Policy-Controlled Signatures*

Pairat Thorncharoensri, Willy Susilo, and Yi Mu

Centre for Computer and Information Security
School of Computer Science & Software Engineering
University of Wollongong, Australia
{pt78,wsusilo,ymu}@uow.edu.au

Abstract. In this paper, we present a new cryptographic primitive called "policy-controlled signature". In this primitive, a signer can sign a message and attach some policies to it. Only a verifier who satisfies the policies attached can verify the authenticity of the message. This type of signature schemes has many applications, in particular to deal with sensitive data, where the signer does not want to allow anyone who is not authorized to verify its authenticity. Nonetheless, there is no existing cryptographic primitives that can offer this feature in the literature. Policy-controlled signatures can be seen to be similar to the notion of designated verifier signatures, as it can also be used to *designate* a signature to multiple recipients. When there is only a single attribute involved in a policy presented by a verifier, then we will achieve a designated verifier signature (with some trivial modifications). Therefore, policy-controlled signatures can be viewed as the generalization of the notion of the designated verifier signatures. We present a formal model to capture this notion. Furthermore, we also present a concrete scheme that is secure in our model. Finally, we briefly mention about an implementation that incorporates P3P to realize policy-controlled signatures.

1 Introduction

Consider the following scenario. Alice is a CIA agent. She would like to convey a sensitive message in regards to the international terrorism to the other secret agents, but this message should not be verifiable by the public. Therefore, this message should be verifiable by $\{\mathsf{CIA\ agents} \vee \mathsf{KGB\ agents} \vee (\mathsf{secret\ agents\ in\ the\ country} \wedge \mathsf{authorized\ agents\ by\ the\ US\ Government})\}$. The first two conditions imply that if the agent is either a CIA agent or a KGB agent, then the message should be verifiable. The third condition implies that if the person is a secret agent in the country where he/she is authorized by the US government, then he/she should be able to verify the message as well. The message should not be verifiable by any other people outside this authorized set. Furthermore, it is also required that upon the verification of the message by the authorized agents, the agents cannot *relay* this conviction and convince any other third party outside the authorized set. This is a typical scenario where policy-controlled signatures

* This work is partially supported by ARC Linkage Project Grant LP0667899.

S. Qing, C.J. Mitchell, and G. Wang (Eds.): ICICS 2009, LNCS 5927, pp. 91–106, 2009.

are useful. Furthermore, there are many other applications that involve contents sensitive, such as medical records, that are only authorized to several people such as medical doctors, etc.

The above scenario seems to be straightforward and it could be solved by using the existing cryptographic primitives. Nonetheless, we shall demonstrate that unfortunately, the existing cryptographic primitives cannot be used to solve this problem. Firstly, the notion that is *close* to this scenario is the notion of policy-based cryptography [1]. In a policy-based cryptography, the sender is equipped with the policy, but not the verifier. Unlike our scenario, policy-based cryptography introduced in [1] allows the sender to sign a message correctly if and only if the sender satisfies some policies. We argue that our scenario is more natural compared to the scenario used in policy-based cryptography [1] since usually the sender should specify which receiver(s) that should be able to verify the authenticity of the message. At a glance, we may be able to achieve the above solution by signing a message with a regular signature scheme, and then encrypt it with a policy-based encryption scheme. Nevertheless, in this solution, a verifier who can decrypt the ciphertext can obtain the signature and make it publicly verifiable. Hence, it violates the requirement as stated above.

At the first glance, this notion seems to be closely related to designated verifier signatures [2]. With a trivial modification, policy-controlled signatures are in fact the generalization of the notion of designated verifier signatures. If the number of verifiers is merely only *one*, then policy-controlled signatures with a trivial modification will achieve a designated verifier signature scheme as defined in [2]. Nevertheless, policy-controlled signatures allow multiple verifiers to verify the signature on some message (and hence, it provides a *similar* property as the designated *multiple* verifiers signatures). The idea is to allow several receivers to satisfy some policies, and therefore, the signature produced by the signer should be verifiable by these receivers. Nevertheless, the other party who does not satisfy the policies should not be able to verify the signature.

This notion resembles a similar idea to the notion of multi-designated verifier signatures. However, in the latter, the collaborations of the verifiers are required in verifying the designated signatures. In contrast to this notion, in policy-controlled signatures, each verifier can verify it individually, as long as it holds the satisfying policies.

Due to the requirements of the verifier to satisfy some policies specified by the signer, we call our primitive as *policy-controlled signatures*. We further argue that this cryptographic primitive is hard to construct. In particular, we should ensure that any coalition of unauthorized verifiers must not be able to verify the authenticity of any signature. If coalition resistance is not required, then we can simply assign a separate policy for each verifier to enable the verifier to test the authenticity of the signature.

This notion can be viewed as the *dual* of secret handshakes, as introduced in [3]. Secret handshakes aim to allow members of the group to identify each other. Secret handshakes ensure that non-group members cannot recognize the handshake and hence are not able to recognize group members. Additionally,

non-group members cannot perform the handshake and hence are unable to trifle group members into thinking they are also members [3]. Secret handshakes are merely encryption schemes that only allow the group members to decrypt the ciphertext.

1.1 Related Work

Since the seminal introduction of digital signature notion [4] and its formalization [5], the notion of digital signatures have been extended to capture different scenarios and situations in real life. Among the different type of signatures, we include some variants of digital signatures that are related to our notion, that include (universal) designated verifier signature [6,7,8,9,10,11,12,13] and policy-based crytography [1,14,15,16].

The notion of designated verifier signatures was put forth by Jakobsson, Sako and Impagliazzo in [2]. In this notion, a signature ensures authenticity and deniability properties at the same time. The deniability property stops further conviction of a validity of a digital signature by a verifier to any other third party. The authenticity is ensured by the signature since the verifier can be sure that the signature is indeed created by the original signer, but nobody else will be convinced with this fact. Recently, the topics on designated verifier signatures have been actively studied [6,7,8,9,10,11].

