^\delta g^\kappa] = [\sum_{i\in\lambda} c_i^{\delta'}(\alpha^\kappa KP(0) + \sum_{j=1}^{\kappa} g^j h^{\kappa-j} KP(j))]_l$$

By subtraction, we have

$$[\tilde{r}_0^\delta\alpha^\kappa] - [\sum_{i\in\lambda} c_i^{\delta'}(\alpha^\kappa KP(0))] = [\sum_{i\in\lambda} c_i^{\delta'}(\sum_{j=1}^{\kappa} g^j h^{\kappa-j} KP(j))] - [\tilde{r}_1^\delta g^\kappa]$$

By distributivity, we have

$$[\alpha^\kappa(\tilde{r}_0^\delta - \sum_{i\in\lambda} c_i^{\delta'} KP(0))] = [\sum_{i\in\lambda} c_i^{\delta'}(\sum_{j=1}^{\kappa} g^j h^{\kappa-j} KP(j)) - \tilde{r}_1^\delta g^\kappa]$$

By division, we have

$$[\alpha^\kappa] = [\frac{\sum_{i\in\lambda} c_i^{\delta'}(\sum_{j=1}^{\kappa} g^j h^{\kappa-j} KP(j)) - \tilde{r}_1^\delta g^\kappa}{\tilde{r}_0^\delta - \sum_{i\in\lambda} c_i^{\delta'} KP(0)}]$$

By the forking lemma [7], the chance of each successful rewind simulation is at least $\xi/4$, where ξ is the probability that $\mathcal{A}$ successfully forges a signature. Hence, the probability that for a given adversary $\mathcal{A}$ we can extract $[\prod_{j\in\tau}\alpha_j]_\nu$ is $\frac{\xi}{4}\frac{1}{2}$. □

Theorem 4. *The FS-LRS is non-slanderable in the ROM, if (k,b)-GMDP is hard.*

Proof. We use the setting of Theorem 1. $\mathcal{A}$ can query any oracle other than to submit a chosen public key pk_π to $\mathcal{CO}$ or include $pk_\pi \in \vec{pk}_\pi$ to $\mathcal{SO}$. It then gives $\mathcal{B}$: the key pk_π, a list of public keys $\vec{pk}_t \ni pk_\pi$ (w.l.o.g., we have $|\vec{pk}_t| = n$), a message M, a description *event*, and a time t. In return, $\mathcal{B}$ generates a signature $\sigma([f]_l, ...)$ using the standard method for the signing oracle, and gives it back to $\mathcal{A}$. Recall, we let $[f]_l = [d * \prod_{j\in\tau} a_j x_{\pi,j}]_l$. $\mathcal{A}$ continues to query various oracles, expect that it is not allowed to submit pk_π to $\mathcal{CO}$.

Suppose $\mathcal{A}$ produces another valid signature $\sigma^* = ([f']_l, .)$ that was not an output from $\mathcal{SO}$ but is linkable to σ. Since they are linkable, we have $[f']_l = [f]_l$ and hence $\frac{[\prod_{i\in\tau} a_i x_{\pi,i}]_l}{[d]_0} = \frac{[\prod_{i\in\tau} a_i x_{\pi,i}]_l}{[d]_0}$. Recall that, by definition of the game, $\sigma^* \neq \sigma^{'}$ which implies that $[\tilde{r}_i^*] \neq [\tilde{r}_i']$ and hence $[r_i^*] \neq [r_i^{'}]$. We then extract $[\prod_{j\in\tau} \alpha_j]_\nu$ from σ^* as outlined in Theorem 1.

The probability that, for a given adversary $\mathcal{A}$, we can extract $[\prod_{j\in\tau} \alpha_j]_\nu$ is $\frac{\xi}{4n}$. □

Theorem 5. *The FS-LRS scheme is forward-secure anonymous in the random-oracle model, if MDDH is hard.*

Proof. We show that the ability of the adversary to make corruption queries at times later than t does not allow it to de-anonymise signatures at time t or earlier, without breaking (κ,b)-MDDH, and hence the system achieves forward-secure anonymity. In this proof, we start by guessing the break point t at which the adversary's will choose to be challenged.

