A User-centric Federated Single Sign-on System

Suriadi Suriadi, Ernest Foo, Audun Jøsang
Information Security Institute, Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia
s.suriadi@isi.qut.edu.au, e.foo@qut.edu.au, a.josang@qut.edu.au

Abstract

Current identity management systems are not concerned with user privacy. Users must assume that identity providers and service providers will ensure their privacy, which is not always the case. This paper proposes an extension of existing Federated Single Sign-On (FSSO) systems that adopts the beneficial properties of the User- Centric Identity Management (UCIM) model. This new identity management system allows the users to control and enforce their privacy requirements while still retaining the convenience of single sign on over a federation of service providers. Coloured Petri Nets are used to formally model the new identity management system to provide assurance that the privacy goals are achieved. To our knowledge, Coloured Petri Nets have not been used to model privacy in identity management systems before.

Keywords: identity management, privacy, private credential, single sign-on, user-centric

1 Introduction

A Federated Single Sign-on (FSSO) system is an identity management system (IMS) that allows the use of the same user's Personal Identification Information (PII) across multiple organizations within a federation. In essence, this allows users to access services from different organizations but they are only required to provide their authentication data once. An FSSO system is made up of a group of Identity Providers (IdPs) and Service Providers (SPs). An IdP provides services related to the management and usage of the user's PII, and an SP provides services that users consume. IdP and SP are roles, therefore, an entity can be an IdP and an SP at the same time. Several existing FSSO systems are: SAML (Security Assertion Markup Language) (OASIS 2005), Liberty ID-FF (Identity Federation Framework) (Alliance 2004), and WS-Federation (Lockhart et al. 2006).

One of the main problems with the FSSO model is user privacy. FSSO systems concentrate users' PII into IdPs. A user can divide their PII between several IdPs, however each IdP still has a chunk of the user's PII and can still track some of the user's activities. Similarly, SPs can also gather information about a user from the information they get from the IdPs. Sharing of user's information by malicious IdPs and SPs can reveal a complete user's identity and activities. In an FSSO system, users have no control over the disclosure of their PII. IdPs and SPs are assumed to be trusted entities, but this is not always the case (Lemos 2007, Vijayan 2006).

An IMS that allows concrete enforcement of user's privacy requirement, instead of a mere 'trust', is needed. The rise in the User-Centric Identity Management (UCIM) system provides the potential solution to this problem. A UCIM system is designed from the *user's perspective* (Jøsang & Pope 2005). It requires that users should have an *effective control* of the use and management of their PII, resulting in a better privacy. The most common UCIM mechanism is the use of a device that can be configured as desired by users to assist them in managing their PII, such as storage of PII, modification of PII, and enforcement of the privacy requirements on the PII.

Therefore, the main contributions of this paper are:

- The proposal of a new system named a User Centric Federated Single Sign On System (UFed) which adopts the UCIM concepts into the existing FSSO systems, focusing on the proposal for a privacy-enhanced single sign-on protocol (UFed SSO)
- The application of a private credential system (Bangerter et al. 2004) and PII negotiation procedure into UFed SSO (Single Sign-on) protocol to achieve a privacy-enhanced federated single sign-on system.
- The formalization of the UFed SSO protocol using Coloured Petri Nets (CPN) modeling to show the achievement (or failure) of the privacy goals of UFed SSO protocol.

The private credential system proposed in (Bangerter et al. 2004) is *applied* in the UFed SSO protocol. Readers who are interested in the details of this credential system and the formal security proofs of the underlying cryptographic schemes should refer to: (Bangerter et al. 2004, Camenisch & Lysyanskaya 2002, 2004, Camenisch & Shoup 2003). The aim of the formal CPN modeling of UFed SSO protocol is to evaluate the success (or the failure) of the protocol in achieving its privacy goals (the exact goals are described in section 6.3). The use of CPN to model, analyze and verify protocols are widely used, such as in (Floreani et al. 1996, Huber 1991, Billington et al. 2004).

UFed can be applied in a situation whereby maintaining privacy is important while having the SSO capability is desirable for the convenience of the user. For example, a digital library may want to federate itself to other digital libraries to provide users with a greater access to library resources without having the user to repeatedly register and sign-on at each library site. However, library borrowing records may contain sensitive information as they could reveal users' interests in various areas, some of which may be sensitive information. Therefore, the ability of UFed to maintain privacy and still provide the convenience of FSSO is desirable.

Overview: The first five sections of this paper address the first contribution of this paper: the UFed architecture. Section 2 provides a description of the related work. In particular, UFed will be compared to PRIME - Privacy and Identity Management for Europe - project (PRIME 2006). The differences and advantages of UFed over PRIME are explained. How UFed is related to an earlier work by (Bhargav-Spantzel et al. 2006) on the concept of *user-centric federated identity management* system is also explained.

Section 3 describes the threat environment for UFed system. In addition to the threats from external entities who can capture and manipulate communications, it is also assumed that the IdPs, the SPs, and the users in UFed are not fully trusted entities. These threats drive the approach to designing and evaluating the UFed system.

Section 4 describes the security requirements for UFed based on the threat environment. The requirements are explained in a layered approach, and how each of these requirements is a response to the threat environment is explained.

Section 5 provides an overview of the UFed architecture, the main entities and their interactions. How the UFed requirements from section 4 can be fulfilled is explained. This section introduces the application of the private credential system (Bangerter et al. 2004). This credential system is explained in an *informal high-level perspective* to allow readers who are not specialists in cryptography to understand the capabilities of this system: partial release of information, blind certification, anonymity revocation, and other features which are useful to achieve many of the UFed requirements.

Section 6 addresses the second contribution of this paper: detailed proposal of the the UFed SSO protocol by applying the private credential system. The setup process before the UFed SSO can be conducted is explained, along with the details of the UFed SSO protocol steps. The messages that should be included in each of the protocol steps are described, as well as how the private credential system is applied.

Section 6.3 addresses the third contribution of this paper: the formalization and evaluation of the UFed SSO protocol using Coloured Petri Net. The goals of modeling the UFed SSO protocol are provided in section 6.3. The assumptions and limitations of the model are explained in section 6.3.1. The detailed construction of the model is described in section 6.3.2. The attacker model, based on the threat environment described in section 3, is introduced in section 6.3.3. Finally, the

evaluation results are provided in section 6.3.4, and the security conclusion gained from the formal evaluation process is provided in section 6.3.5.

Section 7 provides further discussion on several important aspects of UFed, such as how users profiling can be significantly reduced, how a fair and balanced negotiation process could be achieved, and issues related to anonymity revocation. Conclusion and future works are provided in section 8.

2 Related Work

Related to UFed is (PRIME 2006), which is a European government-funded project whose goal mirrors UFed: to develop a privacy enhancing identity management system. PRIME is a UCIM system that aims to give users *'sovereignty over their personal data'* (Camenisch et al. 2005). The main feature of PRIME is the ability to protect user's privacy through several mechanisms: provision for user's consent and control, privacy policy negotiation, and data minimization. These are enforced through several methods: cryptographic applications, trusted platform, and others.

The main difference between PRIME and UFed is that UFed is built based on *existing* FSSO systems while PRIME is not. PRIME nevertheless uses the federated identity management concept, whereby the IdPs issue certificates which the SPs accept as the source of users' identity information. However, how SSO and its related services can be achieved in PRIME is not evident.

The main advantage of designing UFed based on existing FSSO systems is the ability to retain the convenience and ease-of-use that FSSO system provides while providing an improved user-controlled privacy capability. In addition, the efficiency of the system is improved as users only have to be authenticated once at the IdP. As will be explained in section 5 and 6, UFed uses the private credential system (Bangerter et al. 2004) that allows anonymous authentication. However, this credential system requires substantial computing resources. Therefore, by performing this only once in an SSO setting, the overall efficiency of the system is improved. Finally, basing the design of UFed on the existing FSSO systems allows a smoother and easier migration path for those who have implemented the traditional FSSO systems.

Some properties of a user-centric federated identity management system - upon which UFed is based on - have been laid out in (Bhargav-Spantzel et al. 2006). In that paper, the two main classes of federated identity management are explained: the credential-focused and relationship-focused. The existing FSSO systems, such as SAML and Liberty ID-FF, can be classified into the relationship-focused system whereby the users and IdPs interact to generate some short-term security tokens as identity information source. The credential-focused system, on the other hand, uses long-term credentials with the IdPs as identity information source for the SPs. PRIME aligns more closely with the credential-focused system. In (Bhargav-Spantzel et al. 2006), a system called a *universal user-centric* system is proposed. This system has both the credential- and relationship-focused features. However, not much details are provided in this paper on how such a system should be achieved. Therefore, UFed is developed to achieve such a universal user-centric system.

A formal model is needed to verify, with a high assurance, that UFed achieves its key privacy goals. Therefore, the behavior verification of the UFed SSO protocol is modeled and simulated using **Coloured Petri Nets** - CPN (Jensen 1997). CPN allows the modeling of a system's various states and the transitions between these states (Kristensen et al. 1998) in a graphical representation. CPN supports a well-defined script language to describe the various data types, the manipulations and the transformations of them in the model. In short, CPN is a useful formal modeling tool as it has a '*graphical representation* and *well-defined semantics* allowing *formal analysis*' (Jensen 1997).

CPN has been widely used as a formal method tool to model, analyze and verify systems. There have been many industrial applications of CPN (Jensen 1997, Chapter 7). CPN can be used to model and verify systems in many research areas, including communication protocols (Floreani et al. 1996, Huber 1991), software system designs (McLendon & Vidale 1992, Scheschonk & Timpe 1994), and cryptographic protocols (Aly & Mustafa 2004?).

