

Enhanced Anonymous Tag Based Credentials

Shinsuke Tamura*, Shuji Taniguchi

Graduate School of Engineering, University of Fukui, Fukui, Japan

*Corresponding author: tamura@dance.plala.or.jp

Received January 20, 2014; Revised March 09, 2014; Accepted March 17, 2014

Abstract A scheme for anonymous tag based anonymous credentials is enhanced. Different from existing zero knowledge proof based anonymous credential schemes that require numbers of challenges and responses between verifiers and credential holders, the anonymous tag based scheme requires only small number of challenges and responses between verifiers and credential holders. This is because the scheme in this paper can be easily made sound even in environments where verifiers also may behave dishonestly. However, the original scheme has probabilistic features, i.e. verifiers must generate dummy fake challenges; therefore overheads for managing anonymous systems cannot be reduced to the minimum. The enhanced anonymous tag based scheme excludes these probabilistic features from interactions between verifiers and credential holders. The scheme also corrects several design errors included in the original scheme.

Keywords: information security, privacy, anonymous services

Cite This Article: Shinsuke Tamura, and Shuji Taniguchi, "Enhanced Anonymous Tag Based Credentials." *Information Security and Computer Fraud*, vol. 2, no. 1 (2014): 10-20. doi: 10.12691/iscf-2-1-3.

1. Introduction

Credentials are ones that maintain attribute values and enable their holders to convince others that they are eligible for receiving services corresponding to the attribute values. If a credential, which includes the age of a holder as its attribute value, is issued to a person by the government agency, the person can buy whisky from a liquor shop provided that the age in the credential is over 19 for example. Here, an anonymous credential includes a secret of its holder, and only an entity that knows the secret can show the ownership of the credential. In addition, the holder can prove that it knows the secret in the credential without disclosing the secret itself; also the validity of the credential can be verified even if its forms are changed from the one that the authority had issued. Therefore, if the authority gives an anonymous credential to an entity after confirming its eligibility, the entity can convince verifiers which provide services that it is eligible without disclosing its identity by proving that it knows the secret in the credential, i.e. anonymous credential holders can receive services from verifiers while preserving their privacies.

To implement the above anonymous credentials various schemes had been proposed already, and although schemes based on blind signature [1] have limitations, schemes based on zero knowledge proof (ZKP) [2,3] successfully satisfy all primary requirements of anonymous credentials [4-12], e.g. they enable verifiers also to identify dishonest entities while preserving privacies of honest credential holders. However, existing ZKP based anonymous credentials are not practical enough because numbers of challenges and responses

between verifiers and credential holders are required to verify validities of credentials.

In detail, in existing ZKP based anonymous credential schemes, provided that the secret in credential T is R , firstly credential holder P imbeds $f(R)$ instead of R in credential T to conceal R from others. Then at a time when verifier V verifies T , P calculates $g(r)$ while generating its secret integer r to show it with T that includes $f(R)$, and V generates its secret integer q as a challenge. After that, P calculates response $h(R, r, q)$ based on R , r and challenge q , and finally V , which knows $f(R)$ in T and had received $g(r)$ from P , examines whether relation $F(f(R), g(r), q) = G(h(R, r, q))$ holds or not. Here, functions $f(R)$, $g(r)$, $h(R, r, q)$, $F(f(R), g(r), q)$ and $G(h(R, r, q))$ are defined so that V cannot know R or r from $f(R)$, $g(r)$ or $h(R, r, q)$. On the other hand, P cannot calculate $h(R, r, q)$ that makes $F(f(R), g(r), q)$ and $G(h(R, r, q))$ equal without knowing R . Therefore P can convince V that it knows R if $F(f(R), g(r), q)$ coincides with $G(h(R, r, q))$ without disclosing R itself.

But, to disable others to link responses that P calculated at its different verification requests of same credential T , integer r in $h(R, r, q)$ is defined by credential holder P . Therefore, if P successfully estimates integer q , it can find pair $\{R^*, r^*\}$ that satisfies relation $F(f(R), g(r^*), q) = G(h(R^*, r^*, q))$ even if it does not know R , also P can easily estimate q if it is conspiring with V . Then, to disable conspiring entities and verifiers to legitimately carry out credential verification procedures without knowing secrets in credentials, V must generate numbers of challenges, i.e. P must calculate a lot of responses based on its secret information each time when it requests services.

To make anonymous credentials practical enough while reducing the number of challenges and responses between verifiers and credential holders, this paper proposes an

enhanced anonymous tag based credential scheme that disables entities to legitimately carry out credential verification procedures without knowing secrets in credentials even if they are conspiring with verifiers. Here, although the original anonymous tag based credential scheme had reduced numbers of challenges and responses [13], several parts of the scheme include probabilistic features, i.e. a credential holder must respond to several dummy fake challenges made by a verifier to generate evidences that the holder had certainly used the credential. Therefore, overheads for handling anonymous information cannot be reduced to the minimum. The enhanced scheme excludes these probabilistic features from interactions between verifiers and credential holders, and it also corrects several design errors included in the original scheme. As a consequence, the enhanced scheme becomes an efficient and secure component for developing anonymous service systems. For example, more efficient and secure anonymous payment, procurement, voting systems, etc. can be developed based on the proposed scheme.

2. Requirements for Anonymous Credentials

In the remainder, S , V and P represent a credential issuer, a verifier, and a credential holder respectively, namely S issues credentials to credential holders, V verifies validities of credentials shown by credential holders, and P shows its credential to receive services from V . Then, by showing anonymous credential T issued by issuer S , credential holder P can convince any entity (verifier) that it is eligible while preserving its privacy, i.e. anonymous credential T satisfies the following requirements [10].

R1. Unforgeability No one other than issuer S can generate valid credentials, in other words, credential holder P can obtain credential T only from S .

R2. Soundness and non-transferability Credential T is sound, namely entities that do not know secrets imbedded in T cannot prove their ownership of T . Here, it must be noted that this soundness must be satisfied even when the entities are conspiring with verifiers. Also this soundness does not automatically mean that the entity that can prove the ownership is only credential holder P , i.e. entities other than P also can prove the ownership if P informs them of the secrets in T . Non-transferability is the intensified notion of the soundness. Namely, P cannot inform others of secrets in its non-transferable credential T , but it seems impossible to disable P to leak its secrets to others. Therefore this requirement is restated as T can discourage P from informing others of its secrets, and the proposed scheme satisfies this requirement by limiting the number of times that entities can use same credential T .

R3. Anonymity No one except P can identify P from forms of credential T that P shows.

R4. Unlinkability Even if P uses its credential T repeatedly, no one except P can link repeatedly used T .

R5. Revocability S can invalidate credential T , if its holder P behaves or behaved dishonestly while showing T or if S reissued a new credential to P as a replacement of T (because P had forgotten its secrets necessary for using T for example). The proposed scheme satisfies this

requirement while preserving privacies of other credential holders by 2 mechanisms. The 1st mechanism identifies credential holders that had received services while using invalid credentials later, and the 2nd one denies service requests accompanied by invalid credentials.

In addition to the above, to enable any entity to verify the validity of credentials and to maintain attribute values as secrets of credential holders, anonymous credentials must satisfy the following 2 requirements also.

R6. Verifier V can verify the validity of credential T without knowing any secret of S .

R7. No one except P can know exact values of attributes in P 's credential T from its forms shown by P .

Enhanced anonymous tag based credential scheme proposed in this paper successfully and efficiently satisfies all of these requirements as discussed in the following sections.

3. Anonymous Tag Based Credentials

3.1. Anonymous Tags

An anonymous tag consists of a pair of a tag and an associate tag parts as shown in Figure 1 [14], i.e. tag holder P defines integer T_P and put $T_{P \bmod B}$ and $T_{P \bmod B}^R$ in the tag part and the associate tag part respectively. Here, integer R is a secret of P , and B is an appropriate integer common to all tags and large enough. Under these settings, tag $\{T_{P \bmod B}, T_{P \bmod B}^R\}$ is transformed by multiple entities S_1, S_2, \dots as below. In the remainder notation $\bmod B$ is omitted when confusions can be avoided.

$T_{P \bmod B}$ Tag part	$T_{P \bmod B}^R$ Associate tag part
-----------------------------	---

Figure 1. Anonymous tag

1. P generates its secret integer W and transforms tag $\{T_P, T_P^R\}$ to $\{T_P^W, T_P^{RW}\}$ to forward it to other entity S_1 .
2. S_h that receives tag $\{T_P^{WK^{*(h-1)}}, T_P^{RWK^{*(h-1)}}\}$ from S_{h-1} generates its secret integer K_h and transforms it to $\{T_P^{WK^{*(h-1)}(K_h)}, T_P^{RWK^{*(h-1)}(K_h)}\} = \{T_P^{WK^{*(h)}}, T_P^{RWK^{*(h)}}\}$ to forward the result to other entity S_{h+1} (here, $K_*(h)$ represents product $K_1 K_2 \dots K_h$, and as an exception S_1 receives $\{T_P^W, T_P^{RW}\}$ form P).

Namely, anonymous tag $\{T_P, T_P^R\}$ changes its form as $\{T_P^W, T_P^{RW}\}, \{T_P^{W(K_1)}, T_P^{RW(K_1)}\}, \{T_P^{W(K_1)K_2}, T_P^{RW(K_1)K_2}\}$ and $\{T_P^{W(K_1)K_2}K_3}, T_P^{RW(K_1)K_2}K_3}\}$ in this order when it is transformed by integers K_1, K_2 and K_3 that are secrets of 3 entities S_1, S_2 and S_3 . Then, Proposition 1 ensures that anonymous tag $\{T_P, T_P^R\}$ satisfies the following 3 conditions.

C1. Anyone except tag owner P cannot identify P from tag forms $\{T_P^W, T_P^{RW}\}$ or $\{T_P^{WK^{*(h)}}, T_P^{RWK^{*(h)}}\}$.

C2. Anyone except P cannot know that $\{T_P^W, T_P^{RW}\}, \{T_P^{WK^{*(1)}}, T_P^{RWK^{*(1)}}\}, \{T_P^{WK^{*(2)}}, T_P^{RWK^{*(2)}}\}, \dots$ are the ones transformed from the same tag, unless all relevant entities conspire with each other.

