U-Prove Collaborative Issuance Extension

Draft Revision 1

Microsoft Research

Author: Christian Paquin, Greg Zaverucha

June 2014

Summary

This document extends the U-Prove Cryptographic Specification [UPCS] by specifying how the Prover and the Issuer can collaborate to issue tokens encoding attributes unknown to the Issuer.

Contents

Summary 1
1 Introduction 3
1.1 Notation 3
2 Protocol specification 3
2.1 Protocol overview 3
2.2 Pre-issuance phase 4
2.3 Issuance phase 6
2.3.1 Prover steps 6
2.3.2 Issuer steps 6
3 Security considerations 7
References 7

List of Figures

Figure 1: PreIssuanceProofGeneration 5
Figure 2: PreIssuanceProofVerification 6

Change history

Version	Description
Revision 1	Initial draft

1 Introduction

The U-Prove specification [UPCS] requires the Issuer to know the attribute values it issues to Provers. However, it is useful for the Issuer to collaborate with either a registration authority[1] or the Prover to issue tokens on attributes for which it doesn't know the value. This specification describes how to achieve this collaborative issuance protocol.

When the Issuer fully trust the entity providing the attribute values, e.g., if they come from a trusted Attribute Provider in a federation, then the issuance input value γ defined in Figure 8 of [UPCS] can be calculated and simply provided to the Issuer by that trusted entity.[2] If, however, the providing entity is not fully trusted, then a protocol must be run before issuance to convince the Issuer that the unknown attributes have the right properties.

This feature is useful to carry-over attributes from one token to another. For example, an Issuer participating in a trust framework could encode into a new token the Prover's revocation handle encoded in a presented token, without learning its value. The Prover needs to present proof to the Issuer that the carried-over attributes are well-formed and belong to the Prover.

As an example, the scope-exclusive pseudonym feature defined in [UPCS] allows users to present pseudonyms derived from unique attribute values. However, unless the attribute value is a random number, the Issuer is able to track users when collaborating with Relying Parties by brute forcing the presented pseudonyms. The best way to use this feature is for the user to generate a random secret from $\mathbb{Z}_q$ and have it encoded as an attribute in each token. For the "primordial" U-Prove token, the user can present the blinded secret to the Issuer that can remember this value for future issuances. For tokens issued by other Issuers, the user can "carry-over" the attribute value.

1.1 Notation

The notation defined in [UPCS] is used in this document. The key words "MUST", "MUST NOT", "SHOULD", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119].

2 Protocol specification

This section describes the

2.1 Protocol overview

The attributes encoded in γ, to be used in the issued token are divided into three types:

1. Known: attributes known to the Issuer. Represented by the set $K$of indices
2. Unknown: attributes unknown to the Issuer, but that are not committed. The Prover proves knowledge of these attributes, but nothing more about them. Represented by the set U of indices.
3. Committed: attributes for which the Issuer is given commitments, and possibly further proof about the attribute (e.g., that the attribute is contained in another token). Represented by the set C of indicies. This serves as an extension point for other issuance features. The Prover and Issuer may engage in any protocol beforehand, and create a commitment used as input (provided the Prover knows the opening).

[1] Similar to the CA/RA split, in a traditional PKI.
[2] To prevent the Issuer from guessing the encoded attributes, the γ value can be blinded before being provided to the Issuer.

Each attribute is in only one of K, U, C, and every $i \in \{1, \dots, n\}$ is contained in one of the three sets.

There are two parts to the issuance protocol. In the first part, the Prover provides the Issuer with the attribute information (perhaps committed), along with proof the hidden attributes are valid. The output of the first part is a value h_0, a blinded version of the value γ used in the current protocol. The second part uses h_0 to create a U-Prove token. The second part is essentially the U-Prove issuance protocol with a small modification on the Prover's side, to unblind the values derived from γ.

