Overview

Public parameters are of the form

$$g_1^{\alpha}, g_1^{\alpha^2}, \ldots, g_1^{\alpha^N}, g_1^{\alpha^{N+2}}, g_1^{\alpha^{N+3}}, \ldots, g_1^{\alpha^{2N}}$$
$$g_2^{\alpha}, g_2^{\alpha^2}, \ldots, g_2^{\alpha^N}, g_2^{\alpha^{N+2}}, g_2^{\alpha^{N+3}}, \ldots, g_2^{\alpha^{2N}}$$

where α is a uniform nonzero scalar unknown to the adversary. Note that the $(N+1)$ power is omitted.

Consistency Check

Given an alleged set of public parameters $\{\overline{g_1{}^{\alpha}}, \ldots; \overline{g_2^{\alpha}}, \ldots\}$, one can check that it is of the correct form. This is described in a separate document.

Parameter Generation MPC

Notation

For convenience, let Powers(g_1^{α}) be shorthand for

$$g_1^{\alpha}, g_1^{\alpha^2}, \ldots, g_1^{\alpha^N}, g_1^{\alpha^{N+2}}, g_1^{\alpha^{N+3}}, \ldots, g_1^{\alpha^{2N}}, g_2^{\alpha}, g_2^{\alpha^2}, \ldots, g_2^{\alpha^N}, g_2^{\alpha^{N+2}}, g_2^{\alpha^{N+3}}, \ldots, g_2^{\alpha^{2N}}$$

and let Powers$(g_1^{\alpha})^{\beta}$ denote Powers$(g_1^{\alpha\beta})$. Note that Powers$(g_1^{\alpha})^{\beta}$ can be computed from Powers(g_1^{α}) and β without knowing α by raising each element of Powers(g_1^{α}) to the corresponding power of β.

Given a vector of $2N-1$ elements of $\mathbb{G}_1$ and $2N-1$ elements of $\mathbb{G}_2$, the consistency check in the previous section allows verifying that there exists a nonzero α for which these elements are Powers(g_1^{α}). This consistency check is described in a separate document.

Let ZKPoK be a zero knowledge proof of knowledge of exponent scheme where proofs are bound to an arbitrary string (an identity) such that seeing a simulated proof for identity I does not affect soundness of proofs for any identity other than I. Write ZKPoK(x, I) for a zero knowledge proof of knowledge of exponent of g_1^x bound to identity I. In the random oracle model, ZKPoK(x, I) can be a Schnorr signature of message I using private key x.

Protocol

This protocol is similar to the ZCash "Powers of Tau" protocol.

Let there be k parties $P_1, \ldots, P_k$, each with distinct identity $\mathrm{id}_1, \ldots \mathrm{id}_k$. Let x_0 be some public fixed nonzero scalar known in advance (e.g., 2). The first party picks a random scalar δ_1, broadcasts Powers$((g_1^{x_0})^{\delta_1})$ along with $g_1^{\delta_1}$ and ZKPoK$(\delta_1; \mathrm{id}_1)$, and erases δ_1. All parties check the consistency of Powers$(g_1^{x_0\delta_1})$, check that $e(g_1^{\delta_1}, g_2^{x_0}) = e(g_1^{x_0\delta_1}, g_2)$, and check the ZKPoK and abort on failure.

The second party picks a random scalar δ_2, broadcasts $\text{Powers}(g_1^{x_0\delta_1})^{\delta_2}$ along with $g_1^{\delta_2}$ and $\text{ZKPoK}(\delta_2; \text{id}_2)$, and erases δ_2.

All parties check the consistency of $\text{Powers}(g_1^{x_0\delta_1\delta_2})$, check that $e(g_1^{\delta_2}, g_2^{x_0\delta_1}) = e(g_1^{x_0\delta_1\delta_2}, g_2)$, and check that the ZKPoK is a valid proof of knowledge of exponent for $g_1^{\delta_2}$.

In general, if party $i-1$ broadcast $\text{Powers}(g_1^{x_{i-1}})$, then party i will broadcast $\text{Powers}(g_1^{x_{i-1}})^{\delta_i}$ along with $g_1^{\delta_i}$ and $\text{ZKPoK}(\delta_i; \text{id}_i)$. All parties will check the consistency of $\text{Powers}(g_1^{x_{i-1}})^{\delta_i}$, check that $e(g_1^{\delta_i}, g_2^{x_{i-1}}) = e(g_1^{x_{i-1}\delta_i}, g_2)$, and check that the ZKPoK is a valid proof of knowledge of exponent for $g_1^{\delta_i}$.

Finally, after the last party has participated (sending $\text{Powers}(g_1^{x_k})$) and all parties have done the corresponding consistency / pairing / PoK checks, a scalar β is sampled from a public random beacon and the final output parameters are $\text{Powers}(g_1^{x_k})^{\beta}$.

Proof of security

Claim 1. *There exists a simulator* Sim *such that for all PPT adversaries $\mathcal{A}$ corrupting all but one party, assuming $\mathcal{A}$ does not cause an abort,*

$$\{(\mathcal{A}\textit{'s view during real execution, Real output params})\} \approx \left\{\alpha \leftarrow \$ \; : \; \left(\text{Sim}^{\mathcal{A}}(\text{Powers}(g_1^{\alpha})), \text{Powers}(g_1^{\alpha})\right)\right\}$$

Proof. Let P_i be the (sole) honest party. The simulator has been given $\text{Powers}(g_1^{\alpha})$ for some uniformly sampled nonzero α but has not been given α. The simulator gets to choose P_i's message and the random beacon value, it gets black-box access to the adversary, and it can (by programming a random oracle or setting a CRS) simulate valid proofs for the ZKPoK scheme. The goal of the simulator is to choose P_i's message and a random beacon value so as to produce valid transcripts giving $\text{Powers}(g_1^{\alpha})$ as the output parameters, such that the distribution of simulated transcripts is indistinguishable from that of real transcripts generated by running the protocol with the same adversary and an honest P_i (conditioned on having $\text{Powers}(g_1^{\alpha})$ as output).