For modeling and evaluating a protocol, the protocol is first modeled using a CPN tool, capturing the essential details. Next, the possible attack scenarios are added to the model, followed by the

Layer 3	3.1 Registration	3.2 Anonymous Authentication	3.3 Data Storage	3.4 Accountability	Others
Layer 2	2.1 Minimal Data Sharing and Diclosure		2.2 Negotiation		
Layer 1	1.1 User Control		1.2 Network Communication Security		

Figure 1. Layers of Requirements

definition of the insecure state(s). When these have been included in the model, it is simulated (executed) to evaluate if the ***insecure state(s) can be reached***. In our model, the UFed SSO will be modeled and simulated using the CPN Tool[1]. To our knowledge, CPN has not been used to evaluate a privacy-enhanced SSO protocol before.

3 Threat model

The threat model for an identity management system should be as follows:

1. External attackers can monitor and capture all communications between all of the interacting entities, and have knowledge of all the cryptographic algorithms, protocols and other mechanisms employed to secure communications, except the cryptographic key(s) used.
2. The providers (both IdPs and SPs) are not unconditionally trusted entities. There are possibilities that providers could compromise users' information.
3. Users are not to be fully trusted. Users might provide false or misleading information to gain some personal advantages.

We argue that IdPs and SPs are not unconditionally trusted entities because of alarming frequencies of data breaches incidents that have happened (Lemos 2007). This threat model is similar to the *strong trust* model introduced in (Bhargav-Spantzel et al. 2006).

This threat model guides the development of the security requirements in section 4. How these requirements can be achieved leads to the high-level architecture of UFed (section 5). This threat model is also used to guide the development of the formal attacker model in section 6.3.3. Therefore, this threat model provides the *baseline* for the design of the UFed system.

4 UFed security requirements

The security requirements for UFed, based on the threat model are provided. Apart from PRIME, none of the existing SSO systems, to the extent of our current knowledge, have these built-in privacy-respecting requirements in their designs. The requirements are layered (Figure 1): the first layer is for the basic requirements for UFed, the second layer states the essential requirements to enable user-controlled privacy, the third layer states the specific UFed operations requirements.

Requirement 1.1 - user control: User should be in control of their PII in all operations that involve the use of their PII. User consent must be obtained either by pre-configuration (which can be changed as the user wishes to) or per request basis. This requirement is to counter Threat 2 by providing the users with control over the information disclosed to IdPs and SPs, thus, reducing the chance of misuse.

Requirement 1.2 - communication security: The communications between all interacting entities in UFed should be secured against Threat 1.

Requirement 2.1 - minimal data sharing and disclosure: In UFed, user's PII can still be shared but explicit user consent has to be obtained, respecting the concept of *minimum data sharing and disclosure of sensitive information*. This means that whenever possible, the user's sensitive information should not be shared to those that do not need them, but when it is needed, the least revealing level of data disclosure should be opted. Levels of data disclosure are: Level 1

[1] `http://wiki.daimi.au.dk/cpntools/_home.wiki`

- for a non-disclosure of PII value, but a disclosure of its characteristics (for example, instead of revealing the user's date of birth, only a statement that the user is over 18 years old is revealed), and Level 2 - for a disclosure of the PII value. This requirement is to counter Threat 2 by minimizing PII known to providers.

Requirement 2.2 - negotiation: To achieve requirement 2.1, a negotiation procedure is used. Users should be allowed to negotiate on the PII that they want to reveal and at what level they are willing to disclose it. The amount of PII collected should be proportionate to necessity and risks of the transaction (Harper 2006). Sometimes it is at the user's best interest to reveal more of their PII to let other party identify them correctly, especially when misidentification could have serious consequences, such as in health care. However, Threat 2 states that there is a possibility that the revealed PII will be compromised. Thus, a user needs to assess the consequences of compromised PII and the probability of it happening. Negotiation is therefore an important process to agree on which PII to reveal and the disclosure level based on the nature of the transaction, and the calculated risks.

Privacy policy is *not* the main negotiation subject in UFed as it relies on the *trust* that the IdPs and SPs will follow and enforce the agreed policy. However, this contradicts Threat 2 and thus cannot be used.

Requirement 3.1 - user registration: User should be able to negotiate the PII to be registered and their disclosure level as per requirement 2.1 and 2.2. To counter Threat 3, a mechanism should be put to allow certification of critical information for correctness. However, to avoid having the users in a disadvantaged position whereby they have to provide certified PII all the time, there should be a provision to allow users to provide uncertified PII (further discussion on this issue in section 7).

Requirement 3.2 - anonymous authentication: A user should be able to be authenticated anonymously: the authenticator can verify that a user is a valid entity recognized by a trusted authority in the federation without learning the user's identity. This is to counter Threat 2 whereby the IdPs and SPs may use users' identity information for malicious purposes.

Requirement 3.3 - data storage: To counter Threat 2, the storage of the user's PII should be secured from the possibility of the providers compromising the records. User's information can only be used under the collaboration with the owner of the information.

Requirement 3.4 - accountability: For transactions where accountability is important, a provider should maintain the log of the transaction activities. For privacy, the user's activities log files should be maintained as per requirement 3.3. As per Threat 3, users may perform activities that can be considered as a security breach to either the IdP or SP. In such a security breach investigation whereby the user refuses to cooperate, there should be a mechanism to allow the relevant authority to forcefully reveal the log data. If the user was anonymously authenticated as per requirement 3.2, there should be a mechanism to revoke the anonymity.

5 UFed architecture

The architecture of UFed (Figure 2) will be explained to show the interaction between the entities in UFed and how the security requirements can be fulfilled.

There are several additional entities in UFed as compared to traditional FSS: **Attribute Issuer (AI)** is a provider that has the authority to issue attributes about a user, such as the user's credit card information, bank account, and so on. This information is conveyed in the form of certificates that are trusted in the federation. The nature of AIs allows them to collect user's PII and it is of user's best interest to be identified correctly by the AIs, such as banks and birth registry department.

UFed Adapter is a software component that handles the UFed protocols without causing major changes to the existing FSSO protocols. This is a *critical* component for this is where most adaptation is done to bridge the existing FSSO protocols into UFed. The main role of this adapter is *to intercept the normal FSSO protocols flow where appropriate, and to execute the necessary UFed*

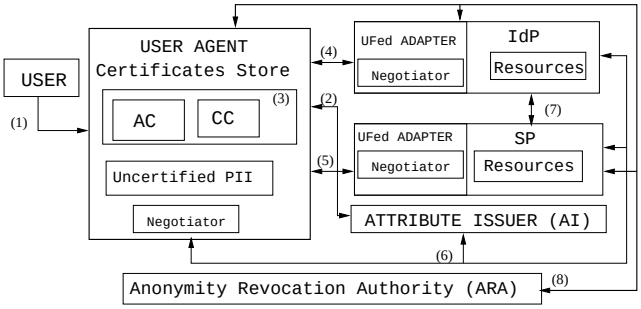

Figure 2. The UFed Architecture

protocols, before continuing the normal FSSO protocols flow as required. **Anonymity Revocation Authority (ARA)** is an entity that is responsible for revoking users anonymity.

5.1 Interaction between UFed entities

Interaction (1) shows the use of User Agent - UA (such as a PDA or a browser plug-in) to help users with their identity management activities. UA has a reasonable computational power to do protocols execution, cryptographic operations, and data storage. Users obtain two types of certificate from AI (2), and store them in the certificate store (3). Attribute Certificate (AC) contains the exact user's PII value. Disclosure of data in this certificate is a Level 2 disclosure. The Character Certificate (CC), also issued by the AI, contains *not* the exact value of a user's attributes, but only the characteristics of the attributes - a Level 1 disclosure.

UA interacts with IdP (4) and SP (5) directly as needed. IdP and SP also interact with each other (7) accordingly. However, the role of IdP in UFed is different from traditional FSSO. In UFed, IdP still provides FSSO services, but it does not store the user's PII permanently. In UFed, IdP only handles per-session registration to maintain users' anonymity. When a user is registered to IdP permanently, there will be a need for a permanent identifier that will be used to identify the user. This conflicts with requirement 3.2 and enables IdP to track the users activities. With per session registration, the users can be assigned a one-time pseudonym to be used for that session only. At the end of the session, the pseudonym is useless. A malicious IdP can retain a user's one-time pseudonym the linked PII for a session, but it still cannot profile the user's activities because the next time the user authenticates to the IdP, a new pseudonym will be used, and the IdP cannot link the PII revealed from previous sessions with the new pseudonym, unless the revealed PII themselves leak linkable information - which is why the use of CC instead of AC is crucial.

For negotiation, a software component called *Negotiator* is used. As per requirement 2.2, *Negotiator* should include the relevant risk calculation capability. UA, IdP and SP should have their own negotiator component that they can set according to their preferences.

If Level 1 disclosure is used, additional steps might have to be executed (6). For example, when an SP requires a user's credit card information to complete a purchase and the user is only willing to provide Level 1 disclosure (the use of CC), the user has to first obtain a CC from the bank to certify that the user has a valid credit card to complete the purchase. This is done for the PII that are not static, such as, the available credit balance on the user's credit card account at the time of purchase. On the other hand, some CC that certify static PII can be pre-obtained, such as a CC to certify that a user is at least 18 years of age.

Accountability (8) involves interactions between the user, IdP, SP and Anonymity Revocation Authority (ARA). While a user can be anonymous in UFed, sometimes the anonymity has to be revoked, such as in a security breach investigation. A set of conditions need to be agreed to determine the circumstances under which the user's anonymity can be revoked. Only an ARA can revoke it.

