Revocation with type-3 pairings

Dmitry Khovratovich

5 May 2017, version 0.3

1 Setup

This section describes how the Prover obtains a non-revocation claim from the Issuer.

1.1 Common parameters for all Issuers

Issuer and Prover mutually trust each other in submitting values of the right format during credential's issuance. This trust can be eliminated at the cost of some extra steps.

Common parameters:

- Groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order q;
- Type-3 pairing operation $e: \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$.
- Generators: g for $\mathbb{G}_1$, g' for $\mathbb{G}_2$.

1.2 Issuer revocation setup

Issuer makes the following steps:

1. Generate random $h, h_0, h_1, h_2, \widetilde{h} \in \mathbb{G}_1$;
2. Generate random $u, \widehat{h} \in \mathbb{G}_2$;
3. Generate random sk, x smaller than q.
4. Compute
$$pk \leftarrow g^{sk}; \qquad\qquad y \leftarrow \widehat{h}^x.$$
.
5. The issuer revocation public key is $pk^R = (h, h_1, h_2, \widetilde{h}, \widehat{h}, u, pk, y)$ and the secret key is (x, sk).

The Issuer fixes the number L of credentials per accumulator. For each accumulator:

1. Generate random $\gamma < q$.
2. Compute $g_1, g_2, \ldots, g_L, g_{L+2}, \ldots, g_{2L}$ where $g_i = g^{\gamma^i}$.
3. Compute $g'_1, g'_2, \ldots, g'_L, g'_{L+2}, \ldots, g'_{2L}$ where $g'_i = g'^{\gamma^i}$.
4. Compute $z = (e(g, g'))^{\gamma^{L+1}}$.
5. Set $V \leftarrow \emptyset$, acc $\leftarrow 1$.

The accumulator public key is (z) and secret key is (γ).

2 Issuance of non-revocation claim

Prover starts:

1. Loads Issuer's revocation key pk^R and generates random s' mod q.
2. Computes $U \leftarrow h_2^{s'}$ taking h_2 from pk^R.
3. Sends U to the Issuer.

Issuer proceeds:

1. Generates random numbers s'', c mod q.
2. Takes m_2 from the primary claim he is preparing for the Prover.
3. Selects the accumulator index A_i and the user index i for the Prover so that i has not been assigned yet for A_i.
4. Computes

$$\sigma \leftarrow \left(h_0 h_1^{m_2} \cdot U_R \cdot g_i \cdot h_2^{s''}\right)^{\frac{1}{x+c}}; \qquad w \leftarrow \prod_{j \in V} g'_{L+1-j+i}; \tag{1}$$

$$\sigma_i \leftarrow g'^{1/(sk+\gamma^i)}; \qquad u_i \leftarrow u^{\gamma^i}; \tag{2}$$

$$\text{acc} \leftarrow \text{acc} \cdot g'_{L+1-i}; \qquad V \leftarrow V \cup \{i\}; \tag{3}$$

$$\text{wit}_i \leftarrow \{\sigma_i, u_i, g_i, w, V\}. \tag{4}$$

5. Sends $(A_i, \sigma, c, s'', \text{wit}_i, g_i, g'_i, i)$.
6. Publishes updated V, acc.

Prover finishes:

1. Computes $s \leftarrow s' + s''$.
2. Stores *non-revocation claim* $C_{NR} \leftarrow (A_i, \sigma, c, s, \text{wit}_i, g_i, g'_i, i)$.

TEST Tests

$$\frac{e(g_i, acc_V)}{e(g, w)} \stackrel{?}{=} z; \tag{5}$$

$$e(pk \cdot g_i, \sigma_i) \stackrel{?}{=} e(g, g'); \tag{6}$$

$$e(\sigma, y \cdot \widehat{h}^c) \stackrel{?}{=} e(h_0 \cdot h_1^{m_2} h_2^s g_i, \widehat{h}). \tag{7}$$

3 Revocation

Issuer revokes ID S, which corresponds to accumulator acc, index i, and valid index set V:

1. Sets $V \leftarrow V \setminus \{i\}$;
2. Computes $\text{acc} \leftarrow \text{acc}/g'_{L+1-i}$.
3. Publishes V, acc.

