U-Prove Bit Decomposition Extension

Draft Revision 1

Microsoft Research

Author: Mira Belenkiy

June 2014

Summary

This document extends the U-Prove Cryptographic Specification [UPCS] by specifying bit decomposition proofs, useful for other extension protocols.

Contents

List of Figures

Change history

Version	Description
Revision 1	Initial draft

1 Introduction

This document extends the U-Prove Cryptographic Specification [UPCS] by specifying bit decomposition proofs, useful for other extension protocols.

The Prover and Verifier have as common input a list of values $C, C_0, C_1, \ldots, C_{n-1} \in G_q$ and a pair of generators $g, h \in G_q$. The Prover wants to show that the C_i are Pedersen Commitments to the bit decomposition of the committed value in C.

$$\pi = PK\left\{\{\alpha_i, \beta_i\}_{i\in[0,n-1]}, \gamma \middle| \left(\forall i: C_i = g^{\propto_i} h^{\beta_i} \cap \alpha_i \in [0,1]\right) \cap C = h^{\gamma} \prod_{i\in[0,n-1]} (C_i)^{2^i}\right\}$$

The Prover knows knows a set of values $\{x_i, y_i\}_{i\in[0,n-1]}, z$ that would satisfy the above relation. The Prover will create a special honest-verifier non-interactive zero-knowledge proof of knowledge using its witness $\{x_i, y_i\}_{i\in[0,n-1]}, z$ that satisfies the above relation. The Prover will create n separate set-membership proofs [EXSM] to show that $\forall i: C_i = g^{\propto_i} h^{\beta_i} \cap \alpha_i \in [0,1]$. The Prover will create a separate equality proof [EXEQ] to show that $C = h^{\gamma} \prod_{i\in[0,n-1]} (C_i)^{2^i}$.

The U-Prove Cryptographic Specification [UPCS] allows the Prover, during the token presentation protocol, to create a Pedersen Commitment and show that the committed value is the equal to a particular token attribute. The Prover MAY use this Pedersen Commitment as either C or any of the C_i for the bit decomposition proof. The Issuance and Token Presentation protocols are unaffected by this extension. The Prover may choose to create a bit decomposition proof after these two protocols complete.

The committed value in C and all of the C_i MUST NOT be hashed. If any of these values are U-Prove token attributes, the attributes also MUST NOT be hashed.

1.1 Notation

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

C	Value of the Prover's Pedersen Commitment.
x	Committed value of Pedersen Commitment C.
y	Opening of Pedersen Commitment C.
C_i	Commitment to the i[th] bit of the decomposition of x.
x_i	The i[th] bit of the decomposition of x, the committed value of Pedersen Commitment C_i.
y_i	The opening of Pedersen Commitment C_i
A	Input to equality proof; C divided by the composition of the C_i.
z	Prover's witness for equality proof, the discrete logarithm of A.
M	Part of set membership proof: "response".
π	Equality proof.
π_i	Set membership proof.

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

The Bit Decomposition proof consists of a straightforward combination of a set membership proof and an equality proof.

To show that each value C_i is a Pedersen Commitment to either 0 or 1, the Prover will create a set membership proof [EXSM] for the set $[0,1]$.

To show that composing the committed values in the C_i results in C, the Prover will create an equality proof [EXEQ]. The Prover knows witnesses $x, y, (x_0, y_0), (x_1, y_1) \dots, (x_{n-1}, y_{n-1})$ that are the openings of $C, C_0, C_1, \dots, C_{n-1}$. The Prover will compute

$$z := y - \sum_{i \in [0,n-1]} 2^i y_i \mod q$$

It is easy to see that the following relation holds:

$$C = h^z \cdot \prod_{i \in [0,n-1]} (C_i)^{2^i}$$

The Prover will create proof of knowledge of the discrete logarithm of $A = C / \prod (C_i)^{2^i}$ in terms of the generator h.

2 Protocol specification

As the bit decomposition proof can be performed independently of the U-Prove token presentation protocols, the common parameters consist simply of the group G_q, two generators g and h, and a cryptographic function $\mathcal{H}$. The commitments $C, C_0, C_1, \dots, C_{n-1}$ and their openings MAY be generated by the Prover.

2.1 Presentation

The presentation protocol consists of creating n set membership proofs for the set $[0,1]$, and an equality proof to prove valid decomposition.

BitDecompositionProve( )

Input

Parameters: desc(G_q), $\text{UID}_{\mathcal{H}}, g, h$
Commitment to x: C
Commitment to decomposition: $C_0, C_1, \ldots, C_{n-1}$
Opening information: x, y
Opening information: $(x_0, y_0), (x_1, y_1) \ldots, (x_1, y_1)$

Computation

For all $i \in [0, n-1]$

$\pi_i := \text{SetMembershipProve}(desc(G_q), \text{UID}_{\mathcal{H}}, g, h, C_i, x_i, y_i, \{0,1\})$

end

$$A := C \cdot \prod_{i \in [0,n-1]} (C_i)^{-2^i}$$

$$z := y - \sum_{i \in [0,n-1]} 2^i y_i \mod q$$

$\mathcal{M} := \emptyset$

$\pi := \text{EqualityProve}(desc(G_q), \text{UID}_{\mathcal{H}}, \{(A, h)\}, \mathcal{M}, z)$

Output

Return $\pi, \pi_0, \pi_1, \ldots, \pi_{n-1}$,

Figure 1: BitDecompositionProve

2.2 Verification

The Verifier verifies the set membership and equality proofs.

BitDecompositionVerify( )

Input

Parameters: desc(G_q), $\text{UID}_{\mathcal{H}}, g, h$
Commitment to x: C
Commitment to decomposition: $C_0, C_1, \ldots, C_{n-1}$
Proof $\pi, \pi_0, \pi_1, \ldots, \pi_{n-1}$,

Computation

$pass := true$

For all i **in** $0 \ldots n-1$

If $\text{SetMembershipVerify}(desc(G_q), UID_{\mathcal{H}}, g, h, C_i, \{0,1\}, \pi_i) = false$
then $pass := false$

end

$$A := C \cdot \prod_{i \in [0,n-1]} (C_i)^{-2^i}$$

$\mathcal{M} := \emptyset$

If $\text{EqualityVerify}(desc(G_q), \text{UID}_{\mathcal{H}}, \{(A, h)\}, \mathcal{M}, \pi) = false$ **then** $pass := false$

Output

Return *pass*

Figure 2: BitDecompositionVerify

3 Security considerations

The bit decomposition proof protocol is a composition of the set membership proof and the equality proof. The following restrictions apply:

1. The Prover and the Verifier MUST NOT know the relative discrete logarithm $\log_g h$ of the generators g and h. This is not an issue if the generators are chosen from the list of U-Prove recommended parameters.

References

[EXEQ] Mira Belenkiy. *U-Prove Equality Proof Extension*. Microsoft, June 2014. http://www.microsoft.com/u-prove.

[EXSM] Mira Belenkiy. *U-Prove Set Membership Proof Extension*. Microsoft, June 2014. http://www.microsoft.com/u-prove.

[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.