個人情報保護フレームワークにおける プログラム変換手法の改良

松永 崇秀[1]　高橋 健一[1]　川村 尚生[1]　菅原 一孔[1]

概要：近年，インターネットの普及に伴い様々なネットワークサービスが利用されている．これらのサービスの中には，サービス提供時に利用者に対して個人情報の提供を要求するものが存在する．しかし，利用者は提供した個人情報が実際にどのように利用されるか知ることができないため，サービス提供者へ個人情報を提供することに不安を感じる．そこで，我々は利用者が個人情報の利用方法を指定することができる仕組みを提案している．提案手法では，利用者が指定した保護方法を個人情報の処理に適用するために，サービス提供者の持つプログラムを変換する．しかし，これまでの変換手法では正しく変換できないプログラムが存在した．そこで，本稿では変換手法を改良し，プログラム変換率の向上を図る．

キーワード：個人情報保護，プログラム変換，セキュリティ，プライバシ

Improvement of Program Conversion on Personal Information Protection Framework

Takahide Matsunaga[1]　Kenichi Takahashi[1]　Takao Kawamura[1]　Kazunori Sugahara[1]

Abstract: Nowadays, various network services have been used with the spread of the Internet. Some of these services request to offer personal information to users when providing services. However, we cannot know how offered personal information is used. Thus, we feel uneasy to offer personal information to service providers. Therefore, we proposed a framework that an user can designate usage procedures of his/her personal information. In this framework, we convert a program processing personal information in order to apply the protection method that an user designate. There, however are some cases that cannot be converted correctly. This paper focus on the improvement of the program conversion method.

Keywords: personal information protection, program conversion, security, privacy

1. はじめに

近年，インターネットの普及に伴い様々なネットワークサービスが利用されている．例えば，Amazon や楽天などのオンラインショップや Gmail などのメールサービス，ホテルの予約やオンラインバンキングなどが挙げられる．これらのサービスは，利用者に対して名前や住所，電話番号やクレジットカード番号などの個人情報の提供を求める．利用者は要求に従い自身の個人情報を提供することでサービスを利用することができる．しかし，サービス提供者の要求する情報の種類や提供の方法に従わなければ，利用者はサービスを利用することができない．例えば，多くのサービスでは情報の登録時において入力必須欄を用意しており，その項目について利用者は情報を提供しなければならない．

提供する情報は HTTPS などにより暗号化してサービス提供者に送信されることが多い．しかし，このような暗号化が行われていなかった場合，たとえ利用者が暗号化することを望んだとしても情報を保護することはできない．また，個人情報の利用方法がサービス提供者に委ねられてい

[1] 鳥取大学大学院工学研究科
Graduate School of Engineering, Tottori University

ることから，利用者には一度サービス提供者に提供した個人情報が実際にどのように利用されているか確認する術がない．

プライバシポリシ [1] を閲覧することにより，利用者はサービス提供者による個人情報の利用方法を知ることができる．しかし，プライバシポリシはサービス提供者が記述通りに情報を利用することを保証するものではない．また，多くの利用者がプライバシポリシを閲覧しないという問題も存在する．そこで，収集する個人情報の利用方法を利用者に提示するフォーマットとして P3P（Platform for Privacy Preferences）[2] が提案されているが，P3P もサービス提供者がプライバシポリシ通りに情報を利用することを保証しない．

一方で，個人情報を送信しないことにより悪意のあるアプリケーションから個人情報を守ること [3] が提案されている．この研究では，取得したアプリケーションルールから生成した制御コマンドを個人情報の代わりに送信することで情報を保護する．しかし，アプリケーションルールはサービス提供者が作成しているため，サービス提供者のことを信頼できない利用者の不安を解消することができない．

このため，現状では利用者はサービス提供者に個人情報を提供することに不安を感じたとしても，個人情報を提供してサービスを利用するか，個人情報を提供せずにサービスを利用しないかという選択しかすることができない．