The notion of ring signatures was introduced and formalized by Rivest, Shamir and Tauman [17] with the aim to provide an unconditional security to the signer in a group. In this notion, a signer (alone without any cooperation with other signers) can generate a signature that is verifiable to have been signed by one of the signers in a ring (a set of signers). Hence, from the verifier's point of view, a ring signature provides authentication of a message by one signer in the ring. Two-party ring signatures are known to give the construction of designated verifier signatures. This notion has also been actively studied in the literature [18,19,20,21,22,23,24,25,26]. In particular, the notion of threshold ring signatures, which provides the integrity of the message, and the authenticity, non-repudiation and anonymity of the multi-signers, was first formalized by Bresson, Stern and Szydlo in [18]. Similar to ring signature schemes, threshold ring signature schemes support multi-signers (cf. the original ring signature schemes).

The notion of policy-based cryptography, which includes policy-based encryption schemes and policy-based signature schemes, was firstly introduced by Bagga and Molva in [1]. In this notion, a sender (or signer, resp.) can only construct an encrypted message (or sign a message, resp.) if he/she satisfies the required policy. A policy-based encryption scheme provides the authorization and the message's integrity, while a policy-based signature provides the message's integrity, and the authenticity and non-repudiation of signer. Bagga and Molva gave an improved construction of policy-based encryption in [14]. This improved policy-based encryption is enhanced from the previous construction by including a public key of user into a user's credential. Here, the user is required to have a credential and a public key to decrypt the ciphertext. The purpose of this improvement is to prevent the collision attack.

1.2 Our Contributions

In this paper, we introduce the notion of policy-controlled signature (PCS) schemes. Our notion allows a receiver to verify the authenticity of the signed message if and only if the receiver satisfies the policy specified by the sender (or signer). We formalize this notion and define its security model and requirements. Furthermore, we instantiate our model with a concrete construction that is proven secure in our model.

Paper Organization
The paper is organized as follows. In the next Section, we will review some preliminaries that will be used throughout this paper. The definition of PCS and its security notions will be presented in Section 3. Next, our concrete scheme will be provided in Section 4.2. Then, proof of the security of our concrete scheme is presented in Section 5. In Section 6, we briefly demonstrate our implementation that incorporates P3P [27]. Finally, we conclude the paper.

2 Preliminaries

2.1 Notation

We will use the following notations throughout this paper. Let PPT denote a probabilistic polynomial-time algorithm. When a PPT algorithm F privately accesses and executes another PPT algorithm E, we denote it by $F^{E(.)}(.)$. We denote by $poly(.)$ a deterministic polynomial function. For all polynomials $poly(k)$ and for all sufficiently large k, if $q \leq poly(1^k)$ then we say that q is polynomial-time in k. We say that a function $f : \mathbb{N} \rightarrow \mathbb{R}$ is *negligible* if, for all constant $c > 0$ and for all sufficiently large n, $f(n) < \frac{1}{n^c}$. Denote by $l \xleftarrow{\$} L$ the operation of picking l at random from a (finite) set L. A collision of a function $h(.)$ refers to the case when there are message pair m, n of distinct points in its message space such that $h(m) = h(n)$. We denote by $||$ the concatenation of two strings (or integers).

2.2 Bilinear Pairing

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic multiplicative groups generated by g_1 and g_2, resp. The order of both generators is a prime p. Let $\mathbb{G}_T$ be a cyclic multiplicative group with the same order p. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a bilinear mapping with the following properties:

1. *Bilinearity:* $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for all $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, $a, b \in \mathbb{Z}_p$.
2. *Non-degeneracy:* There exists $g_1 \in \mathbb{G}_1$ *and* $g_2 \in \mathbb{G}_2$ such that $\hat{e}(g_1, g_2) \neq 1$.
3. *Computability:* There exists an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1 \in \mathbb{G}_1,\ g_2 \in \mathbb{G}_2$.

Note that there exists $\varphi(.)$ function which maps $\mathbb{G}_1$ to $\mathbb{G}_2$ or vice versa in one time unit.

2.3 Complexity Assumptions

Definition 1 (Computation Diffie-Hellman (CDH) Problem). *Given a 3-tuple* $(g, g^x, g^y \in \mathbb{G}_1)$ *as input, output* $g^{x \cdot y}$. *An algorithm* $\mathcal{A}$ *has advantage* ϵ' *in solving the CDH problem if*

$$\Pr\left[\mathcal{A}(g, g^x, g^y) = g^{x \cdot y}\right] \geq \epsilon'$$

where the probability is over the random choice of $x, y \in \mathbb{Z}_q^*$ *and the random bits consumed by* $\mathcal{A}$.

Assumption 1 ((t, ϵ')-Computation Diffie-Hellman Assumption). We say that the (t, ϵ')-CDH assumption holds if no PPT algorithm with time complexity $t(.)$ has advantage at least ϵ' in solving the CDH problem.

Definition 2 (Decision Bilinear Diffie-Hellman (DBDH) Problem) *Given a random 4-tuple* $(g, g^a, g^b, g^c) \in \mathbb{G}_1$ *and a random integer* $Z \in \mathbb{G}_T$ *as input, decide whether or not* $Z = \hat{e}(g, g)^{abc}$. *An algorithm* $\mathcal{A}$ *is said to* $(\mathtt{t}, \epsilon')$ *solves the DBDH problem in* $\mathbb{G}_1, \mathbb{G}_T$, *if* $\mathcal{A}$ *runs in time* $\mathtt{t}$, *and*

$$\left|\Pr\left[\mathcal{A}\left(g, g^a, g^b, g^c, Z = \hat{e}(g, g)^{abc}\right) = 1\right] - \Pr\left[\mathcal{A}\left(g, g^a, g^b, g^c, Z = \hat{e}(g, g)^{d}\right) = 1\right]\right| \geq \epsilon',$$

where the probability is taken over the random choices of $a, b, c, d \in \mathbb{Z}_p$, $g \in \mathbb{G}_1$, *and the random bits consumed by* $\mathcal{A}$.