Let $\gamma = g_0 g_1^{x_1} \dots g_n^{x_n} g_t^{x_t} [h_d]_{\mathbf{d}}$ and $h_0 = \gamma^{\beta_0}$, where β_0 is a secret blinding value chosen by the Prover. For each $i \in C$, let $C_i = g^{x_i} g_1^{r_i}$. Let $C_\gamma = \gamma g^\rho$, a commitment to γ, where g is the base in the system parameters, and ρ is chosen at random in $\mathbb{Z}_q^*$. Let $C_{h_0} = C_\gamma^{\beta_0}$, and note that $C_{h_0} = h_0 g^{\rho\beta_0}$ (a commitment to h_0). The proof the Prover computes is described by:

PK{ ($\beta_0, \rho, x_1, \dots, x_n, \{r_i : i \in C\}$) :

$C_{h_0} = C_\gamma^{\beta_0}$	(needed for the multiply proof, C_{h_0} commits to h_0)
$C_{h_0}/h_0 = g^d$	(proves thath_0 is computed correctly)
$\wedge\ C_\gamma = (g_0 g_1^{x_1} \dots g_n^{x_n} g_t^{x_t} [h_d]_{\mathbf{d}}) g^\rho$	(proves that C_γ and γ are computed correctly)
$\wedge\ C_i = g^{x_i} g_1^{r_i}$ for all $i \in C$ }	(proves that the x_i values match the commitments)

2.2 Pre-issuance phase

The optional phase is performed when the Issuer needs to be convinced that the provided γ value is well formed. If this value is provided by a party trusted by the issuer, then it can be skipped.

The Prover generates a pre-issuance proof by following the steps from Figure 1; the Issuer verifies it by following the steps from Figure 2.

Some notes:

- The methods ComputeXi and ComputeXt are defined in [UPCS].
- If attributes are carried from one token to another, then the source token is presented to the Issuer, following the steps from Figure 9 of [UPCS], and by committing the attributes to be carried-over.

PreIssuanceProofGeneration( )

Input

Issuer parameters: $IP = \text{UID}_\text{P}, \text{desc}(G_q), \text{UID}_\mathcal{H}, (g_0, g_1, \dots, g_n, g_t), (e_1, \dots, e_n), S$
Ordered indices of known attributes: $K \subset \{1, \dots, n\}$
Ordered indices of unknown attributes: $U \subset \{1, \dots, n\} - K$
Ordered indices of committed attributes: $C \subset \{1, \dots, n\} - (K \cup U)$
Application-specific attribute information: $(\text{A}_1, \dots, \text{A}_n), TI \in \{0,1\}^*$
Device-protected Boolean: $\mathbf{d}$
$[\text{Device parameters: } g_d, h_d]_\mathbf{d}$
For each $i \in \{1, \dots, n\}$, $x_i := \text{ComputeXi}(IP, \text{A}_i)$
$x_t := \text{ComputeXt}(IP, TI, \mathbf{d}, [g_d]_\mathbf{d})$
Attribute values $(x_1, \dots, x_n)$
$\gamma = g_0 g_1^{x_1} \dots g_n^{x_n} g_t^{x_t} [h_d]_\mathbf{d}$
Commitments $\{C_i\}_{i \in C}$, and their openings $\{r_i\}_{i \in C}$
Message: $m \in \{0,1\}^*$

Computation

Generate $\beta_0, \tilde{\beta}_0, \rho, \tilde{\rho}, \tilde{d}$ at random from $\mathbb{Z}_q^*$
$h_0 := \gamma^{\beta_0}$.
$C_\gamma := \gamma g^\rho$
$C_{h_0} := C_\gamma^{\beta_0}$
For each $i \notin K$, generate $\tilde{x}_i$ at random from $\mathbb{Z}_q^*$
For each $i \in C$, generate $\tilde{r}_i$ at random from $\mathbb{Z}_q^*$
$\tilde{D} := g^{\tilde{d}}$.
$\tilde{C}_\gamma := (\prod_{i \notin K} g_i^{\tilde{x}_i}) g^{\tilde{\rho}}$
$\tilde{T} := C_\gamma^{\tilde{\beta}_0}$
For each $i \in C$, $\tilde{C}_i := g^{\tilde{x}_i} g_1^{\tilde{r}_i}$
$c := H\left(IP, h_0, C_\gamma, C_{h_0}, \langle \{C_i\}_{i \in C} \rangle, \tilde{D}, \tilde{C}_\gamma, \tilde{T}, \langle \{\tilde{C}_i\}_{i \in C} \rangle, m\right)$
$s_{\beta_0} := \tilde{\beta}_0 - \beta_0 c \mod q$
$s_d := \tilde{d} - (\beta_0 \rho) c \mod q$
$s_\rho := \tilde{\rho} - \rho c \mod q$
For each $i \notin K$, $s_{x_i} := \tilde{x}_i - x_i c \mod q$
For each $i \in C$, $s_{r_i} := \tilde{r}_i - r_i c \mod q$

