アドバンスト事例紹介

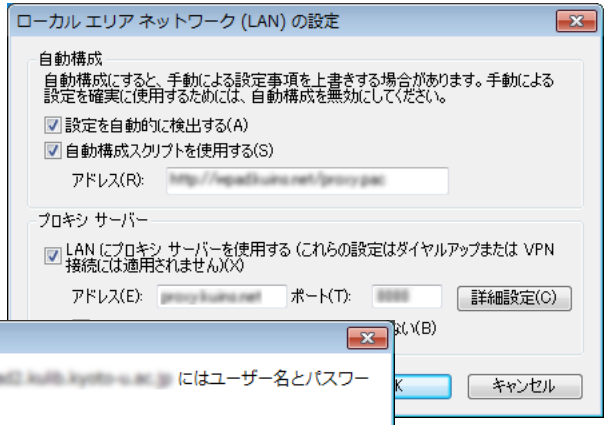
京都大学 古村隆明

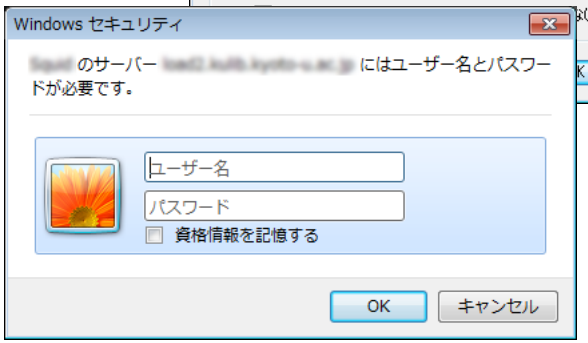
目次

- Shibboleth認証対応のフォワードプロキシ
- Windowsログオン・Shibboleth連携
- ICカード・証明書認証に向けて

Shibboleth対応フォワードプロキシ

- フォワードプロキシ
  ＝ブラウザに設定するプロキシ
- 利用目的はいろいろ





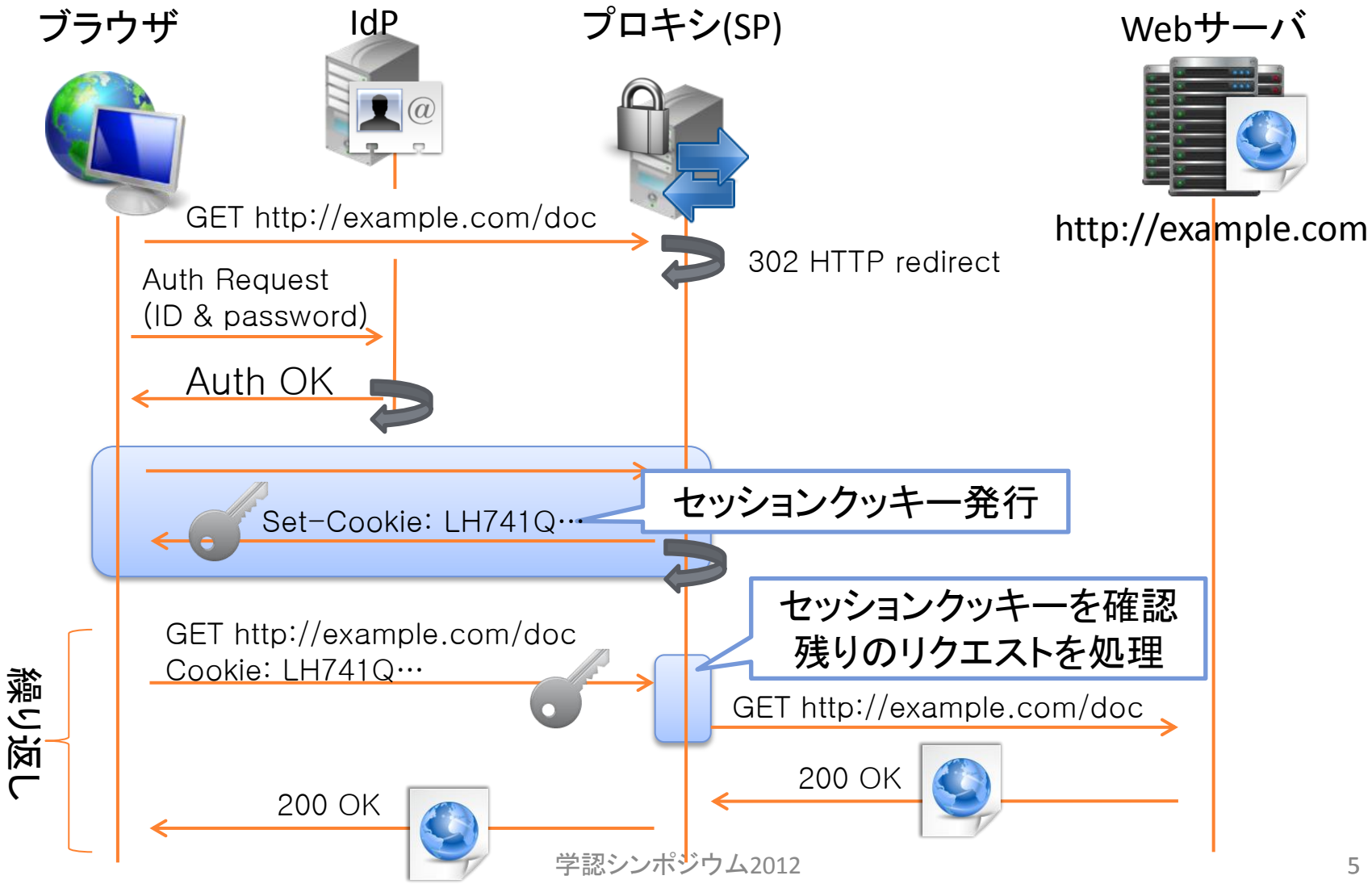
- 通常の認証方式は2種類
  - BASIC認証
  - Digest認証

- Shibboleth認証に対応させた
- 2012年度中に利用開始予定

shibproxyの実装

- Shibboleth SP 2.4.3
  - プロキシとしてリクエストを受けた際に
    未認証であればIdPへリダイレクトして認証処理
  - 認証完了後にブラウザからのリクエストを処理続行
  - diff –u スタイルで880 行の変更
- Apache 2.2
  - ソースコードの修正なし
  - mod_proxy に対する設定
  - mod_rewrite に対する設定