Assumption 2 (Decision Bilinear Diffie-Hellman Assumption). We say that the (t, ϵ')-DBDH assumption in $\mathbb{G}_1, \mathbb{G}_T$ holds if there is no PPT algorithm that (t, ϵ') solves the DBDH problem.

3 Policy-Controlled Signature Schemes (PCS)

Model

Let TA denote a trusted authority who issues credentials associated with policies. Let CA denote a certificate authority who generates system parameters and certifies public keys for all parties. There are two main players in a policy-controlled signature scheme, namely a signer and a verifier. A signer S generates a signature that can be verified *only* when a verifier V holds a credential satisfying the policy. V holds credentials issued by TA.

Let A denote an assertion issued by TA. Each assertion A may be a hash value of some statements, such as "CIA agent". We define POL to be a policy which contains a set of assertions $POL = \bigwedge_{i=1}^{a}[\bigvee_{j=1}^{a_i}[\bigwedge_{k=1}^{a_{i,j}} A_{i,j,k}]]$ where i, j, k are indexes. In general, a policy POL can be represented in the disjunctive normal form (DNF) or the conjunctive normal form (CNF) or any combination of both forms. The policy POL is in the DNF when $a = 1$ and in CNF when $\forall i, \forall j : a_{i,j} = 1$. For example, a policy in DNF is as follows $POL =$ "$(A_{1,1,1} \bigwedge A_{1,1,2}) \bigvee (A_{1,2,1} \bigwedge A_{1,2,2}) \bigvee A_{1,3,1} \bigvee (A_{1,4,1} \bigwedge A_{1,4,2})$".

For simplicity, let $VCR_{i,j,k}$ be a credential for an assertion and let $\overline{POL} = [\{VCR_{1,1,1}, ..., VCR_{a,1,a_{i,1}}\}, ..., \{VCR_{1,a_i,1}, ..., VCR_{a,a_i,a_{i,j}}\}]$ be a set of the entire possible set of credential that satisfy the policy POL, where i, j, k are indexes for assertion associated to credential. Let $\{VCR_{i,j,k}\} = VCR_{1,j,1}, ..., VCR_{a,j,a_{i,j}}$ be a set of credentials, which it may or may not be a set of credentials in $\overline{POL}$. Let $\{VCR\} = VCR_{1,1,1}, ..., VCR_{a,a_i,a_{i,j}}$ be the entire credentials, where i, j are indexes.

Without losing generality, we assume that all parties must comply with the registration protocol with a certificate authority CA to obtain a certificate on their respective public keys. In the following, we provide a definition of policy-controlled signature scheme as follows.

Definition 3. *A policy-controlled signature scheme Σ is an 6-tuple (Setup, $TKeyGen$, $SKeyGen$, $CreGen$, $Sign$, $Verify$) such that*

Signature Scheme Setup:

- *System Parameters Generation (Setup):*
  Setup is a PPT algorithm that, on input a security parameter $\mathcal{K}$, outputs the system parameters param.
- *TA Key Generator ($TKeyGen$) :*
  $TKeyGen$ is a PPT algorithm that, on input the system parameters param*, outputs strings (sk_{TA}, pk_{TA}) where they denote a secret key and a public key of trusted authority, respectively. That is $\{pk_{TA}, sk_{TA}\} \leftarrow TKeyGen($*param*$)$.*
- *Signer Key Generator ($SKeyGen$) :*
  $SKeyGen$ is a PPT algorithm that, on input the system parameters param *and a public key of the trusted authority pk_{TA}, outputs strings (sk_S, pk_S) where they denote a secret key and a public key of a signer, respectively. That is $\{pk_S, sk_S\} \leftarrow SKeyGen($*param*$, pk_{TA})$.*
- *Verifier Credential Generator ($CreGen$) :*
  $CreGen$ is a PPT algorithm that, on input the system parameters param*, the TA's public key, the policy POL, outputs verifier credential strings $\{VCR_{i,j,k}\}$ where i, j, k are an index of credential strings. That is $\{VCR_{i,j,k}\} \leftarrow CreGen($*param*$, pk_{TA}, POL)$.*

Policy-controlled Signature Signing ($Sign$):
On input the system parameters param*, the trust authority's public key pk_{TA}, the signer's secret key sk_S, the signer's public key pk_S, a message M and the policy POL, $Sign$ outputs signer's signature σ. That is $\sigma \leftarrow Sign($*param*$, M, sk_S, pk_S, pk_{TA}, POL)$.*

Policy-controlled Signature Verification ($Verify$):
On input the system parameters param*, the trust authority's public key pk_{TA}, the signer's public key pk_S, the policy POL, a set of credentials $\{VCR_{i,j,k}\} \in \overline{POL}$, a message M and a signature σ, $Verify$ outputs a verification decision $d \in \{Accept, Reject\}$. That is $d \leftarrow Verify($*param*$, M, \sigma, pk_{TA}, pk_S, POL, \{VCR_{i,j,k}\})$.*

3.1 Security Model

Queries: To model the ability of the adversaries in breaking the security of PCS schemes, the following queries is prescribed.

$\mathcal{SSO}$ **oracle:** At most q_{SS}, $\mathcal{A}$ can make a query for a signature σ on its choice of a message M. As a response, $\mathcal{SSO}$ runs the $Sign$ algorithm to generate a signature σ on a message M corresponding with pk_{TA}, pk_S and POL. $\mathcal{SSO}$ then returns σ, M to $\mathcal{A}$.

$\mathcal{VCO}$ **oracle:** At most q_{VC}, $\mathcal{A}$ can make a query for the credential VCR_i corresponding to the assertion A_i. As a response, $\mathcal{VCO}$ replies $\mathcal{A}$ with a corresponding credential VCR_i.