Notation	Description	Notation	Description
$m_a, m_b...$	Data item - plain text	*HiddenSign*	A protocol execution that allows the signer to sign data items, some of which may have been blinded using commitment scheme:S=*HiddenSign*(C,m_j;K_{Sign})
K_{Enc}	Encryption key, public		
K_{Dec}	Decryption key, private		
Enc	An encryption algorithm		
E	Encrypted text,E=*Enc*(m_i;K_{Enc})	*PK*	Proof of knowledge protocol, e.g. PK{(m_a): F($m_a, m_b...m_i$) = 1} The data on the left hand side of the colon m_a is the data item that a Prover needs to prove the knowledge of and which a Verifier will not learn in the process. A function *F*($m_a, m_b...m_i$) is the function to be executed to enable the proof of the knowledge of data item m_a. The data items $m_b...m_i$ are known to the Verifier. The actual protocol involves one or more message exchange(s)
K_{Sign}	Signing key, private		
K_{Verify}	Signature verification key, public		
Sign	Signing algorithm		
VerifySign	Signature verification algorithm		
S	Signature, e.g. S=*Sign*(m;K_{Sign})		
Commit	Commitment algorithm		
r	Random value		
C	Commitment value, e.g. C=*Commit*(m_i,r)		

Table 1. Notation Descriptions

5.2 Security requirements fulfillment

To fulfill requirement 1.2 for communication security, techniques such as encryption and digital signatures should be used to secure and authenticate communication channel. Onion routing is also needed in order to prevent traffic analysis.

To satisfy the user-controlled requirement 1.1 in UFed, a usable UA and protocols that allow enforcement of user privacy requirements in various aspects of UFed operations are needed. The former requires good user interface design for the UA, while the latter requires a set of protocols to be designed for UFed operations.

The application of **private credential system** (Bangerter et al. 2004) can be used to fulfill some of the UFed requirements as it provides several useful properties: partial release of certified information, blind certification, proving relationships between data items, and conditional anonymity revocation. These properties are explained further in this section *informally* to allow non-specialists in cryptography to understand the capability of this credential system.

Assumptions: We assume that public key cryptography is used and there is an existing public key infrastructure (PKI) that can be used by the credential system. The details of the underlying PKI infrastructure is *out of the scope* of this paper. However, a proposal for the certification framework for such a credential system is provided in (Camenisch et al. 2006). The key-size and the encryption/signing algorithms used are of sufficient length and strength to prevent reasonable adversaries from breaking the cryptographic schemes. The details of the underlying cryptographic schemes (including the keying requirements and key generation algorithms) are provided in: the SRSA-CL (Strong RSA) signature (Camenisch & Lysyanskaya 2002), Camenisch and Shoup verifiable encryption (Camenisch & Shoup 2003) and BM-CL (Bilinear Mapping) signature scheme (Camenisch & Lysyanskaya 2004). It is again *beyond of the scope* of this paper to provide such details.

Table 1 describes the notations used[2]. A message for a party X is encrypted using X's public key (publicly known). To decrypt, only X who owns the corresponding private key (only known to X - private) can decrypt it. An X's signature on a message can only be produced using X's signing key (private). Verification of the signature can be done by anybody using the corresponding verification key (public).

The private credential mechanism uses zero knowledge proof interactive protocol (PK) that is

[2]Notations follow those used in (Bangerter et al. 2004)

executed by a Prover and a Verifier. At the end of the protocol, the verifier can be convinced (or not) that the prover has the knowledge of a secret value without the verifier learning it. The value of m_a in the example given in Table 1 is the secret value that a prover needs to prove the knowledge of, and which the verifier will not learn in the process.

A commitment scheme is used in this credential system. A commitment C of a secret value m_i is *hiding*: it does not show any computational information on m_i. It is also *binding*: it is computationally impossible to find another m_i' and r' as inputs to the same *Commit* algorithm that gives a value C' which is equal to C. Commitment scheme is used in the zero-knowledge proof and hidden signature protocols.

$Cert_1$ is a certificate issued by an AI1: $Cert_1 = Sign(m_a, m_b, ...m_j; K_{SignAI1})$. The $K_{SignAI1}$ is private to AI1, while the signature verification key $K_{VerifyAI1}$ is assumed to be publicly known and authenticated. The PII $m_a.....m_j$ are issued by AI1, with m_a as the user pseudonym issued by AI1. Several PK protocols are introduced in this credential system (Bangerter et al. 2004) to provide several useful properties:

Protocol 1 - partial release of PII: This credential system allows a partial release of the PII in a certificate as needed. For example, a user has $Cert_1$ and only wants to release PII $m_g...m_j$. The protocol represented by the following notation can be executed:

$$\text{PK}\{ (Cert_1, m_a...m_f): \text{VerifySign}(Cert_1, m_a...m_f, m_g...m_j; K_{VerifyAI1}) = 1\}$$

Normally, showing a certificate reveals *all* information in the certificate. In contrast, Protocol 1 allows only selected data in $Cert_1$ to be released. Successful execution of Protocol 1 allows user to prove the possession of a valid certificate issued by AI1 and reveals only PII $m_g...m_j$, hence, fulfillment of data minimization requirement 2.1. Users can reveal only a limited amount of their PII during registration, hence, fulfillment of requirement 3.1. If the user's pseudonym m_a is also not revealed, then the user was authenticated anonymously: the verifier only knows that the user is a valid entity known by AI1 without learning the pseudonym. This satisfies requirement 3.2. As a result, if a user shows this certificate multiple times, they are not linkable since m_a is not known.

Protocol 2 - blind certification: Assume that a user wants to obtain an additional certificate from another AI called AI2 based on the user's year of birth (PII m_j) from the primary certificate $Cert_1$. However, the user does not want to reveal the pseudonym and any other data items $m_a...m_i$, while still needing the new certificate to contain the pseudonym m_a. Before AI2 can issue the secondary certificate, it needs assurance that the hidden pseudonym m_a was issued by AI1. This can be achieved by 'blind certification' protocol represented by the following notation:

$$\text{PK}\{ (Cert_1, \text{r}, m_a...m_i): C_a=\text{Commit}(m_a,\text{r}) \wedge \text{VerifySign}(Cert_1, m_a...m_i, m_j; K_{VerifyAI1})=1\}$$

The above protocol allows users to show that C_a is a commitment to the private pseudonym m_a, and a successful execution of it proves that the user owns a valid certificate with a valid pseudonym, while not revealing any PII except m_j. Based on this information, AI2 can issue a new certificate that contains the same pseudonym m_a and add a new data item: a statement that the user is 18 years old or older - PII m_u. A new secondary certificate - called $SecCert_2$ can thus be issued by executing: $SecCert_2$=HiddenSign(C_a, m_u;$K_{SignAI2}$). The user can verify the correctness of the secondary certificate by executing the following: VerifySign($SecCert_2$, m_a, m_u;$K_{VerifyAI2}$) = 1. This protocol is useful in allowing users to obtain multiple certificates while still respecting requirement 2.1 for data minimization, and 3.2 for anonymous authentication.

Protocol 3 - proving relations between PII: This credential system allows the proving of relationships between data items (PII) from various certificates without revealing the value of the data items. This is useful because a user may have to show multiple certificates to reveal a set of PII to a provider and prove that all of the certificates belong to the same user with the same pseudonym without revealing the pseudonym itself. Assume that a user needs to reveal the user's state of residence (PII m_e) from $Cert_1$ and the user is older than 18 years old from PII m_u from $SecCert_2$, while proving that both certificates contains the same pseudonym m_a and still keeping the other PII secret. The protocol represented by the following notation can be executed:

PK{ ($Cert_1$,m_a..m_d,m_f..m_j,$SecCert_2$): VerifySign($Cert_1$,m_a..m_d,m_f..m_j,m_e;$K_{VerifyAI1}$)=1 ∧
VerifySign($SecCert_2$,m_a,m_u;$K_{VerifyAI2}$) = 1}

The above protocol allows the recipient of the certificates to be sure that the user owns both certificates and, by using the technique of demonstrating *relation between data items* (in this case equality of item m_a in both certificates), knows that these two certificates contain the same pseudonym without learning its value. Along with a good negotiation protocol, this mechanism is useful for fulfilling data minimization (2.1) and registration (3.1) requirements.

Protocol 4 - conditional anonymity revocation: Requirement 3.4 states that user anonymity should be revocable. To this end, the protocol represented by the following notation can be used:

PK{($Cert_1$,m_a...m_i): VerifySign($Cert_1$,m_a...m_i,m_j;$K_{VerifyAI1}$)=1 ∧
E = Enc(m_a, *Conditions*;K_{ARAEnc})}

The above protocol proves that the user is the legitimate owner of the certificate, while disclosing only PII m_j. In addition, the pseudonym is encrypted using *verifiable encryption* technique which provides assures that the encrypted text is an encryption of the data item m_a from certificate $Cert_1$. A set of conditions are included to specify the conditions upon which the data item m_a can be revealed. By doing so, users can remain anonymous unless the conditions are fulfilled.

Finally, to fulfill requirement 2.2, *Negotiator* component still needs to be developed. The required negotiation protocol should take into account the trust factor, the type of PII to be disclosed, the disclosure level, the recipient of the data, risks and necessities. Further research is still needed to determine the best way to design this component to support the required features. Similarly, to fulfill requirement 3.3, the best data storage method to handle the storage of user's PII in the AI needs a further research, especially in ensuring the enforcement of user's involvement, such as an explicit consent, before any PII stored can be used in an enforceable manner.

6 UFed Single Sign-on

Two processes are required for UFed SSO: the setup and the SSO process. The UFed SSO is based on the SAML 2.0 SSO protocol. Other similar protocols, such as the Liberty ID-FF SSO protocol (Alliance 2004) can be adapted to support the UFed SSO protocol.

6.1 Setup

The setup process involves users obtaining several primary certificates from one or more AIs. As these are primary certificates, off-line documents, such as birth certificate, can be used as the basis for issuing the certificates. Primary certificates contain the user's pseudonym, which is unique within the realm of the AI that issued the certificate, and each AI's identifier is unique within a federation only. The combination of the user's pseudonym and the AI's identifier that issued the primary certificates provide a unique identifier of a user within that federation.

