平成28年度第1回学術情報基盤オープンフォーラム@NII
2016/5/26(Thu)

Shibboleth認証におけるクライアント証明書の利用

金沢大学　松平　拓也

KANAZAWA UNIVERSITY

国立大学法人 金沢大学

- 構成員
  - 学生（院生含む）
    - 約 10,300名
  - 教職員（非常勤含む）
    - 約3,800名

2015年3月14日・北陸新幹線開業

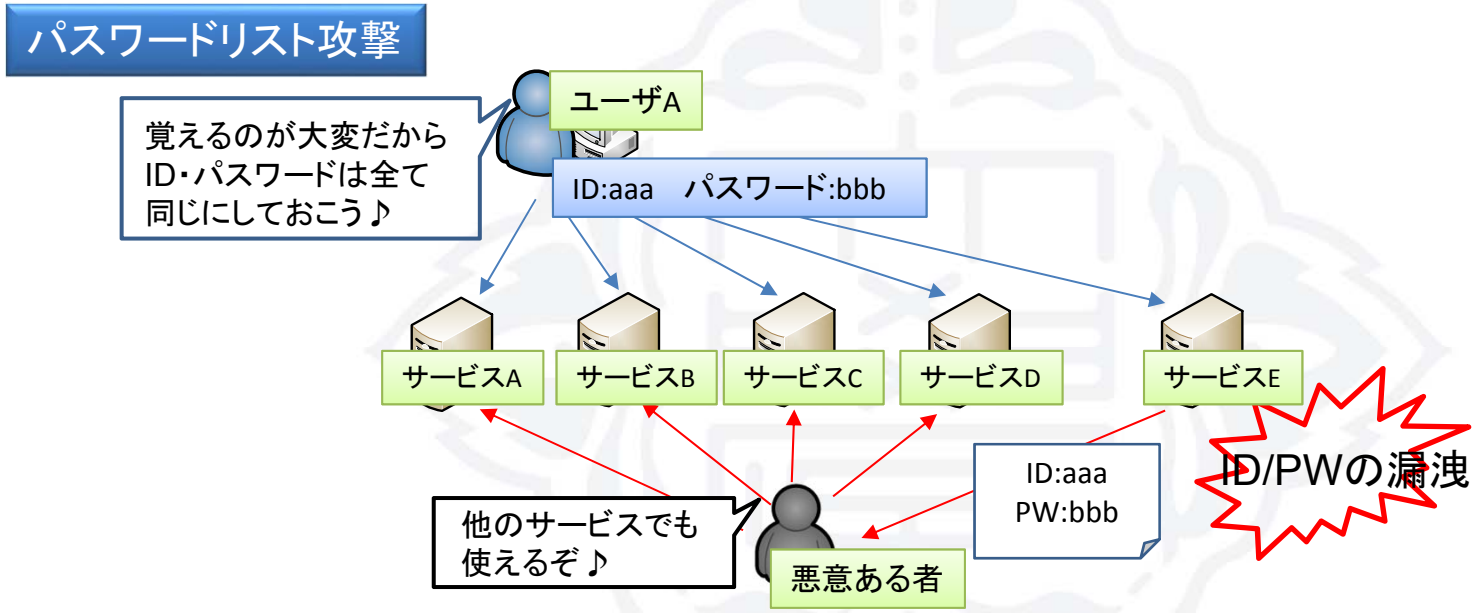
KANAZAWA UNIVERSITY

金沢大学統合認証基盤（KU-SSO）

- Shibbolethによるシングルサインオンを実現
  - Kanazawa University Single Sign On（KU-SSO）
    - 30以上の学内情報システムをShibboleth SP化
      - 予算執行支援、給与明細、教務システム、教員DB等、機微な情報を取り扱うシステムも多い
- KU-SSOの認証方式
  - 金沢大学ID・パスワードによる認証
    - パスワード認証の強度はパスワードの強度に依存
    - そのため、パスワードポリシーは徹底
      - 全アカウントに対してパスワードポリシーを満たすようパスワード点検（更新）作業を実施
        （未実施のアカウントのパスワードはリセット）