$\mathcal{VSO}$ **oracle:** At most q_{VS}, $\mathcal{A}$ can make a query for the verification of a signature σ to $\mathcal{VSO}$ with a signature σ as input. As a response, $\mathcal{VSO}$ returns with `Accept` or `Reject` corresponding to a validation of signature σ.

Unforgeability: The unforgeability property in our model aims to provide a security against existential unforgeability under adaptive chosen message and credential exposure attack. It intentionally prevents an attacker accesses to credential queries to generate a policy-controlled signature σ_* on a new message M^*. Formally, the unforgeability provides an assurance that one, with an access to signing queries, credential queries, and the signer public parameters pk_S, should be unable to produce a policy-controlled signature on a new message M^* even with arbitrarily choosing policy POL, message M and the entire credentials $\{VCR\}$ as inputs.

We denote by CM-A the adaptively chosen message and credential exposure. We also denote by EUF-PCS the existential unforgeability of PCS scheme. Let $\mathcal{A}^{CM-A}_{EUF-PCS}$ be the adaptively chosen message and credential exposure adversary and let $\mathcal{F}$ be a simulator. The following game between $\mathcal{F}$ and $\mathcal{A}$ is defined to describe the existential unforgeability of PCS scheme:

Given a choice of messages M and an access to queries $\mathcal{SSO}$ and $\mathcal{VCO}$, $\mathcal{A}$ arbitrarily make queries to oracles in an adaptive way. At the end of the above queries, we assume that $\mathcal{A}$ outputs a forged signature σ_* on a new message M^* with respect to the public key pk_S and policy POL^*. We denote that $\overline{POL^*}$ is the entire possible set of credential of a policy POL^*. We say that $\mathcal{A}$ wins the game if:

1. $Accept \leftarrow Verify(M^*, \sigma_*, pk_S, POL^*, \{VCR_{i,j,k}\} \in \overline{POL^*})$.
2. $\mathcal{A}$ never made a request for a policy-controlled signature on input M^*, pk_S, $POL^*, \{VCR_{i,j,k}\} \in \overline{POL^*}$ to $\mathcal{SSO}$ oracle.

Let $Succ^{CM-A}_{EUF-PCS}(.)$ be the success probability function of that $\mathcal{A}^{CM-A}_{EUF-PCS}$ wins the above game.

Definition 4. *We say that PCS scheme is* ($\mathtt{t}$,q_H,q_{SS},q_{VC},ϵ)*-secure existentially unforgeable under a chosen message and credential exposure attack if there are no PPT adversary* $\mathcal{A}^{CM-A}_{EUF-PCS}$ *such that the success probability* $Succ^{CM-A}_{EUF-PCS}(k) =$

ϵ is non-negligible in k, where $\mathcal{A}^{CM-A}_{EUF-PCS}$ runs in time at most $\mathtt{t}$, make at most q_H to the random oracle, and at most q_{SS}, and q_{VC} queries to queries $\mathcal{SSO}$ and $\mathcal{VCO}$, respectively.

Coalition-resistant: In this section, we will discuss about coalition-resistant property of PCS schemes. Coalition-resistant property aims to prevent an attacker as a group of corrupted credential holders (verifiers) to verify a policy-controlled signature σ_* on a message M^* with a policy POL which an attacker does not have enough credential to satisfy the policy POL.

The condition "verifier does not have enough credential to satisfy the policy POL" is elaborated as follows:

At least one set of credential of assertions A in the policy $POL = \bigwedge_{i=1}^{a}[\bigvee_{j=1}^{a_i}[\bigwedge_{k=1}^{a_{i,j}} A_{i,j,k}]]$ is not given to the verifier such that the verifier does not have sufficient credentials to verify a policy-controlled signature on a message M with the policy POL. For example:

1. In case of $a = 1; a_i = 1; a_{i,j} > 1$, the verifier does not have one (or more) credential VCR_l of assertions $A_1, ..., A_{a_{i,j}}$.
2. In case of $a = 1; a_i > 1; a_{i,j} > 1$, the verifier does not have one set of credential $VCR_{l,1}, ...,VCR_{l,a_i}$ of assertions $[A_{1,j,1}, ..., A_{1,j,a_{i,j}}]_{1 \leq j \leq a_i}$ that satisfy the first cases.
3. In the case of $a > 1; a_i > 1; a_{i,j} > 1$, the verifier has one set of credential (e.g., $[VCR_{l,j,1}, ..., VCR_{l,j,a_{i,j}}]_{1 \leq j \leq a_i}$) of assertions $[A_{i,j,1}, ..., A_{i,j,a_{i,j}}]_{1 \leq i \leq a, 1 \leq j \leq a_i}$ that does not satisfy the second case.

Formally, the coalition-resistant provides an assurance that one, with an access to signing queries, credential queries, and the signer public parameters pk_S, should be unable to distinguish a valid signature out of two policy-controlled signature on a message M^* even by arbitrarily choosing policy POL^*, message M^* and the entire credential $\{VCR\}$ except one set of credentials that make unsatisfiable to policy POL^* as inputs. This intentionally prevents a distinguisher to distinguish a valid signature from a (simulated) invalid signature on any message M with a new policy POL^*.

Let *CRI-PCS* denote the existential coalition-resistant of PCS scheme. Let $\mathcal{A}^{CMP-A}_{CRI-PCS}$ be the adaptively chosen message and chosen policy attack and let $\mathcal{F}$ be a simulator. The following experiment between $\mathcal{F}$ and $\mathcal{A}$ is prescribed the existential coalition-resistant of PCS scheme. The experiment is divided into two phases. We run them as follows:

1. **Phase** 1**:** With any adaptive strategies, $\mathcal{A}$ arbitrarily sends requests of query to $\mathcal{SSO}$, $\mathcal{VCO}$ queries. The queries response as per their design.
2. **Challenge:** At the end of the first phase, $\mathcal{A}$ decides to challenge and then outputs M^* and $POL^* = \bigwedge_{i=1}^{a}[\bigvee_{j=1}^{a_i}[\bigwedge_{k=1}^{a_{i,j}} A_{i,j,k}]]$ such that:
   a. On input POL^* and M^*, $\mathcal{A}$ never issued a request for a policy-controlled signature to $\mathcal{SSO}$ queries.

b. On input POL^*, $\mathcal{A}$ can issue a request of credentials to $\mathcal{VCO}$ queries, however, $\mathcal{A}$ must not make sufficient requests of credentials to satisfy the policy POL^* as mentioned clearly above.