そこで，我々は利用者自身が個人情報の処理方法を決めることができる仕組みを提案している [4], [5]．ネットワークサービスでは，一般的に利用者が提供した個人情報はサービス提供者が持つプログラムで処理される．そこで，このプログラムの処理方法を利用者が指定した方法に書き換える．これにより，利用者が指定した方法でサービス提供者に個人情報の処理を行わせる．すなわち，個人情報を提供する利用者自身が個人情報の利用方法を決定するため，利用者は安心して個人情報を提供することができる．我々はこれまでに提案手法のプロトタイプを作成し，プログラムに利用者の意図を反映させることが可能であることを確認した．しかし，これまでの実装方法では提案手法を適用できないプログラムが存在した．そこで，本稿ではこの仕組みを実現するためのプログラム変換手法の改良について述べる．

以降，2 章で我々が提案するフレームワークについて述べ，3 章で本稿で行ったプログラム変換の改良について述べる．また，4 章では提案手法の実装および実験について述べる．

2. 個人情報保護フレームワーク

ネットワークサービスでは，サービス提供者が要求する個人情報をサービス提供者が定める方法に従って提供することで，利用者はサービスを利用することができる．しかし，個人情報の処理はサービス提供者に委ねられているため，利用者は提供する個人情報の利用方法を把握することができず，一度提供した個人情報を利用者自身で保護することができない．また，個人情報の処理方法のすべてをサービス提供者が決めているため，個人情報の漏洩などが起きた場合，すべての責任がサービス提供者に集中することとなる．

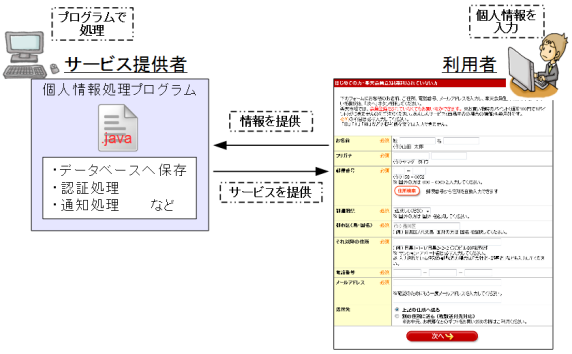

図 1　ネットワークサービスの利用モデル
Fig. 1　Example of a web based network service

上記の問題が発生する原因の 1 つに，利用者が提供した個人情報の処理がサービス提供者のプログラム（以下，個人情報処理プログラム）に委ねられていることが挙げられる（**図 1**）．そこで，利用者が指定した方法で個人情報を処理できるようにサービス提供者の持つプログラムを変換し，利用者の意図を個人情報の処理に反映させることができる**個人情報保護フレームワーク**を提案している．本フレームワークにおいて，利用者は自身の納得できる方法で個人情報を保護することができるため，より安心してサービスを利用できるようになる．また，個人情報の処理に利用者も関与しているため，サービス提供者への負担の集中も軽減できる．

2.1 概要

本フレームワークでは，サービス提供者が使用する個人情報処理プログラムに利用者が指定した処理方法を適用することで，利用者の意図を個人情報の処理に反映させる．このため，利用者は自身が指定した処理方法をサービス提供者に伝えることができる必要がある．このことを実現するためには，以下の要件を満たす必要がある．

【要件 1】　どの個人情報に対してどのような保護方法を適用できるか知ること

【要件 2】　指定する保護方法がサービス提供者のプログラムに適用できること

【要件 1】を解決するため，プログラム中での情報の利用方法を示した**利用ポリシ**をあらかじめ準備する．利用ポリシはサービス提供者が作成し，個人情報処理プログラムとともに保持しておく．利用者は利用ポリシを参照することにより，サービス提供者による個人情報の利用方法と各