ID・パスワード認証から多要素認証へ

- ID・パスワードに関わるセキュリティインシデント報告の増加
  - 総当り攻撃や辞書攻撃に加え、パスワードリスト攻撃による被害の急増



パスワードを使い回している限り、ポリシーを厳しくしてもパスワードリスト攻撃の被害を防げない！

多要素認証への移行が重要

KANAZAWA UNIVERSITY

多要素認証導入に向けて

- 多要素認証とは
  - 本人しか知らない知識（パスワード、PINなど）、本人しか持っていない所有物（ICカード、スマートフォンなど）、本人の生体的特徴（指紋、静脈など）の3つの要素のうち、2要素以上を必要とする認証方式

    例：ICカードとPIN（所有物＋知識）

- 多要素認証導入の課題
  - 特定の所有物（スマートフォン、ICカードなど）がないと認証できない
    - 大学には様々な身分の構成員がおり、全員に同一の所有物を所持させることは困難（如何にして使えないユーザを作らないかが重要）
  - ID・パスワード認証より手間がかかる
    - 全サービスで多要素認証を要求するのはユーザに対する負担が大きい

金沢大学における多要素認証導入方針

1. 複数の多要素認証方式を選択できるようにする
   - 全員に同じ多要素認証方式を指定するのではなく、複数の多要素認証方式を用意し、ユーザが選択できるようにする
     - トータルで全構成員が多要素認証を扱える環境を整備する

2. サービスの重要度に応じて認証レベルを変更できるようにする
   - 従来の認証レベルで十分なサービスはID・パスワード認証で対応
   - 高いレベルを要求するサービスにおいては、ユーザの利用環境に応じて多要素認証を要求
     - サービスの重要度に応じて多要素認証を要求することで、ユーザの利便性を維持する

KANAZAWA UNIVERSITY

検証中の多要素認証方式とその対象

・tiqr認証
スマートフォン（所有物）＋PIN（知識）

・学生（主）・教職員
※ある学内アンケートでは、新入生の9割以上がスマホを所持
・ノートPC

・YubiKey認証
YubiKeyデバイス（所有物）＋ID・パスワード（知識）

・スマホ、ICカードを持っていないユーザ
・出張先でKU-SSOを（頻繁に）利用するユーザ
⇒ tiqr認証とICカード認証を補う役割

・UPKIパス認証
ICカード/クライアント証明書（所有物）＋PIN（知識）

・教職員（主）・学生
・デスクトップPC

KU-SSOを利用する全てのユーザが多要素認証できる環境を構築

KANAZAWA UNIVERSITY

Shibbolethのクライアント証明書認証

- クライアント証明書とは？
  - 個人（クライアント）に対して発行される電子的な身分証明書
    - 公開鍵暗号方式を利用しており、証明書の偽造は非常に困難
    - 第三者（認証局（CA））が証明書を発行し、証明書の正当性を保証

      ⇒ なりすまし、不正アクセスに対して非常に有効

- Shibbolethにおけるクライアント証明書認証対応
  - IdPv3でもほぼv2と同様の設定で対応可能

IdP（v3）の設定（1）

1. idp.propertiesにクライアント証明書認証を使う設定を記載

```
# Regular expression matching login flows to enable, e.g. IPAddress|Password
idp.authn.flows= Password|X509
```