C3. P can identify that $\{T_P^{WK^{*(h)}}, T_P^{RWK^{*(h)}}\}$ is generated from its tag $\{T_P, T_P^R\}$.

Proposition 1

Under the strong RSA and the decisional Diffie-Hellman (DDH) assumptions, only entities that know R or

W can identify the correspondence between $\{T_P, T_P^R\}$ and $\{T_P^W, T_P^{RW}\}$ provided that integer B is sufficiently large. Under the same assumptions, only entities that know R or all K_1, K_2, \dots, K_h can identify the correspondence between $\{T_P^W, T_P^{RW}\}$ and $\{T_P^{WK^*(h)}, T_P^{RWK^*(h)}\}$.

Proof

An entity that knows W can identify the correspondence between $\{T_P, T_P^R\}$ and $\{T_P^W, T_P^{RW}\}$ by simply calculating $\{T_P^W, T_P^{RW}\}$ from $\{T_P, T_P^R\}$ by using W, also if an entity knows R, it is easy to know that $\{T_P^W, T_P^{RW}\}$ and $\{T_P^{WK^*(h)}, T_P^{RWK^*(h)}\}$ are generated from $\{T_P, T_P^R\}$, i.e. associate tag part values T_P^{RW} and $T_P^{RWK^*(h)}$ in $\{T_P^W, T_P^{RW}\}$ and $\{T_P^{WK^*(h)}, T_P^{RWK^*(h)}\}$ are calculated as T_P^W and $T_P^{WK^*(h)}$ to the power of R.

On the other hand, under the strong RSA and DDH assumptions, it is computationally infeasible to calculate R or W from $T_P^R \pmod B$ or $T_P^W \pmod B$ even if T_P is known. Therefore, entities that do not know R or W cannot know that $\{T_P^W, T_P^{RW}\}$ is generated from $\{T_P, T_P^R\}$. In the same way, an entity that does not know one of K_1, K_2, \dots, K_h cannot know that $\{T_P^{WK^*(h)}, T_P^{RWK^*(h)}\}$ is generated from $\{T_P^W, T_P^{RW}\}$ without knowing R. Q.E.D.

3.2. Anonymous Credentials Based on Anonymous Tags

Despite tag $\{T_P^W, T_P^{RW}\}$ is anonymous, entity P in the above can convince any entity V that P is the holder of $\{T_P^W, T_P^{RW}\}$ without disclosing its secret integer R along the scheme of Diffie and Hellman [15] as below.

1. V generates its secret integer X and calculates $(T_P^W)^X$ and $(T_P^{RW})^X$ and shows $(T_P^W)^X$ to P.
2. P calculates $\alpha = (T_P^{WX})^R$ by using its secret integer R.
3. V convinces itself that P is the holder of $\{T_P^W, T_P^{RW}\}$ if $\alpha = (T_P^{RW})^X$.

Table 1. List of Symbols

T_P	a publicly known integer defined by S and unique to a credential issued to P
T_A	a publicly known integer defined by S and unique to attribute A
$S(d, x)$	RSA signing function of S, i.e. $S(d, x) = x^d \pmod B$ and d is a secret signing key of S
$S(d_1 \parallel d_2, x)$	signature pair $\{S(d_1, x) = x^{d_1}, S(d_2, x) = x^{d_2}\}$
B	a publicly known integer constructed as a product of S's 2 large secret prime numbers
R, w, W	integers that are secrets of P
k, c	publicly known integers defined by S and common to all credentials
C_w, K_w	$C_w = c^w \pmod B, K_w = k^w \pmod B$
U(n)	an unique integer assigned to n-th service requests of all credential holders as a common base of used seals
U(V, m)	an unique integer that is assigned to the m-th service of V as a base of a used seal

Namely, when $(T_P^W)^X$ is given P that knows R can easily calculate α that coincides with $(T_P^{RW})^X$, i.e. $(T_P^{WX})^R = (T_P^{RW})^X$. On the other hand as in the previous section, for an entity that does not know R it is computationally infeasible to know R from T_P^W, T_P^{RW} and T_P^{WX} , as a consequence an entity that does not know R or X cannot calculate α that coincides with $(T_P^{RW})^X$ from $(T_P^W)^X$ or T_P^{RW} . Therefore, if attribute values are not considered, anonymous credentials can be constructed from anonymous tags when mechanisms that disable entities to forge or modify tags are established as discussed in the

remainder of this subsection. Table 1 shows symbols used in this section for constructing enhanced anonymous tag based credentials (some of the symbols will be used in later sections).

Let T_P, k and c be integers defined by issuer S, and R and w be secret integers defined by credential holder P. Then, provided that B is a publicly known integer that is constructed as a product of sufficiently large S's secret 2 prime numbers, d_1 and d_2 are 2 secret signing keys of S and $S(d_1 \parallel d_2, x)$ is a pair of RSA signatures $\{S(d_1, x) = x^{d_1} \pmod B, S(d_2, x) = x^{d_2} \pmod B\}$, signature pair $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R \pmod B)$ is an anonymous tag based anonymous credential generated by S and given to P. Where, T_P, k and c are publicly known and they are defined so that to know integers $q_1, q_1^*, q_2, q_2^*, q_3, q_3^*$ that satisfy relations $T_P = k^{q_1}, T_P^{q_1^*} = k, k = c^{q_2}, k^{q_2^*} = c, c = T_P^{q_3}, c^{q_3^*} = T_P$ is computationally infeasible for entities other than S. Also, different from k and c that are common to all credentials, T_P and R are unique to credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$, and T_P and T_{P^*} that are assigned to different credential holders P and P* are defined so that knowing integers q_4 and q_4^* that satisfy $T_P = T_{P^*}^{q_4}, T_P^{q_4^*} = T_{P^*}$ is computationally infeasible for entities other than S. Regarding K_w and C_w , P calculates them as $K_w = k^w \pmod B$ and $C_w = c^w \pmod B$ based on publicly known k, c and same secret integer w.

Then, credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ is valid when $S(d_1, T_P^{R+1} K_w C_w^R)$ and $S(d_2, T_P^{R+1} K_w C_w^R)$ are consistent signatures on same value $L^{u+1} k^u c^{uv}$ for some integers $\{L, u, v\}$. Here, credentials must be defined as signature pairs; if they are constituted as simple signatures, anyone can calculate $Q_1 = S(d_1^*, L^{v+1} k^u c^{uv})$ (d_1^* is a publicly known verification key) while generating integers L, u and v arbitrarily, and claim that $L^{v+1} k^u c^{uv}$ is a consistent signature, i.e. it is a signature on $Q_1 = (L^{(v+1)} k^u c^{uv})^{d_1^*} \pmod B = (L^{d_1^*})^{(v+1)} (k^{d_1^*})^u (c^{d_1^*})^{uv} \pmod B$ (in other words, $S(d_1, Q_1) = S(d_1, S(d_1^*, L^{v+1} k^u c^{uv})) = L^{v+1} k^u c^{uv}$). Signature pairs disable entities to dishonestly construct credentials in this way. About signing keys d_1 and d_2 , their confidentiality can be maintained despite 2 verification keys d_1^* and d_2^* are disclosed because the signer is only S, i.e. other entities do not know any signing key.

Also, because integer B is a product of S's 2 secret large prime numbers, anyone other than S cannot find integer pair $\{y, y^*\}$ that satisfies relation $T^{yy^*} \pmod B = T \pmod B$ for a given integer T. Therefore, P cannot decompose $T_P^{R+1} K_w C_w^R$ into $\{T^*, T^{y^*-1}, K_{w^*}, C_{w^*}^{y-1}\}$ ($T^* \neq T_P$) while choosing integer X arbitrarily, calculating K_{w^*} and C_{w^*} as $K_{w^*} = k^X$ and $C_{w^*} = c^X$ and defining $T^* = \{T_P^{R+1} K_w C_w^R / (K_{w^*} C_{w^*}^{y-1})\}^{y^*}$ so that relation $T^* (y^*-1) + 1 K_{w^*} C_{w^*}^{y-1} = \{T_P^{R+1} K_w C_w^R / (K_{w^*} C_{w^*}^{y-1})\}^{y^*} (K_{w^*} C_{w^*}^{y-1}) = T_P^{R+1} K_w C_w^R$ holds and pairs $\{K_{w^*}, C_{w^*}\}$ and $\{T^{y^*-1}, C_{w^*}^{y-1}\}$ are calculated as k and c to the power of same X and T* and C_{w^*} to the power of same (y-1) respectively.

Uniqueness of P's secret integer R can be maintained as below. Namely to generate integer R, firstly P asks multiple mutually independent authorities S_1, S_2, \dots, S_H to generate their secret integers r_1, r_2, \dots, r_H and calculate $Z^1 \pmod B, Z^2 \pmod B, \dots, Z^H \pmod B$ to inform issuer S of them, where Z is a publicly known integer that is common to all credentials. After that, S calculates $Z^1 Z^2 \dots Z^H = Z^{1+r_2+\dots+r_H} = Z^R$, and if Z^R did not appear before, each S_h informs P of its secret integer r_h so that P can calculate $R = r_1 + r_2 + \dots + r_H$, but when Z^R appeared already, S_1, S_2, \dots, S_H

generate other secret integers. Then, because Z is common to all credentials, different integers are assigned to different credentials. On the other hand, P can make R confidential unless all S_1, S_2, \dots, S_H conspire, because any entity other than S_h cannot calculate r_h from Z^h .

By using the above Z^R , S can also confirm that P had honestly included R in credential $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ at a time when it issues the credential. Namely, if S asks P to calculate Z^R again P can calculate it correctly only when the credential includes R as will be discussed in Sec. 3.3 where credential holders are forced to honestly calculate used seals of their anonymous credentials. Here, R must be constructed by multiple independent entities as above. If P itself generates secret integer R and discloses Z^R to make R unique, issuer S can know R when other entity P_* shows same value Z^R by chance after P had disclosed it, provided that S and P_* are conspiring and P_* informs S of R .

S issues anonymous credential $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ to credential holder P through the following procedure, and Figure 2 depicts the procedure.

Credential issuing procedure

1. After verifying eligibility of P based on P 's identity, S generates integer T_P to inform P of it together with integers k and c that are common to all credentials.