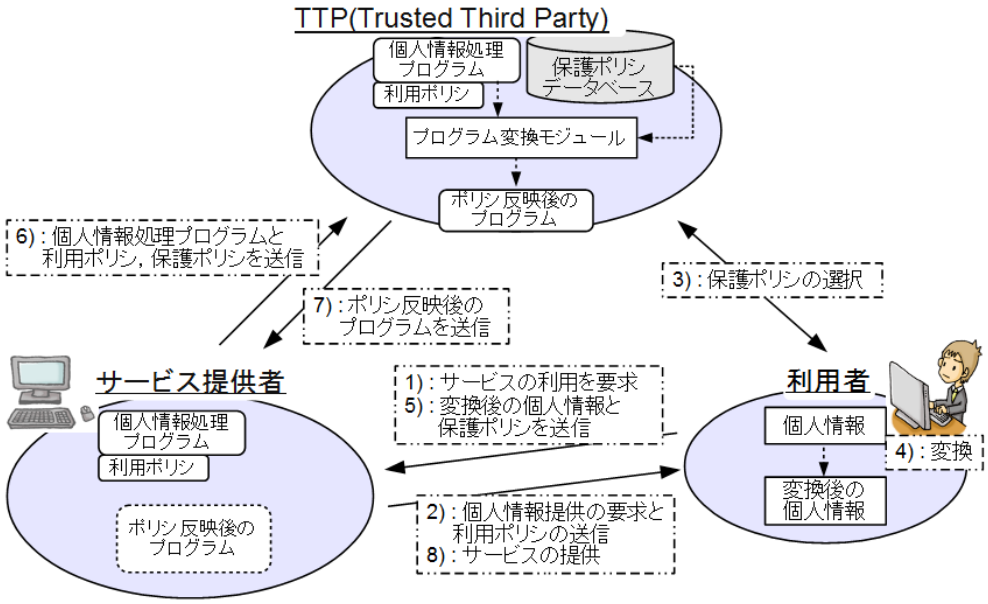

図 2　個人情報保護フレームワーク

Fig. 2　Overview of the Personal Information Protection Framework

個人情報に適用可能な保護方法を知ることができる．

【要件 2】について，一般の利用者にはプログラムに関する知識はなく，保護方法を自身で定義することは難しい．また，利用者が自由に保護方法を定義すると，サービス提供者のプログラムに適用できない可能性や保護方法適用後のプログラムが予期しない動作をする可能性（例えば，マルウェアのような動作をするなど）が考えられる．このため，個人情報の保護方法およびプログラムの変換方法を定義した**保護ポリシ**をあらかじめ準備する．保護ポリシは信頼できる第三者機関 (TTP：Trusted Third Party) が設置した保護ポリシデータベースで管理されるものとする．TTP が確認した保護ポリシのみを利用することで，上記のような可能性を排除する．

利用ポリシを用いることで，個人情報処理プログラムに適用可能である保護ポリシを絞り込むことができる．利用者は適用可能な保護ポリシの中から自身の安心できる保護方法が記述された保護ポリシを選択する．選択した保護ポリシを個人情報に適用することで，利用者は自身の選択した方法で個人情報を変換する．また，保護ポリシが適用された個人情報を個人情報処理プログラムで正しく処理できるようにするために，利用者が選択した保護ポリシに従って個人情報処理プログラムを変換する．変換後のプログラムで個人情報の処理を行う．これにより，保護ポリシが適用された個人情報を正しく処理することができる．個人情報保護フレームワークの処理の流れを**図 2** に示す．

(1) 利用者はサービス提供者にサービスの利用を要求する．
(2) サービス提供者は利用者に対して個人情報の提供を要求するとともに利用ポリシを送信する．これにより，プログラム中での個人情報の利用方法と処理内容を利用者に伝える．
(3) 利用者は利用ポリシを参照することで，TTP からサービス提供者のプログラムに適用可能な保護ポリシの一覧を取得し，自身の安心できる保護方法が記述された保護ポリシを選択する．
(4) 利用者は自身の個人情報に保護ポリシを適用する．
(5) 利用者は保護ポリシを適用した個人情報と選択した保護ポリシをサービス提供者に送信する．
(6) サービス提供者は個人情報処理プログラムと利用ポリシ，および利用者が選択した保護ポリシを TTP に送信する．
(7) TTP はサービス提供者から受け取った個人情報処理プログラムを保護ポリシに従って変換し，変換後の個人情報処理プログラムをサービス提供者に送信する．
(8) サービス提供者は変換後の個人情報処理プログラムで個人情報の処理を行い，利用者にサービスを提供する．

本フレームワークにより，利用者は自身が選択した方法で個人情報の処理を行わせることができる．これにより，利用者は保護方法の決定権を持つことができるため，より安心してネットワークサービスを利用できるようになる．

2.2　保護ポリシ

本フレームワークにおいて，利用者は自身の安心できる個人情報の処理方法を保護ポリシにより指定する．保護ポリシでは

- 個人情報の保護方法
- 個人情報およびプログラムの変換手法

を定義する．

利用者は保護ポリシ中の保護方法を見ることにより，自

身の安心できる保護方法が記述された保護ポリシを選択する．また，変換手法に従って個人情報および個人情報処理プログラムを変換することで，利用者が選択した方法で個人情報を処理させることができる．保護ポリシはパーサによる解析を可能とするために XML 形式で記述する．具体的な記述方法については 3.1 節で述べる．