After that $\mathcal{F}$ chooses a random bit $b \overset{\$}{\leftarrow} \{0,1\}$. If $b = 1$ then, on input a policy POL^*, a signer public key pk_S and a message M^*, $\mathcal{F}$ makes a request for a policy-controlled signature to $\mathcal{SSO}$ queries and responds $\mathcal{A}$ with σ^* as an output from $\mathcal{SSO}$ queries. Otherwise, on input a policy POL^*, a signer public key pk_S, a message M^*, a valid policy-controlled signature σ^* on message M^* with policy POL^* and a set of credentials $\{VCR_{i,j,k}\} \in \overline{POL^*}$, $\mathcal{F}$ computes a (simulated) invalid policy-controlled signature σ^* and responds $\mathcal{A}$ with σ^*.

3. **Phase 2:** In this phase, $\mathcal{A}$ can return to *Phase* 1 or *Challenge* as many as $\mathcal{A}$ wants, on one condition, $\mathcal{A}$ must have at least one set of challenges M^*, POL^*, σ^* such that
   a. On input POL^* and M^*, $\mathcal{A}$ never issued a request for a policy-controlled signature to $\mathcal{SSO}$ queries.
   b. On input POL^*, $\mathcal{A}$ can issue a request of credentials to $\mathcal{VCO}$ queries, however, $\mathcal{A}$ must not have enough credentials to satisfy the policy POL^* and to verify σ^* as mentioned clearly above.
4. **Guessing:** On the challenge M^*, POL^*, σ^*, $\mathcal{A}$ finally outputs a guess b'. The distinguisher wins the game if $b = b'$.

Let $Succ^{CMP-A}_{CRI-PCS}(.)$ be the success probability function of that $\mathcal{A}^{CMP-A}_{CRI-PCS}$ wins the above game.

Definition 5. *We say that PCS scheme is (*$\mathtt{t}$*,*q_H*,*q_{SS}*,*q_{VC}*,*ϵ*)-secure existentially coalition-resistant under a chosen message and chosen policy attack if there are no PPT distinguisher* $\mathcal{A}^{CMP-A}_{CRI-PCS}$ *such that the success probability* $Succ^{CMP-A}_{CRI-PCS}(k) = |\Pr[b = b'] - \Pr[b \neq b']| = \epsilon$ *is non-negligible in* k*, where* $\mathcal{A}^{CMP-A}_{CRI-PCS}$ *runs in time at most* $\mathtt{t}$*, make at most* q_H *to the random oracle, and at most* q_{SS}*, and* q_{VC} *queries to queries* $\mathcal{SSO}$ *and* $\mathcal{VCO}$*, respectively.*

Invisibility: In this section, we will elaborate the invisibility property of PCS schemes. Intuitionally, the invisibility property is to prevent an attacker, who does not have any credential to satisfy a policy POL, to verify a policy-controlled signature σ on a message M with respect to a policy POL. Formally, the invisibility provides an assurance that one, with an access to signing queries, verification queries and the signer public parameters pk_S, should be unable to distinguish a valid policy-controlled signature on a message M^* from an invalid one even with arbitrarily choosing policy POL^* and message M^* as inputs.

Let *INV-PCS* denote the existential invisibility privacy of PCS scheme. Let $\mathcal{A}^{CMP-A}_{INV-PCS}$ be the adaptively chosen message and chosen policy distinguisher and let $\mathcal{F}$ be a simulator. The following experiment between $\mathcal{F}$ and $\mathcal{A}$ is prescribed as the existential invisibility privacy of PCS scheme. The experiment is divided into two phases. We run them as follows:

1. **Phase** 1**:** With any adaptive strategies, $\mathcal{A}$ arbitrarily sends requests of query to $\mathcal{SSO}$ and $\mathcal{VSO}$. The queries response as their design.
2. **Challenge:** At the end of the first phase, $\mathcal{A}$ decides to challenge and then outputs M^* and $POL^* = \bigwedge_{i=1}^{a}[\bigvee_{j=1}^{a_i}[\bigwedge_{k=1}^{a_{i,j}} A_{i,j,k}]]$ such that, on input POL^* and M^*, $\mathcal{A}$ never issued a request for a policy-controlled signature to $\mathcal{SSO}$ queries. After that $\mathcal{F}$ chooses a random bit $b \xleftarrow{\$} \{0,1\}$. If $b = 1$ then, on input a policy POL^*, a signer public key pk_S and a message M^*, $\mathcal{F}$ makes a request for a policy-controlled signature to $\mathcal{SSO}$ queries and responds $\mathcal{A}$ with σ^* as an output from $\mathcal{SSO}$ queries. Otherwise, on input a policy POL^*, a signer public key pk_S, a message M^*, a valid policy-controlled signature σ^* on message M^* with policy POL^*, $\mathcal{F}$ computes a (simulated) invalid policy-controlled signature σ^* and responds $\mathcal{A}$ with σ^*.
3. **Phase** 2**:** In this phase, $\mathcal{A}$ can return to *Phase* 1 or *Challenge* as many as it want. With one condition, $\mathcal{A}$ must have at least one set of challenge M^*, POL^*, σ^* such that:
   - **a.** On input POL^* and M^*, $\mathcal{A}$ never issued a request for a policy-controlled signature to $\mathcal{SSO}$ queries.
   - **b.** On input POL^*, M^* and σ^*, $\mathcal{A}$ never issued a request for the verification of the policy-controlled signature σ^* to $\mathcal{VSO}$ queries.
4. **Guessing:** On the challenge M^*, POL^*, σ^*, $\mathcal{A}$ finally outputs a guess b'. The distinguisher wins the game if $b = b'$.