2. P generates its secret integers w and R , and calculates $K_w = k^w$, $C_w = c^w$ and pair T_P^R and C_w^R to send them to S , where $\{T_P, T_P^R\}$ constitutes an anonymous tag.

3. S generates its secret integers $\{\Gamma_1, \Gamma_2, \Gamma_3\}$ and $\{\Psi_1, \Psi_2, \Psi_3\}$ to calculate triplets $\{k^{\Gamma_1}, c^{\Gamma_2}, (kc)^{\Gamma_3}\}$, $\{K_w^{\Gamma_1}, C_w^{\Gamma_2}, (K_w C_w)^{\Gamma_3}\}$, $\{T_P^{\Psi_1}, C_w^{\Psi_2}, (T_P C_w)^{\Psi_3}\}$ and $\{(T_P^R)^{\Psi_1}, (C_w^R)^{\Psi_2}, ((T_P C_w)^R)^{\Psi_3}\}$ and shows $\{k^{\Gamma_1}, c^{\Gamma_2}, (kc)^{\Gamma_3}\}$ and $\{T_P^{\Psi_1}, C_w^{\Psi_2}, (T_P C_w)^{\Psi_3}\}$ to P .

4. P calculates $\beta_1 = (k^{\Gamma_1})^w$, $\beta_2 = (c^{\Gamma_2})^w$, $\beta_3 = ((kc)^{\Gamma_3})^w$, $\delta_1 = (T_P^{\Psi_1})^R$, $\delta_2 = (C_w^{\Psi_2})^R$ and $\delta_3 = ((T_P C_w)^{\Psi_3})^R$.

5. If relations $\beta_1 = K_w^{\Gamma_1}$, $\beta_2 = C_w^{\Gamma_2}$, $\beta_3 = (K_w C_w)^{\Gamma_3}$, $\delta_1 = (T_P^R)^{\Psi_1}$, $\delta_2 = (C_w^R)^{\Psi_2}$ and $\delta_3 = ((T_P C_w)^R)^{\Psi_3}$ hold, S generates credential $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ and gives it to P .

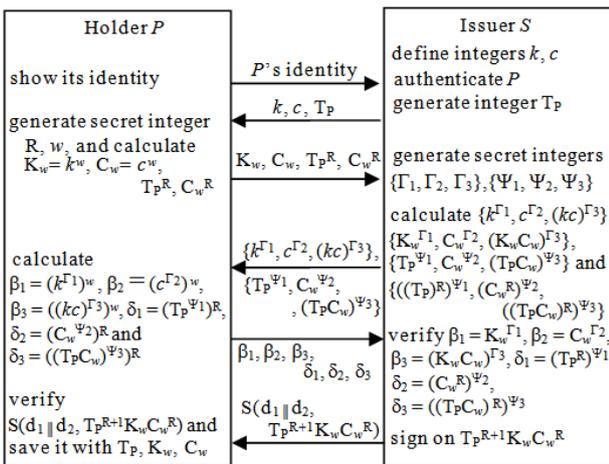


Figure 2. Issuing anonymous credentials

Here, it must be noted that although P discloses T_P^R, k^w, c^w and C_w^R , S cannot know R or w under the strong RSA assumption. Nevertheless, S can force P to calculate K_w and C_w as k and c to the power of same w through the scheme of Diffie and Hellman [15], i.e. Proposition 2 holds. In the same way, S can confirm that T_P^R and C_w^R are integers that coincide with T_P and C_w to the power of

same unknown integer R . Lastly as stated in Proposition 3, the above credential issuing procedure ensures that entities can obtain consistent credentials only from issuer S when they are eligible provided that S is honest. Of course S can issue credentials to even ineligible entities at its will, but these credentials bring losses only to S itself.

Proposition 2

In the credential issuing procedure, P must calculate pairs $\{K_w, C_w\}$ and $\{T_P^R, C_w^R\}$ as k and c and T_P and C_w to the power of same integers w and R respectively, although w and R are secrets of P .

Proof

Although P does not know Γ_3 , it can define ϕ arbitrarily and calculate γ and β_3 as $\gamma = (kc)^w / \phi$ and $\beta_3 = ((kc)^{\Gamma_3})^w$ respectively to consistently report ϕ and γ as values of K_w and C_w instead of k^w and c^w , i.e. relation $(\phi\gamma)^{\Gamma_3} = ((kc)^w)^{\Gamma_3} = ((kc)^{\Gamma_3})^w = \beta_3$ holds. However P , which does not know q_2 nor q_2^* that satisfy relations $k = c^{q_2}$ and $k^{q_2^*} = c$, cannot know at least one of integers σ_1 and σ_2 that satisfy $\phi = k^{\sigma_1}$ and $\gamma = c^{\sigma_2}$. As a consequence, P cannot calculate $\beta_1 = (k^{\Gamma_1})^{\sigma_1}$ and $\beta_2 = (c^{\Gamma_2})^{\sigma_2}$ so that both relations $\beta_1 = \phi^{\Gamma_1}$ and $\beta_2 = \gamma^{\Gamma_2}$ hold. In the same way, P must calculate T_P^R and C_w^R as T_P and C_w to the power of same integer R . Q.E.D.

Proposition 3 (Unforgeability)

Under the strong RSA assumption, entity P can obtain consistent credential $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ only from issuer S when it is eligible.

Proof

Firstly, S examines eligibility of P before entering the procedure, and signs on $T_P^{R+1} K_w C_w^R$ by using its secret keys d_1 and d_2 . Then through the legitimate way, P can obtain its credential only from S when it is eligible. On the other hand through illegitimate ways, P can declare any integer Q_* as the signature on pair $\{Q_1 = S(d_1, Q_*), Q_2 = S(d_2, Q_*)\}$, namely anyone knows public verification key pair d_{1*} and d_{2*} , and relations $Q_* = S(d_1, S(d_{1*}, Q_*))$ and $Q_* = S(d_2, S(d_{2*}, Q_*))$ hold. Also, from $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ and $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ issued to P and P_* , anyone can generate signature pair on $T_P^{R+1} T_P^{R+1} K_w K_w^R C_w^R C_w^R$ as product $S(d_1 || d_2, T_P^{R+1} K_w C_w^R) S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$ (it must be noted that $S(d_j, x) S(d_j, y) = S(d_j, xy)$, i.e. RSA signing functions are multiplicative).

However, in the former case P that does not know signing key pair $\{d_1, d_2\}$ cannot make Q_1 and Q_2 equal. About the latter case, consistent credentials must be decrypted to forms that are decomposed into the product of $\{L^{v+1}, k^u, c^{uv}\}$ for some integers L, u and v according to proposition 2. But anyone except S cannot know $q_1, q_1^*, q_2, q_2^*, q_3, q_3^*, g_1, g_1^*, g_3, g_3^*, q_4$ nor q_4^* even if P and P_* conspire (here, relations $T_P = k^{q_1}, T_P^{q_1^*} = k, k = c^{q_2}, k^{q_2^*} = c, c = T_P^{q_3}, c^{q_3^*} = T_P, T_P^* = k^{g_1}, T_P^{g_1^*} = k, c = T_P^{g_3}, c^{g_3^*} = T_P^*, T_P = T_P^{q_4}$ and $T_P^{q_4^*} = T_P^*$ are assumed). Therefore, under the strong RSA assumption and a condition where to find integer pair $\{y, y^*\}$ that satisfies relation $L^{y^*} = L$ for a given integer L is computationally infeasible, P or P_* cannot calculate pair $\{L, v\}$ that satisfies relation $L^{v+1} c^{uv} = T_P^{R+1} T_P^{R+1} K_w K_w^R C_w^R C_w^R / k^u$ for given u to make $T_P^{R+1} T_P^{R+1} K_w K_w^R C_w^R C_w^R$ consistent, in other words to decompose $T_P^{R+1} T_P^{R+1} K_w K_w^R C_w^R C_w^R$ into $\{L^{v+1}, k^u, c^{uv}\}$, either. Q.E.D.

After having obtained anonymous credential $S(d_1 || d_2, T_P^{R+1} K_w C_w^R)$, the credential verification procedure shown

in Figure 3 enables P to convince anyone that it is an eligible entity ensured by S without disclosing its identity nor linkages among its contiguous credential verification requests. Namely, Propositions 4-6 ensure that the procedure accepts P only when it knows integer R , and P can conceal R from others.

Credential verification procedure

1. P generates its secret integer W and calculates $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W = S(d_1 \| d_2, (T_P^{R+1} K_w C_w^R)^W)$ to show it to verifier V together with T_P^W, K_w^W, C_w^W and C_w^{RW} .

2. V examines whether $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W$ is the consistent signature pair on $(T_P^{R+1} K_w C_w^R)^W$ or not by using public verification key pair d_{1*} and d_{2*} . After that, V decomposes it into T_P^W, T_P^{RW}, K_w^W and C_w^{RW} based on the information given by P .

3. To confirm that P had calculated pairs $\{K_w^W = k^{wW}, C_w^W = c^{wW}\}$ and $\{T_P^{RW}, C_w^{RW}\}$ from integer pairs $\{k, c\}$ and $\{T_P^W, C_w^W\}$ while using P 's same secret integers wW and R respectively, V generates its secret integers $\{\Theta_1, \Theta_2, \Theta_3\}$ and $\{\Lambda_1, \Lambda_2, \Lambda_3\}$, calculates triplets $\{k^{\Theta_1}, c^{\Theta_2}, (kc)^{\Theta_3}\}$, $\{K_w^{W\Theta_1}, C_w^{W\Theta_2}, (K_w^W C_w^W)^{\Theta_3}\}$, $\{(T_P^W)^{\Lambda_1}, (C_w^W)^{\Lambda_2}, (T_P^W C_w^W)^{\Lambda_3}\}$ and $\{(T_P^{RW})^{\Lambda_1}, (C_w^{RW})^{\Lambda_2}, (T_P^{RW} C_w^{RW})^{\Lambda_3}\}$ and shows $\{k^{\Theta_1}, c^{\Theta_2}, (kc)^{\Theta_3}\}$ and $\{T_P^{W\Lambda_1}, C_w^{W\Lambda_2}, (T_P^W C_w^W)^{\Lambda_3}\}$ to P .