2.3 利用ポリシ

利用ポリシは，

- サービス提供者のプログラムによる個人情報の利用方法の確認
- 適用可能な保護ポリシの選択

を可能とするために準備する．

利用ポリシは，プログラム中で利用する情報 1 つにつき 1 つ用意する．例えば，1 つのプログラム中で ID とパスワードを利用する場合，そのプログラムに対する利用ポリシは 2 つ存在することになる．利用ポリシはサービス提供者がサービスを提供する Web ページのヘッダにリンクとして記述する．ヘッダを見ることにより，利用者はいつでも利用ポリシを参照できる．これにより，利用者は個人情報処理プログラムにおける個人情報の具体的な処理方法を知ることができる．

利用者は利用ポリシを見ることで，変換対象の情報がプログラム中でどの変数に格納され，どの操作で使用されているか知ることができる．これにより，保護対象の情報と操作を特定することができるため，プログラムに適用可能な保護ポリシを選択することができるようになる．

利用ポリシも保護ポリシと同様に，パーサによる詳細な解析を可能とするために XML 形式で定義される．利用ポリシの具体的な記述方法についても 3.1 節で述べる．

2.4 プログラム変換モジュール

サービス提供者が持つプログラムを利用者が指定した方法で変換することにより，利用者の意図を個人情報の処理に反映させる．ここで，プログラムの変換は TTP が所持するプログラム変換モジュールにより行う．**図 3** にプログラム変換モジュールによるプログラム変換の流れを示す．

プログラム変換モジュールはまず，保護ポリシと利用ポリシ，および個人情報処理プログラムを読み込む．次に，保護ポリシと利用ポリシの結びつけを行い，プログラムを変換するためのルール（変換ルール）を生成する．保護ポリシには個人情報の保護方法とプログラムの変換手法が定義されており，利用ポリシにはプログラム中での個人情報の処理内容が定義されている．このため，保護ポリシと利用ポリシを結びつけることで，プログラム中のどの処理 (変数) をどのように変換すればよいか判断できるようになる．その後，プログラムの解析を行うことで変換する箇所を特定する．解析後，結びつけを行った変換ルールを変換

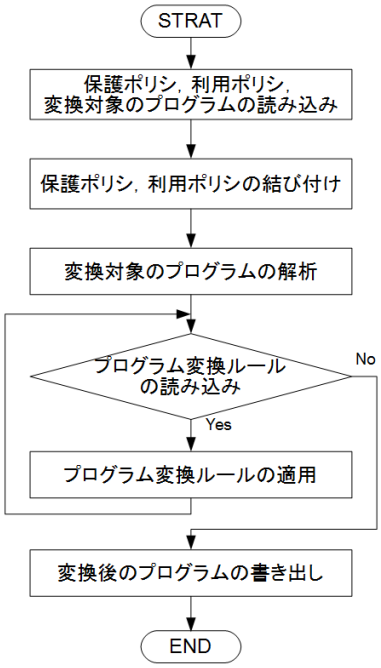

図 3 プログラム変換の流れ
Fig. 3 The flow of program conversion

対象のプログラムに適用し，プログラムを変換する．すべての変換ルールの適用後，プログラムを書き出す．これにより，利用者が安心できる処理方法が適用された個人情報処理プログラムを生成する．

3. プログラム変換手法の改良

我々の提案する個人情報保護フレームワークでは，サービス提供者が個人情報処理時に使用するプログラムを利用者が指定した方法に従って変換することにより，利用者の意図を個人情報の処理に反映させる．これまでに提案手法のプロトタイプを作成し，プログラム変換により利用者が指定した保護方法を個人情報の処理に反映できることを確認した．しかし，シリアライズを使用しているプログラムや変換対象が複数のクラスに跨っているプログラムなど，正しく変換を行うことができないプログラムが存在した．このため，ポリシおよびプログラム変換モジュールの実装を見直すことで，本フレームワークがより現実的かつ実用的になるようにプログラム変換手法の改良を行う．