4 Presentation of non-revocation proof

This phase is a part of the entire presentation protocol, which instructs the prover to maintain sets $\mathcal{C}$ and $\mathcal{T}$, which are filled by all primary claims and their non-revocation complements used in the presentation.

4.1 Preparation

Verifier starts:

1. Loads Issuer's public revocation key $p^R = (h, h_1, h_2, \widetilde{h}, \widehat{h}, u, pk, y)$.

Prover continues:

1. Loads Issuer's public revocation key $p^R = (h, h_1, h_2, \widetilde{h}, \widehat{h}, u, pk, y)$.
2. Loads the non-revocation claim $C_{NR} \leftarrow (A_i, \sigma, c, s, \mathrm{wit}_i, g_i, g_i', i)$;
3. Obtains recent V, acc (from Verifier, Sovrin link, or elsewhere).
4. Updates C_{NR}:

$$\begin{aligned} w &\leftarrow w \cdot \frac{\prod_{j \in V \setminus V_{old}} g'_{L+1-j+i}}{\prod_{j \in V_{old} \setminus V} g'_{L+1-j+i}}; \\ V_{old} &\leftarrow V. \end{aligned}$$

5. Selects random $\rho, \rho', r, r', r'', r''', o, o'$ mod q;
6. Computes

$$\begin{aligned} E &\leftarrow h^{\rho}\widetilde{h}^{o} & D &\leftarrow g^{r}\widetilde{h}^{o'}; & (8) \\ A &\leftarrow \sigma\widetilde{h}^{\rho} & \mathcal{G} &\leftarrow g_i\widehat{h}^{r}; & (9) \\ \mathcal{W} &\leftarrow w\widehat{h}^{r'} & \mathcal{S} &\leftarrow \sigma_i\widehat{h}^{r''} & (10) \\ \mathcal{U} &\leftarrow u_i\widehat{h}^{r'''} & & & (11) \end{aligned}$$

and adds these values to $\mathcal{C}$.

7. Computes

$$\begin{aligned} m &\leftarrow \rho \cdot c; & t &\leftarrow o \cdot c; & (12) \\ m' &\leftarrow r \cdot r''; & t' &\leftarrow o' \cdot r''; & (13) \end{aligned}$$

8. Generates random $\widetilde{\rho}, \widetilde{o}, \widetilde{o'}, \widetilde{c}, \widetilde{m}, \widetilde{m'}, \widetilde{t}, \widetilde{t'}, \widetilde{m_2}, \widetilde{s}, \widetilde{r}, \widetilde{r'}, \widetilde{r''}, \widetilde{r'''}$, mod$q$.
9. Computes

$$\overline{T_1} \leftarrow h^{\widetilde{\rho}}\widetilde{h}^{\widetilde{o}} \qquad \overline{T_2} \leftarrow E^{\widetilde{c}}h^{-\widetilde{m}}\widetilde{h}^{-\widetilde{t}} \qquad (14)$$

$$\overline{T_3} \leftarrow e(A, \widehat{h})^{\widetilde{c}} \cdot e(\widetilde{h}, \widehat{h})^{\widetilde{r}} \cdot e(\widetilde{h}, y)^{-\widetilde{\rho}} \cdot e(\widetilde{h}, \widehat{h})^{-\widetilde{m}} \cdot e(h_1, \widehat{h})^{-\widetilde{m_2}} \cdot e(h_2, \widehat{h})^{-\widetilde{s}} \qquad (15)$$

$$\begin{aligned} \overline{T_4} &\leftarrow e(\widetilde{h}, \mathrm{acc})^{\widetilde{r}} \cdot e(1/g, \widehat{h})^{\widetilde{r'}} & \overline{T_5} &\leftarrow g^{\widetilde{r}}\widetilde{h}^{\widetilde{o'}} & (16) \\ \overline{T_6} &\leftarrow D^{\widetilde{r''}}g^{-\widetilde{m'}}\widetilde{h}^{-\widetilde{t'}} & \overline{T_7} &\leftarrow e(pk \cdot \mathcal{G}, \widehat{h})^{\widetilde{r''}} \cdot e(\widetilde{h}, \widehat{h})^{-\widetilde{m'}} \cdot e(\widetilde{h}, \mathcal{S})^{\widetilde{r}} & (17) \\ \overline{T_8} &\leftarrow e(\widetilde{h}, u)^{\widetilde{r}} \cdot e(1/g, \widehat{h})^{\widetilde{r'''}} & & & (18) \end{aligned}$$

and add these values to $\mathcal{T}$.