The simulator first does the following:

- Runs $\mathcal{A}.P_1$ to get $\overline{\text{Powers}(g_1^{x_1})}, \overline{\text{ZKPoK}(\delta_1; \text{id}_1)}, \overline{g_1^{\delta_1}}$.
- Rewinds to extract δ_1 from the ZKPoK
- ...
- Runs $\mathcal{A}.P_{i-1}$ to get $\overline{\text{Powers}(g_1^{x_{i-1}})}, \overline{\text{ZKPoK}(\delta_{i-1}; \text{id}_{i-1})}, \overline{g_1^{\delta_{i-1}}}$.
- Rewinds to extract δ_{i-1} from the ZKPoK

Note that an honest P_i would abort unless all consistency checks pass, so since we assume $\mathcal{A}$ does not cause an abort, each $\overline{\text{Powers}(g^{x_j})}$ is in fact $\text{Powers}(g^{x_j})$ (for some x_j). Furthermore an honest P_i would abort if any of the proofs of knowledge of exponent for $\overline{g_1^{\delta_j}}$ fail, so by soundness of the ZKPoK scheme the extracted δ_j is the correct exponent in $\overline{g_1^{\delta_j}}$. Finally, since honest

P_i would check that $e(g_1^{\delta_j}, g_2^{x_{j-1}}) = e(g_1^{x_j}, g_2)$ for all $1 \leq j < i$, we know that $x_{i-1} = x_0\delta_1 \ldots \delta_{i-1}$. In particular, the simulator knows x_{i-1}, the exponent in the set of powers sent by P_{i-1}.

The simulator now picks a random nonzero scalar r. The simulated P_i now broadcasts

$$\mathrm{Powers}(g_1^{\alpha})^{rx_{i-1}}\,,\ (g_1^{\alpha})^r, \mathrm{ZKSim}((g_1^{\alpha})^r; \mathrm{id}_i)$$

where ZKSim is the ZK simulator for the ZKPoK scheme.

The simulator now runs the remaining $\mathcal{A}.P_{i+1}, \ldots, \mathcal{A}.P_k$ and extracts $\delta_{i+1}, \ldots, \delta_k$ as before. Note that P_i's simulated ZKPoK is tied to id_i and so giving it to the adversary does not impact soundness of the remaining adversary-produced ZKPoK, which are tied to $\mathrm{id}_j \neq \mathrm{id}_i$.

As before, the consistency checks ensure that the last party's message is $\mathrm{Powers}(g_1^{\alpha r x_{i-1}\Delta})$, where $\Delta = \delta_{i+1} \ldots \delta_k$. Finally the simulator programs the random beacon value to be $\hat{\beta} = \frac{1}{rx_{i-1}\Delta}$. Now the final output is $\mathrm{Powers}(g_1^{\alpha r x_{i-1}\Delta})^{\hat{\beta}} = \mathrm{Powers}(g_1^{\alpha})$.

We now compare the real view of the adversary to a view simulated by this simulator. All players' messages before P_i are sampled identically (by running the adversary) in the two views.

In the real view, P_i's message is sampled as follows:

$$\left\{\delta_i \leftarrow \$, \mathrm{Powers}(g^{x_{i-1}}) \leftarrow \mathcal{A}.P_{i-1}\ :\ (\mathrm{Powers}(g^{x_{i-1}})^{\delta_i},\ g_1^{\delta_i},\ \mathrm{ZKPoK}(\delta_i; \mathrm{id}_i))\right\}$$

In the simulated view, P_i's message is sampled as follows, where r is an already-fixed nonzero scalar:

$$\left\{\overline{\mathrm{Powers}(g^{x_{i-1}})} \leftarrow \mathcal{A}.P_{i-1}\ :\ (\mathrm{Powers}(g^{\alpha})^{rx_{i-1}},\ g_1^{\alpha r},\ \mathrm{ZKSim}(g_1^{\alpha r}; \mathrm{id}_i))\right\}$$

α is uniform, nonzero, and independent of x_{i-1}, so in both the real and simulated views, P_i's message is distributed as

$$\left\{\begin{array}{c}\mathrm{Powers}(g^u) \text{ for uniform nonzero } u \text{ independent of } x_{i-1},\\ \text{the unique } g_1^z \text{ such that } e(g_1^z, g_2^{x_{i-1}}) = e(g_1^u, g_2),\\ \text{a ZKPoK of exponent of } g_1^z \text{ using identity } \mathrm{id}_i\end{array}\right\}$$

noting that the real and simulated ZKPoKs are indistinguishable by the zero-knowledge property of the ZKPoK scheme. Note also that the distribution of P_i's message is independent of r.

All messages up to and including P_i's are indistinguishable in the two views, so the remaining parties' messages (which are in both views generated by running the adversary on all messages so far) are indistinguishable as well.

Finally, the real random beacon value β is a uniform nonzero scalar, and the simulated beacon value is

$$\hat{\beta} = \frac{1}{rx_{i-1}\Delta}$$

where Δ is extracted from the adversary's ZKPoKs after P_i. r is uniform nonzero and independent of x_{i-1} and Δ (since P_i's message is independent of r). Therefore $\hat{\beta}$ is uniform nonzero.

The output parameters in the simulated view are uniquely determined by the messages and the random beacon. Thus the real and simulated views are identical. $\square$