3.1 ポリシの改良

プログラムの変換は，「プログラム中で対象の情報が使用される部分（操作）」を保護ポリシに従って書き換えることで行う．このため，[5] では “操作” を基準に保護ポリシと利用ポリシを定義していた．しかし，この方法だと「処理 A において対象の情報は変数 X で使用され，処理 A 中の変数 Y は〜で・・・」というように，1 つの処理で使用されている変数のすべてを利用ポリシ中に定義する必要があった．これにより，記述が冗長（難解）になることやポ

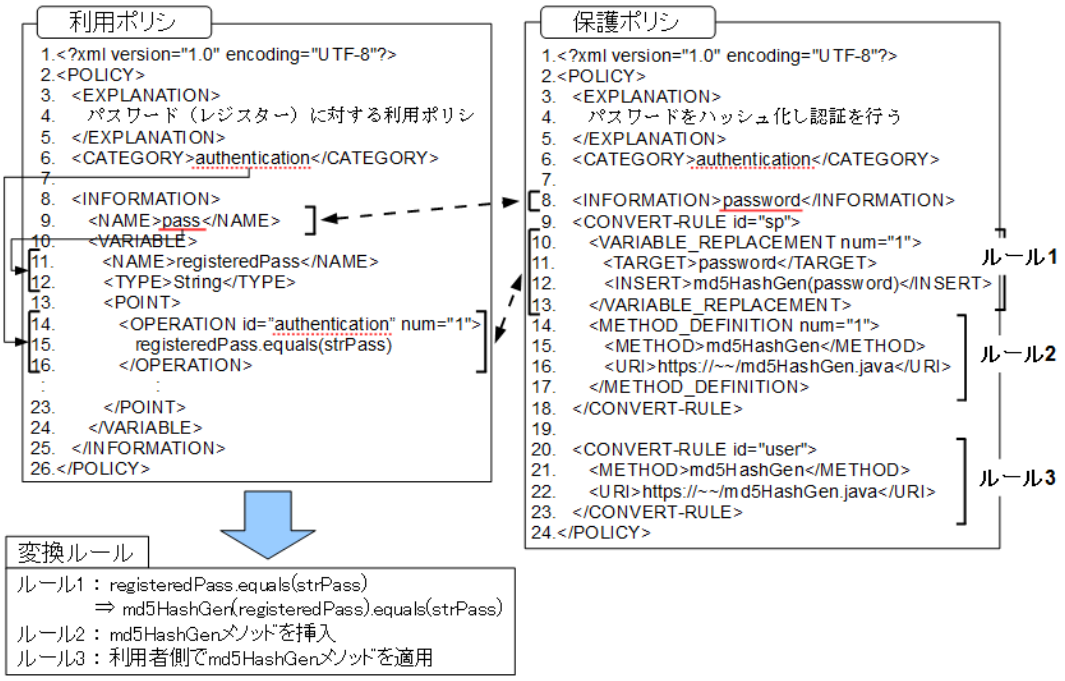

図 4 ポリシの結びつけ例

Fig. 4 Association of an use policy and a protection policy

リシの結びつけに失敗することがあった．また，変換対象の変数が複数の処理で用いられている場合，処理ごとに同じ変数を定義する必要があったため，変換箇所を特定するのが困難になることがあった．そこで，利用ポリシの定義基準を “操作” から “情報” へ変更する．改良した利用ポリシと保護ポリシの例を**図 4** に示す．

それぞれのポリシ中の<EXPLANATION>（3〜5 行目）には，利用者に提示するための説明が自然言語で記述されている．利用ポリシ中の<INFORMATION>（8〜25 行目）には，そのポリシで定義する情報が実際にどのように利用されるかが定義される．図 4 の例では，この利用ポリシで扱う個人情報は “pass” という識別子で表現されており（9 行目），プログラム中で String 型の変数 registeredPass として扱われていることがわかる（11，12 行目）．また，“authentication” に対応する操作が，プログラム中で「registeredPass.equals(strPass)」として実現されていること（14〜16 行目）がわかる．

以前の記述方法では，情報が使用される箇所が増えるたびに処理と変数の組み合わせの記述が増えていた．しかし，“情報” を基準にポリシを定義しなおしたことにより「変換対象の情報が使用される処理」を一度に定義できるようになったため，以前より簡潔に記述できるようになった．これにより，冗長な記述を減らすことができ，ポリシが難解になることを防ぐことができる．