4. P calculates $\zeta_1 = (k^{\Theta_1})^{wW}, \zeta_2 = (c^{\Theta_2})^{wW}, \zeta_3 = ((kc)^{\Theta_3})^{wW}, D = (T_P^{W\Lambda_1})^R, \rho_1 = (C_w^{W\Lambda_2})^R$ and $\rho_2 = ((T_P^W C_w^W)^{\Lambda_3})^R$.

5. If relations $\zeta_1 = K_w^{W\Theta_1}, \zeta_2 = C_w^{W\Theta_2}, \zeta_3 = (K_w^W C_w^W)^{\Theta_3}, D = (T_P^{RW})^{\Lambda_1}, \rho_1 = (C_w^{RW})^{\Lambda_2}$ and $\rho_2 = (T_P^{RW} C_w^{RW})^{\Lambda_3}$ hold, V determines that P is an eligible entity that knows R .

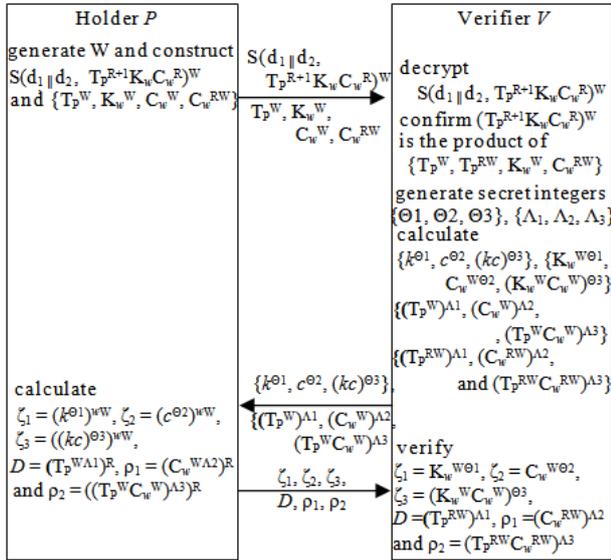


Figure 3. Verifying anonymous credentials

Proposition 4 (Soundness)

Under the strong RSA assumption, only entities that know integers w, W and R can prove the ownership of $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W$ to verifier V provided that V is honest.

Proof

It is apparent that entities that know w, W and R can calculate $\zeta_1, \zeta_2, \zeta_3, D, \rho_1$ and ρ_2 that coincide with $(k^{wW})^{\Theta_1}, (c^{wW})^{\Theta_2}, ((kc)^{wW})^{\Theta_3}, (T_P^{RW})^{\Lambda_1}, (C_w^{RW})^{\Lambda_2}$ and $(T_P^{RW} C_w^{RW})^{\Lambda_3}$ from triplets $\{k^{\Theta_1}, c^{\Theta_2}, (kc)^{\Theta_3}\}$ and $\{(T_P^W)^{\Lambda_1}, (C_w^W)^{\Lambda_2}, (T_P^W C_w^W)^{\Lambda_3}\}$ without knowing $\Theta_1, \Theta_2, \Theta_3, \Lambda_1, \Lambda_2$ or Λ_3 . On the other hand under the strong RSA assumption, any entity except ones that know w, W and R and verifier V

that knows $\Theta_1, \Theta_2, \Theta_3, \Lambda_1, \Lambda_2$ and Λ_3 cannot calculate at least one of $\zeta_1, \zeta_2, \zeta_3, D, \rho_1$ and ρ_2 consistently. Here, Proposition 6 ensures P decomposes $(T_P^{R+1} K_w C_w^R)^W$ into exactly T_P^W, T_P^{RW}, K_w^W and C_w^{RW} . Q.E.D.

Proposition 5 (Anonymity and unlinkability)

Under the strong RSA and DDH assumptions, no one except P can identify P from its showing credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W$. Provided that W_1, W_2, W_3, \dots are integers secrets of P , no one other than P can know $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_1}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_2}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_3}, \dots$ are different forms of a same credential either.

Proof

Under the strong RSA assumption, anyone except P cannot know T_P, T_P^R or R from any of $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_1}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_2}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_3}, \dots$, i.e. no one other than P itself can identify P from them. Also under DDH assumption, it is computationally infeasible to know correspondences among $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R), S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_1}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_2}, S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^{W_3}, \dots$, provided that W_1, W_2, W_3, \dots are secrets of P . Q.E.D.

Proposition 6

Under the strong RSA assumption, entity P that shows credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W$ can prove its ownership only when it calculates $D = (T_P^{W\Lambda_1})^R$ as $T_P^{W\Lambda_1}$ to the power of exactly R .

Proof

To use R^* instead of R ($R^* \neq R$) while satisfying relations $\zeta_1 = K_w^{W^*\Theta_1}, \zeta_2 = C_w^{W^*\Theta_2}, \zeta_3 = (K_w^{W^*} C_w^{W^*})^{\Theta_3}, D = (T_P^{R^*W^*\Lambda_1})^{\rho_1}, \rho_1 = (C_w^{R^*W^*\Lambda_2})^{\rho_2}$ and $\rho_2 = (T_P^{R^*W^*} C_w^{R^*W^*})^{\Lambda_3}$, P must decompose $(T_P^{R+1} K_w C_w^R)^W$ into $\{T_P^{W^*}, T_P^{R^*W^*}, K_w^{W^*}, C_w^{R^*W^*}\}$. In addition, pair $\{K_w^{W^*}, C_w^{W^*}\}$ must be calculated as k and c to the power of same integer w^*W^* . This means $T_P^{W^*(R^*+1)} = (T_P^{R+1} K_w C_w^R)^W / k^{w^*W^*} c^{w^*W^*R^*} = T_P^{(R+1)W} k^{wW-w^*W^*} c^{wWR-w^*W^*R^*} = L$. But because $R \neq R^*, L$ becomes a function of at least T_P and k or T_P and c (i.e. L is a function of at least T_P and k if $wW \neq w^*W^*$, and it is a function of T_P and c if $wW = w^*W^*$). Then, under the strong RSA assumption and in a condition where to find integer pair $\{y, y^*\}$ which satisfies relation $L^{y^*} = L$ is computationally infeasible, P that does not know $q_1, q_1^*, q_2, q_2^*, q_3$ nor q_3^* (it is assumed that $T_P = k^{q_1}, T_P^{q_1^*} = k, k = c^{q_2}, k^{q_2^*} = c, c = T_P^{q_3}$ and $c^{q_3^*} = T_P$) cannot find integers T^* and W^* that satisfy relation $T_P^{W^*(R^*+1)} = L$. Q.E.D.

Then, it is concluded that anonymous credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)$ satisfies the 1st requirement (unforgeability), a part of the 2nd requirement (soundness), and the 3rd and the 4th requirements (anonymity and unlinkability). Also verifier V does not need to know any secret of issuer S for verifying $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)^W$ because verification keys d_{1*} and d_{2*} are publicly known and $(T_P^{R+1} K_w C_w^R)^W$ is decomposed into T_P^W, T_P^{RW}, K_w^W and C_w^{RW} by credential holder P itself, i.e. the 6th requirement is also satisfied.

But it must be noted that Proposition 4 does not ensure the complete soundness of P 's credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)$, namely it cannot disable other entity P^* to use $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)$ as below.

Threat-1 P^* that steals $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R)$ can legitimately carry out the verification procedure without knowing secret R when it is conspiring with verifier V , i.e. if V informs P^* of integers $\Theta_1, \Theta_2, \Theta_3, \Lambda_1, \Lambda_2$ and Λ_3 , it is easy for P^* to calculate $\zeta_1, \zeta_2, \zeta_3, D, \rho_1$ and ρ_2 consistently.

Threat-2 P^* that is conspiring with other verifier V^* can impersonate P by exploiting decomposition $\{T_P^W, K_w^W,$

C_w^W, C_w^{RW} } that P had shown with $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$ at its past visit to V .

Threat-3 P_* can use $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ if P informs it of secret R , i.e. $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ is transferable.

Used seals constructed in the next subsection based on Proposition 6 enable developments of simple mechanisms that remove the above 3 threats and that make credentials also revocable. In addition, mechanisms in Sec. 7 enable credential holders to maintain attribute values as their secrets. Namely, the enhanced anonymous tag based scheme satisfies all requirements in Sec. 2.

About anonymity of credentials, when 2 entities P and P_* use their credentials $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$ and $S(d_1 \parallel d_2, T_{P_*}^{R_*+1} K_{w_*} C_{w_*}^{R_*})^{W_*}$ while defining integer pairs $\{w, W\}$ and $\{w_*, W_*\}$ in the way relation $wW = w_*W_*$ holds by chance, issuer S can detect this fact by duplicated appearances of $K_w^W = K_{w_*}^{W_*}$ ($= k^{wW}$) and if S is conspiring with P_* , it can know wW ($= w_*W_*$) by asking it to P_* . But S cannot identify P from wW , i.e. S cannot extract W from wW to calculate T_j^W for each T_j it had generated so that it can compare T_j^W with T_P^W that P shows together with $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$.

3.3. Used Seals of Anonymous Credentials

A used seal of anonymous credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ owned by credential holder P is defined as U^R , where verifier V defines base integer U and P calculates used seal U^R by using its secret R . Then, because R is unique and known only to P , verifier V or issuer S can use U^R as an evidence that P had certainly used $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ to receive services from V . Here, V cannot calculate R from U^R of course. In addition, although V does not know R , it can force P to honestly calculate U^R by extending the interactions in the credential verification procedure where V shows $T_P^{W\Lambda_1}$ and P calculates $D = (T_P^{W\Lambda_1})^R$ as below, i.e. by adding operations 3_* , 4_* and 5_* to steps 3, 4 and 5 in the credential verification procedure.

Then as a direct result of additions of 3_* , 4_* and 5_* , threat-1 about the soundness in the previous subsection can be excluded. Namely, even if entity P_* and verifier V are conspiring they cannot calculate used seal U^R that is consistent with $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$, and V itself must compensate accompanying losses as will be discussed in Sec. 4.

Proposition 7 ensures this extension forces P to calculate used seal U^R honestly.

Used seal generation procedure

3_* . V generates its secret integer Ω , to calculate $(UT_P^{W\Lambda_1})^{\Lambda_1\Omega}$ in addition to $T_P^{W\Lambda_1}$ and $T_P^{RW\Lambda_1}$, and shows it with U and $T_P^{W\Lambda_1}$ to P , where U is the base of the used seal and Λ_1 is the secret integer V generates at step 3 in the credential verification procedure.