From the possession of primary certificates, users can obtain a set of secondary certificates by using the blind certification mechanism (Protocol 2). Primary certificates would normally be in the form of AC, while secondary certificates could be in the form of AC or CC.

Assume that a user has $Cert_1$ as defined in section 5.2. If m_a is the user's pseudonym, m_j is the PII to be based on to produce m_u as the data item to be included in the secondary certificate issued by AI2, then the protocol represented by this notation can be executed:

PK{ ($Cert_1$, r, m_a...m_i): C_a=Commit(m_a,r) ∧ VerifySign($Cert_1$,m_a...m_i,m_j;$K_{VerifyAI1}$)=1}
$SecCert_2$=HiddenSign(C_a, m_u;$K_{SignAI2}$)
User verifies the secondary certificate: VerifySign($SecCert_2$,m_a, m_u;$K_{VerifyAI2}$)=1

The above notation shows that AI2 has to verify that m_a was issued by AI1, and create a new $SecCert_2$ by using hidden signing technique to sign the obfuscated m_a and the new data item m_u. User can verify the correctness of $SecCert_2$ by using the AI2's public signature verification key.

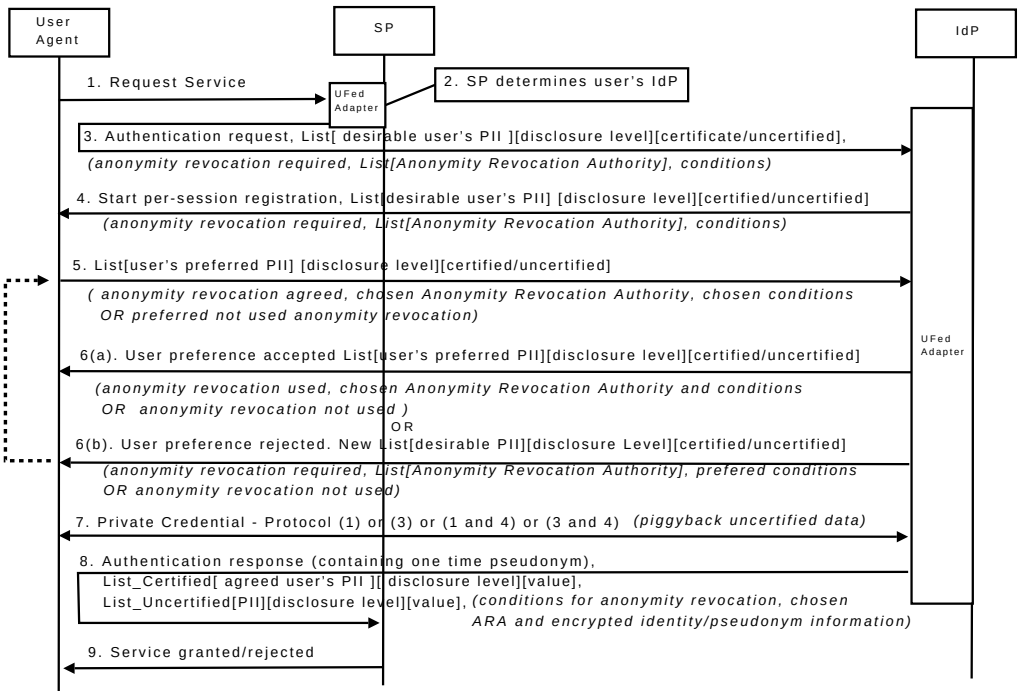

Figure 3. The UFed SSO

The setup process will not be modeled and simulated as it mainly involves the use of Protocol 2 - Blind Certification. The security proofs for this blind certification are provided in their respective concrete implementation schemes: the SRSA-CL (Camenisch & Lysyanskaya 2002) and the BM-CL (Camenisch & Lysyanskaya 2004) signature schemes. Therefore, it will be redundant to do another formal formal security proofs of these signature schemes.

6.2 Single sign-on

The scenario is based on the SAML 2.0 Web Browser SSO Profile (there are other variations of SAML 2.0 SSO). The SSO process (Figure 3) follows closely to the normal SSO protocol. The messages with number 1,2,3, 8 and 9 are the normal SAML SSO messages. Message 3 and 8 may have to be modified to reflect the use of UFed SSO, while message 4 to 7 are UFed-specific. The user is anonymous to IdP and SP throughout the whole process. The user has $Cert_1$ and $SecCert_2$ from the previous setup operation. The detailed message flows are as follows:

1. User requests a service from SP
2. SP determines the user's IdP. There are several methods to do this, but it could be as simple as asking the user to choose from a list of IdPs in the federation that the SP belongs to.
3. SP requests for the user to be authenticated by the chosen IdP by sending an <AuthnRequest> message. If the SP supports the UFed SSO protocol, then the UFed adapter should also include additional information:
   - The SP should indicate that the UFed SSO will be used in the <AuthnRequest> message.
   - SP should specify a list of the desirable user's PII that it wishes to have, the disclosure level (Level 1 or 2 - see section 4), and whether the required information has to be certified or not. This list may or may not be respected by the user and the IdP.
   - Optionally, the SP can request for the user's anonymity to be revocable by providing a list of the Anonymity Revocation Authority (ARA) that it has relationships with and

acceptable to be used within that federation, as well as a set of preferred conditions under which the user's anonymity can be revoked.

4. At the IdP side, the UFed Adapter intercepts the normal protocol flows. The user's and IdP's negotiator component start a per-session registration negotiation process. The IdP assesses the required PII from the SP and generates its own list of PII requirement for this per-session registration which may or may not be the same with the one form the SP. Similarly, if the IdP decides that user's anonymity revocation is acceptable, it will include the list of ARA and the conditions upon which the anonymity can be revoked that the SP provided.
5. Upon receiving the preferred PII list, the user should assess the list and determine its own preference on PII to be released, their disclosure level, and whether the PII should be certified or not. Users should be able to state their preference to have the anonymity revocation enabled (or not), the chosen ARA, as well as the preferred conditions for anonymity revocation. The user preference is sent to the IdP.
6. There are two possible scenarios:
   (a) The IdP accepts the user's preference, thus proceeding to the next step.
   (b) The IdP rejects the user's preference. In this case, the IdP should take into consideration the user's preference and create a new IdP preference list (the PII, disclosure level, certification requirement, and others). At this point, the IdP's new preference should be sent back to the user, and the protocol goes back to step 5 again. This negotiation process can be done over several cycle with a pre-determined maximum negotiation cycle.
7. When a set of PII and other related information are agreed, the user and the IdP should start one of the private credential protocols as mentioned in section 5.2. One or more of the private credential protocols can be executed:
   (a) If the agreed set of PII require only one certificate (for example PII m_e to m_h from $Cert_1$), then Protocol 1 (partial release of PII) can be executed:

   PK{ ($Cert_1$, m_a...m_d, m_i...m_j):
   VerifySign($Cert_1$, m_a...m_d,m_i...m_j, m_e...m_h; $K_{VerifyAI1}$) = 1}

   This protocol requires the IdP to verify that $Cert_1$ was issued by AI1 without learning any PII (including the pseudonym m_a) except those that were agree to be revealed.
   (b) If several certificates are needed (requiring data item PII m_e to m_h from $Cert_1$ and m_u from $SecCert_2$), then Protocol 3 (proving relations between data items) can be executed to verify that all certificates belong to the same user:

   PK{ ($Cert_1$, m_a..m_d,m_i..m_j, $SecCert_2$):
   VerifySign($Cert_1$,m_a..m_d,m_i..m_j,m_e..m_h; $K_{VerifyAI1}$) = 1}∧
   VerifySign($SecCert_2$, m_a, m_u; $K_{VerifyAI2}$)=1}

   The above notation shows that the IdP verifies that $Cert_1$ was issued by AI1 and $SecCert_2$ was issued by AI2, and that both certificates contain the same pseudonym m_a without learning any value except those that were agreed to be revealed.
   (c) If there should be a provision for anonymity revocation, then the combination of either Protocol 1 and 4 (conditional anonymity revocation) can be executed:

   PK{ ($Cert_1$, m_a..m_d, m_i..m_j):
   VerifySign($Cert_1$,m_a..m_d,m_i..m_j,m_e..m_h;$K_{VerifyAI1}$) = 1} ∧
   E = Enc(m_a,*Conditions*;K_{ARAEnc}) }

   or Protocol 3 and 4:

   PK{($Cert_1$, m_a..m_d,m_i..m_j,$SecCert_2$):
   VerifySign($Cert_1$,m_a...m_d,m_i...m_j,m_e...m_h;$K_{VerifyAI1}$) = 1}∧
   VerifySign($SecCert_2$, m_a, m_u; $K_{VerifyAI2}$) = 1} ∧
   E = Enc(m_a, *Conditions*; K_{ARAEnc}) }

The above notations show that in addition to what IdP verifies as per 7(a)) and 7(b), the IdP also has the encrypted text of the user's pseudonym. If the *Conditions* were fulfilled, the ARA can decrypt text and revoke the user's anonymity.

If some uncertified PII have been agreed, they should be piggybacked with one of the above protocol messages.

8. Upon successful execution of the above protocol, the IdP should assign a one-time pseudonym for the user. The IdP returns an authentication response message containing a one-time pseudonym and the user's PII released from the previous step to the SP. If anonymity revocation is used, the authentication response message should include the conditions upon which the anonymity can be revoked, the encrypted user identity, and the chosen ARA.
9. Based on the received authentication response, SP decides to grant/refuse services to user.

If a user who has completed the above UFed SSO goes to other SPs in the federation, step 4 to 7 are not executed anymore, as long as the user session with the IdP is still active. This increases the efficiency of UFed SSO process.

The UFed SSO allows a logical separation of the roles between AI and IdP. The user's identifier is only known to the AI who originally issued it, but during the transactions, the identifier is not known to the IdP. Even if AI and IdP are the same physical entity, the use of anonymous authentication only allows the IdP to know that the user is known by itself, but not the exact user identity. However, it is equally important that the revealed PII in the per-session registration do not leak any linkable information to IdP. Paranoid use of CC instead of AC would help.