2. Apacheのssl.confにクライアント証明書認証の処理を記載

```
<Location /idp/Authn/X509>
    SSLCACertificateFile /opt/shibboleth-idp/credentials/Kanazawa-CA.crt ← 認証局のCA証明書
    SSLVerifyClient require  ← クライアント証明書を検証
    SSLVerifyDepth 3
    SSLRequireSSL
    SSLOptions +ExportCertData +StdEnvVars
    SSLUserName SSL_CLIENT_S_DN_CN  ← “CN”の値を”REMOTE_USER”環境変数にセット
    SSLRequire %{SSL_CLIENT_S_DN_O} eq “Kanazawa University“← “O”の値が”Kanazawa University ”かどうかをチェック
</Location>
```

IdP（v3）の設定（2）

3. /TOMCAT_HOME/webapps/idp/WEB-INF/web.xmlにパラメータを追加

```
<!-- Servlet protected by container used for X.509 authentication -->
  <servlet>
     <servlet-name>X509AuthHandler</servlet-name>
     <servlet-class>net.shibboleth.idp.authn.impl.X509AuthServlet</servlet-class>
     <load-on-startup>3</load-on-startup>
  </servlet>
  <servlet-mapping>
     <servlet-name>X509AuthHandler</servlet-name>
     <url-pattern>/Authn/X509</url-pattern>
  </servlet-mapping>
```

4. relying-party.xmlにデフォルトの認証手段を指定（パスワード認証）

```
<bean id="shibboleth.DefaultRelyingParty" parent="RelyingParty">
   <property name="profileConfigurations">
      <list>
         <bean parent="Shibboleth.SSO" p:postAuthenticationFlows="attribute-release"
                              p:defaultAuthenticationMethods="urn:oasis:names:tc:SAML:1.0:am:password" />
          ---省略---
           <bean parent="SAML2.SSO" p:postAuthenticationFlows="attribute-release"
          p:defaultAuthenticationMethods="urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport" />
          ---省略---
      </list>
   </property>
</bean>
```

SPの設定

1. IdPに対して認証方式の指定

shibboleth2.xml

```
<SessionInitiator type="Chaining" Location="/Login" isDefault="true" id="Intranet“
        authnContextClassRef="urn:oasis:names:tc:SAML:2.0:ac:classes:X509"
          relayState="cookie" entityID="https://IdPserver/idp/shibboleth">
          <SessionInitiator type="SAML2" acsIndex="1" template="bindingTemplate.html"/>
          <SessionInitiator type="Shib1" acsIndex="5"/>
</SessionInitiator>
```

または、shib.conf

```
<Location /secure>
  AuthType shibboleth
  ShibCompatWith24 On
  ShibRequestSetting requireSession 1
  ShibRequestSetting authnContextClassRef urn:oasis:names:tc:SAML:2.0:ac:classes:X509
  require shib-session
</Location>
```

Shibbolethでは特に複雑な設定を行う必要なし！

クライアント証明書の配布・利用方法

- デバイスへ格納して配布したい
  - 利用者がクライアント証明書を雑に扱う危険性の排除
  - 格納するデバイスは本人が既に所持している（かつ本人を特定できる）もの

    ⇒ ICカード（職員証・学生証）の利用が最適！
- 金沢大学の職員証・学生証はICカード
  - 入退館、出席管理、生協マネー、証明書発行など、大学での活動に必要不可欠
  - 金沢大学のICカードはFelica（FCFフォーマット）
    - FCFフォーマット
      - 大学等教育機関におけるICカードのデファクトスタンダード
      - クライアント証明書のような大きなデータは格納できない