4_* . P calculates $U_* = U^R$ and $B = \{(UT_P^{W\Lambda_1})^{\Lambda_1\Omega}\}^R$ in addition to $D = (T_P^{W\Lambda_1})^R$.

5_* . V examines whether relations $D = (T_P^{RW})^{\Lambda_1}$ and $B = U_*^{\Lambda_1\Omega} D^{\Omega}$ hold or not.

Proposition 7

Under the strong RSA assumption, P that uses credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ in the used seal generation procedure must calculate U_* , D and B honestly as $U_* = U^R$, $D = (T_P^{W\Lambda_1})^R$ and $B = \{(UT_P^{W\Lambda_1})^{\Lambda_1\Omega}\}^R$,

respectively from U , $T_P^{W\Lambda_1}$ and $(UT_P^{W\Lambda_1})^{\Lambda_1\Omega}$ shown by verifier V , provided that V is honest.

Proof

Proposition 6 ensures that P calculates D as $(T_P^{W\Lambda_1})^R$ based on its secret R , because P cannot prove the ownership of $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ if $D \neq (T_P^{W\Lambda_1})^R$. About U_* , if P calculates U_* as U^Q instead of U^R ($Q \neq R$), because relation $D = (T_P^{RW})^{\Lambda_1}$ is ensured, P must calculate B as $(U^Q)^{\Lambda_1\Omega} D^{\Omega}$. But to calculate $(U^Q)^{\Lambda_1\Omega} D^{\Omega}$, P must decompose $(UT_P^{W\Lambda_1})^{\Lambda_1\Omega}$ into $U^{\Lambda_1\Omega}$ and $(T_P^{W\Lambda_1})^{\Lambda_1\Omega}$, which is computationally infeasible for an entity that does not know Λ_1 or Ω under the strong RSA assumption. Q.E.D.

Here, P can calculate used seal U^Q instead of U^R dishonestly when it conspires with verifier V , but as will be discussed in the next section U^Q becomes inconsistent with any credential, as a result, V must compensate accompanying losses by itself.

As discussed in the following sections P calculates used seals of same credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ every time it shows the credential based on different base values $U(1)$, $U(2)$, $U(3)$, ---. But if S or V defines $U(i)$ as $U(j)^G$, S or V that knows G can have clues to identify the linkage between $U(i)^R$ and $U(j)^R$, i.e. relation $U(i)^R = U(j)^{RG}$ holds. This threat can be removed when bases of used seals are defined through cooperation among multiple independent authorities. Namely, when base $U(i)$ is defined as the sum or product of $U_1(i)$, $U_2(i)$, ---, $U_H(i)$ defined by independent authorities S_1, S_2, \dots, S_H , S or V cannot define $U(i)$ at its will without conspiring with all authorities.

4. Non-Transferability and Revocability

Used seals discussed in the previous section endow anonymous tag based credentials with non-transferability and revocability.

4.1. Mechanisms for Satisfying Non-Transferability

As same as threat-1 at the end of Sec. 3.2, threat -2 and 3 also can be excluded by using used seals. About threat-2, entity P_* , which is conspiring with verifier V_* and steals C_w^W and decomposition $\{T_P^W, T_P^{RW}, K_w^W, C_w^{RW}\}$ that P had shown with $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$ at its past visit to verifier V , can easily use credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ while changing its form to $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^{WW_*}$ ($= S(d_1 \parallel d_2, (T_P^{R+1} K_w C_w^R)^{WW_*})$). Namely, P_* that generates W_* and knows $\{T_P^W, T_P^{RW}, K_w^W, C_w^{RW}\}$ can extract $C_w^{WW_*}$ and consistently decompose $(T_P^{R+1} K_w C_w^R)^{WW_*}$ into $\{T_P^{WW_*}, T_P^{RW_*}, K_w^{WW_*}, C_w^{RW_*}\}$, and based on them, conspiring P_* and V_* can carry out the credential verification procedure legitimately. Here, P_* shows $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^{WW_*}$ instead of $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$ so that V_* will not be accused of having accepted same credential forms repeatedly even when the dishonest use of $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)$ is detected.

However, if P was required to calculate used seal $U(V, m)^R$ as an evidence that it had used credential $S(d_1 \parallel d_2, T_P^{R+1} K_w C_w^R)^W$ for receiving V 's m -th service, at a time when P_* requests V_* 's m -th service, either P_* or V_* cannot calculate used seal $U(V_*, m_*)^R$, because R is known only to P and P did not calculate $U(V_*, m_*)^R$ before. Here, issuer S defines $U(V, m)$ as an integer unique to V 's m -th

service in advance. Then, threat-2 is removed. In detail, at a time when P_* uses P_* 's credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$, it calculates an inconsistent value as the used seal or uses $U(V, m)^R$ that was calculated by P already, and in the former case V_* cannot identify the liable entity based on the inconsistent used seal as discussed in the next subsection (this means V_* cannot impute the liability to P). In the latter case, although $U(V, m)^R$ becomes consistent if $V_* = V$, V that had accepted the same used seal repeatedly is regarded as dishonest. As a consequence, dishonest V_* or V is forced to compensate corresponding losses by itself.

In environments where all verifiers are connected to issuer S through online communication channels, the above $U(V, m)$ that is bound to verifier V can be replaced with integer $U(n)$ that is common to n -th service requests of all credential holders. Namely, provided that $U(n)^R$ is a used seal that P had calculated at its past visit to V , although P_* can show $U(n)^R$ to V_* as a consistent used seal even if V_* and V are not same because $U(n)$ is not bound to V , V_* can detect the multiple appearance of $U(n)^R$ by examining records in S 's database. In other words, if V_* accepts P_* 's showing $U(n)^R$, it is regarded as an entity responsible for the illegitimate use of $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$ that had accepted same used seal $U(n)^R$ repeatedly. This integer $U(n)$ can be used also for limiting the number of times that an entity can show a same credential as below.

To protect credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$ from threat-3, the enhanced anonymous tag based credential scheme limits the number of times that entities can use same credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$ so that credential holder P is discouraged from informing others of its secret R . Namely, in the credential verification procedure, P is requested also to declare n , the number of service requests that it had made before by using same credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$, and verifier V informs P of $U(n)$ so that P can calculate used seal $U(n)^R$, where $U(n)$ is an integer mentioned in the above. Then, because $U(n)^R$ is unique to pair $\{P, n\}$, V can disable P to use $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$ more than N -times, i.e. the multi-show restriction procedure shown below can be constructed.

Multi-show restriction procedure

1. P requests services from V while showing credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)^{W_n}$ together with integer n .
2. V examines whether $n \leq N$ or not, and rejects P 's request when $n > N$. If $n \leq N$, V shows $U(n)$ to P as a base of a credential used seal.
3. P calculates used seal $U(n)^R$ to report it to V .
4. V examines whether pair $\{n, U(n)^R\}$ exists in issuer S 's database already or not, and rejects P 's request if it exists. If $\{n, U(n)^R\}$ does not exist V accepts P 's request and registers pair $\{n, U(n)^R\}$ in S 's database.

Here, entities other than P cannot extract a sequence of used seals $U(1)^R, U(2)^R, U(3)^R$ -- that P had calculated of course because they do not know R . But if some verifiers are not connected to issuer S through online channels, V cannot know used seals memorized by other verifiers immediately, as a consequence, V or S detects excessive uses of credentials only later after used seals of all credential holders have been gathered from distributed verifiers. Nevertheless, S can identify credential holders liable for excessive uses as discussed just below.

Now, when discussions about threat-1, 2 and 3 are combined, V must ask P to calculate used seal $U(n)^R$ in online environments and 2 used seals $U(n)^R$ and $U(V, m)^R$

in offline environments at a time when P uses $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)$ for its n -th time as a request for V 's m -th service.

4.2. Mechanisms for Satisfying Revocability

Anonymous tag based credentials can be endowed with revocability in 2 ways. In the 1st way, issuer S identifies entities that behave or behaved dishonestly (e.g. entities that do not pay fees for services) later after they had successfully authenticated, of course while preserving privacy of honest entities, and in the 2nd way, S rejects service requests from entities that behaved dishonestly in the past or that are using credentials replaced with new ones.

To identify dishonest entities, when credential holder P requests the m -th service of verifier V while showing credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)^W$ for its n -th time, V memorizes $\{U(n), U(n)^R\}$ or $\{U(V, m), U(V, m)^R\}$ as a part of a service record. Here $U(n)$ and $U(V, m)$ are integers defined in the previous subsection, and $U(n)^R$ and $U(V, m)^R$ are used seals of credential $S(d_1\|d_2, T_P^{R+1}K_wC_w^R)^W$ calculated by P . Under these settings, issuer S can identify P if any dishonesty is detected later in the service record that includes $\{U(n), U(n)^R\}$ or $\{U(V, m), U(V, m)^R\}$ by the following procedures. The procedure below is for offline environments, i.e. for a case where a service record includes pair $\{U(V, m), U(V, m)^R\}$.

Dishonest entity identification procedure (for offline environments)

1. S informs each entity P_* of $U(V, m)$ in pair $\{U(V, m), U(V, m)^R\}$ included in the dishonest service record.
2. P_* calculates used seal $U(V, m)^{R*}$ from $U(V, m)$ and its credential $S(d_1\|d_2, T_{P_*}^{R*+1}K_wC_w^{R*})$.
3. S identifies P_* as the entity that is liable for the dishonest service record when $U(V, m)^{R*}$ coincides with $U(V, m)^R$ (i.e. when $R = R^*$).

In the above, P_* is forced to calculate $U(V, m)^{R*}$ honestly as discussed in Sec. 3.3. On the other hand, S cannot identify P_* or P_* 's past service requests from $U(V, m)^{R*}$ if P_* is honest, because P_* did not calculate $U(V, m)^{R*}$ before, i.e. different values are assigned to different service requests as bases of used seals. Here, although P can calculate a used seal that is different from $U(V, m)^R$ while showing a credential of other entity, S can detect this dishonesty if it asks P to calculate initial used seal U_0^R in addition to $U(V, m)^R$, where S asks all credential holders to calculate their initial used seals based on U_0 when it issues credentials. Also, S can reduce inconveniences that P_* must calculate used seals every time when dishonest records are detected by asking all credential holders to calculate used seals related to dishonest records at each end of its service period.