TEST Tests that for

$$\begin{array}{cccc} \widetilde{\rho} = \rho & \widetilde{o} = o & \widetilde{o'} = o' & \widetilde{c} = c \\ \widetilde{m} = m & \widetilde{m'} = m' & \widetilde{t} = t & \widetilde{t'} = t' \\ \widetilde{m_2} = m_2 & \widetilde{s} = s & \widetilde{r} = r & \widetilde{r'} = r' \\ \widetilde{r''} = r'' & \widetilde{r'''} = r''' & & \end{array}$$

the following holds:

$$E \overset{?}{=} h^{\widetilde{\rho}}\widetilde{h}^{\widetilde{o}} \qquad 1 \overset{?}{=} E^{\widetilde{c}}h^{-\widetilde{m}}\widetilde{h}^{-\widetilde{t}} \qquad (19)$$

$$\frac{e(h_0\mathcal{G},\widehat{h})}{e(A,y)} \overset{?}{=} e(A,\widehat{h})^{\widetilde{c}} \cdot e(\widetilde{h},\widehat{h})^{\widetilde{r}} \cdot e(\widetilde{h},y)^{-\widetilde{\rho}} \cdot e(\widetilde{h},\widehat{h})^{-\widetilde{m}} \cdot e(h_1,\widehat{h})^{-\widetilde{m_2}} \cdot e(h_2,\widehat{h})^{-\widetilde{s}} \tag{20}$$

$$\frac{e(\mathcal{G},\mathrm{acc})}{e(g,\mathcal{W})z} \overset{?}{=} e(\widetilde{h},\mathrm{acc})^{\widetilde{r}} \cdot e(1/g,\widehat{h})^{\widetilde{r'}} \qquad D \overset{?}{=} g^{\widetilde{r}}\widetilde{h}^{\widetilde{o'}} \tag{21}$$

$$1 \overset{?}{=} D^{\widetilde{r''}} g^{-\widetilde{m'}} \widetilde{h}^{-\widetilde{t'}} \qquad \frac{e(pk\cdot\mathcal{G},\mathcal{S})}{e(g,g')} \overset{?}{=} e(pk\cdot\mathcal{G},\widehat{h})^{\widetilde{r''}} \cdot e(\widetilde{h},\widehat{h})^{-\widetilde{m'}} \cdot e(\widetilde{h},\mathcal{S})^{\widetilde{r}} \tag{22}$$

$$\frac{e(\mathcal{G},u)}{e(g,\mathcal{U})} \overset{?}{=} e(\widetilde{h},u)^{\widetilde{r}} \cdot e(1/g,\widehat{h})^{\widetilde{r'''}} \tag{23}$$

After all claims are processed, Prover generates c_H:

$$c_H \leftarrow H(\mathcal{T},\mathcal{C},n_1).$$

4.2 Last preparation steps

Prover finalizes:

1. Computes

$$\begin{aligned}
\widehat{\rho} &\leftarrow \widetilde{\rho} - c_H\rho \bmod q & \widehat{o} &\leftarrow \widetilde{o} - c_H\cdot o \bmod q\\
\widehat{c} &\leftarrow \widetilde{c} - c_H\cdot c \bmod q & \widehat{o'} &\leftarrow \widetilde{o'} - c_H\cdot o' \bmod q\\
\widehat{m} &\leftarrow \widetilde{m} - c_H m \bmod q & \widehat{m'} &\leftarrow \widetilde{m'} - c_H m' \bmod q\\
\widehat{t} &\leftarrow \widetilde{t} - c_H t \bmod q & \widehat{t'} &\leftarrow \widetilde{t'} - c_H t' \bmod q\\
\widehat{m_2} &\leftarrow \widetilde{m_2} - c_H m_2 \bmod q & \widehat{s} &\leftarrow \widetilde{s} - c_H s \bmod q\\
\widehat{r} &\leftarrow \widetilde{r} - c_H r \bmod q & \widehat{r'} &\leftarrow \widetilde{r'} - c_H r' \bmod q\\
\widehat{r''} &\leftarrow \widetilde{r''} - c_H r'' \bmod q & \widehat{r'''} &\leftarrow \widetilde{r'''} - c_H r''' \bmod q.
\end{aligned}$$

   and add them to $\mathcal{X}$.