金沢大学を含む多くの大学でICカードによるクライアント証明書認証を行うには？

UPKIパスの利用

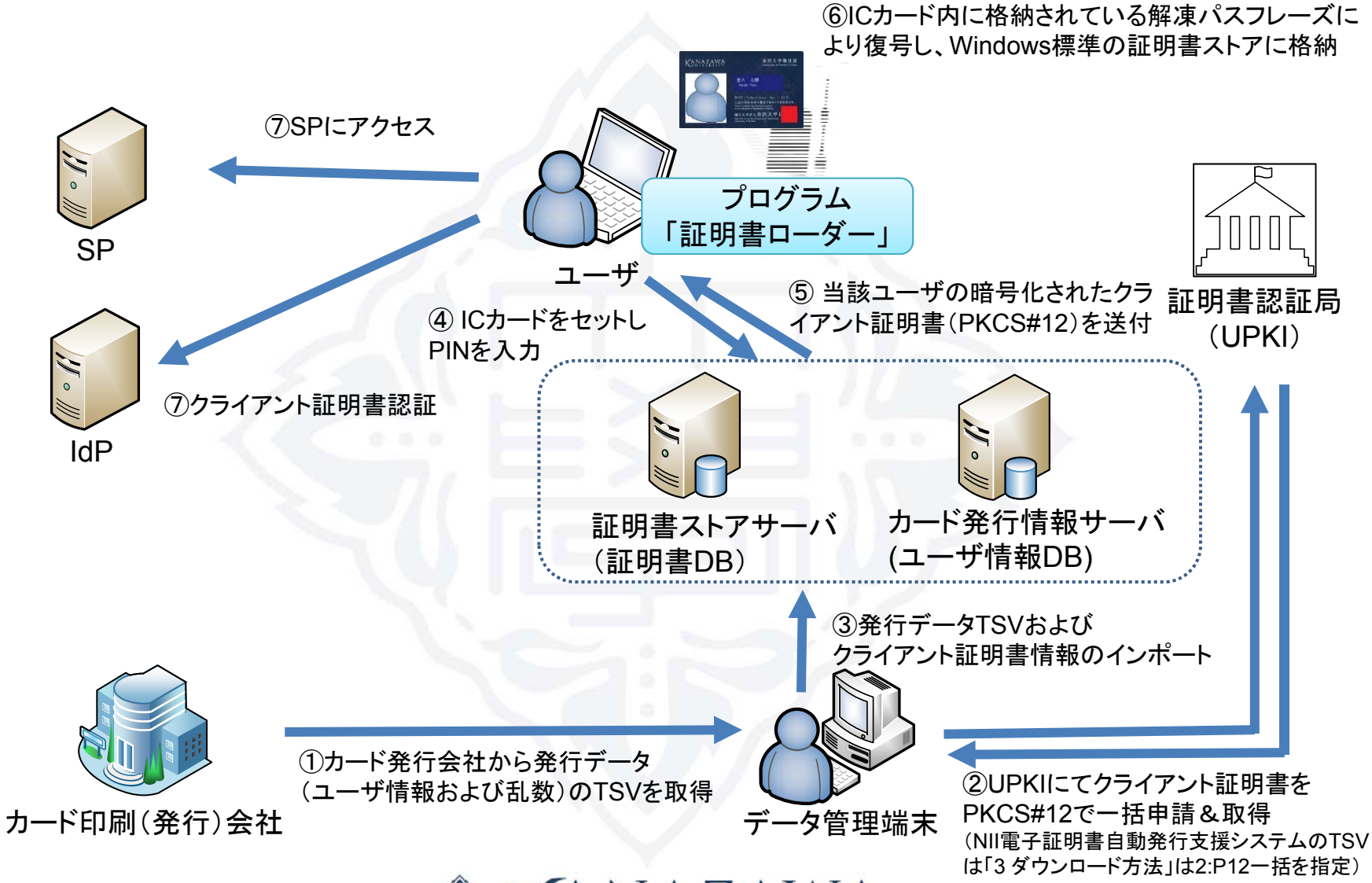
KANAZAWA UNIVERSITY

UPKIパス

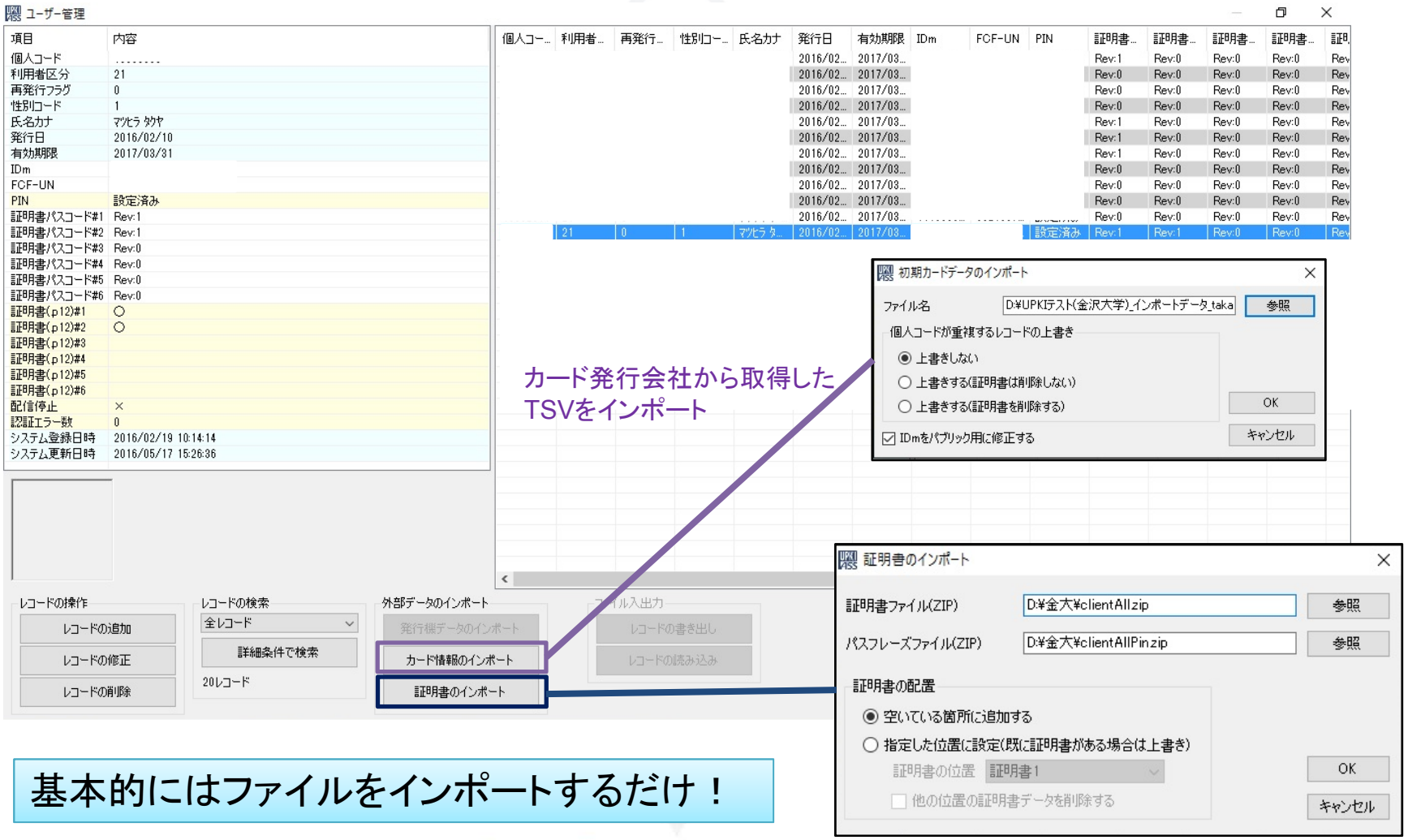
- UPKIパス方式とは？
  - 証明書をサーバに格納させておき、ICカードをリーダにかざした際に、クライアントPCに一時的にダウンロード/インストールする方式
- UPKIパス方式のメリット
  1. Felicaで利用可能（FCFフォーマットVer.3が必要）
  2. 証明書の更新における作業がサーバ側のみ
     - サーバ側の証明書を更新するだけでICカード側は変更する必要なし
     - ICカードを紛失しても、証明書をサーバから削除すればよい