また，保護ポリシの定義内容を利用ポリシに合わせて変更した．以前の記述方法では，利用者側で行う変換（Web ページの入力フォームに入力された情報に対する変換）とサービス提供者側の変換（個人情報処理プログラムの変換）を並列に記述していたため，記述が冗長になり変換手法の導出が困難になることがあった．そこで，変換のためのルールに id を付与した（図 4 右の保護ポリシ中の 9，20 行目）．id によりこれらを区別し，必要な記述量を利用者側とサービス提供者側で最低限に抑えることで，ポリシが難解になることを防ぐ．これにより，ポリシの記述量を以前の約半分に抑えることができた．

3.2 ポリシの結びつけ

利用ポリシと保護ポリシを結びつけることで，プログラムを変換するためのルールを導出する．図 4 左の利用ポリシで定義されるプログラムに対して，図 4 右の保護ポリシが利用者によって選択されたとする．このとき，利用ポリシと保護ポリシの<INFORMATION>タグを参照し，それぞれの識別子を結びつける．利用ポリシにより，サービス提供者のプログラム中で利用者が登録したパスワードは String 型の registeredPass 変数として扱われており，利用ポリシ中で “pass” という識別子で表現されていることがわかる．また，保護ポリシでは変換対象の情報が “password” という識別子で定義されており，“authentication” で使用される情報に対して md5HashGen メソッドを適用することが 10〜13 行目に定義されている．

これらを結びつけることで，プログラム中で「registeredPass」が authentication として使用される箇所を「md5HashGen(registeredPass)」に置き換える変換ルール（ルール 1）を得ることができる．また，変換前のプログ

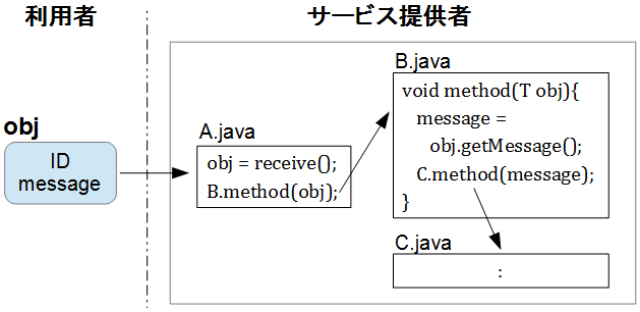

図 5 プログラム変換の失敗例
Fig. 5 Failure of program conversion

ラムには md5HashGen メソッドは存在しないため，<URI>（16 行目）からメソッドを取得しプログラムへ挿入する変換ルール（ルール 2）を得る．さらに，<CONVERT-RULE>の id が “user” である箇所（20〜23 行目）には，利用者が入力フォームに入力した情報を変換するための方法が定義されるため，ここからルール 3 を得る．

3.3 プログラム変換モジュールの改良

[5] では，86 通りの変換のうち 79 通りは意図した通りに変換することができた．しかし，残りの 7 通りは変換に失敗した．変換に失敗したプログラムは以下の条件を含む場合であった．

失敗例 1 変換対象の情報がオブジェクトのフィールドとして送られてくる場合

失敗例 2 変換対象の情報が複数のクラスに跨って使用される場合

失敗例を**図 5** に示す．失敗例 1 は，変換対象の変数（“message”）がオブジェクト（“obj”）のフィールドとして送られる時に発生する．この時，サービス提供者のプログラムが受け取る情報は obj であるため，利用ポリシには message ではなく，受け取る情報である obj を定義する．このため，プログラム変換モジュールは保護ポリシの内容と obj を結びつけ，obj を変換する．しかし，実際に変換をすべきであるのは message であるため，正しい結果を得られず，変換に失敗する．

失敗例 2 は，変換対象の情報が複数のクラスで利用される場合に発生する．例えば，図 5 のようなデータフローでかつ C.java で目的の処理を行う場合，A.java 中のメソッド呼び出し部分に保護ポリシを適用することで，変換は成功する．しかし，変換される前の情報に対して行う必要のある処理が B.java にあった場合，B.java に渡された情報は既に変換された情報であるため，本来の処理を行うことができない．

これらの失敗は，プログラム変換モジュールで変換するプログラムを特定のプログラムに固定しているために発生している．変換対象の情報がオブジェクトのフィールドとして送られてきたとしても，実際に処理を行うプログラム

