Summary
This document extends the U-Prove Cryptographic Specification [UPCS] by specifying an efficient revocation mechanism based on a dynamic accumulator. This scheme requires a designated verifier that shares the Revocation Authority’s private key. Unlike many accumulator schemes based on bilinear pairings, this scheme is built using a prime-order group (like the ones defined in [UPCS]) and is therefore suitable for system that require standard constructions.
Contents
Summary ............................................................................................................................................. 1
1 Introduction ...................................................................................................................................... 3
  1.1 Notation ..................................................................................................................................... 3
  1.2 Feature overview .......................................................................................................................... 3
2 Protocol specification ........................................................................................................................ 4
  2.1 Revocation Authority setup .......................................................................................................... 4
  2.2 Token issuance ............................................................................................................................. 4
  2.3 Revocation list management ......................................................................................................... 4
  2.4 Token presentation ....................................................................................................................... 6
  2.5 Revocation verification .................................................................................................................. 7
References .............................................................................................................................................. 9

List of Figures
Figure 1: Function RASetup .................................................................................................................. 4
Figure 2: Function ComputeAccumulator .............................................................................................. 5
Figure 3: Function ComputeWitness .................................................................................................... 5
Figure 4: Function UpdateWitness ....................................................................................................... 6
Figure 5: Function GenerateNonRevocationProof .............................................................................. 7
Figure 6: Function VerifyNonRevocationProof .................................................................................... 8

Change history

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft Revision 1</td>
<td>09/11/2013</td>
<td>Initial</td>
</tr>
<tr>
<td></td>
<td>02/26/2014</td>
<td>Added clarification to Figure 3</td>
</tr>
</tbody>
</table>
1 Introduction
This document extends the U-Prove Cryptographic Specification [UPCS] by specifying an efficient revocation mechanism based on a dynamic accumulator. This scheme requires a designated verifier that shares the Revocation Authority’s private key. Unlike many accumulator schemes based on bilinear pairings, this scheme is built using a prime-order group (like the ones defined in [UPCS]) and is therefore suitable for systems that require standard constructions.

1.1 Notation
In addition to the notation defined in [UPCS], the following notation is used throughout the document.

\[ a \notin A \]
Indicates that element \( a \) is not in set \( A \).

The key words “MUST”, “MUST NOT”, “SHOULD”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC 2119].

1.2 Feature overview
Revocation is an important feature of a credential system; this document specifies a scheme based on a dynamic accumulator using a standard prime order group instead of bilinear pairings.\(^1\)

A cryptographic accumulator allows the aggregation of a large set of elements into one constant-size value, and the ability to prove that an element has been aggregated or not in the accumulator.

The accumulator scheme is built on the prime order groups defined in [UPCS]; it is therefore possible to use the scheme as a revocation mechanism for U-Prove tokens. The Revocation Authority is a new party that manages the revocation list and validates the users’ non-revocation proofs. Each token encodes a unique user identifier UID; a non-revocation proof consists of proving that the value UID has not been accumulated in the accumulator. In order to create an efficient non-revocation proof, users periodically obtain revocation witnesses (computed on-demand by the Revocation Authority, or by users as the revocation list is updated); users can then compute in constant time the non-revocation proof.

We detail the DA revocation extension in five steps.

1 Revocation Authority setup: The Revocation Authority generates its public parameters and secret key, and makes the public parameters available to users.
2 Token issuance: The user obtains U-Prove tokens encoding her unique identifier UID from the Issuer.
3 Revocation list management: Periodically, the Revocation Authority updates the revocation accumulator, and the user obtains non-revocation witnesses from the Revocation Authority, or computes them using the revocation list update.
4 Token presentation: The user presents a U-Prove token to the Verifier, including a non-revocation proof (using the non-revocation witnesses). The Verifier validates the presentation proof.
5 Revocation verification: The Verifier sends the non-revocation proof to the Revocation Authority that verifies that the undisclosed UID does not appear on the current revocation list.