       ⇒ カードを回収して再配布する必要がない
  3. カードが安価であり、かつカードリーダも安価なPaSoRiに対応

UPKIパス利用イメージ



データ管理端末画面（管理者作業）



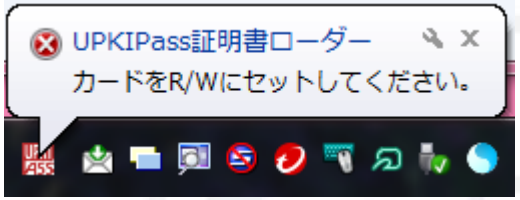
基本的にはファイルをインポートするだけ！

NII電子証明書自動発行支援システムから取得したファイルをインポート

ユーザの動作（1）

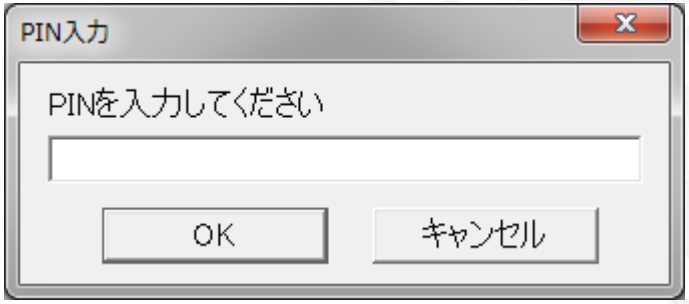
① UPKIパス証明書ローダーのインストール
・setup.exeを実行

② ICカードをカードリーダにセット



ICカードがリーダにセットされていない場合に表示

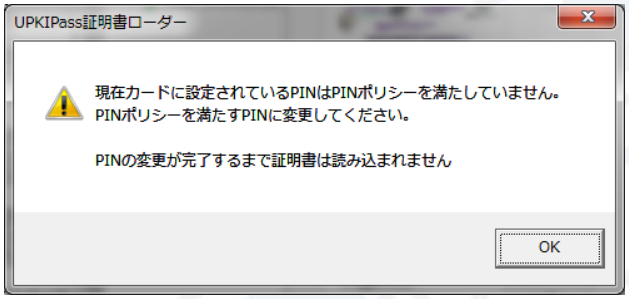
③ PINの入力
・各ユーザに対応するPINを入力



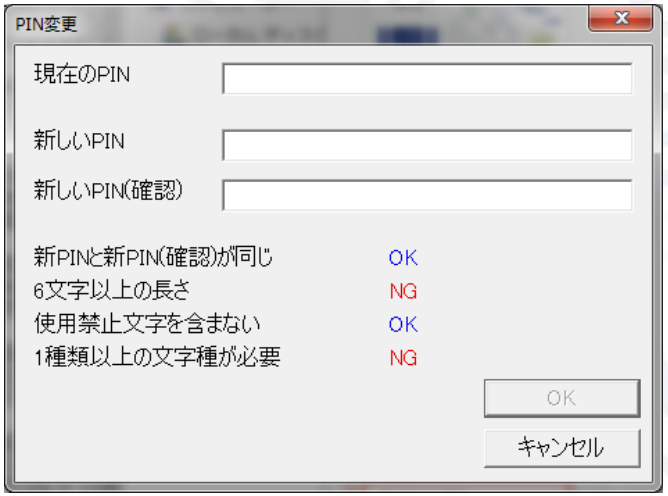
ユーザの動作（2）

④ PINの設定（初回時のみ）

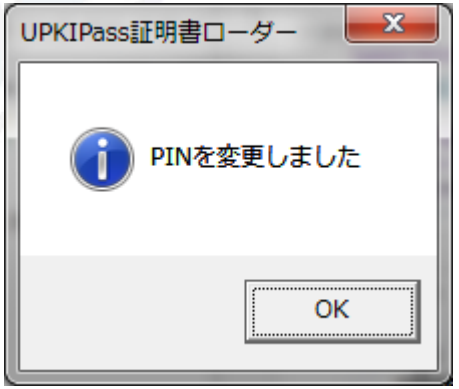
・各ユーザにPINを設定してもらうため、初期PINをポリシー違反に設定