In a case where a dishonest service record includes pair $\{U(n), U(n)^R\}$, credential holder P_* is asked to calculate $U(n)^{R*}$. But P_* had calculated $U(n)^{R*}$ already if it made more than n requests before, therefore S can identify P_* as the credential holder that corresponds to the service record accompanied by $\{U(n), U(n)^{R*}\}$ despite P_* is honest. Random integer $r(P_*)$ that is secret of P_* copes with this inconvenience.

Dishonest entity identification procedure (for online environments)

1. S informs each entity P_* of $U(n)$ in pair $\{U(n), U(n)^R\}$ included in the dishonest service record.

2. P_* generate random secret integer $r(P_*)$, and provided that P_* shows its credential in form $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})^{W^*}$, calculates $U_{r(P_*)} = U(n)^{r(P_*)}$, $U_R = U(n)^{Rr(P_*)}$, $L_1 = T_{P_*}^{W^*r(P_*)}$ and $L_2 = T_{P_*}^{R^*W^*r(P_*)}$.

3. S examines whether relations $U_{r(P_*)} = U(n)^\pi$, $U_R = U(n)^{R\pi}$, $L_1 = T_{P_*}^{W^*\pi}$ and $L_2 = T_{P_*}^{R^*W^*\pi}$ hold for same unknown π or not, and generates its secret integers τ and ξ to calculate L_1^τ and $(U_{r(P_*)} L_1)^{\tau\xi}$.

4. P_* calculates $D_* = (L_1^\tau)^{R^*}$ and $B_* = \{(U_{r(P_*)} L_1)^{\tau\xi}\}^{R^*}$ together with $U_{R^*} = U_{r(P_*)}^{R^*}$.

5. S determines P_* is liable when one of relations $D_* = L_2^\tau$ and $B_* = U_{R^*}^{\tau\xi} D_*^\xi$ does not hold or $U_{R^*} = U(n)^{R^*r(P_*)}$ coincides with $U_R = U(n)^{Rr(P_*)}$ (i.e. $R^* = R$).

In the above, relations $D_* = L_2^\tau$ and $B_* = U_{R^*}^{\tau\xi} D_*^\xi$ correspond to $D = (T_P^{RW})^{\Lambda_1}$ and $B = U_*^{\Lambda_1 \Omega} D^\Omega$ in the used seal generation procedure, therefore P_* is forced to calculate $U_{R^*} = U_{r(P_*)}^{R^*} = U(n)^{R^*r(P_*)}$ honestly. Also, S can verify relations $U_{r(P_*)} = U(n)^\pi$, $U_R = U(n)^{R\pi}$, $L_1 = T_{P_*}^{W^*\pi}$, and $L_2 = T_{P_*}^{R^*W^*\pi}$ in the same way as pair $\{K_w, C_w\}$ in Sec. 3.2 was examined, and this means P_* calculates U_R as $U(n)^{Rr(P_*)}$. Then, S can conclude that P_* is liable for the dishonest record when U_{R^*} coincides with U_R , namely used seals $U(n)^{R^*r(P_*)}$ and $U(n)^{Rr(P_*)}$ of credentials $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})$ and $S(d_1 \| d_2, T_{P_*}^{R+1} K_w C_w^R)$ are same. On the other hand, P_* can disable others to identify its service requests, because $r(P_*)$ is secret of P_* and used seal $U(n)^{R^*r(P_*)}$ does not coincide with $U(n)^{R^*}$ even if P_* had calculated $U(n)^{R^*}$ at its past request.

In environments where online channels between issuer S and verifiers are available, S can also reject service requests from credential holder P that had behaved dishonestly in the past or that is using a credential replaced with a new one. Firstly, issuer S registers pair $\{U(n), U(n)^R\}$ in its invalid credential list when dishonesties are detected in the service record corresponding to $\{U(n), U(n)^R\}$. In a case where P had obtained a new credential as the replacement of $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})$, S constructs record $\{U(n), U(n)^R\}$ in the invalid credential list from initial used seal $\{U_0, U_0^R\}$ that P had left when S issued $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})$. Then at a time when entity P_* shows its credential $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})^{W^*}$ to verifier V_* , the following procedure enables V_* to reject P_* 's request if $P_* = P$ in the same way as the previous procedure identified dishonest entities.

Invalid credential rejection procedure

1. P_* that requests V_* 's service by showing credential $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})$ in form $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})^{W^*}$ calculates $U_{r(P_*)} = U(n)^{r(P_*)}$, $U_R = U(n)^{Rr(P_*)}$, $L_1 = T_{P_*}^{W^*r(P_*)}$ and $L_2 = T_{P_*}^{R^*W^*r(P_*)}$. Here, $r(P_*)$ is a secret integer generated by P_* .

2. V_* examines whether relations $U_{r(P_*)} = U(n)^\pi$, $U_R = U(n)^{R\pi}$, $L_1 = T_{P_*}^{W^*\pi}$ and $L_2 = T_{P_*}^{R^*W^*\pi}$ hold for same unknown π or not, and generates its secret integer τ and ξ to calculate L_1^τ and $(U_{r(P_*)} L_1)^{\tau\xi}$.

3. P_* calculates $D_* = (L_1^\tau)^{R^*}$ and $B_* = \{(U_{r(P_*)} L_1)^{\tau\xi}\}^{R^*}$ together with $U_{R^*} = U_{r(P_*)}^{R^*}$.

4. V_* accepts P_* 's request when relations $D_* = L_2^\tau$ and $B_* = U_{R^*}^{\tau\xi} D_*^\xi$ hold and $U_{R^*} = U(n)^{R^*r(P_*)}$ does not coincide with $U_R = U(n)^{Rr(P_*)}$ (i.e. R^* and R do not match).

Because P_* must calculate used seals $U(n)^{R^*r(P_*)}$ for each $\{U(n), U(n)^R\}$ in the invalid credential list at its

every service request, overheads accompanying the procedure cannot be ignored when the number of dishonest or secret forgetting entities is large. Therefore, the above procedure is practical only when the number of dishonest or secret forgetting credential holders is small. An example that satisfies this condition is the authentication of clients in cloud computing systems.

5. Features of Anonymous Tag Based Credentials

As discussed already, the proposed enhanced anonymous tag based anonymous credential scheme satisfies all requirements shown in Section 2 except the 7th one (the performance about this requirement will be discussed in Sec.7). A distinctive property of the scheme is it can easily satisfy the complete soundness of credentials even in environments where verifiers also may behave dishonestly. As a consequence when compared with existing ZKP based schemes, which require numbers of interactive or non-interactive challenges and responses between verifiers and credential holders, procedures in the proposed scheme require credential holder P to calculate only values $\zeta_1 = (k^{\Theta_1})^{wW}$, $\zeta_2 = (c^{\Theta_2})^{wW}$, $\zeta_3 = ((kc)^{\Theta_3})^{wW}$, $D = (T_P^{W\Lambda_1})^R$, $\rho_1 = (C_w^{W\Lambda_2})^R$, $\rho_2 = (T_P^{WC_w^W})^{\Lambda_3 R}$, $U_* = U(n)^R$ and $B = \{U(n)T_P^W\}^{\Lambda_1 \Omega}$, i.e. only 8 challenges and responses are necessary if all verifiers are connected to issuer S through online communication channels (when online channels are not available 10 challenges and responses are necessary because verifiers must ask P to calculate 2 used seals $U(n)^R$ and $U(V, m)^R$). Here, the number of interactions between verifiers and credential holders is limited to 1.5 rounds, i.e. verification requests from credential holders, challenges from verifiers and responses from credential holders. Therefore, anonymous tag based credentials enable developments of highly efficient and secure anonymous service systems.

When compared with the original scheme [13], probabilistic features in generating used seals are excluded in the enhanced anonymous tag based scheme. To force P , a holder of credential $S(d_1 \| d_2, T_{P_*}^{R^{*+1}} K_{w_*} C_{w_*}^{R^*})^W$, to honestly calculate used seal U^R , verifier V in the original scheme generates multiple secret integers X_1, X_2, \dots, X_M , and calculates $(T_P C_w)^{W(X_1)}$, $(T_P C_w)^{W(X_2)}$, \dots , $(T_P C_w)^{W(X_M)}$ to ask P to calculate $(T_P C_w)^{W(X_1)R}$, $(T_P C_w)^{W(X_2)R}$, \dots , $(T_P C_w)^{W(X_M)R}$ and $\{U(T_P C_w)^{WXY}\}^R$ by showing $\{(T_P C_w)^{W(X_1)}, (T_P C_w)^{W(X_2)}, \dots, (T_P C_w)^{W(X_M)}, U(T_P C_w)^{WXY}\}$ while randomly changing their positions. Namely, if P honestly calculates $\{U(T_P C_w)^{WXY}\}^R$, V can know U^R as $\{U(T_P C_w)^{WXY}\}^R / (T_P C_w)^{WXYR}$. On the other hand if P tries to calculate it dishonestly, it may calculate $(T_P C_w)^{W(X_q)}$ inconsistently because it cannot identify $U(T_P C_w)^{WXY}$ in $\{(T_P C_w)^{W(X_1)}, \dots, (T_P C_w)^{W(X_M)}, U(T_P C_w)^{WXY}\}$ and its requests is rejected as discussed in Sec. 3.2. But this means P can calculate the used seal dishonestly with probability $1/M$. As a consequence, V must generate a large number of integers X_1, X_2, \dots, X_M to make the probability $1/M$ small enough, which reduces the efficiency. Different from the original scheme, in the proposed scheme, the above probabilistic feature is excluded completely from interactions between verifiers and credential holders.

About the configuration of credentials, $S(d_1||d_2, T_P^{R+1}K_wC_w^R)^W$ in the proposed scheme is configured as $S(d, T_P^{R+Gg+G+1})^W$ in the original scheme, where integers G and g are defined by issuer S and common to all credentials, and G is publicly known and g is a secret of S . But $S(d, T_P^{R+Gg+G+1})^W$ cannot disable P to calculate used seal as U^{R^*} instead of U^R while using integer $R^* (\neq R)$. In the enhanced scheme, K_w^W and C_w^{RW} included in credential $S(d_1||d_2, T_P^{R+1}K_wC_w^R)^W$ disable P to dishonestly calculate used seals as shown in Proposition 7. Also, signature pair $S(d_1||d_2, x)$ completely disables entities to forge credentials.