shibproxyの動作概要



目次

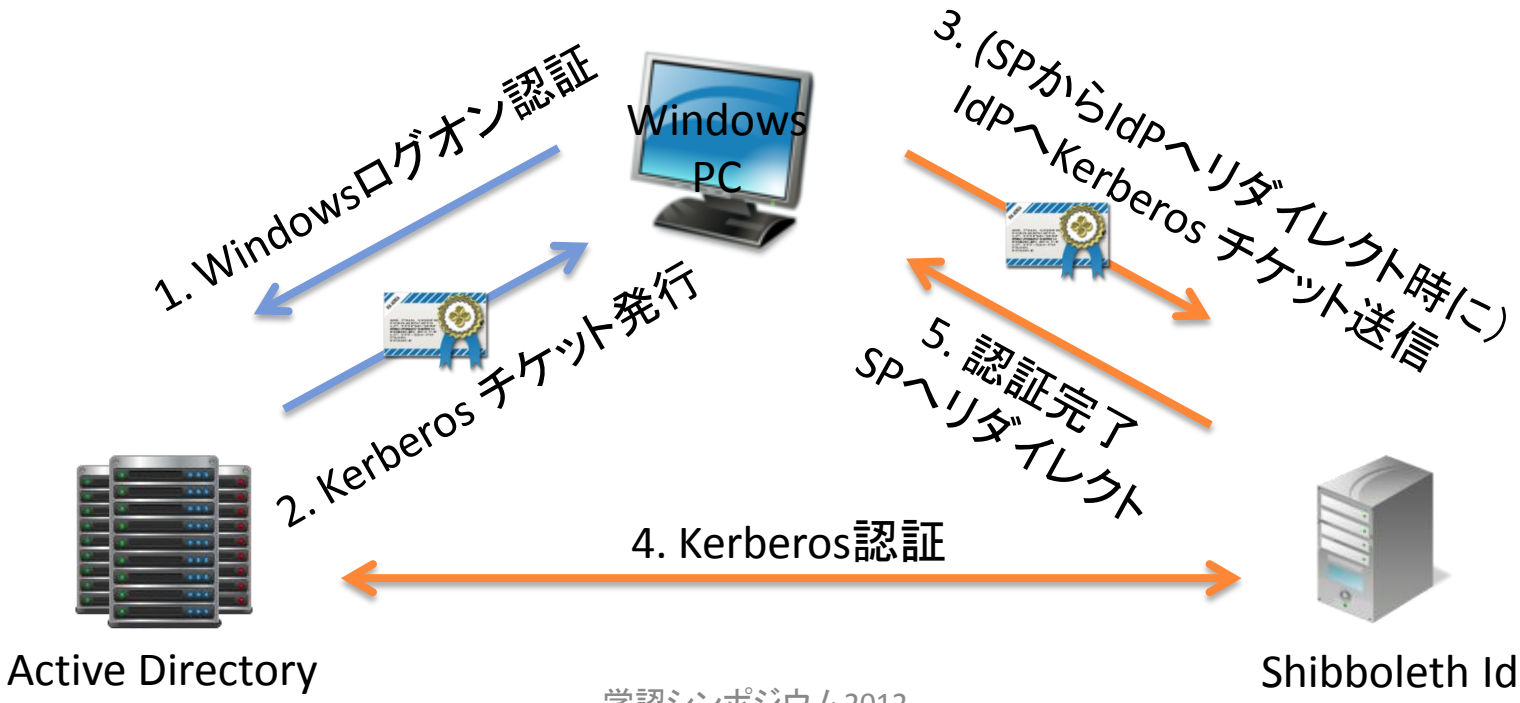
- Shibboleth認証対応のフォワードプロキシ
- Windowsログオン・Shibboleth連携
- ICカード・証明書認証に向けて

Windowsログオン連携

- WindowsログオンとShibboleth 認証を連携
  - Windowsログオン操作でShibboleth IdP へのシングルサインオンを実現する
  - WebでのSSOを超えたSSO
- 学内オープンスペースの Windows 端末での利用を検討中 (2012年度中)
- Kerberos認証を利用してIdPとActive Directory (AD)を連携
- Internet Explorer と Firefox で動作確認済み

Windowsログオン連携フロー

- Windowsログオン時にADからチケット発行(1～2)
- Kerberosチケットを利用してIdPへログイン (3～5)