\(^1\) Many accumulator-based schemes rely on bilinear pairings, common in the cryptographic literature, but not yet popular in industry systems and standards.
2 Protocol specification

2.1 Revocation Authority setup
The Revocation Authority generates its key and parameters as specified in Figure 1. The group $G_q$ MUST match
the one specified in the parameters of Issuers that will issue tokens revocable by this Revocation Authority.

\[ RASetup(\ ) \]

Input
Parameters: desc($G_q$), $g$ (the $G_q$ generator)

Computation
Choose $\delta, r_1$ and $r_2$ at random from $\mathbb{Z}_q^*$
Compute $P := g, H := g^{r_1}, K := H^\delta, G := g^{r_2}$

Output
Private key: $\delta$
Public key: $P, H, K, G$

Figure 1: Function RASetup

2.2 Token issuance
Token issuance follows the same steps as in \[UPCS\]. One of the attributes, called the revocation attribute and
denoted $x_{id}$ (where $id$ is an index value between 1 and the number of attributes in the tokens), is reserved for
revocation.

2.3 Revocation list management
The Revocation Authority computes the accumulator corresponding to a set of revoked attribute values\(^2\) (see
Figure 2); the accumulator is re-computed when values are added or removed from the revocation list. Users
periodically obtain revocation witnesses corresponding to their revocation attribute $x_{id}$ allowing them to create
non-revocation proofs. If the revocation list is secret, or for better efficiency, the witnesses are computed by the
Revocation Authority (see Figure 3); otherwise, the witnesses are computed by users and updated when values
are added or removed from the revocation list (see Figure 4).

\(^2\) Initially, this set can be empty.
\begin{figure}
\centering
\begin{align*}
\textbf{ComputeAccumulator}() \\
\text{Input} & \quad \text{Revocation Authority private key: } \delta \in \mathbb{Z}_q^* \\
& \quad \text{Revocation Authority public key: } P, H, K, G \in G_q \\
& \quad \text{List of revoked attribute values: } R = \{x_1, \ldots, x_m\} \in \mathbb{Z}_q - \{\delta\} \\
\text{Computation} & \quad V := P^{\prod_{x \in R}(\delta + x)} \\
\text{Output} & \quad \text{Return accumulator } V 
\end{align*}
\caption{Function ComputeAccumulator}
\end{figure}

\begin{figure}
\centering
\begin{align*}
\textbf{ComputeWitness}() \\
\text{Input} & \quad \text{Revocation Authority private key: } \delta \in \mathbb{Z}_q^* \\
& \quad \text{Revocation Authority public key: } P, H, K, G \in G_q \\
& \quad \text{List of revoked attribute values: } R = \{x_1, \ldots, x_m\} \in \mathbb{Z}_q - \{\delta\} \\
& \quad \text{Target attribute value: } x_{id} \notin R \\
& \quad \text{Current accumulator: } V \in G_q \\
\text{Computation} & \quad d = f(\delta) \mod (\delta + x_{id}) \in \mathbb{Z}_q \quad \text{where } f(\delta) = \prod_{x \in R}(\delta + x) \\
& \quad W = P^{\prod_{x \in R}(\delta + x) - d}/(\delta + x_{id}) \\
& \quad Q = VW^{-x_{id}}p^{-d} \\
\text{Output} & \quad \text{Revocation witness for target } x_{id}: (d, W, Q)_{x_{id}} 
\end{align*}
\caption{Function ComputeWitness}
\end{figure}

Note that the computation of \( d \) is a polynomial division of polynomial \( f(\delta) = \prod_{x \in R}(\delta + x) \) over polynomial \( (\delta + x_{id}) \) in polynomial ring \( \mathbb{Z}_q[\delta] \). As the denominator is a polynomial of degree 1, the result \( d \) is a polynomial of degree 0, i.e. a constant, in the polynomial ring. So \( d \) can be computed from just the set \( R \) and does not depend on \( \delta \).
**UpdateWitness**

**Input**
- Revocation Authority private key: \( \delta \in \mathbb{Z}_q^* \)
- Revocation Authority public key: \( P, H, K, G \in G_q \)
- Revocation attribute value: \( x_{id} \)
- New revoked value: \( x' \)
- Boolean: add (if true, \( x' \) will be added to the witness, otherwise it will be removed)
- Old accumulator: \( V \in G_q \)
- Old witness: \( (d, W, Q)_{x_{id}} \)
- Updated accumulator: \( V' \in G_q \)