Let $Succ_{INV-PCS}^{CMP-A}(.)$ be the success probability function of that $\mathcal{A}_{INV-PCS}^{CMP-A}$ wins the above game.

Definition 6. *We say that PCS scheme is* ($\mathtt{t}$,q_H,q_{SS},q_{VS},ϵ)*-secure existentially invisibility under a chosen message and chosen policy attack if there are no PPT distinguisher* $\mathcal{A}_{INV-PCS}^{CMP-A}$ *such that the success probability* $Succ_{INV-PCS}^{CMP-A}(k) = |\Pr[b = b'] - \Pr[b \neq b']| = \epsilon$ *is non-negligible in* k*, where* $\mathcal{A}_{INV-PCS}^{CMP-A}$ *runs in time at most* $\mathtt{t}$*, make at most* q_H *to the random oracle, and at most* q_{SS}*, and* q_{VS} *queries to* $\mathcal{SSO}$ *and* $\mathcal{VSO}$*, respectively.*

Theorem 1. *The invisibility of of policy-controlled signature schemes implies the coalition-resistant property of policy-controlled signature schemes.*

Proof. Assume that an invisibility adversary $\mathcal{A}_I$ solves the invisibility of PCS scheme, we will show that an adversary $\mathcal{A}_{CR}$ can solve the coalition-resistant of PCS scheme by using $\mathcal{A}_I$. Let $\mathcal{S}$ be a simulator and then $\mathcal{S}$ runs the existentially coalition-resistant game defined earlier with $\mathcal{A}_{CR}$. Meanwhile, $\mathcal{A}_{CR}$ runs the existentially invisibility game defined earlier with $\mathcal{A}_I$. On access to the $\mathcal{SSO}$ and the $\mathcal{VCO}$ queries constructed by $\mathcal{S}$, $\mathcal{A}_{CR}$ passes the $\mathcal{SSO}$ queries from $\mathcal{S}$ to $\mathcal{A}_I$. Then, by using $\mathcal{VCO}$ queries from $\mathcal{S}$, $\mathcal{A}_{CR}$ construct $\mathcal{VSO}$ queries for $\mathcal{A}_I$ by making queries to $\mathcal{VCO}$ for credentials to verify a signature queried by $\mathcal{A}_I$. At the end of phase 2, $\mathcal{A}_I$ outputs for challenge with M^* and POL^* to $\mathcal{A}_{CR}$. $\mathcal{A}_{CR}$ then relays this challenge to $\mathcal{S}$. $\mathcal{S}$ responses with σ^*. $\mathcal{A}_{CR}$ returns σ^* to $\mathcal{A}_I$. Finally, $\mathcal{A}_I$ outputs a decision b' and give to $\mathcal{A}_{CR}$. Then $\mathcal{A}_{CR}$ returns b' to $\mathcal{S}$.

From the above experiment, it is clearly shown that if $\mathcal{A}_I$ can solve the invisibility of PCS scheme, then $\mathcal{A}_{CR}$ can solve the coalition-resistant of PCS scheme via $\mathcal{A}_I$. Hence, the coalition-resistant is a stronger model of the invisibility in the notion of PCS scheme. □

4 PCS Scheme

4.1 High Level Idea

Prior to presenting our concrete construction of policy-controlled signature schemes, we will first describe our idea and intuition behind our construction for clarity. Intuitively, we can achieve a policy-controlled signature scheme by combining the idea of policy-based encryption schemes [1], a general signature scheme and a designated verifier signature scheme as follows. Firstly, we combine a designated verifier signature on a message into a policy-based ciphertext such that only a verifier, who has satisfied the policy, can verify the authenticity of the signature. This will constitute the part of policy-controlled signatures, which we refer to as "the encrypted designated verifier signature". Then, a signature scheme is used to sign the concatenation of the policy and the encrypted designated verifier signature. The encrypted designated verifier signature and the signature from the above construction constitutes a policy-controlled signature. The purpose of the above construction is to ensure the authentication of the signer such that the verifier is convinced that the signer has actually generated this policy-controlled signature. The signature can be publicly verifiable, however, one could not be convinced that the signature is indeed associated to a message, unless one has the credentials satisfying the policy to verify the encrypted designated verifier signature. Hence, without revealing the verifier's private information (the credentials associated to the policy), the verifier should not be able to convince other party that a signer generated the policy-controlled signature.

4.2 The Construction

In this section, we present our concrete construction of PCS schemes. Let $H_0 : \{0,1\}^* \rightarrow \mathbb{G}_1$; $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$; $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$ be three distinct random one-way functions that map any string to group $\mathbb{G}_1$ and let $h : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$ be a collision-resistant hash function. We denote by $\mathbb{G}_1$ and $\mathbb{G}_T$ groups of prime order p. Assume that there exists an efficient computationally bilinear mapping function $\hat{e}$ which maps $\mathbb{G}_1$ to $\mathbb{G}_T$. The above mapping function is defined as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$. The scheme is described as follows.

Setup: On input a security parameter $\mathcal{K}$, a trusted third party randomly chooses a prime $p \approx poly(1^{\mathcal{K}})$. Select a random generator $g \in \mathbb{G}_1$ and a bilinear mapping function $\hat{e}$. Select hash functions $H_0(.), H_1(.), H_2(.), h(.)$. We denote by $\mathsf{param} = (p, \hat{e}, g, H_0, H_1, H_2, h)$ the system parameters. Then, *Setup* returns param.

TKeyGen: On input a system parameters param, a trusted authority TA randomly generates a private key sk_{TA} and a public key pk_{TA} as follows: select two random integers $\mu, \gamma \in \mathbb{Z}_p$. Let $U = g^{\mu}; W = g^{\gamma}$ denote a public key. Therefore, *TKeyGen* returns $sk_{TA} = (\mu, \gamma)$ as a private key of the trusted authority and $pk_{TA} = (U, W)$ as a public key of the trusted authority.