図 6 保護方法選択のためのユーザインターフェース
Fig. 6 User interface for selecting a protection method

では対象の情報自体が参照される．このため，そのプログラムを指定できれば，変換対象の情報を正しく変換することができる．また，プログラム中の目的の処理をピンポイントで指定し変換することで，その処理までに別の処理が行われていたとしても影響をなくすことができる．

上記失敗を解決するため，「実際に処理を行う部分」を特定できるように改良を行った．具体的には，メソッド間やクラス間における情報の受け渡しの定義を利用ポリシに追加した．これにより，実際に処理を行う部分までたどってプログラムを変換することができるようになるため，上記失敗を解決できる．

4. 実装・実験

提案手法が実現可能であることを確認するために，本フレームワークで想定する三者（TTP，利用者，サービス提供者）の実装を行った．

4.1 TTP の実装

TTP では，保護ポリシの管理とプログラムの変換を行う．本フレームワークにおいて，利用者は個人情報の保護方法を自身で選択するために，TTP から保護ポリシを取得する．このため，利用者が保護ポリシデータベースを参照するための仕組みが必要となる．そこで，TTP に “保護方法選択ページ” を設置し，利用者が保護ポリシデータベースを参照できるようにした．TTP が提供する保護方法選択ページの例を**図 6** に示す．

図 6 の左上のフレームには利用者側のアドオン（後述）により解析された利用ポリシの一覧が表示される．すなわち，利用者が個人情報入力ページで要求される個人情報の一覧が表示される．図 6 は，サービス利用時に ID とパスワードが要求されるサービスの例を示している．この画面において，ID/Password の横に配置された選択ボタンを押すと，選択した個人情報に対して適用可能な保護ポリシの

一覧が左下のフレームに表示される．保護ポリシは TTP 内の保護ポリシデータベースで管理しており，MySQL で実装している．一覧の中の保護ポリシをクリックすると，その保護ポリシが個人情報をどのように保護するものであるかが右下のフレームに表示される．利用者はこれを見ることで，自身の安心できる保護方法が記述された保護ポリシを選択し，右下のフレームの決定ボタンを押す．最後に，左上のフレームの完了ボタンを押すことで，保護ポリシが利用者に送信される．

また，TTP はプログラム変換モジュールにより個人情報処理プログラムを変換するサービスを提供する．TTP は個人情報処理プログラムと利用ポリシ，保護ポリシの参照先を受け取ると，プログラム変換モジュールにこれらの情報を与え，個人情報処理プログラムを変換する．プログラムの変換後，サービス提供者にポリシ適用後のプログラムを返す．

4.2 利用者の実装

利用者側では，保護ポリシの取得，個人情報へ保護ポリシを適用，ポリシ適用後の個人情報の送信機能を実現する必要がある．

これらの機能は Web サービスでの利用を想定し，Google Chrome[6] のアドオンとして実装した．本フレームワークを利用したいと考える利用者は，あらかじめこのアドオンをインストールする．サービス提供者の個人情報入力ページで右クリックをするとアドオンが起動する．アドオンは個人情報入力ページのヘッダに示された URI から利用ポリシを取得し，TTP の保護方法選択ページ（図 6）へアクセスする．利用者は 4.1 節で述べた手順で保護ポリシを選択し，取得する．

利用者が個人情報入力ページで個人情報を入力し送信ボタンを押すと，入力した個人情報は取得した保護ポリシに従って変換される．変換は保護ポリシの<URI>に記述されたアドレスから変換に必要なプログラムを取得し，これを実行することで行う．これにより，利用者の個人情報に保護ポリシが適用され，変換後の個人情報を生成する．

その後，選択した保護ポリシの参照先と変換後の個人情報をサービス提供者に送信する．

4.3 サービス提供者の実装

サービス提供者は，利用者への利用ポリシの通知，保護ポリシとポリシ適用後の個人情報の受信，変換後のプログラムの実行をできる必要がある．

サービス提供者は利用者からサービス利用の要求があった場合，利用者に対して利用ポリシを通知する．利用者への利用ポリシの通知は，そのページで入力を求める個人情報に対する利用ポリシの URI を Web ページのヘッダに記述しておくことで実現する．