6. Adding Attribute Values to Credentials

Attribute values can be attached to credentials discussed in Sec. 3 by simply constructing them with identity and attribute parts pairs as shown in Figure 4. Namely, provided that a is an value of attribute A , and T_A is a publicly known integer corresponding to attribute A , entity P can prove that it deserves attribute value a without disclosing its identity by calculating $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$ based on credential $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})$ given from S and its secret integer W , and by showing it together with $T_P^W, K_w^W, C_w^W, C_w^{RW}, T_{Aw}^W$ and T_{Aw}^{aW} . Here, $T_P^{R+1}K_wC_w^R$ constitutes the identity part and $T_P, K_w = k^w, C_w = c^w$ and R are defined as in Sec. 3.2. On the other hand, T_{Aw}^{a+1} constitutes the attribute part, and T_A is defined by S so that no one other than S can know integers $v_1, v_1^*, v_2, v_2^*, v_3, v_3^*$ that satisfy relations $T_A = T_P^{v_1}, T_A^{v_1^*} = T_P, T_A = k^{v_2}, T_A^{v_2^*} = k, T_A = c^{v_3}, T_A^{v_3^*} = c$, and P calculates T_{Aw} as T_A^w while using its secret integer w that it had used for calculating $K_w = k^w$ and $C_w = c^w$.

$T_P^{R+1}K_wC_w^R$ Identity part	T_{Aw}^{a+1} Attribute part
--------------------------------------	----------------------------------

Figure 4. Identity and attribute parts

The identity part in $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})$ plays the same roles as credential $S(d_1||d_2, T_P^{R+1}K_wC_w^R)$ in previous sections did, i.e. the identity part conceals the identity of credential holder P while ensuring P is eligible, therefore identity part $T_P^{R+1}K_wC_w^R$ must conceal R from anyone except P . On the other hand the purpose of attribute part T_{Aw}^{a+1} at this stage is not to conceal attribute value a , instead, the role of T_{Aw}^{a+1} is to disable P to use attribute value a illegitimately, i.e. it disables P to claim a^* that is different from a as the attribute value. Therefore different from integer R , attribute value a in this section is made publicly known.

Credential verification procedure that enables entities to determine whether P deserves attribute value a or not without knowing P can be constructed as below.

Attribute value verification procedure 1

1. P shows credential $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$ with $T_P^W, K_w^W, C_w^W, C_w^{RW}, T_{Aw}^W, T_{Aw}^{aW}$ and attribute value a to verifier V (W is P 's secret integer as same as in the credential verification procedure in Sec. 3.2).

2. V verifies signature pair $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$ and decomposes

$(T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$ into $T_P^W, T_P^{RW}, K_w^W, C_w^{RW}, T_{Aw}^W$ and T_{Aw}^{aW} based on the information given by P .

3. To determine whether P is an eligible holder of $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$ or not, V examines the consistency of decomposition $\{T_P^W, T_P^{RW}, K_w^W, C_w^{RW}\}$ as in steps 3-5 of the credential verification procedure.

4. V confirms that P had calculated T_{Aw}^W as T_A to the power of unknown w that was used for calculating $K_w^W = k^{wW}$ and $C_w^W = c^{wW}$, and determines that P deserves a when relation $T_{Aw}^{aW} = (T_{Aw}^W)^a$ holds.

From discussions in previous sections, it is apparent that the above procedure ensures P is the eligible holder of $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1})^W$, i.e. only P that knows R can use the credential. In addition, attribute part $(T_{Aw}^{a+1})^W$ disables P to declare attribute value a dishonestly as follow, i.e. when P decomposes $T_{Aw}^{(a+1)W}$ into $T_{Aw}^{w^*}$ and $T_{Aw}^{a^*w^*}$ while finding integers a^* and W^* that are different from a and W , although relation $T_{Aw}^{a^*w^*} T_{Aw}^{w^*} = T_{Aw}^{(a+1)W}$ may hold, V detects the dishonesty because $T_{Aw}^{w^*}$ and K_w^W are T_A and k to the power of different integers.

Credential $S(d_1||d_2, T_P^{R+1}K_wC_w^R)$ can have also multiple attribute values. For example, to add an expiration time as new attribute E to the above credential, S defines new integer T_E , P calculates $T_{Ew} = T_E^w$ and T_{Ew}^{e+1} based on attribute value e to construct the credential as $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1} T_{Ew}^{e+1})$. But it must be noted that verifier V must examine all attribute values included in a credential even when it does not require all attribute values. If attribute value e in $S(d_1||d_2, T_P^{R+1}K_wC_w^R T_{Aw}^{a+1} T_{Ew}^{e+1})^W$ is not examined for example, P that possesses an illegitimate credential can declare the value of $T_{Ew}^{(e+1)W}$ arbitrarily to make other parts of the credential consistent.

7. Concealing Attribute Values

In the previous attribute value verification procedure, credential holder P must disclose attribute value a to convince verifier V that it deserves a . However, attribute values may include clues to identify P , or in face to face environments, P may not want to disclose attribute values even if it is anonymous. Therefore sometimes anonymous credentials are required to conceal also attribute values. To conceal attribute values, this section constructs a mechanism which enables verifier V to confirm that relation $b < a$ (or $b > a$) holds for some value b it defines, of course without knowing a itself.

Before constructing the attribute value verification procedure in which credential holder P can conceal its attribute value a , it must be noted that P cannot show even pair $\{T_{Aw}^W, T_{Aw}^{aW}\}$, because usually relatively small number of values are assigned to an attribute (e.g. if an attribute is the age of a person, only about 100 values are effective), and V can know a from the pair by calculating $T_{Aw}^{Wa^*}$ from T_{Aw}^W for every possible attribute value a^* to compare the result with T_{Aw}^{aW} . To disable V to know a in this way, P constructs attribute values as pairs of real values and large dummy secret random values as shown in Figure 5. In the figure real attribute value a and large dummy value a' are located at the upper and the lower digits of attribute value \underline{a} , i.e. a' is defined so that relation $a' < max$ holds for some integer max . As a consequence,

issuer S issues credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$ instead of $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$ in the remainder.

Here, value \underline{a} also must be concealed from S , because if S knows \underline{a} , later it can identify \underline{a} even from pair $\{T_{Aw}^W, T_{Aw}^{aW}\}$ by calculating $T_{Aw}^{W\underline{a}}$ from T_{Aw}^W for every \underline{a} it had issued. P can conceal \underline{a} from others while convincing them that a in it is correct and a' in it is less than max by using reference credential $S(e_1 \| e_2, T_P^{Re} K_w C_w^{Re} T_{Aw}^{Q_1+1})$ that includes integer Q_1 as below, where the reference credential enables P to convince others that relation $max/2 < Q_1 < max$ holds while concealing Q_1 itself, and e_1 and e_2 in it are signing keys of S that are different from d_1 and d_2 , as will be discussed at the end of this section.

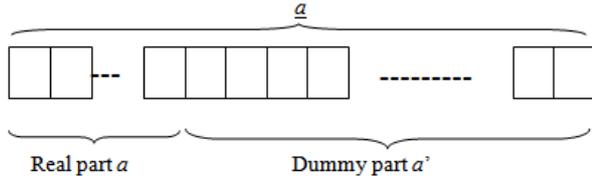


Figure 5. Real and dummy parts in an attribute value

Attribute value generation procedure

- P shows reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q_1+1})$ with $\{T_P^{Re}, K_w, C_w, C_w^{Re}, T_{Aw}, T_{Aw}^{Q_1}\}$ to S , and convinces S that decomposition $\{T_P, K_w, C_w, C_w^{Re}, T_{Aw}, T_{Aw}^{Q_1}\}$ of $T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q_1+1}$ is consistent.
- P generates random integer a' that satisfies relations $max/2 < a' < max$ and $a' < Q_1$, and calculates $T_{Aw}^a, T_{Aw}^{a'}$ and $T_{Aw}^{Q_1-a'}$ to show them together with $T_{Aw}^{Q_1}, a$ and $Q_1 - a'$ to S .
- S confirms that $Q_1 - a'$ is less than $max/2$, $T_{Aw}^{Q_1}$ is certainly calculated from reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q_1+1})$, $T_{Aw}^{Q_1-a'}$ equals to $T_{Aw}^{Q_1}/T_{Aw}^{a'}$, and T_{Aw}^a and $T_{Aw}^{Q_1-a'}$ are T_{Aw} to the power of a and $(Q_1 - a')$, respectively.
- S calculates $T_{Aw}^{a+1} = T_{Aw}^a T_{Aw}^{a'} T_{Aw} = T_{Aw}^{a+a'+1}$ to include it in credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$.

In the above, S issues credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$ while identifying P , therefore P can show reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q_1+1})$ without changing its form. Also P in step 3 can convince others that $T_{Aw}^{Q_1}$ is certainly included in reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q_1+1})$ because T_{Aw}, K_w and C_w are calculated as $T_{Aw} = T_A^w, K_w = k^w, C_w = c^w$ by same unknown integer w , and this ensures that $T_{Aw}^{Q_1}$ is calculated as T_{Aw} to the power of integer Q_1 that satisfies relation $max/2 < Q_1 < max$ although Q_1 is P 's secret as will be discussed later. Then, S can confirm that $a' < max$ without knowing a' as follows.

Namely, if relation $max/2 < a' < Q_1 < max$ holds, apparently $(Q_1 - a')$ becomes less than $max/2$ because $max/2 < Q_1 < max$ is ensured, but when $a' > max$ (this means $a' > Q_1$), $(Q_1 - a')_{\text{mod } B}$ is calculated as $(Q_1 - a')_{\text{mod } B} = B + (Q_1 - a') = Q_1 + (B - a')$, i.e. $(Q_1 - a')_{\text{mod } B}$ becomes greater than $max/2$ because $Q_1 > max/2$ and $B - a' > 0$. On the other hand, S cannot know the value of a' , because P discloses a' in the forms $(Q_1 - a')$, $T_{Aw}^{a'}$ and $T_{Aw}^{Q_1-a'}$. But it must be noted that P cannot assign a value that is less than $max/2$ to a' although relation $a' < max$ holds.