SKeyGen: On input a system parameters param and a public key of the trusted authority pk_{TA}, a signer S randomly generates a private key sk_S and a public key pk_S as follows: select a random integer $x \in \mathbb{Z}_p$. Let $X = g^x; \hat{X} = W^x$ denote a public key. Therefore, *SKeyGen* returns $sk_S = x$ as a private key of the signer and $pk_S = (X, \hat{X})$ as a public key of the signer.

CreGen: Let P be a statement in the policy, e.g., $P =$'CIA agent'. An assertion A of P is computed as follows: $A = H_2(P)$. On input a system parameters param, the trusted authority's public key pk_{TA}, the trusted authority's private key sk_{TA} and a set of assertions $A_1, ..., A_n$ that verifier is satisfied to obtain, a trusted authority TA randomly generates each verifier's credential strings $VCR_i = (CV_i, CR_i, CG_i)$ where i is an index of credentials as follows: TA randomly selects $\nu_i \in \mathbb{Z}_p^*$ and computes each credential $CV_i = U^{1/\nu_i}$; $CR_i = g^{(\mu\gamma)/\nu_i} A_i^{\mu}$; $CG_i = g^{\nu_i}$ and then returns $VCR_i = (CV_i, CR_i, CG_i)$ to the verifier as a credential of assertion A_i. Verifier checks the validity of VCR_i as follows:

$$\hat{e}(CR_i, g) \stackrel{?}{=} \hat{e}(A_i, U)\hat{e}(W, CV_i),$$
$$\hat{e}(CV_i, CG_i) \stackrel{?}{=} \hat{e}(U, g).$$

Sign: Given param, pk_{TA}, sk_S, pk_S, $POL = \bigwedge_{i=1}^{a}[\bigvee_{j=1}^{a_i}[\bigwedge_{k=1}^{a_{i,j}} A_{i,j,k}]]$ and a message M, S computes a policy-controlled signature σ on a message M as follows:

$$r, t_1, ..., t_a \stackrel{\$}{\leftarrow} \mathbb{Z}_p,\ t = \oplus_{i=1}^{a} t_i,\ \sigma_1 = g^r,$$
$$\sigma_2 = X^r,\ \sigma_3 = \hat{X}^r,$$
$$M' = M||\sigma_1||\sigma_2||\sigma_3||t||t_1||...||t_a||pk_S||pk_{TA}||POL,$$

for $i = 1$ to a, for $j = 1$ to a_i:

$$\alpha_{i,j} = h(\hat{e}((\prod_{k=1}^{a_{i,j}} A_{i,j,k})^{r \cdot x}, U)||H_0(M')||i||j),$$
$$R_{i,j} = t_i \oplus h(\hat{e}(g^{\alpha_{i,j}}, H_0(M')^x)||i||j),$$

Then compute

$$\breve{M} = \sigma_1||\sigma_2||\sigma_3||t||t_1||...||t_a||pk_S||pk_{TA}||POL||[R_{i,1}||...||R_{i,a_i}]_{1 \le i \le a},$$
$$\sigma_4 = H_1(\breve{M})^x.$$

The policy-controlled signature on a message M is

$$\sigma = (H_0(M'), \sigma_1, \sigma_2, \sigma_3, \sigma_4, [R_{i,1}, ..., R_{i,a_i}]_{1 \le i \le a}).$$

$Verify$ **:** Let $\{VCR_{i,j,k}\} = VCR_{1,j,1}, ..., VCR_{a,j,a_{i,j}}$ be a set of credentials in $\overline{POL}$ that verifier possessed. Given pk_S, pk_{TA}, pk_V, $\{VCR_{i,j,k}\} \subset \overline{POL}$, POL, σ and a message M, a verifier V first checks whether

$$\hat{e}(\sigma_2, g) \stackrel{?}{=} \hat{e}(\sigma_1, X),\ \hat{e}(\sigma_3, g) \stackrel{?}{=} \hat{e}(\sigma_2, W).$$

holds or not. If not, then V outputs `reject`. Otherwise, V computes as follows: for $i = 1$ to a:

$$\alpha_{i,j} = h((\prod_{k=1}^{a_{i,j}} (\hat{e}(CR_{i,j,k}, \sigma_2)\hat{e}(CV_{i,j,k}, \sigma_3)^{-1}))||H_0(M')||i||j)$$
$$\hat{t}_i = R_{i,j} \oplus h(\hat{e}(H_0(M')^{\alpha_{i,j}}, X)||i||j).$$

After that, compute

$$\hat{t} = \oplus_{i=1}^{a} \hat{t}_i,\ \hat{M} = M||\sigma_1||\sigma_2||\sigma_3||\hat{t}||\hat{t}_1||...||\hat{t}_a||pk_S||pk_{TA}||POL.$$

Then, V checks whether $H_0(\hat{M}) \stackrel{?}{=} H_0(M')$, $\hat{e}(\sigma_4, g) \stackrel{?}{=} \hat{e}(H_1(\sigma_1||\sigma_2||\sigma_3||\hat{t}||\hat{t}_1||...||\hat{t}_a||pk_S||pk_{TA}||POL||[R_{i,1}||...||R_{i,a_i}]_{1 \le i \le a}), X)$ holds or not. If not, then it outputs `reject`. Otherwise, it outputs `accept`.

5 Security Analysis

Theorem 2. *Our policy-controlled signature scheme is existentially unforgeable under an adaptive chosen message and credential exposure attack if the CDH assumption holds in the random oracle model.*

Theorem 3. *In the random oracle model, the proposed policy-controlled signature scheme is existentially coalition-resistant against adaptively chosen message and chosen policy attack $\mathcal{A}_{CRI-PCS}^{CMP-A}$ attack if the DBDH assumption is hold.*

Due to the page limitation, please find the proof for Theorem 2 and Theorem 3 in the full version of this paper [28].