図 7 パスワードの認証結果
Fig. 7 The result of password authentication

保護ポリシおよび保護ポリシ適用後の個人情報の受信は HTTP POST で実現する．POST された内容から保護ポリシの参照先（URI）と保護ポリシ適用後の個人情報を取得する．保護ポリシの参照先が送られてこなかった場合は，通常通り個人情報の処理を行う．送られてきた場合は，これらの情報をローカルに保存した後，TTP が提供するプログラムの変換サービスを利用するために，TTP へ個人情報処理プログラムと利用ポリシ，保護ポリシの参照先を送信する．これらの情報の送信は Java で実装した SSL ソケット通信により実現する．

その後，サービス提供者は TTP から変換後の個人情報処理プログラムが送られてくるのを待つ．プログラムの受信後，そのプログラムを実行する．これにより，変換後の個人情報処理プログラムで保護ポリシ適用後の個人情報を処理する．

4.4 動作実験

個人情報の変換処理を正しく行うことができるかを実験により確認した．実験では Google の会員登録ページおよびログインページのソースコードを取得し，この入力フォームに個人情報を入力することで行った．ただし，これらのページで使用される個人情報処理プログラムは取得できなかったため，本来の個人情報処理プログラムの代用として，1 つのソースファイルで受信および認証を行うプログラムと，複数のソースファイルを用いてかつシリアライズにより情報を受信するプログラムを用意した．サービス提供者が生のパスワードを受け取り，パスワード認証に成功したか否かを表示するだけのサービスを想定して実験を行った．

実験では，まず会員登録ページから各個人情報の登録を行った．このとき，パスワードとして「123456」を登録した．次にログインページからアドオンを起動し，「ハッシュ変換により個人情報を保護する保護ポリシ」をパスワードに対して選択した．その後，上記のパスワードを入力して

送信したところ，サービス提供者にはハッシュ変換後の「e10adc・・・」が送信され，この情報により正しく認証ができていた（**図 7**）．これにより，保護ポリシによって個人情報処理プログラムが変換され，正しくパスワードの処理ができていることを確認できた．

5. おわりに

サービス提供者による個人情報の取り扱い方を利用者が指定することができる個人情報保護フレームワークを提案している．本稿では，プログラム変換手法を改良した．この結果，今まで対応できなかったプログラムの変換ができることを確認した．今後の課題として，改良したポリシとプログラム変換モジュールを用いてプログラム変換率の評価を行うことが挙げられる．

評価は，GitHub[7] でキーワードで検索した結果の上位から Java 言語で記述されているプログラムが存在するプロジェクトを取得して行うことを想定している．取得したプロジェクト内に利用ポリシを格納するディレクトリを用意し，この中に主要なプログラムに対する利用ポリシをあらかじめ用意する．保護ポリシは現在，ハッシュ変換や暗号化などの 7 種類を準備している．これらのプログラムと保護ポリシの組み合わせに対して，どのくらいの割合で変換が成功するか確かめる．

参考文献

[1] IBM：プライバシー・ポリシーの定義，入手先 ⟨https://publib.boulder.ibm.com/tividd/td/ITPME/SC23-1284-00/ja_JA/HTML/p12plmst22.htm⟩（参照 2016-04-15).

[2] W3C: Platform for Privacy Preferences (P3P) Project, available from ⟨http://www.w3.org/P3P/⟩ (accessed 2016-04-15).

[3] 田丸修平，岩谷晶子，高汐一紀，徳田英幸：プライバシを考慮したパーソナライゼーションを実現するアプリケーションフレームワーク，情報処理学会研究報告システムソフトウェアとオペレーティング・システム（OS），pp.49-56（2003）.

[4] 松永崇秀，高橋健一，川村尚生，菅原一孔：個人情報保護を目的としたフレームワークの提案，コンピュータセキュリティシンポジウム 2015(CSS2015)，pp.162-169 (2015).

[5] 松永崇秀，山口哲敬，高橋健一，川村尚生，菅原一孔：利用者による個人情報保護手法の決定を可能とするフレームワークの提案，情報処理学会論文誌，Vol.57，No.9，to appear（Sep. 2016）.

[6] Google：Chrome ブラウザ，入手先 ⟨https://www.google.co.jp/chrome/browser/desktop/⟩ (参照 2016-07-28)

[7] GitHub: Explore GitHub, available from ⟨https://github.com/explore⟩ (accessed 2016-07-28).