6.3 Formal Model of UFed SSO

A formal model is needed to verify, with a high assurance, that the UFed SSO protocol proposed in section 6.2 can achieve the key privacy goals. It is also needed to identify the main factors that may jeopardize the overall security and privacy goals of UFed.

Goals of Formal UFed SSO Modeling The main goals that UFed tries to achieve, and which the formal model of the UFed SSO proves are:

1. **Goal 1**: User's anonymity throughout multiple SSO sessions is maintained - assuming that no anonymity revocation has taken place.
2. **Goal 2**: The ability of IdPs and SPs to link up users' information gathered from multiple SSO sessions to profile user activities is reduced to a minimum. While user's identity may not be revealed, other revealed PII may be unique enough to allow information correlation.

The main focus on the formal model of the UFed SSO is to model the protocol described in section 6.2, and to simulate the protocol execution to prove the achievement of the goals mentioned.

6.3.1 Assumptions and Limitations of the UFed SSO CPN Model

We assume that a federation has been setup between the IdP and SP. Members of the federation have agreed on the name of every possible PII labels and the corresponding format (data type) for PII values. For example, members of the federation will use 'DOB', instead of 'Date of Birth' for the PII label. The format of the value of 'DOB' is 'DD/MM/YYYY' instead of 'MM/DD/YYYY'. This assumption is essential to ensure that the attributes can be understood and to enable meaningful PII negotiation during the per-session registration stage.

We assume that there are working rules that govern how a negotiation should be conducted and how an agreement can be reached - see section 7 for further discussion on this issue. Members of the federation have agreed on a list of Anonymity Revocation Authorities (ARA) that can be used within that federation. It is assumed that some communication security mechanisms are employed to counter Threat 1 (section 3).

Limitations The UFed SSO CPN models the *essential* protocol messages and flows only to ensure that the model is accurate enough to capture the UFed SSO and is still simple enough to be understandable without the unnecessary complications of inconsequential details. To keep the model manageable, not every aspect of the UFed SSO protocol is modeled. Therefore, the UFed SSO CPN model has some limitations, though these limitations do not compromise the evaluation of the goals of the UFed SSO protocol.

The simulation of the negotiation protocol leaves many of the negotiation process details out while still retaining the essence of what a negotiation process should provide. The actual negotiation should take into account the risk, trust, necessities and other factors as detailed in section 4. However, for simplicity, the behavior of the negotiation protocol in this simulation is randomized. This should not affect the final evaluation result because the key feature that this model needs to show is how the negotiation results in a set of agreed PIIs to be disclosed, the disclosure level, and whether certified or uncertified PII are required. The limit of negotiation cycle (see step 6b of the UFed SSO) is not modeled. Instead, a random behavior of IdP is set to accept or reject user's preference with equal probability.

The UFed SSO CPN only models the end-result of the private credential protocols execution. That is, after a set of PII and other information has been negotiated, the model hides the operation of the private credential protocol and simulate only the end-result of executing such a protocol: the selected certified data items as negotiated previously are released to the IdP. The security proofs of this private credential protocol has already been detailed in (Camenisch & Lysyanskaya 2002, 2004).

Uncertified data is not included in the model as they may be inaccurate, hence, not useful for the IdP and SP for information correlation and thus not a serious threat to privacy.

6.3.2 UFed SSO CPN Model

The model is generated using the Coloured Petri Net Tool (CPN-Tool). CPN-Tool is a tool that can be used to edit, simulate and analyze a Coloured Petri Net. The model for the UFed SSO protocol is divided into four main sections: the declarations, main UFed SSO model, detailed model of the operations involved in processing an authentication request (which includes the negotiation process and the per session registration), and finally, the detailed model of the operations required on the user side to determine its PII disclosure preference.

Symbols in Coloured Petri Nets The states of a system are represented by ellipses or circles called *places*. The actions applied to the system are represented by rectangles called *transitions*. The *arcs* and the *arc expressions* describe how the state of a system changes from one state to another. Each *place* contains zero or more *tokens* - represented by a number at the edge of the corresponding *place*.

Each token contains data value of one of the types that have been declared (see Figure 4 - top box). *Places* with token(s) represent the state of the system at any given time. For token(s) to move from one *place* to another, a *transition* must be fired. A *transition* is enabled when the *place(s)* that act as the inputs to the *transition* have sufficient token(s) that match the *arc expressions*.

Declarations Refer to Figure 4 (top box) for data type declaration. The data type `LEVEL` refers to the disclosure level of a PII value, as described in section 4. The data type (or in CPN terminology, the colour set - hence 'colset') `A_LEVEL` refers to the disclosure level of an PII that a user deems acceptable. The data type `PII_SET` refers to a structure that consists of: the PII identification, the disclosure level for that PII, and whether that PII should come from a certified or uncertified source. This data set is used heavily during the negotiation process. For the purpose of this simulation, we use five different PII, each representing various attributes of a user. In this model, *the PII with the ID of **1** represents a user's actual identity/pseudonym.* A set of `PII_SET` is collected into a list with the data type `PII_SET_LIST`.

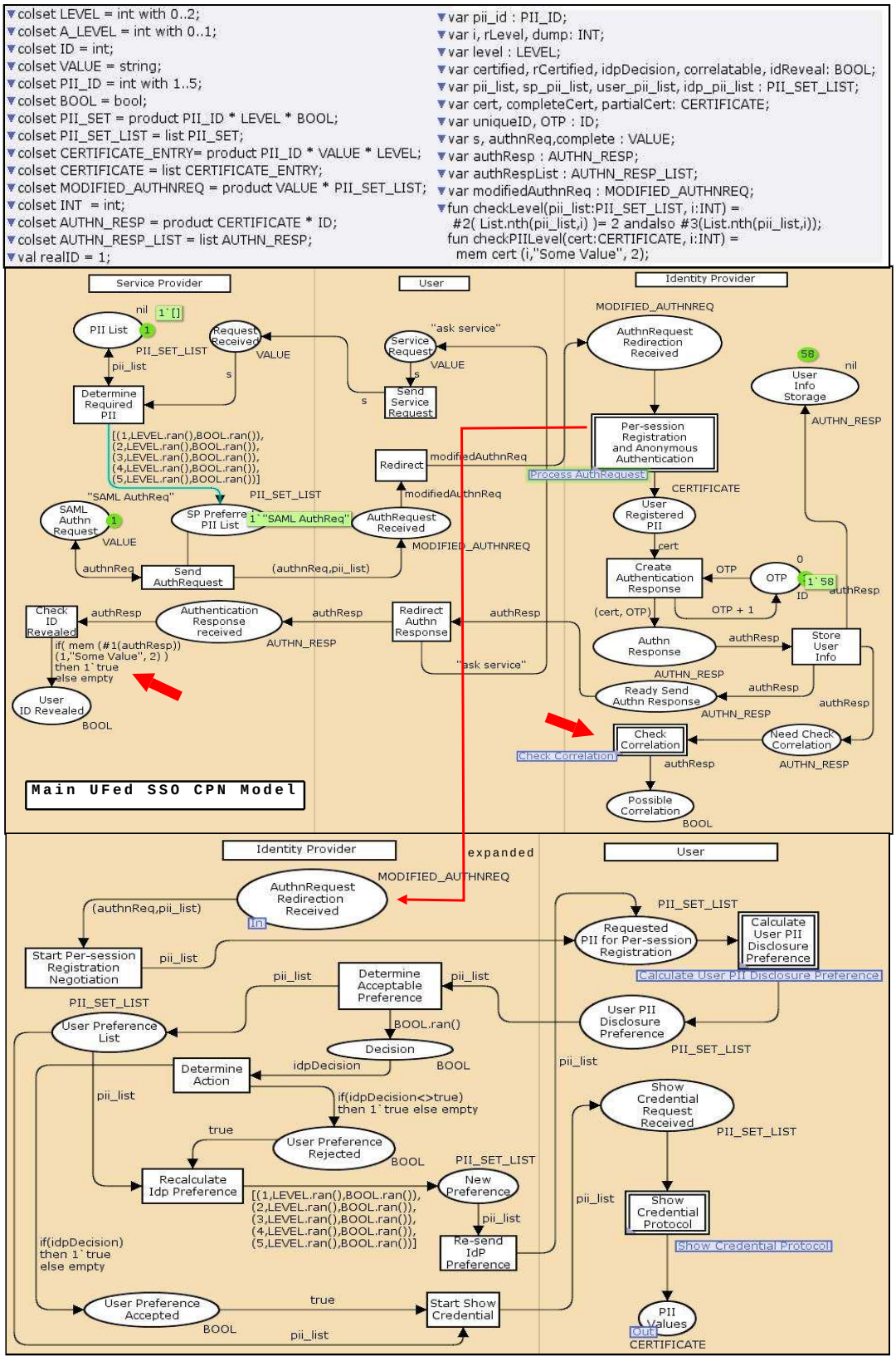

Figure 4. Main UFed SSO Model and Declarations

`CERTIFICATE_ENTRY` refers to an entry of a certified PII data, which consists of the PII identification, its value, and the disclosure level. A set of certified PII data creates a `CERTIFICATE.`

The data type `MODIFIED_AUTHREQ` refers to the SAML 2.0 authentication request message. However, in UFed, the authentication request message from an SP to an IdP may include a list of preferred PII and its related information. Therefore, the original authentication request message has to be modified to depict the UFed authentication request message.

Similarly, the data type `AUTHN_RESP` refers to the original SAML 2.0 authentication response message. However, in the UFed SSO CPN model, only the important element of this response message that is included: the one time pseudonym that the IdP generates, as well as the certified information that is included in the authentication response message. The actual authentication response message contains other supplementary information.