After all claims are processed this way, Prover sends $(c_H,\mathcal{X},\{Pr_C\},\{Pr_p\},\mathcal{C})$ to the Verifier.

4.3 Verification of non-revocation proof

Verifier computes

$$\widehat{T_1} \leftarrow E^{c_H}\cdot h^{\widehat{\rho}}\cdot \widetilde{h}^{\widehat{o}} \qquad \widehat{T_2} \leftarrow E^{\widehat{c}}\cdot h^{-\widehat{m}}\cdot \widetilde{h}^{-\widehat{t}} \tag{24}$$

$$\widehat{T_3} \leftarrow \left(\frac{e(h_0\mathcal{G},\widehat{h})}{e(A,y)}\right)^{c_H} \cdot e(A,\widehat{h})^{\widehat{c}} \cdot e(\widetilde{h},\widehat{h})^{\widehat{r}} \cdot e(\widetilde{h},y)^{-\widehat{\rho}} \cdot e(\widetilde{h},\widehat{h})^{-\widehat{m}} \cdot e(h_1,\widehat{h})^{-\widehat{m_2}} \cdot e(h_2,\widehat{h})^{-\widehat{s}} \tag{25}$$

$$\widehat{T_4} \leftarrow \left(\frac{e(\mathcal{G},\mathrm{acc})}{e(g,\mathcal{W})z}\right)^{c_H} \cdot e(\widetilde{h},\mathrm{acc})^{\widehat{r}} \cdot e(1/g,\widehat{h})^{\widehat{r'}} \qquad \widehat{T_5} \leftarrow D^{c_H}\cdot g^{\widehat{r}}\widetilde{h}^{\widehat{o'}} \tag{26}$$

$$\widehat{T_6} \leftarrow D^{\widehat{r''}}\cdot g^{-\widehat{m'}}\widetilde{h}^{-\widehat{t'}} \qquad \widehat{T_7} \leftarrow \left(\frac{e(pk\cdot\mathcal{G},\mathcal{S})}{e(g,g')}\right)^{c_H} \cdot e(pk\cdot\mathcal{G},\widehat{h})^{\widehat{r''}} \cdot e(\widetilde{h},\widehat{h})^{-\widehat{m'}} \cdot e(\widetilde{h},\mathcal{S})^{\widehat{r}} \tag{27}$$

$$\widehat{T_8} \leftarrow \left(\frac{e(\mathcal{G},u)}{e(g,\mathcal{U})}\right)^{c_H} \cdot e(\widetilde{h},u)^{\widehat{r}} \cdot e(1/g,\widehat{h})^{\widehat{r'''}} \tag{28}$$

and adds these values to $\widehat{T}$.

4.4 Final hashing

After all claims are processed this way:

1. Verifier computes
$$\widehat{c_H} \leftarrow H(\widehat{\mathcal{T}},\mathcal{C},n_1).$$
2. If $c_H = \widehat{c_H}$ output VERIFIED else FAIL.

5 Differences from the original

This section lists the changes to the type-1 pairing-based revocation scheme:

1. A new group $\mathbb{G}_2$ is introduced with generator g'.
2. A new variable $\widehat{h}$ is introduced.
3. Variable u now belongs to $\mathbb{G}_2$.
4. Variable y is now computed from $\widehat{h}$.
5. New variables $g'_1, \ldots, g_{2L}$ are introduced in $\mathbb{G}_2$.
6. v_R is replaced by s.
7. $w, \mathrm{acc}, \sigma_i$ are computed using g', not g.
8. $\mathcal{W}, \mathcal{S}, \mathcal{U}$ are computed using $\widehat{h}$.
9. The second argument of the pairing function is never h or $\widetilde{h}$, both cases are now using $\widehat{h}$.