Given an MDDH instance $([\vec{e}]_0, [\vec{\alpha}]_{\nu/\kappa}, [\beta]_{l-\nu}, [\gamma]_l)$, $\mathcal{B}$ is asked to decide whether $[\gamma]_l = [\prod_{i\in k} \alpha_i^{e_i} \beta]_l$. Where $\vec{e}$ produces the same selection of keys as $\tau = PK(t)$. $\mathcal{B}$ picks $[g]_0, [h]_0 \in_R [\mathbb{G}]_0$ and sets $[h]_b = \mathcal{E}([h]_0, [1]_b)$. $\mathcal{B}$ simulates:

- *Random Oracles* H_i: For query input H_0("GENERATOR-g"), $\mathcal{B}$ returns $[g]_0$. For query input H_b("GENERATOR-h"), $\mathcal{B}$ returns $[h]_b$. For other queries, $\mathcal{B}$ randomly picks $[\lambda]_0 \in_R [\mathbb{G}]_0$, sets $[a]_i = \mathcal{E}([\lambda]_0, [1]_i)$ and returns $[a]_i$.
- *Joining Oracle* $\mathcal{JO}$: Assume $\mathcal{A}$ can only query $\mathcal{JO}$ for a maximum n' times, where $n' \geq n$. W.l.o.g., $(1, ..., n)$ will be the indices for which $\mathcal{B}$ does not know the private keys and embeds the challenge, and $(n+1, ..., n')$ be the indices for which a private key is known. For the first n indices, $\mathcal{B}$ chooses $[\vec{x}]_0, [\vec{y}]_0 \in_R [\mathbb{G}^k]_0$ and sets $[\vec{Z}]_b = ([Z_1]_b, ..., [Z_k]_b)$ where $[Z_i]_b = [g\alpha_i x_i]_b + [hy_i]_b$. For the remaining indices it generates the public/private key pair as in the scheme. Upon the jth query, $\mathcal{B}$ returns the matching public key.
- *Corruption Oracle* $\mathcal{CO}$: On input a public key pk_i obtained from $\mathcal{JO}$, and a time t, $\mathcal{B}$ checks whether it is corresponding to $[n+1, n']$, if yes, then $\mathcal{B}$ returns the private key. Otherwise, $\mathcal{B}$ calls $\mathcal{O}(\tau, \nu)$ returns $sk_i = ([\prod_{j\in\tau} x_{i,j}\alpha_j]_\nu, ..., [\prod_{j\in\tau} y_{i,j}]_0)$.
- *Signing Oracle* $\mathcal{SO}$: On input a signing query for event *event*, a set of public keys $\vec{pk}_t = \{[Z_1]_l, ..., [Z_n]_l\}$, the public key for the signer $[Z_\pi]_l$ where $\pi \in [1, n]$, a message M, and a time t, $\mathcal{B}$ simulates as follows:

  1. If the query of $H_{l-\nu}(t||event)$ has not been made, carry out the H-query of $t||event$ as described above. Set $[d]_{l-\nu}$ to $H_{l-\nu}(t||event)$. Note that $\mathcal{B}$ knows the $[\lambda]_0$ that corresponds to $[d]_{l-\nu}$.
  2. $\mathcal{B}$ randomly chooses $[\tilde{r}_0]_0 \in_R [\mathbb{G}]_0$ and $[\tilde{r}_1]_\nu \in_R [\mathbb{G}]_\nu$ and $[c_i]_0 \in_R [\mathbb{G}]_0$ for all $i \in [1, n]$ and sets the H_0 oracle output of

$$H_0\Big(\vec{pk}_t||event||[f]_l||M||\mathcal{E}([\tilde{r}_0]_0, [h^\kappa]_l) + \mathcal{E}([\tilde{r}_1]_\nu, [g^\kappa]_{l-\nu}) + \sum_{i=1}^n \mathcal{E}([pk_i']_l, [c_i]_0)||\mathcal{E}([d]_{l-\nu}, [\tilde{r}_1]_\nu) + \mathcal{E}([f]_l, \textstyle\sum_{i=1}^n [c_i]_0)||t\Big)$$

  3. $\mathcal{B}$ returns the signature $\sigma = ([f]_l, [\tilde{r}_0]_0, [\tilde{r}_1]_\nu, [c_1]_0, ..., [c_n]_0)$. $\mathcal{A}$ cannot distinguish between $\mathcal{B}$'s simulation and real life.