**Computation**
- If add = true
  - \( d' := d(x' - x_{id}) \)
  - \( W' := VW(x' - x_{id}) \)
  - \( Q' := V'W - x_{id}p - d' \)
- Else
  - \( d' := d(x' - x_{id})^{-1} \)
  - \( W' := (V'^{-1}W)(x' - x_{id})^{-1} \)
  - \( Q' := V'W - x_{id}p - d' \)

**Output**
- Updated revocation witness: \( (d', W', Q')_{x_{id}} \)

**Figure 4:** Function UpdateWitness

### 2.4 Token presentation

The presentation proof is generated according to the needs of the application following [UPCS], but additionally \( x_{id} \) is a committed attribute. The (public) output \( \tilde{c}_{id} = g^{x_{id}}g_1^{o_{id}} \) and the (private) opening information \( (x_{id}, o_{id}) \), are input to the non-revocation proof generation defined in Figure 5.
GenerateNonRevocationProof( )

Input
Parameters: UID_p, desc(G_q), UID_K, g_1
Commitment to x_id: ̃c_id
Opening information: x_id, a_id
Revocation Authority public key: P, H, K, G
Revocation witness: (d, W, Q) x_id

Computation
Generate t_0, t_1, t_2, t_3, k_0, ..., k_8 at random from ℤ_q
B := g^{k_0} g_1^{t_0}
X := W H^{t_1}
Y := Q K^{t_2}
R := G^{d} H^{t_2}
S := G^{d^{-1}} H^{t_3}
T_1 := G^{k_4} H^{k_8}
T_2 := G^{k_5} H^{k_8} R^{-k_0}
T_3 := G^{k_6} H^{k_8}
T_4 := H^{k_8} S^{-k_8}
Γ := X^{-k_0} H^{k_5} K^{k_8} P^{-k_5}
c' := H(g, g_1, P, H, K, G, ̃c_id, B, X, Y, R, S, T_1, T_2, T_3, T_4, Γ)
r' := -c' x_id + k_0 \mod q
s_0 := -c' x_id + k_0 \mod q
for each i ∈ {1, 2, 3}, s_i := -c' t_i + k_i \mod q
s_4 := -c' t_2 x_id + k_4 \mod q
s_5 := -c' d + k_5 \mod q
s_6 := -c' d^{-1} + k_6 \mod q
s_7 := -c' t_1 x_id + k_7 \mod q
s_8 := -c' t_3 d + k_8 \mod q
Delete t_0, t_1, t_2, t_3, k_0, ..., k_8

Output
Return c', r', s_0, ..., s_8, X, Y, R, S

Figure 5: Function GenerateNonRevocationProof.

2.5 Revocation verification
The revocation verification function, defined in Figure 6, is run after the corresponding presentation proof has been successfully verified. Inputs are the commitment ̃c_id from the presentation proof and the non-revocation proof. If the presentation and non-revocation proofs are valid, then the verifier has assurance that ̃c_id is a valid commitment to the attribute x_id and that x_id is not in the revocation list.
\textbf{VerifyNonRevocationProof( )}

Input
Parameters: desc(\(G_q\)), \(g_1\)
Commitment to \(x_{id}: \tilde{c}_{id}\)
Non-revocation proof: \(c', r', s_0, \ldots, s_8, X, Y, R, S\)
Revocation Authority public key: \(P, H, K, G\)
Revocation Authority private key: \(\delta\)

Computation
\[ B := g^{s_0}g^r \tilde{c}_d \]
\[ T_1 := G^{s_1}H^s R^c \]
\[ T_2 := G^{s_7}H^4 R^{-s_0} \]
\[ T_3 := G^{s_6}H^3 S^c \]
\[ T_4 := G^{-c}H^8 S^{-s_5} \]
\[ \Gamma := X^{-s_0}H^{s_5}K^5 p^{-s_5}(V^{-1}Y)^c \]
Verify that \(c' = \mathcal{H}(g, g_1, P, H, K, G, \tilde{c}_{id}, B, X, Y, R, S, T_1, T_2, T_3, T_4, \Gamma)\)
Verify that \(Y = X^\delta\)

\textit{Figure 6: Function VerifyNonRevocationProof}
References