6 Implementation

In this section, we briefly discussed our implementation to realize policy-controlled signatures. We incorporated the P3P privacy policy description language [27] and extended the description to capture the notion of policy-controlled signatures. We augment the XML digital signature with the P3P privacy policy to ensure that the policies attached will be enforced. An example of a policy-controlled signature given in our earlier scenario is provided in Figure 1. We note that in this section, we only demonstrate how the policy can be written in the P3P description language and we do not describe its enforcement, as it is not in the scope of this paper. The purpose of this section is just merely to demonstrate that our primitive is also applicable in practice.

```
<?xml version="1.0" encoding="UTF-8"?>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
. . .
<SignatureValue>NC3E˜LE=</SignatureValue>
<KeyInfo>
<POLICIES>
    <POLICY discuri="http://p3pbook.com/privacy.html" name="policy">
    <RECIPIENT>
        <DATA ref=‘‘CIA Agent’’/>
        <DATA ref=‘‘KGB Agent’’/>
        <DATA ref=‘‘Agents’’ AND ref=‘‘Authorized by US’’/>
    </RECIPIENT>
    </POLICY>
</POLICIES>
```

Fig. 1. P3P for policy-controlled signatures

7 Conclusion

In this paper, we introduced the notion of policy-controlled signatures that allows a signer to control the verification of his signature by attaching policies. If the verifier satisfies the policies provided, then the verifier can test the authenticity of the message. Otherwise, the verifier cannot do so. Only a verifier who has credentials satisfying the policies provided by the signer can verify the policy-controlled signatures. We argue that this cryptographic primitive has many applications, in particular the ones that involve sensitive information. We presented a security model for PCS schemes, together with a concrete construction that is secure in our model.

References

1. Bagga, W., Molva, R.: Policy-based cryptography and applications. In: S. Patrick, A., Yung, M. (eds.) FC 2005. LNCS, vol. 3570, pp. 72–87. Springer, Heidelberg (2005)
2. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
3. Balfanz, D., Durfee, G., Shankar, N., Smetters, D., Staddon, J., Wong, H.: Secret Handshakes from Pairing-based Key Agreements. In: 2003 IEEE Symposium on Security and Privacy, pp. 180–196 (2003)
4. Diffie, W., Hellman, M.E.: New directions in cryptography IT-22(6), 644–654 (November 1976)
5. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks 17(2), 281–308 (April 1988); Special issue on cryptography
6. Laguillaumie, F., Vergnaud, D.: Multi-designated verifiers signatures: anonymity without encryption. Inf. Process. Lett. 102(2-3), 127–132 (2007)

7. Laguillaumie, F., Vergnaud, D.: Designated verifier signatures: Anonymity and efficient construction from any bilinear map. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 105–119. Springer, Heidelberg (2005)
8. Li, Y., Lipmaa, H., Pei, D.: On delegatability of four designated verifier signatures. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 61–71. Springer, Heidelberg (2005)
9. Lipmaa, H., Wang, G., Bao, F.: Designated verifier signature schemes: Attacks, new security notions and a new construction. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 459–471. Springer, Heidelberg (2005)
10. Huang, X., Mu, Y., Susilo, W., Zhang, F.: Short designated verifier proxy signature from pairings. In: Enokido, T., Yan, L., Xiao, B., Kim, D.Y., Dai, Y.-S., Yang, L.T. (eds.) EUC-WS 2005. LNCS, vol. 3823, pp. 835–844. Springer, Heidelberg (2005)
11. Susilo, W., Zhang, F., Mu, Y.: Identity-based strong designated verifier signature schemes. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 313–324. Springer, Heidelberg (2004)
12. Steinfeld, R., Bull, L., Wang, H., Pieprzyk, J.: Universal designated-verifier signatures. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 523–542. Springer, Heidelberg (2003)
13. Laguillaumie, F., Libert, B., Quisquater, J.-J.: Universal designated verifier signatures without random oracles or non-black box assumptions. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 63–77. Springer, Heidelberg (2006)
14. Bagga, W., Molva, R.: Collusion-free policy-based encryption. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 233–245. Springer, Heidelberg (2006)
15. Bagga, W., Crosta, S., Molva, R.: Proof-carrying proxy certificates. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 321–335. Springer, Heidelberg (2006)
16. Bagga, W., Crosta, S., Michiardi, P., Molva, R.: Establishment of ad-hoc communities through policy-based cryptography. Electr. Notes Theor. Comput. Sci. 171(1), 107–120 (2007)
17. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
18. Bresson, E., Stern, J., Szydlo, M.: Threshold ring signatures and applications to ad-hoc groups. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 465–480. Springer, Heidelberg (2002)
19. Tsang, P.P., Wei, V.K., Chan, T.K., Au, M.H., Liu, J.K., Wong, D.S.: Separable linkable threshold ring signatures. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 384–398. Springer, Heidelberg (2004)
20. Herranz, J., Sáez, G.: Forking lemmas for ring signature schemes. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 266–279. Springer, Heidelberg (2003)
21. Zhang, F., Kim, K.: Id-based blind signature and ring signature from pairings. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 533–547. Springer, Heidelberg (2002)
22. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 60–79. Springer, Heidelberg (2006)
23. Shacham, H., Waters, B.: Efficient ring signatures without random oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 166–180. Springer, Heidelberg (2007)

24. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 181–200. Springer, Heidelberg (2007)
25. Liu, J.K., Wong, D.S.: On the security models of (threshold) ring signature schemes. In: Park, C.-s., Chee, S. (eds.) ICISC 2004. LNCS, vol. 3506, pp. 204–217. Springer, Heidelberg (2005)
26. Liu, J.K., Wei, V.K., Wong, D.S.: A separable threshold ring signature scheme. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 12–26. Springer, Heidelberg (2004)
27. W3C: Platform for privacy preferences (p3p) project, `http://www.w3.org/P3P/`
28. Thorncharoensri, P., Susilo, W., Mu, Y.: Policy-controlled signatures (full version). can be obtained from the first author (2009)