At some point, $\mathcal{A}$ requests to be challenged on $e, t, n, \vec{pk}_t, M$. $\mathcal{B}$ sets $H_{l-\nu}(e||t) = [\beta]_{l-\nu}$, samples $i \in [n]$, sets $[f]_l = [\gamma \prod_{i\in\tau} x_i' + \beta \sum_{i=1}^{\kappa-1} g^{i-\kappa} h^{\kappa-i} KP(i)]_l$, and then performs the remaining steps of the signing oracle as above. Notice that if $[\gamma]_l$ is equal to $[\prod_{i=1}^{\kappa} a_i^{e_i} \beta]_l$ then this signature is normally

formed; however, if $[\gamma]_l$ is a random group element than the linking element is random, while the rest of the signature is independent of the signer. If $\mathcal{A}$ successfully guesses i then $\mathcal{B}$ guesses that $[\gamma] = [\alpha\beta]$, otherwise $\mathcal{B}$ guesses that $[\gamma]$ is random. □

7. Conclusions

We have presented the first linkable ring signature scheme with both unconditional anonymity and forward-secure key update. By expanding upon of our work in [3], we have shown how a combinatorial trick can allow bilinear pairings to synthesise a polynomial time granularity for forward security. This is a powerful tool which has direct applications in elegantly addressing a number of simultaneous constraints in remote electronic voting. We have also presented a comprehensive security model which better reflects real requirements than existing definitions. We then proved our construction secure under these definitions by reducing to natural bilinear or multilinear generalisations of the computational and decisional Diffie–Hellman Problems.

Author Contributions: Conceptualization, X.B. and T.H.; Formal Analysis, X.B. and T.H.; Writing, X.B. and T.H.

Funding: Xavier Boyen is a Future Fellow of the Australian Research Council, under ARC grant FT140101145.

Conflicts of Interest: The authors declare no conflict of interest.

List of Symbols

KP $(i, \tau, \vec{x}, \vec{y}) = \sum_{j \in \binom{\tau}{i}} \sum_{j' \in \binom{\tau/j}{k-i}} (\prod_{\iota \in j} x_\iota \prod_{\iota \in j'} y_\iota)$. We sometimes just write $KP(i)$, when $\tau, \vec{x}, \vec{y}$ are clear from context.

PK The function which when given a time period t returns the set of indices needed to generate the public key for that time period.

κ The cardinality of τ.

$\mathcal{CO}$ The *Corruption Oracle*, on input a previously joined public key pk_i, returns the matching secret key sk_i at the time t.

$\mathcal{E}$ An $l-$linear map over additive cyclic groups $[\mathbb{G}]_1, .., [\mathbb{G}]_l$ of prime order q.

$\mathcal{HO}$ An Helper Oracle—Used in the definition of k-SMDP, (k,b)-GMDP, k-SMDDH and (k,b)-GMDDH.

$\mathcal{H}$ The *Random Oracle*, on input x, returns h independently and uniformly at random. If an x is repeated, the same h will be returned again.

$\mathcal{JO}$ The *Joining Oracle*, upon request, adds a new user to the system, and returns the public key pk of the new user at the time t.

$\mathcal{O}$ Landau's symbol—An asymptoptic bound on the function's growth.

$\mathcal{SO}$ The *Signing Oracle*, on input an event-id *event*, a group size n, a set $\vec{pk}_t$ of n public keys, a set of public keys of possible signers $\vec{pk}_\Pi \subset \vec{pk}_t$, a message M, and a time t, returns a valid signature σ'.

$\mathcal{T}$ The number of time periods.

ν The level at which the genuine signer should know the secret key, $\kappa + (\mathcal{T} \bmod b) - 1$.

τ The set of indices used to generate a public key for a given time period.

b The initial public key level.

k The combinatorial constant.

l The multilinear map size.

t The current time period.

References

1. Liu, J.K.; Au, M.H.; Susilo, W.; Zhou, J. Linkable ring signature with unconditional anonymity. *IEEE Trans. Knowl. Data Eng.* **2014**, *26*, 157–165.
2. Fujisaki, E.; Suzuki, K. Traceable ring signature. In Proceedings of the 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, 16–20 April 2007; pp. 181–200.
3. Boyen, X.; Haines, T. Forward-Secure Linkable Ring Signatures. In *Australasian Conference on Information Security and Privacy* Springer: Cham, Switzerland, 2018; pp. 245–264.