ポリシーを満たすPINを設定

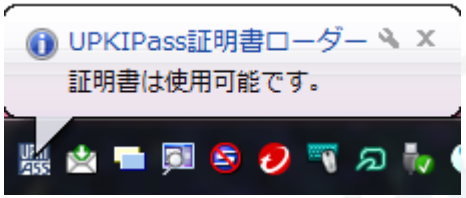


PINの登録完了

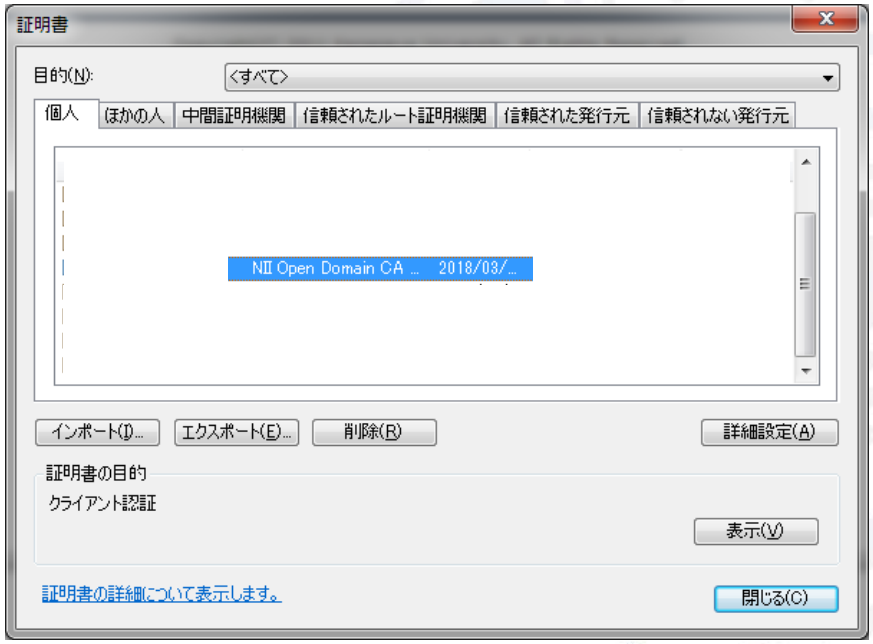


ユーザの動作（3）

⑤ クライアント証明書がPCに格納



クライアント証明書が正常に格納された場合に表示



クライアント証明書が格納されていることが確認できる

ユーザの動作（4）

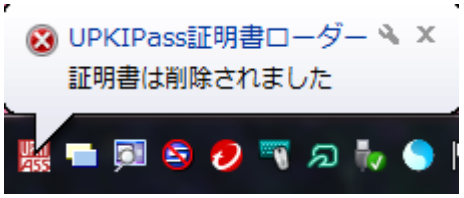
⑤ クライアント証明書認証を実施

SPに
アクセス

クライアント証明書認証

サービス
開始

⑥ カードをはずすと証明書が削除



まとめと今後の課題

- まとめ
  - ID・パスワード認証は危険
    - 多要素認証への移行が必須
  - 金沢大学では多要素認証方式を検証中
    - tiqr認証、YubiKey認証、UPKIパス認証
  - Shibboleth IdP v3でもクライアント証明書認証は容易に実現可能
  - UPKIパスとUPKIが提供するクライアント証明書を利用
    - ICカードで安価にShibbolethのクライアント証明書認証環境を実現
- 今後の課題
  - 各種多要素認証方式の実運用化