Several variables are declared. These variables are those that will be used in the UFed SSO CPN model. Each variable is declared according to a data type that has been specified earlier. For example, the variable 'authResp' has been declared to be of type `AUTHN_RESP`.

There are also two functions declared: `checkLevel` and `checkPIILevel`. These functions will be explained at their appropriate point in the later discussion of the UFed SSO CPN model.

Main UFed SSO CPN Model Figure 4 (middle box) shows the main diagram for the UFed SSO CPN model. The model is divided into three sides: the User, Identity Provider, and Service Provider. The model starts at a *place* named 'Service Request' at the 'User' side. The user sends a service request to the SP. This corresponds to step 1 of the UFed SSO protocol (see section 6.2).

Step 2 in the UFed SSO protocol is not modeled as the process that an SP conducts to determine a user's IdP is inconsequential to the main goals of the UFed SSO protocol as described earlier.

Step 3 is modeled through several *transitions*: the 'Determine Required PII' and 'Send AuthRequest' transitions. The authentication request is represented by the variable 'authReq', which represents the normal SAML 2.0 Authentication Request message, with minor modification to indicate that a UFed SSO is to be used. The list of the desirable user's PIIs, their disclosure level, their certification status are represented by the variable 'pii_list' (type `PII_SET_LIST)`. The behavior of how the SP determines the preferred user's PII is randomized[3]. This model assumes that no anonymity revocation is used. The authentication request from the SP is redirected to the IdP through the user.

Step 4 to 7 are encapsulated in the 'Per-session Registration and Anonymous Authentication' *transition*. This *transition* is invoked after the IdP receives the authentication request from the SP. The detail of this *transition* is explained later. At this point, it is assumed that the transition will output a collection of certified PII data which are now registered to the IdP. These certified PIIs and their corresponding disclosure level are the result of the negotiation process and the execution of the private credential protocol. They are represented by the variable 'cert' (type `CERTIFICATE).`

Step 8 is modeled by the *transition* 'Create Authentication Response'. For simplicity, the authentication response is modeled as containing the 'cert', and a One-Time-Pseudonym (OTP). The one-time property of the pseudonym is modeled in such a way that for every new authentication response that the IdP creates, the OTP will be the value of the 'previous OTP + 1', starting from the value of 0 - in practice, we would want the OTP to be long enough, consisting of digits and alphabets, to ensure uniqueness. The authentication response is then sent to the SP, redirected through the user.

Step 9 is modeled by the transition 'Determine Service Granted/Rejected' at the SP side. Upon receiving the authentication response, the SP should consult its own access control policy to determine if the user is allowed access to the requested service or not.

[3]An entry in the variable 'pii_list' (2, LEVEL.ran(), BOOL.ran()) means that for PII with the ID 2 (which represents an attribute name), the preferred disclosure level is randomized (which could contain the value from 0 to 2 as we have declared the data type LEVEL to range from 0 to 2 only). Similarly, whether we want that attribute to be certified or not is also randomized

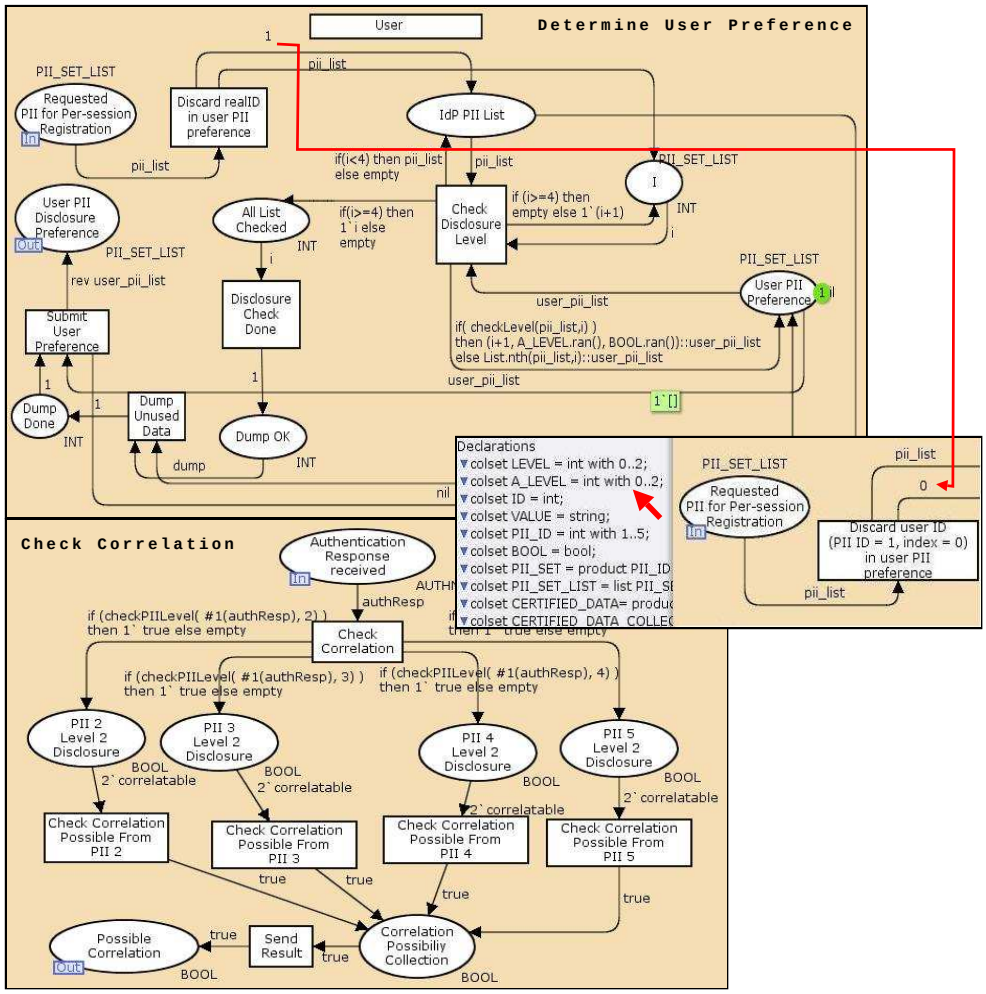

Figure 5. UFed SSO CPN Model Subprocesses

UFed Negotiation and Per-session Registration Step 4 to 7 of the UFed SSO protocol is encapsulated in the 'Per-session Registration and Anonymous Authentication' transition. Figure 4 (bottom box) shows the expanded CPN Model for this transition.

In step 4, after the IdP receives the authentication request and the SP's PII preference, it will start the per-session registration negotiation. The IdP will take into account the SP's PII preference in determining the preferred PII for per-session registration. In this model, we assume that the IdP is satisfied with the SP's PII preference and just send that PII list ('pii_list') to the user.

In step 5, after the user receives the request to start per-session registration and the list of the PIIs that the IdP would like the user to reveal, the user will determine his/her own PII disclosure preference. How the user should determine his/her PII preference is *crucial* in achieving the main goals of UFed SSO. A detailed model of how the user can achieve this is encapsulated in the transition 'Calculate User PII Disclosure Preference'. This transition will be explained in detail later. At this point, assume that the transition will result in another PII list that expresses the user's preference. This PII list will also have the data type of `PII_SET_LIST` . This list will then be sent to the IdP.

There are two possible variations in step 6. After the IdP receives the user's PII disclosure preference, it will decide whether to accept the user's preference or not. If it is accepted, then step 6a will be executed (the transition 'Determine Action' will move to the 'User Preference Accepted' state). The IdP will keep this PII preference list and continue the SSO process. Otherwise, if the IdP rejects the user's PII disclosure preference, then the 'Determine Action' transition will move

to 'User Preference Rejected' state, and then move to 'Recalculate IdP Preference' state. The new IdP preference will be re-sent to the user (back to step 5).

In our model, the behavior of how IdP determines whether to accept or reject the user's preference is randomized with equal probability. Similarly, how the IdP re-calculates its PII preference is also randomized. In practice, we would want the IdP to take into consideration the original SP PII preference, the user's PII preference, and other factors in determining the new preferred PII list.

After a set of PII has been agreed, step 7 will be executed. Step 7 mainly involves the 'Private Credential Protocol' transition, which will trigger one of the private credential protocols (as explained in section 5.2). As we assumed that no anonymity revocation is required, either protocol 1 or 3 can be executed. As mentioned previously, the detailed private credential protocol will not be explained because the security proof of such a protocol has been shown before in their respective papers - see (Camenisch & Lysyanskaya 2002, 2004). This transition, in short, should be taken as a black box.

However, what is modeled here is the end result of executing such a protocol: taking in the input of the set of agreed PII to be released (as a result of the negotiation done previously), the private credential protocol executed between a user and an IdP will output a collection of certified data according to the agreed disclosure level for each of the agreed PII to the IdP. The output will be of data type `CERTIFICATE`, which is represented by the variable 'cert'.

If there are some uncertified data required, then the output of this transition will also include a collection of the uncertified data. However, as mentioned previously, uncertified data is not of our concern, and thus not modeled. The IdP will then register these released data about the user and continue to step 8.

Determining User PII Disclosure Preference Figure 5 (top box) shows the detailed modeling of the calculation at the user's side to determine the PII disclosure preference in a secure manner. After receiving the IdP's PII disclosure preference, the first filter that the user has to do is to discard the request to disclose the user's actual identity/pseudonym (that is, the PII with the ID of 1). This is reflected in the transition 'Discard real ID in user PII preference'. What this transition does is to prevent the inclusion of the PII ID 1 in the final user PII preference list by starting the user's PII preference from the PII with ID 2. This is reflected by the output arch from the transition to the place with the name 'Counter' that contains the **index** value of 1 - index value of 0 refers to PII ID 1, while index value of 1 refers to PII ID 2, and so on.

