Frost	CLSAG	Use of Symbol
$\mathbb{G}$	$\mathtt{G}$	Denotes a group
$\mathtt{p_i}$	$\mathtt{pk_i}$	Denotes some public key
$\mathtt{s}$	$\mathtt{sk}$	Used to denote secret key
$\mathtt{s_i}$	$\mathtt{sk}$	Denotes the i^{th} secret share, assigned to the P_i^{th} participant
$\mathtt{t}$	-	Number of Threshold Participants
$\mathtt{n}$	-	Number of Total Participants
$\mathtt{P_i}$	-	Denotes the i^{th} participant
$\overrightarrow{C}$	-	Public Commitment of the form $\langle \phi_0, \phi_1, ..., \phi_{t-1} \rangle$
$\mathtt{SA}$	$licshares\mathtt{Sign}$	Signature Aggregator
$\mathtt{Y_i}$	-	Public Key Shares assigned to P_i
-	CO	Denotes the corruption oracle.
ρ	ρ_i	Queries to an oracle

Table 1: Notation employed for cryptographic components.

It should be noted that, when reading both the Frost and clsag papers, if the variables or notation denoted below is not found in the table, then it likely stands for a generic value which is defined immediately within the same sentence. These will not be included in the Stone below.

1. It should be noted that oracle queries of all types may or may not possess a number of parameters specified in the same line. Their 'type' is also specified, albeit in a sort of non-obvious way. Regardless of the inconsistency, ρ can be safely interpreted as a vector of some length, with ρ_i or $[\rho]_i$ denoting the i^{th} entry, or the i^{th} query to the oracle. Both are equivalent under the vector interpretation.

2. It should be noted that random oracles and Hash functions seem to have extremely similar notation, so caution should be warranted. In theoretical proofs, it seems to matter little, but universally, H does not necessarily mean a hash function.

3. The notation is fairly distinct between the two papers, with the exceptions noted above. The main difference seems to be the bulk of the notation used in the proofs, particularly in paramaterization of quantities with numerous parameters. (If it is helpful, I can parse through and find the notation with numerous parameters, and detail the nature of the parameters throughout the two papers. This would be more or less just a little cheat sheet that may help with reading.)

1 Notes for the Frost Paper

1. In Frost KeyGen, the notation is very consistent. The only clarifying factor would be that the subscripts that are indexed as a_{ij} should have some separating symbol, like $a_{i,j}$.
2. In round 2, modulus operator in the exponent should be contained in round braces. The comment from above also holds for the product indexing.
3. In the signing protocol on page 13, it looks like set theory notation mixed in with casual notation. Recommend replacing the ':' with the verbal equivalent of 'such that' or encasing in brackets. Other instances occur, so it is recommended to fix those as well.
4. On page 15, the summation in 7.c is not indexed. Shoddy notation.
5. I think that the proof for Theorem 6.1 can be streamlined a bit.
6. Lemma A.1, needs left(, right). The entire paper is beautifully written. It would benefit from some elaboration throughout and an adjustment in formatting to increase readability. Beside the comments above, it is a shining example of a well written paper. It might be worth leaving proof sketches out of the upper section and just moving the whole proof up.

2 Notes on the original clsag Paper

1. I would recommend a slight change in the notation in the preliminaries. Just cleaning up the notation.
2. 3.1, citation needed. Not sure where to find this result. I believe it, but I would like a paper.
3. The linkable ring signature definition is a beast and a half.
4. Typo on page 8, 'we' shows up twice in the first paragraph.

3 Linkability and Forgeability Definitions

3.1 Linkability

1. A careful thing to note, appearing in the clsag definition of *pigeonhole linkability*, is that the adversary does not require key generation, corruption or signature oracle access. It is the case that the adversary

behaves exactly the same with or without them? Where does this independence come from?

2. ACST linkability allows certain oracle queries, under some constraints. My immediate question is: does the adversary from pigeonhole linkability possess the ability to retrieve the same information as the in ACST without an oracle? Presummably, if the adversary from the pigeonhole linkability does not need them, but does this imply that he has them and chooses not to use them? Would it not make for a more powerful attacker potentially, giving oracle access regardless? I think clarification should be made regardless, as the ambiguity leads to two attackers, distinct in their capability, but equal in their attack power.

