Σ-Protocol Example

David Mis

February 12, 2014

The purpose of this document is to demonstrate my understanding of proofs-of-knowledge based on Σ-protocols. I'll do this by describing Protocol 1 of Camenisch et al, which is treated as a black-box and the details are left to the reader. Some placeholder names have been changed for clarity since we are considering Camenisch's scheme out of context.

Let n be an RSA modulus, and QR_n be the group of quadratic residues modulo n. Let g, h be quadratic residues in QR_n.

The protocol is executed between a prover P and a verifier V. Both P and V know n, g and h.

The protocol is set up in the following manner: P secretly chooses four random integers a, b, c, and d. P then sets $X_1 = g^a h^b$ and $X_2 = g^c h^d$. P sends X_1 and X_2 to V, and wants to prove that they are formed correctly – that is, X_1 and X_2 are both the product of a power of g and a power of h. Using Camenisch's notation, P and V will engage in the Σ-protocol:

$$PK\{(\alpha, \beta, \gamma, \delta) : X_1 = (g)^\alpha (h)^\beta \wedge X_2 = (g)^\gamma (h)^\delta\}. \tag{1}$$

The above notation denotes a protocol where P proves to V that it knows values for α, β, γ and δ subject to the two equations on the right. However, U can not reveal any information about α, β, γ or δ during the proof.

Camenisch actually proposes the following protocol

$$PK\{(\alpha, \beta, \gamma, \delta) : X_1^2 = (g^2)^\alpha (h^2)^\beta \wedge X_2^2 = (g^2)^\gamma (h^2)^\delta\}. \tag{2}$$

It is clear that the two protocols are equivalent. It may be possible to execute the latter protocol faster, but I will focus on the former since it is slightly simpler to describe.

The Σ-protocol is executed over three rounds – commitment, challenge and response. During the commitment round, P chooses four new random integers R_1, R_2, R_3, R_4. P then commits to these random values by setting

$$\begin{aligned} t_1 &= g^{R_1}, && (3)\\ t_2 &= h^{R_2}, && (4)\\ t_3 &= g^{R_3}, && (5)\\ t_4 &= h^{R_4}. && (6) \end{aligned}$$

P sends t_1, t_2, t_3 and t_4 to V.

Now the challenge round begins, which is quite simple: V chooses four random challenges k_1, k_2, k_3 and k_4, then sends them all to P.

Finally, the response round begins. P sets the following values:

$$\begin{aligned} s_1 &= R_1 + ak_1 + ak_2, && (7)\\ s_2 &= R_2 + bk_1 + bk_2, && (8)\\ s_3 &= R_3 + ck_3 + ck_4, && (9)\\ s_4 &= R_4 + dk_3 + dk_4. && (10) \end{aligned}$$

P sends all values to V, who can now check whether $g^{s_1}h^{s_2} \stackrel{?}{=} t_1(X_1^{k_1})t_2(X_1^{k_2})$ and $g^{s_3}h^{s_4} \stackrel{?}{=} t_3(X_2^{k_3})t_4(X_2^{k_4})$. V accepts the proof if both equations hold and rejects otherwise.

If P executed the protocol correctly, then V will accept the proof since:

$$\begin{aligned} (g^{s_1})(h^{s_2}) &= (g^{R_1+ak_1+ak_2})(h^{R_2+bk_1+bk_2})\\ &= (g^{R_1})(g^{ak_1})(g^{ak_2})(h^{R_2})(h^{bk_1})(h^{bk_2})\\ &= t_1(g^{ak_1})(h^{bk_1})t_2(g^{ak_2})(h^{bk_2})\\ &= t_1(g^a h^b)^{k_1} t_2(g^a h^b)^{k_2}\\ &= t_1(X_1^{k_1})t_2(X_1^{k_2}) \end{aligned} \tag{11}$$

A similar series of substitutions holds for the other equation in question.

I will now argue that if V accepts the proof, then U must know a and b. If V accepts the proof, then P has produced an s_1 and s_2 that satisfy $g^{s_1}h^{s_2} = t_1(X_1^{k_1})t_2(X_1^{k_2})$. From this, both P and V know $s_1 = R_1+ak_1+ak_2$. Since P knows s_1, R_1 and k_1 by construction, P must also know a (even if P forgot a, it could derive it from this equation). A similar argument shows U must know b.

Finally, I claim the above protocol does not reveal any information about a to V. As stated before, V knows $s_1 = R_1 + ak_1 + ak_2$, but it is not able to recover a since it does not know R_1. Furthermore, V is not able to easily recover R_1 from t_1 due to the discrete logarithm problem. Similar arguments hold for the remaining values b, c and d.