While using attribute values constructed by the above procedure, the following procedure enables entities to convince others that their attribute values are greater (or

less) than given values without disclosing the attribute values themselves. In the procedure, P convinces V that attribute value a in credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$ is greater than b that is designated by V , under the condition that all integers $\underline{a}, \underline{b}, Q$ and $\underline{a}^2, \underline{b}^2, Q^2$ are less than $B/2$, and although Q is P 's secret integer, P can convince V relations $Q < B/2$ and $Q^2 < B/2$ by reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$. Here, V defines \underline{b} from b in the same way as P constructs \underline{a} from a , but \underline{b} is not a secret value different from \underline{a} .

Attribute value verification procedure 2

- P generates its secret integers W and W^* , and from its credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})$ and reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$ calculates $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})^W$ and $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})^{W^*}$ to show them to V , with $\{T_P^W, K_w^W, C_w^W, C_w^{RW}, T_{Aw}^W, T_{Aw}^{aW}\}$ and $\{T_P^{W^*}, K_w^{W^*}, C_w^{W^*}, C_w^{ReW^*}, T_{Aw}^{W^*}, T_{Aw}^{QW^*}\}$.

- V decrypts $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})^W$ to $(T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})^W$, decomposes the result to $T_P^W, T_P^{RW}, K_w^W, C_w^{RW}, T_{Aw}^W$ and T_{Aw}^{aW} , and confirms that the decomposition is consistent. In the same way, it decrypts $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})^{W^*}$ to $(T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})^{W^*}$, decomposes the result to $T_P^{W^*}, T_P^{ReW^*}, K_w^{W^*}, C_w^{ReW^*}, T_{Aw}^{W^*}$ and $T_{Aw}^{QW^*}$, and confirms the decomposition is consistent.

- V generates \underline{b} and calculate $U^{\underline{b}}$, where U is the base of used seals.

- P generates used seals $U^{\underline{a}}$ from pair $\{T_{Aw}^W, T_{Aw}^{aW}\}$ in $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1})^{W^*}$ to calculate $U^{\underline{a}/U^{\underline{b}}} = U^{(\underline{a}-\underline{b})}$. P also calculates used seal $U^{(\underline{a}-\underline{b})Q}$ from pair $\{T_{Aw}^{W^*}, T_{Aw}^{QW^*}\}$ in $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})^{W^*}$ and base value $U^{(\underline{a}-\underline{b})}$, and discloses $U^{\underline{a}}, U^{(\underline{a}-\underline{b})Q}$ and $(\underline{a}-\underline{b})Q$.

- V confirms that $U^{(\underline{a}-\underline{b})Q}$ is certainly U to the power of $(\underline{a}-\underline{b})Q$, and determines that $a > b$ when $(\underline{a}-\underline{b})Q$ is less than $B/2$.

At step 4 in the above procedure, V can force P to honestly calculate used seals $U^{\underline{a}}$ and $U^{(\underline{a}-\underline{b})Q}$ as shown in Proposition 7. In addition, relations $\underline{a} < B/2, \underline{b} < B/2, Q < B/2, \underline{a}^2 < B/2, \underline{b}^2 < B/2$ and $Q^2 < B/2$ are ensured, where, S can confirm that $\underline{a} < B/2$ and $\underline{a}^2 < B/2$ at a time when it had issued the credential because P discloses a and relation $a' < max$ is ensured, and \underline{b} is generated by S itself. Conditions about Q are also ensured by reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$ as shown later. Therefore, $(\underline{a}-\underline{b})Q_{\text{mod } B}$ becomes less than $B/2$ if \underline{a} is greater than \underline{b} . On the other hand if \underline{a} is less than \underline{b} , $(\underline{a}-\underline{b})Q$ is calculated as $(\underline{a}-\underline{b})Q_{\text{mod } B} = B + (\underline{a}-\underline{b})Q$, as a consequence, $(\underline{a}-\underline{b})Q_{\text{mod } B}$ becomes greater than $B/2$. Then by relation $(\underline{a}-\underline{b})Q < B/2$, V can confirm that $\underline{a} > \underline{b}$ and as a consequence $a > b$.

Regarding the discussion at the end of Sec. 6 that verifier V must examine consistencies of all attribute values, based on Proposition 4, V can confirm consistencies of irrelevant attribute values without executing the above procedure. For example, in a case where V does not need to examine attribute value \underline{e} in credential $S(d_1 \| d_2, T_P^{R+1} K_w C_w^R T_{Aw}^{a+1} T_{Ew}^{e+1})^W$ owned by P , V can confirm that P honestly extracts $T_{Ew}^{(e+1)W}$ from the credential by asking P to prove that it knows \underline{e} included in $T_{Ew}^{(e+1)W}$. Here according to Proposition 5, P can maintain attribute value \underline{e} as its secret of course.

P can convince V that $Q < B/2$ and $Q^2 < B/2$ without disclosing Q by obtaining reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$ in the course of member registration procedure, in which issuer S accepts P as an eligible entity.

Here, S uses signing key e_1 and e_2 that are different from d_1 and d_2 to discriminate reference credentials from usual ones. Now P obtains reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$ without disclosing Q while convincing S that Q satisfies the above conditions through cut and choose protocol [16].

Reference credential issuing procedure

1. P calculates $T_A^{u_1} = G_1, T_A^{u_2} = G_2, \dots, T_A^{u_N} = G_N$ while generating a set of secret integers $\{u_1, \dots, u_N\}$ and shows G_1, G_2, \dots, G_N to S .

2. S chooses single integer G_q from $\{G_1, \dots, G_N\}$.

3. P discloses each u_j except u_q .

4. For each disclosed u_j , S examines relations $u_j < B/2$, $u_j^2 < B/2$ and $T_A^{u_j} = G_j$, and when the relations are satisfied, as the value of T_{Aw}^{Q+1} it includes $T_A^{u_{q+1}}$ in reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$.

In the above, apparently P can conceal $u_q (= Q)$ from S . On the other hand, although P can define u_q illegitimately, it must disclose u_q if S does not choose it, and this means S can convince itself that u_q satisfies the conditions $u_q < B/2$ and $u_q^2 < B/2$ with probability $(N-1)/N$. P can generate reference credential $S(e_1 \| e_2, T_P^{Re+1} K_w C_w^{Re} T_{Aw}^{Q+1})$ in the attribute value generation procedure in the same way. Here, although S must ask P to calculate a large number of integers $\{T_A^{u_1}, \dots, T_A^{u_N}\}$ to make the above probability large enough, this probabilistic feature appears only in member registration procedures, i.e. probabilistic features are excluded from credential verification processes. Then, the enhanced anonymous tag based credential scheme can maintain its efficiency because member registration procedures are carried out only once for each entity. Here, the above cut and choose protocol can be replaced by other ZKP protocols of course.

8. Conclusion

A scheme for anonymous credentials based on anonymous tags was enhanced to make it more secure and efficient. Namely, design errors in the original scheme were corrected, and probabilistic features were excluded from interactions between verifiers and credential holders. Although probabilistic features remain in member registration procedures when credential holders want to conceal their attribute values, the scheme still can maintain its efficiency because member registration procedures are required only once for individual entities. Therefore, developments of secure and efficient anonymous service systems e.g. e-cash systems [13],

anonymous data collection systems [17], etc. become possible.

References

- [1] Chaum, D., "Untraceable electronic mail, return address and digital pseudonyms," *Communications of the ACM*, 24 (2), 84-88. 1981.
- [2] Blum, M., Feldman, P. and Micali, S., "Non-interactive zero-knowledge and its applications," in *20th Annual ACM Symposium on Theory of Computing*, 103-112. 1988.
- [3] Goldwasser, S., Micali, S. and Rackoff, C., "The knowledge complexity of interactive proof system," *SIAM Journal on Computing*, 18 (1), 291-304. 1989.
- [4] Damgard, I. B., "Payment systems and credential mechanism with provable security against abuse by individuals," in *CRYPTO'88*, 328-335. 1990.
- [5] Camenisch, J. and Lysyanskaya, A., "An efficient system for non-transferable anonymous credential with optimal anonymity revocation," in *EUROCRYPT'01*, 93-118. 2001.
- [6] Camenisch, J. and Lysyanskaya, A., "Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation," in *EUROCRYPT 2001*, 93-118. 2001.
- [7] Camenisch, J. and Lysyanskaya, A., "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *CRYPTO 2002*, 61-76. 2002.
- [8] Camenisch, J. and Lysyanskaya, A., "Signature schemes and anonymous credentials from bilinear maps," in *CRYPTO 2004*, 56-72. 2004.
- [9] Belenkiy, M., Chase, M., Kohlweiss, M. and Lysyanskaya, A., "P-signatures and noninteractive anonymous credentials," in *Theory of Cryptography Conference*, 356-374. 2008.
- [10] Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A. and Shacham, H., "Randomizable proofs and delegatable anonymous credentials," in *29th Annual International Cryptology Conference on Advances in Cryptology*, 108-125. 2009.
- [11] Shahandashti, S. F. and Safavi-Naini, R., "Threshold attribute-based signatures and their application to anonymous credential systems," in *2nd International Conference on Cryptology in Africa: Progress in Cryptology*, 198-216. 2009.
- [12] Sudarsono, A., Nakanishi, T. and Funabiki, N., "Efficient proofs of attributes in pairing-based anonymous credential system," *Springer Lecture Notes in Computer Science 6794/2011*, 246-263. 2011.
- [13] Tamura, S., *Anonymous security systems and applications: requirements and solutions*, Information Science Reference, 2012.
- [14] Tamura, S., Haddad, H. A., Kouro, K., Tsurugi, H., Md. Rokibul Alam, K., Yanase, T. and Taniguchi, S., "An information system platform for anonymous product recycling," *Journal of Software*, 3 (3), 46-56. 2008.
- [15] Diffie, W. and Hellman, M. E., "New directions in cryptography," *IEEE Trans. on Information Theory*, IT-22 (6), 644-654. 1976.
- [16] Chaum, D., Fiat, A. and Naor, M., "Untraceable electronic cash," in *CRYPTO'88*, 319-327. 1988.
- [17] Tamura, S. and Taniguchi, S., "A scheme for collecting anonymous data," in *IEEE-ICIT*, 1210-1215. 2013.