3. There is a claim in Definition 6 that ACST linkability and Pigeon hole linkability are not the same, due to differing success conditions. However, is the overlap frequent enough that they only differ in a small number of cases? (Cases that will never see practical use?)

4. A common theme I see in the papers I’ve read is the use of common venacular in one subfield placed into another unfamiliar field. This can lead to confusion during an initial read through and needlessly obfuscates the process. For example, in point (30 on page 17, it is stated that M picks two random tapes, **h** and **h’** to simulate an oracle response. But are these not just vectors of some length with entries in a familiar field? The word tape could be simply elaborated on. A change as simple as, ’let h and h' be vectors of length n, in field f, which are used as tapes to simulate oracle responses.’ This sort of proof modification would benefit, I think, readers from all familiar backgrounds.

4 Linkability: Definitions and Results

Here I will include definitions of linkability that I find, and hopefully use it as a sort of handbook to streamline related proofs. In the original clsag paper, there are two definitions of linkability which are presented: ACST linkability and Pigeonhole linkability. It is not immediately clear what exactly the connection between the two are, but I suppose that is the point of this section.

To give a brief summary, the definitions of linkability might lend themselves well to a certain intuitive understanding, like evenness or oddness, but the variations are less so. So I think it is prudent to pull the definitions apart a little bit, over explain what is going on and then show how one might prove linkability in a concise way. It seems that each type of linkability requires different initial hypotheses, but seem to have similar proof styles, with slight variation. Now, let us inspect these and continue forward.

4.1 bg-Linkability

Brandon proposed a new definition for linkability which I am going to attempt to formalize here. If his intuition is correct, then it can reduce down to both ACST and Pigeonhole linkability. How he stated it in his message:

1. With Oracle access, output k+1 unlinked ring signatures, such that:

$$\left|\left(\bigcup_i \mathtt{ring}_i\right) \setminus \mathtt{U}\right| = k$$

2. We have to be careful with the setup, since it should contain the same components as the other two definitions of linkability, but appropriately so. The unfortunate thing here is that the game style setup will be unavoidable.

3. As such, the new definition should mirror the other two, carefully so, in such a way that specific subcases of the new definition reduce down two the definitions of linkability.

So, let us start with pigeonhole linkability, which asserts the following:

Definition 4.1 *q-Pigeonhole Linkability*

We say that any PPT algorithm A that can succeed at outputting q public keys $\{pk_i\}_{i=0}^{q-1}$ *and* $q+1$ *valid unlinked signatures triples* $\{(m_j, Q_j, \sigma_j)\}_{j=0}^{q}$*, such that:*

$$\bigcup_j Q_j \subseteq \{pk_i\}_{i=0}^{q-1}$$

in time, at most, t, and the probability of at least ϵ *is a* (t, ϵ, q)*-solver of pigeonhole linkability.*

*We say that a scheme is q-**pigeonhole linkable** if every PPT algorithm A that is a* (t, ϵ, q)*-solver of q pigeonhole linkability has a negligible acceptance probability* ϵ*.*

Note that the adversary is not granted signing or corruption oracle access here.

Definition 4.2 *ASCT Linkability*

A linkable ring signature is linkable if for any PPT adversary A, and for any polyomial $n(\cdot)$*, the probability that A succeeds in the following game is negligible"*

1. *(Initialization Phase) Key pairs* $\{(PK_i, SK_i)_{i=1}^{n(k)}\}$ *are generated by executing* $KeyGen(1^k)$*, and the set of public key* $S = \{PK_i\}_{i=1}^{n(k)}$ *is given to A.*

2. *(Probing Phase) A is given access to a signing oracle* $SO(\cdot,\cdot,\cdot)$*, where* $SO(s, M, R)$ *outputs* $Sign_{s,SK_s}(M, R)$ *and we require that* $R \subseteq S$ *and* $PK_s \in R$*. A is also given access to a corrupt oracle* $CO(\cdot)$*, where* $CO(i)$ *outputs* SK_i*.*