Output pre-issuance values

Blinding value: β_0
Pre-issuance proof: $\left(IP, h_0, C_\gamma, C_{h_0}, c, s_{\beta_0}, s_d, s_\rho, \{s_{x_i}\}_{i \notin K}, \{s_{r_i}\}_{i \in C}\right)$

Figure 1: PreIssuanceProofGeneration

PreIssuanceProofVerification( )

Input

Issuer parameters: $IP = \text{UID}_\text{P}, \text{desc}(G_q), \text{UID}_\mathcal{H}, (g_0, g_1, \dots, g_n, g_t), (e_1, \dots, e_n), S$
Ordered indices of known attributes: $K \subset \{1, \dots, n\}$
Ordered indices of unknown attributes: $U \subset \{1, \dots, n\} - K$
Ordered indices of committed attributes: $C \subset \{1, \dots, n\} - (K \cup U)$
Known application-specific attribute information: $\{A_i\}_{i \in K}, TI \in \{0,1\}^*$
Device-protected Boolean: $\mathbf{d}$
$[\text{Device parameters: } g_d, h_d]_\mathbf{d}$
For each $i \in K$, $x_i := \text{ComputeXi}(IP, \text{A}_i)$
$x_t := \text{ComputeXt}(IP, TI, \mathbf{d}, [g_d]_\mathbf{d})$
Attribute commitments: $\{C_i\}_{i \in C}$
Pre-issuance proof: $\left(h_0, C_\gamma, C_{h_0}, c, s_{\beta_0}, s_d, s_\rho, \{s_{x_i}\}_{i \notin K}, \{s_{r_i}\}_{i \in C}\right)$
Message: $m \in \{0,1\}^*$

Computation

$\gamma_k := g_0 (\prod_{i \in K} g_i^{x_i}) g_t [h_d]_\mathbf{d}$
$D' := \left(\frac{C_{h_0}}{h_0}\right)^c g^{s_d}$
$C_\gamma' := (C_\gamma / \gamma_k)^c \left(\prod_{i \notin K} g_i^{s_{x_i}}\right) g^{s_\rho}$
$T' := C_\gamma^c C_\gamma^{s_{\beta_0}}$
For each $i \in C$, $C_i' := C_i^c g^{s_{x_i}} g_1^{s_{r_i}}$
$c' := H\left(IP, h_0, C_\gamma, C_{h_0}, \{C_i : i \in C\}, D', C_\gamma', T', \{C_i' : i \in C\}, m\right)$
Verify that $c' = c$.

Figure 2: PreIssuanceProofVerification

2.3 Issuance phase

2.3.1 Prover steps

The Prover needs to unblind the values derived from γ provided by the Issuer. It needs to modify the following steps from Figure 8 of [UPCS] using the β_0 value obtained in Figure 1:

- Precomputation step: $h := \gamma^{\alpha(\beta_0^{-1})}$
- Second message steps:
  - $\sigma_z' := \sigma_z^{\alpha(\beta_0^{-1})}$
  - $\sigma_b' := (\sigma_z')^{\beta_1} t_2 \sigma_b^{\alpha(\beta_0^{-1})}$

2.3.2 Issuer steps

The Issuer performs the steps from Figure 8 of [UPCS], setting its input value γ to either the blinded version $\gamma_0^{\beta_0}$ received from a trusted party, or to the value h_0 received in the pre-issuance phase.

3 Security considerations

+++

References

[RFC2119] Scott Bradner. *RFC 2119: Key words for use in RFCs to Indicate Requirement Levels*, 1997. ftp://ftp.rfc-editor.org/in-notes/rfc2119.txt.

[UPCS] Christian Paquin, Greg Zaverucha. *U-Prove Cryptographic Specification V1.1 (Revision 3)*. Microsoft, December 2013. http://www.microsoft.com/u-prove.