4. Garg, S.; Gentry, C.; Halevi, S. Candidate multilinear maps from ideal lattices. In Proceedings of the 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, 26–30 May 20132; pp. 1–17.
5. Langlois, A.; Stehlé, D.; Steinfeld, R. GGHlite: More efficient multilinear maps from ideal lattices. In Proceedings of the 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, 11–15 May 2014; pp. 239–256.
6. Liu, J.K.; Wei, V.K.; Wong, D.S. Linkable spontaneous anonymous group signature for ad hoc groups. In *Information Security and Privacy*; Springer: Berlin, Germany, 2004.
7. Pointcheval, D.; Stern, J. Security proofs for signature schemes. In *Eurocrypt*; Springer: Berlin, Germany, 1996; Volume 96, pp. 387–398.
8. Pedersen, T.P. Non-interactive and information-theoretic secure verifiable secret sharing. In Proceedings of the Annual International Cryptology Conference (CRYPTO '91), Santa Barbara, CA, USA, 11–15 August 1991.
9. Chaum, D.; van Heyst, E. Group signatures. In *Lecture Notes in Computer Science*; Davies, D.W., Ed.; Springer: Berlin, Germany, 1991; Volume 547, pp. 257–265.
10. Rivest, R.L.; Shamir, A.; Tauman, Y. How to leak a secret. In Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, 9–13 December 2001.
11. Boneh, D.; Silverberg, A. Applications of multilinear forms to cryptography. *Contemp. Math.* **2003**, *324*, 71–90.
12. Boneh, D.; Wu, D.J.; Zimmerman, J. Immunizing multilinear maps against zeroizing attacks. *IACR Cryptol. ePrint Arch.* **2014**, *2014*, 930.
13. Cheon, J.H.; Han, K.; Lee, C.; Ryu, H.; Stehlé, D. Cryptanalysis of the multilinear map over the integers. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015; pp. 3–12.
14. Coron, J.S.; Lepoint, T.; Tibouchi, M. Practical multilinear maps over the integers. In *Advances in Cryptology—CRYPTO*; Springer: Berlin, Germany, 2013; pp. 476–493.
15. Gentry, C.; Gorbunov, S.; Halevi, S. Graph-induced multilinear maps from lattices. In *Theory of Cryptography*; Springer: Berlin, Germany, 2015; pp. 498–527.
16. Adida, B. Helios: Web-based open-audit voting. In Proceedings of the USENIX Security, San Jose, CA, USA, 28 July–1 August 2008.
17. Demirel, D.; Van De Graaf, J.; Araújo, R. Improving helios with everlasting privacy towards the public. In Proceedings of the eVOTE/Trustworthy Elections (USENIX), Bellevue, WA, USA, 6–7 August 2012.
18. Tsoukalas, G.; Papadimitriou, K.; Louridas, P.; Tsanakas, P. From helios to zeus. *USENIX J. Elect. Technol. Syst.* **2013**, 1, 1–17.
19. Adida, B. *Helios v3 Verification Specs*; Technical Report; Helios Voting: Boston, MA, USA, 2010
20. Zhandry, M. Adaptively secure broadcast encryption with small system parameters. *IACR Cryptol. ePrint Arch.* **2014**, *2014*, 757.
21. Hu, Y.; Jia, H. Cryptanalysis of GGH map. In Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, 8–12 May 2016; pp. 537–565.
22. Albrecht, M.R.; Bai, S.; Ducas, L. A subfield lattice attack on overstretched NTRU assumptions. In Proceedings of the 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2016.
23. Cheon, J.H.; Jeong, J.; Lee, C. An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low level encoding of zero. *LMS J. Comput. Math.* **2016**, *19*, 255–266.
24. Cramer, R.; Ducas, L.; Peikert, C.; Regev, O. Recovering short generators of principal ideals in cyclotomic rings. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, 8–12 May 2016.
25. Miles, E.; Sahai, A.; Zhandry, M. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Proceedings of the 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2016.

26. Paneth, O.; Sahai, A. On the equivalence of obfuscation and multilinear maps. *IACR Cryptol. ePrint Arch.* **2015**, *2015*, 791.

27. Barak, B.; Goldreich, O.; Impagliazzo, R.; Rudich, S.; Sahai, A.; Vadhan, S.P.; Yang, K. On the (im)possibility of obfuscating programs. *J. ACM* **2012**, *59*, 6.