3. *(Output Phase) A outputs* (M^*, σ^*, R^*) *and succeeds if* $Verify_{R^*}(M^*, \sigma^*) = 1$*, A never queried* $(\cdot, M^*, R^*)$ *to its signing oracle, and* $R^* \subseteq S \setminus C$*, where C is the set of corrupted users.*

Definition 4.3 *Broad-Linkability*

We will begin this definition by parameterizing the adversarial capability:

Let us define a PPT adversary $\mathcal{A}$ with the following capacity:

1. *$\mathcal{A}$ has access to a set of oracles Θ.*
2. *For each oracle θ in Θ, $\mathcal{A}$ can query θ up to $Q(\theta)$ times.*
3. *$\mathcal{A}$ has access to an initial set of information, I.*

The adversary $\mathcal{A}$ is said to win the linkability game $G_\ell(N)$, given that they are able to accomplish one of the following with non-negligible probability:

1. *If $\mathcal{A}$ is able to produce N unlinked signatures.*
2. *If $\mathcal{A}$ is able to output a valid signature.*

This definition is extremely broad, but it covers each case of without difficulty. We can see that by restricting the capacity of $\mathcal{A}$ in a particular way, we are able to produce exactly the adversary in each sub-definition.

This feels like a far more natural way to define things, since we may impose the conditions of linkability more naturally on the adversary without having to adjust the win conditions of a game. Notably, we can compress the notation for an adversary, defining an arbitrary adversary as $\mathcal{A}(\Theta, I)$. Then, simply by invoking the initial setup in each definition of linkability to fit each sub-definition, giving us adversaries $\mathcal{A}_{ASCT}$ and $\mathcal{A}_p$. Thus, we have that if Broad Linkability is satisfied with a $\mathcal{A}_{ASCT}$ adversary, then it is ASCT linkable, with the same conditions for pigeon-hole linkability.

While the bulk of this definition is contained in notation, it does provide a bit more structure for further definitions to adhere to, so I think it is worth sharpening and further inspecting. Also, it should be noted that the query amount Q can be contained in the information set I, without adding further parameters to the adversary.

Furthermore, this definition can have the game changed, as the win conditions are contained in $G_\ell(N)$. So, the game of linkability can be easily substituted with the game of forgability without altering the core structure of the definition.

5 Unforgability: Definitions and Results

Let us start with the definition found in the paper, Short Linkable Ring Signatures, given as follows:

A linkable ring signature scheme is unforgeable if for any PPT adversary A and for any polynomial $n(\cdot)$, the probability that A succeeds in the following game is negligible:

1. (Initialization Phase) Key pairs $\{(PK_i, SK_i)_{i=1}^{n(k)}\}$ are generated by executing $KeyGen(1^k)$, and the set of public key $S = \{PK_i\}_{i=1}^{n(k)}$ is given to A.

2. (Probing Phase) A is given access to a signing oracle $SO(\cdot,\cdot,\cdot)$, where $SO(s, M, R)$ outputs $Sign_{s,SK_s}(M, R)$ and we require that $R \subseteq S$ and $PK_s \in R$. A is also given access to a corrupt oracle $CO(\cdot)$, where $CO(i)$ outputs SK_i.

3. (Output Phase) A outputs (M^*, σ^*, R^*), and succeeds if $Verify_{R^*}(M^*, \sigma^*) = 1$, A never queried $(\cdot, M^*, R^*)$ to its signing oracle and $R^* \subseteq S\ C$, where C is a set of corrupted users.

Now, the first problem that I have with this definition is that the 'game' style format, to me obfuscates alot of whats going on. It certainly lends itself to better intuitive understanding, but it seems it would force proofs to be trapped in the 'game' format, as opposed to being proper mathematical proofs. It seems that the reason for this lies in the fact that the overall definition can seem unwieldy unless written in the game format. However, if this definition is industry standard, then there may be no need to modify it except to scratch my notational itch.

However, it may be beneficial to re-express the definition in a more readable format overall.