The next filter that the user needs to do is to check the requested disclosure level for each entry in the variable 'pii_list' starting from PII ID 2. The transition 'Check Disclosure Level' calls the function 'checkLevel' that has been declared previously (see Figure 4 - top box). This function simply checks if there is any request that requires disclosure level 2 *and* requires certified data. If it is true, then the user changes its disclosure preference level to the 'acceptable' one - indicated by randomizing the preferred disclosure level from the data type `A_Level` (either 0 or 1).

Otherwise, if there is no request for level 2 disclosure of certified data for that particular PII ID, then we take the original IdP preference for that particular PII ID entry. This process is done repetitively until we reach the final PII ID 5. As this transition is meant to be stateless, some unused data are discarded and the initial state is restored.

At this point, the user's preference calculation is completed. The calculated user PII disclosure preference is then sent to the IdP (as per step 5).

6.3.3 UFed SSO CPN - Attacker Model

The attacker model is modeled following the Threat model as stated in section 3. However, we only model Threat 2 for the following reasons:

- As mentioned in section 6.3.1, we assume that some communication security mechanisms are employed to prevent Threat 1. Therefore, the attack model from Threat 1 will not be included.
- How we counter Threat 3 is already implicitly included in the 'normal' UFed SSO model. That is, how can we prevent the user from providing false PII is done by requiring certified data.

When the IdP finally agrees to the user PII preference, it is assumed that the IdP is satisfied that some of the important PII - which the users may have motivation to falsify - have been agreed to be certified.

Therefore, the attacker model will focus on Threat 2: attacks from the IdP and SP. In line with the main goals of UFed SSO that we aim to prove (see section 6.3.2), there are two attacks that the IdP and SP do at the end of the UFed SSO:

1. Attack 1: The IdP and/or SP will try to retrieve the user's actual identity (PII ID of 1) from the collection of certified PII released to the IdP in the authentication response message for malicious purpose.
2. Attack 2: Even if the user's ID is not revealed to them, the IdP and/or SP will try to find ways to correlate the user's PII released to them (from PII ID 2 to 5).

The attacker model has been included in Figure 4 (middle box) - see the bold arrows. While both of the attacks can be done by both the SP and IdP, the model shows that the SP conducts Attack 1, while the IdP conducts the Attack 2.

At the SP side, the Attack 1 is modeled as follows: the SP, at step 3 will always require certified the level 2 disclosure for user's actually identity (PII ID 1) in its PII preference list. At the end of the SSO where the SP receives the authentication response from the IdP, it will try to retrieve the user's actual identity information (modeled by the transition 'Check ID Revealed'). If it is successful then a token will be added to the place named 'User ID Revealed'.

At the IdP side, the Attack 2 is modeled as follows: from the result of the negotiation, the IdP will store all of the received user's information, and then try to find a way to make correlation from the received information. How an IdP could possible correlate received information is detailed in the transition 'Check Correlation' (Figure 5 - bottom box).

For every received authentication response, the IdP will find if there is any *certified* PII (from PII ID 2 to 5) that has level 2 disclosure. If there is, then a token will be added to its relevant place, e.g. 'PII 2 Level 2 Disclosure'. As long as there are two tokens at this place, then we claim that it is possible for the IdP to start doing some correlation of information. For example, assume PII ID 2 refers to a user's residential address. When the IdP has two records, each containing the same address value, this address can then be used as the 'key' to link up all other information. If correlation is possible, a token will be added to the place named 'Possible Correlation'.

6.3.4 UFed SSO CPN - Evaluation Result

To evaluate the protocol against the main goals stated in section 6.3, we need to determine the insecure state(s) of the UFed SSO CPN model. From the description that has been forwarded so far, it is obvious that the insecure states are when there are tokens in two places:

1. Place with name 'User ID Revealed'. If there is at least one token in this place at the end of the simulation, then **Goal 1** is unachievable.
2. Place with name 'Possible Correlation'. If there is at least one token in this place at the end of the simulation, then **Goal 2** is unachievable.

Evaluation Result: Achievement of Goal 1 and Goal 2 Figure 4 (middle box) shows the simulation result. The simulation was run to achieve roughly 58 SSO sessions - this is represented by the number of token in the 'User Info Storage' place as at the end of every SSO session, a token will be added to that place. Based on how we have modeled the UFed SSO, the simulation result shows that the two insecure states *cannot be reached* (no tokens), and thus, the UFed SSO protocol is considered to be able to achieve the two main goals: to protect users' anonymity and to prevent correlation of their information.

Avoiding Insecure States We have made some changes to the model to show how UFed SSO protocol *can be insecure*. Figure 5 (middle box) shows two minor changes that have been made that

will cause the system to go into insecure states. The first modification is the data type `A_LEVEL` declaration, which now ranges from disclosure level 0 to 2. The second modification is when the user does not correctly filter out the request for certified level 2 disclosure of their identity PII - it is shown by starting the user's PII preference from index 0 (which refers to PII ID 1 - the user's real identity/pseudonym).

After applying the modifications, we run the simulation again (this time amounting to 57 SSO sessions), the UFed SSO protocol arrived at the insecure states: the 'User ID Revealed' and 'Possible Correlation' places, are filled with tokens (not shown in Figure 4).

6.3.5 Security Conclusion of the UFed SSO Protocol

Based on the formal modeling and simulation of the UFed SSO protocol, it is obvious that the security of the protocol relies on *how well the users manage to negotiate the required PII to be disclosed and how strict the user is in determining their PII disclosure preference.*

The private credential mechanism as detailed in section 5.2 provides the necessary tool to enhance users' privacy by giving them the control to release only certain data items in a certificate, and to show the relationships between data items. However, there are many implementation issues that need to be considered to maintain the users' privacy in such an SSO system.

From the result of the simulation, we conclude that while the private credential mechanism is indispensable for achieving a secure and privacy-respecting UFed SSO, the negotiation process is just *equally* important. There are several issues that need to be considered in developing a fair negotiation process, a point that will be discussed later in section 7.

7 Discussion of UFed

In this section, several issues related to UFed's operations will be discussed: problems related with using one-time pseudonym (OTP), how the design of UFed could prevent SPs and IdPs from illegally correlating user's information, user's negotiation power, and preventing abuse of the certified PII mechanism by providers.

One Time Pseudonym and accountability: One-time pseudonym could pose a problem in the accountability area. Requirement 3.4 states that user's activities should be logged. However, since one-time pseudonym is used, the linking of the log information (which is only linked with one-time pseudonym) to the user's actual identity could be problematic. The solution is straight forward: if accountability is not important, requirement 3.4 states that it should not be logged. Otherwise, the execution of Protocol 4 at step 7 during SSO stage would provide a verifiable encrypted text of the user's actual pseudonym. This text is stored by the SP, and should be linked to the one-time pseudonym for that session and other log data. When anonymity revocation is needed, the SP could find the cipher text linked with the one time pseudonym and request an ARA to decrypt it, given that the ARA is satisfied that the conditions linked to the anonymity revocation have been fulfilled.

One Time Pseudonym and account linkage: FSSO allows *existing* user's account to be linked between providers: given that a user already has accounts with both IdP and SP, these accounts can be linked so that users can be identified according to the original identifiers they use at each site. Account linkage is privacy-intrusive. IdP and SP have to know the actual value of the user's pseudonym. Account linkage violates requirement 3.2, and it has a severe implication to user's privacy. With account linkage, it is technically trivial for IdP and SP to collude and link users information (Jøsang et al. 2007). The only guarantee that IdP and SP will not collaborate to link users information is a mere 'trust' (Jøsang et al. 2007). In UFed, account linkage is not possible due to the use of one-time pseudonym and anonymous authentication. It is a deliberate decision to make account linkage unachievable in UFed.

One Time Pseudonym and Profiling: Without account linkage, gathering and linking of users information to create users profiles are made harder. Both IdP and SP do not know the users' real identity due to anonymous authentication, and the use of one time pseudonym disables the

IdPs and SPs ability to deduce if any two or more sessions were performed by the same user since the users are anonymous. If the IdPs and SPs collaborate, they can only, at most, profile a user's activities for that particular session only, as for the subsequent sessions the user will be known with different one time pseudonyms.

However, as shown and simulated in section 6.3.3 and 6.3.4, the revealed PII could unintentionally reveal linkable information. For example, it is common for providers to link user's information based on the user's address, name and date of birth. While the user's pseudonym is not revealed, other revealed information, when combined, could be just as identifying.

The key to prevent this from happening is a paranoid use of level 1 disclosure. Users also have to be selective in the disclosure of any PII, regardless of the disclosure level. The behavior of the negotiator component plays a key role in this respect.

Even if the revealed information do not leak any strong user-identifiable information, the AI has the user's actual pseudonym and other identifiable information. However, since each AI issues a different pseudonyms to a user, the ability to link the information based on pseudonym has been decreased. But similar to previous case, user's name, address and date of birth, if they are known to the AIs, could still uniquely identify a user. However, the separation of role between AI and IdP (as explained in section 6.2) is such that the AIs would only know the user's PII, but not able to track their activities, and thus limiting the ability to profile user's behavior, such as shopping preferences. However, for best privacy, technical solutions to prevent AIs linking information are needed.

One potential solution is to enable the UA to intelligently divide the disclosure of users' PII at each AI in such a way that combination of PII values at any AI do not result in a strong user-identifiable information. Alternatively, a secure data storage can be used whereby the stored user's information can only be used with an explicit user consent (requirement 3.3). Both approaches require further research to produce concrete solutions to this problem.

One Time Pseudonym and further attribute request: In the UFed SSO, SP can only provide services to users based on the known users attributes that IdP provides in step 8. If additional information is required, the SP has to obtain them from the user. If the user already has a certificate that can be shown, Protocol 1 or 2 can be executed to show some of the credentials in the certificate between the user and the SP.

However, some attributes need to be obtained real time. For example, a service provider may want to be sure that a user's account balance is greater than a certain amount at the time a transaction takes place. The 'simple' way is to have the SP contact the user's AI - a bank in this case - directly. However, that requires the revelation of the user's actual identity to the SP before it can ask the bank of the user's account balance status. This is what we want to avoid.

Therefore, the user may probably have to go to the relevant AI to obtain a new certificate to certify that the user's account balance is indeed above certain amount, real time. The new certificate should **not** be linked to the one time pseudonym that the user is known at the SP side as the AI will then be able to link the one time pseudonym with the user's actual identity (since the user has to most likely reveal the identity to the AI to obtain the new certificate). A proper protocol needs to be designed to achieve such a capability.

Negotiation and certified PII abuse: Users are generally powerless in deciding whether a PII is to be revealed or remained private. Frequently, it is either the user provides the required PII or accepts refusal of service. The negotiator component and the design of UFed provide the technical solutions to reduce this power imbalance. This shift is reflected in the European government-funded identity management system PRIME where the user's ability to negotiate is built in its design.

Without the negotiator component, the use of the credential mechanism as introduced in section 5.2 could potentially put users at a disadvantaged position. Providers (IdPs and SPs) could take full advantage of this system by requiring users to provide *only* certified PII. An unlikely opposite situation where the providers have less power than the users is just as undesirable because users would most likely abuse the situation for malicious purposes.

As argued in section 6.3.5, one of the key factors in achieving the privacy goals of the UFed SSO protocol lies in the negotiation process. The use of negotiator component is expected to put an enforceable limit to the abuse as explained previously. The negotiation process should be a fair one.

An ideal behavior is to allow the user to provide either certified Level 1 disclosure of the required PII or to provide the uncertified PII value, both are supported by UFed. Alternatively, there should be a standard on how negotiation process should be conducted in an Internet environment to achieve a fair and balanced result (Suriadi et al. 2007). Either way, such a negotiator component is yet to be developed.

Anonymity Revocation: The conditional anonymity revocation (see Protocol 4 in section 5.2) may be problematic. The fulfillment of the conditions in this scheme is not cryptographically enforceable other than through *trust* that the authority will be able to satisfactorily assess the fulfillment of them, and that it will only revoke the anonymity when the 'conditions' are met (Bangerter 2004). The lack of conditions fulfillment enforcement may lead to the abuse of this capability. Further research is required to solve this problem.

8 Conclusion and Future Work

In this paper, UFed is described as a potential solution for providing a better identity management system to allow user-controlled privacy. UFed is aimed to provide the conveniences of an FSSO system, while still being user-centric to enable user-controlled privacy. The formal CPN modeling of the UFed SSO protocol shows that while the use of the private credential mechanism as proposed by Bangeter et al (Bangerter et al. 2004) is important in enabling a user-controlled privacy, the negotiation process is just as important. To our knowledge, this is the first time that the CPN modeling has been used to model privacy in an identity management system.

Some important future works will be in the area of developing the negotiator component so that the result of a negotiation will be fair and balanced. The protocol to enable secure and privacy-respecting attribute request from the SP to the user needs to be developed, as well as the secure data storage mechanism. The anonymity revocation mechanism needs to be improved to prevent abuse. How to improve the efficiency of the UFed SSO protocol requires further research as the execution of private credential mechanism is performance taxing, and may not work well with mobile agents such as the UA component.

References

Alliance, L. (2004), *Liberty ID-FF Architecture Overview Version 1.2-errata-v1.0*. Last access 15 January 2008 from: `http://www.projectliberty.org/liberty/content/download/318/2366/file/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf`.

Aly, S. & Mustafa, K. (2004?), Protocol verification and analysis using colored petri nets, Technical Report TR-04-003, School of Computer Science, Telecommunications and Information Systems (CTI) - DePaul University. `http://www.cti.depaul.edu/research/Pages/TechnicalReports.aspx`.

Bangerter, E. (2004), A cryptographic framework for the controlled release of certified data (transcript of discussion), *in* B. Christianson, B. Crispo, J. A. Malcolm & M. Roe, eds, 'Security Protocols Workshop', Vol. 3957 of *Lecture Notes in Computer Science*, Springer, pp. 43–50.

Bangerter, E., Camenisch, J. & Lysyanskaya, A. (2004), A cryptographic framework for the controlled release of certified data, *in* B. Christianson, B. Crispo, J. A. Malcolm & M. Roe, eds, 'Security Protocols Workshop', Vol. 3957 of *Lecture Notes in Computer Science*, Springer, pp. 20–42.

Bhargav-Spantzel, A., Camenisch, J., Gross, T. & Sommer, D. (2006), User centricity: a taxonomy and open issues, *in* A. Juels, M. Winslett & A. Goto, eds, 'Digital Identity Management', ACM, pp. 1–10.

Billington, J., Gallasch, G. E. & Han, B. (2004), 'A coloured petri net approach to protocol verification.', *Lectures on Concurrency and Petri Nets: Advances in Petri Nets — Volume 3098 of Lecture Notes in Computer Science / Jrg Desel, Wolfgang Reisig, Grzegorz Rozenberg (Eds.)* pp. 210–290.

Camenisch, J. & Lysyanskaya, A. (2002), A signature scheme with efficient protocols, *in* S. Cimato, C. Galdi & G. Persiano, eds, 'SCN', Vol. 2576 of *Lecture Notes in Computer Science*, Springer, pp. 268–289.

Camenisch, J. & Lysyanskaya, A. (2004), Signature schemes and anonymous credentials from bilinear maps, *in* M. K. Franklin, ed., 'CRYPTO', Vol. 3152 of *Lecture Notes in Computer Science*, Springer, pp. 56–72.

Camenisch, J., Shelat, A., Sommer, D., Fischer-Hübner, S., Hansen, M., Krasemann, H., Lacoste, G., Leenes, R. & Tseng, J. C. (2005), Privacy and identity management for everyone, *in* V. Atluri, P. Samarati & A. Goto, eds, 'Digital Identity Management', ACM, pp. 20–27.

Camenisch, J. & Shoup, V. (2003), Practical verifiable encryption and decryption of discrete logarithms, *in* D. Boneh, ed., 'CRYPTO', Vol. 2729 of *Lecture Notes in Computer Science*, Springer, pp. 126–144.

Camenisch, J., Sommer, D. & Zimmermann, R. (2006), A general certification framework with applications to privacy-enhancing certificate infrastructures, *in* S. Fischer-Hübner, K. Rannenberg, L. Yngström & S. Lindskog, eds, 'SEC', Vol. 201 of *IFIP*, Springer, pp. 25–37.

Floreani, D. J., Billington, J. & Dadej, A. J. (1996), Designing and verifying a communications gateway using coloured petri nets and design/cpn, *in* 'Proceedings of the 17th International Conference on Application and Theory of Petri Nets', Springer-Verlag, London, UK, pp. 153–171.

Harper, J. (2006), *Identity Crisis - How Identification Is Overused and Misunderstood*, Cato Institute, Washington D.C., USA.

Huber, P. (1991), A formal executable specification of the ISDN basic rate interface., *in* 'Proceedings of the 12th International Conference on Application and Theory of Petri Nets, 1991, Gjern, Denmark', pp. 1–21. NewsletterInfo: 39.

Jensen, K. (1997), *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use*, Vol. 1 of *Monographs in Theoretical Computer Science*, 2nd corrected printing edn, Springer-Verlag, Berlin.

Jøsang, A. & Pope, S. (2005), User centric identity management, *in* A. Clark, K. Kerr & G. Mohay, eds, '4th Asia Pacific Information Technology Security Conference Refereed R&D Stream (AusCERT 2005)', Gold Coast, Australia, pp. 77–89.

Jøsang, A., Zomai, M. A. & Suriadi, S. (2007), Usability and privacy in identity management architectures, *in* 'ACSW '07: Proceedings of the fifth Australasian symposium on ACSW frontiers', Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 143–152.

Kristensen, L. M., Christensen, S. & Jensen, K. (1998), 'The practitioner's guide to coloured Petri nets', *International Journal on Software Tools for Technology Transfer* **2**(2), 98–132.

Lemos, R. (2007), 'Reported data leaks reach high in 2007', *SecurityFocus* . Last access 15 January 2008 from:`http://www.securityfocus.com/brief/652`.

Lockhart, H., Andersen, S. et al. (2006), *Web Services Federation Language (WS-Federation) version 1.1*. Last access 15 January 2008 from: `http://dev2dev.bea.com/webservices/specs/WS-Federation-V1-1.zip`.

McLendon, W. W. & Vidale, R. F. (1992), Analysis of an ada system using coloured petri nets and occurrence graphs, *in* 'Proceedings of the 13th International Conference on Application and Theory of Petri Nets', Springer-Verlag, London, UK, pp. 384–388.

OASIS (2005), *Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0*, OASIS. Last access 15 January 2008 from: `http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf`.

PRIME (2006), *PRIME (Privacy and Identity Management for Europe) - Framework V2*, PRIME Consortium. Last access 15 January 2008 from:`https://www.prime-project.eu/prime_products/reports/fmwk/pub_del_D14.1.b_ec_wp14.1_V1_final.pdf`.

Scheschonk, G. & Timpe, M. (1994), Simulation and analysis of a document storage system, *in* 'Proceedings of the 15th International Conference on Application and Theory of Petri Nets', Springer-Verlag, London, UK, pp. 454–470.

Suriadi, S., Ashley, P. & Jøsang, A. (2007), Future standardization areas for identity management systems, *in* '2nd Workshop on Standards for Privacy in User-Centric Identity Management'. Last access 15 January 2008 from: `https://www.prime-project.eu/events/standardisation-ws2/positionpapers/Position_Paper_from_SuriadiAshleyJosang.pdf`.

Vijayan, J. (2006), 'Wells fargo discloses another data breach', *Computer World* . Last access 15 January 2008 from: `http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9002944`.