目次

- Shibboleth認証対応のフォワードプロキシ
- Windowsログオン・Shibboleth連携
- ICカード・証明書認証に向けて

ICカード・証明書認証に向けて

- シングルサインオンにより利便性向上

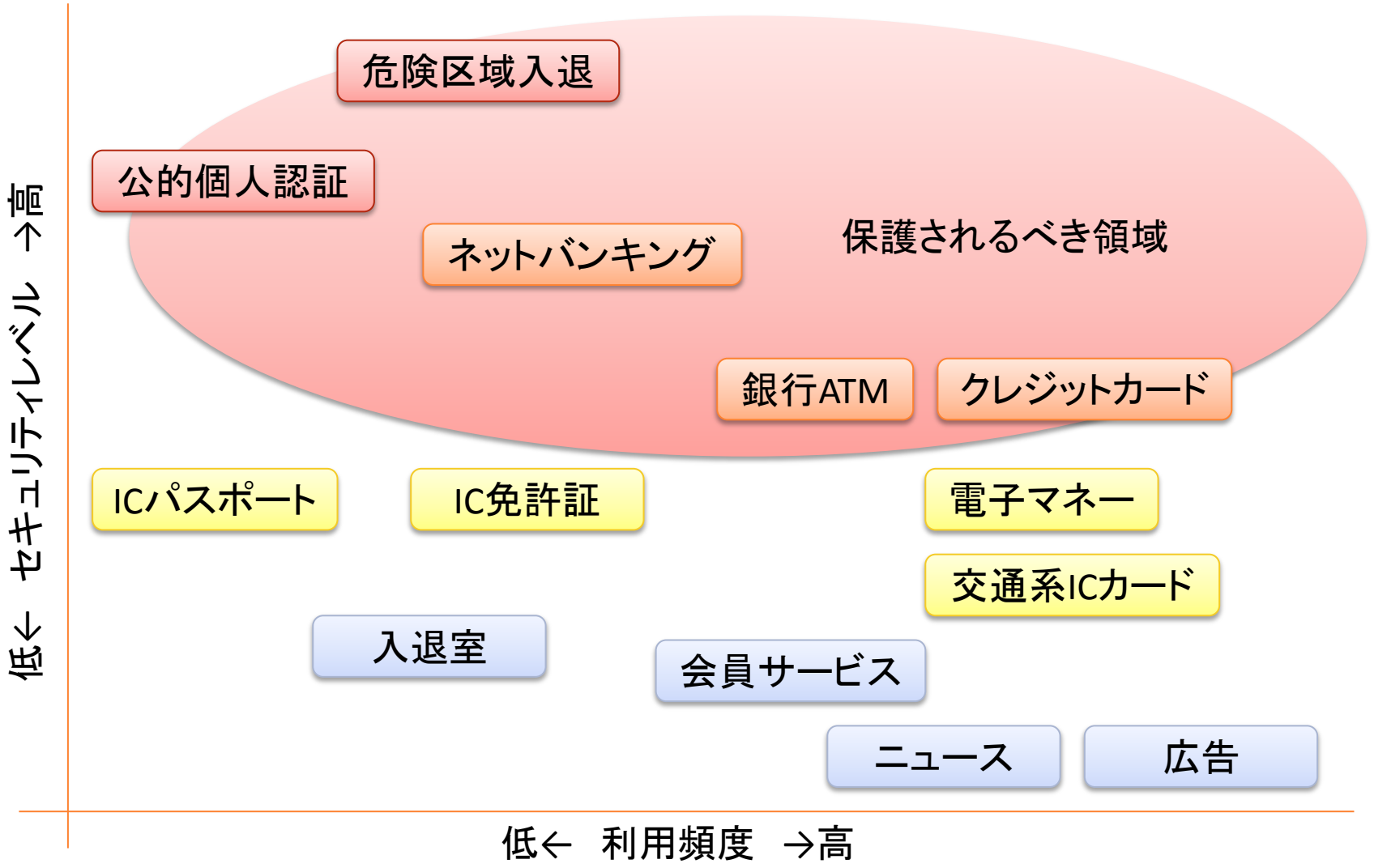
→ID・パスワード流出時のリスクは増大

- サービスごとのセキュリティレベルの違い

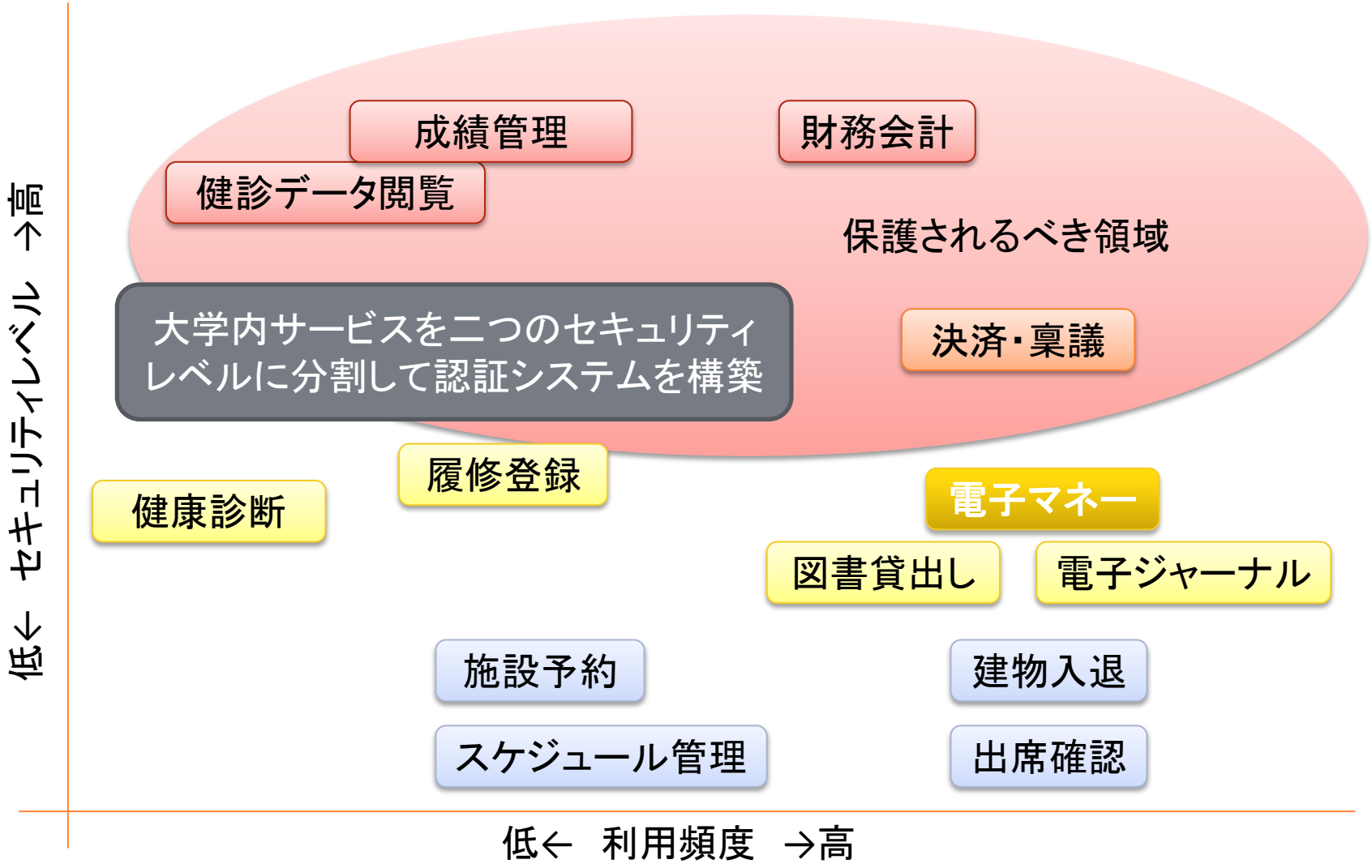
→レベルの違いに合わせた認証方式が必要

- 複数方式を組合わせて認証（多要素認証）

サービスとセキュリティレベル



大学内サービスとセキュリティレベル



京都大学での二つの認証方式

1. ID・パスワードでの認証
2. 電子証明書での認証
   (教職員に配布したIC認証カード内に格納されている電子証明書を利用)

   – 教職員用グループウェアで運用中
     • リバースプロキシ型SSO (非Shibboleth)
   – Shibboleth での運用を検討中

グループウェアでの適用例

1. ID・パスワードで認証し
   シングルサインオン

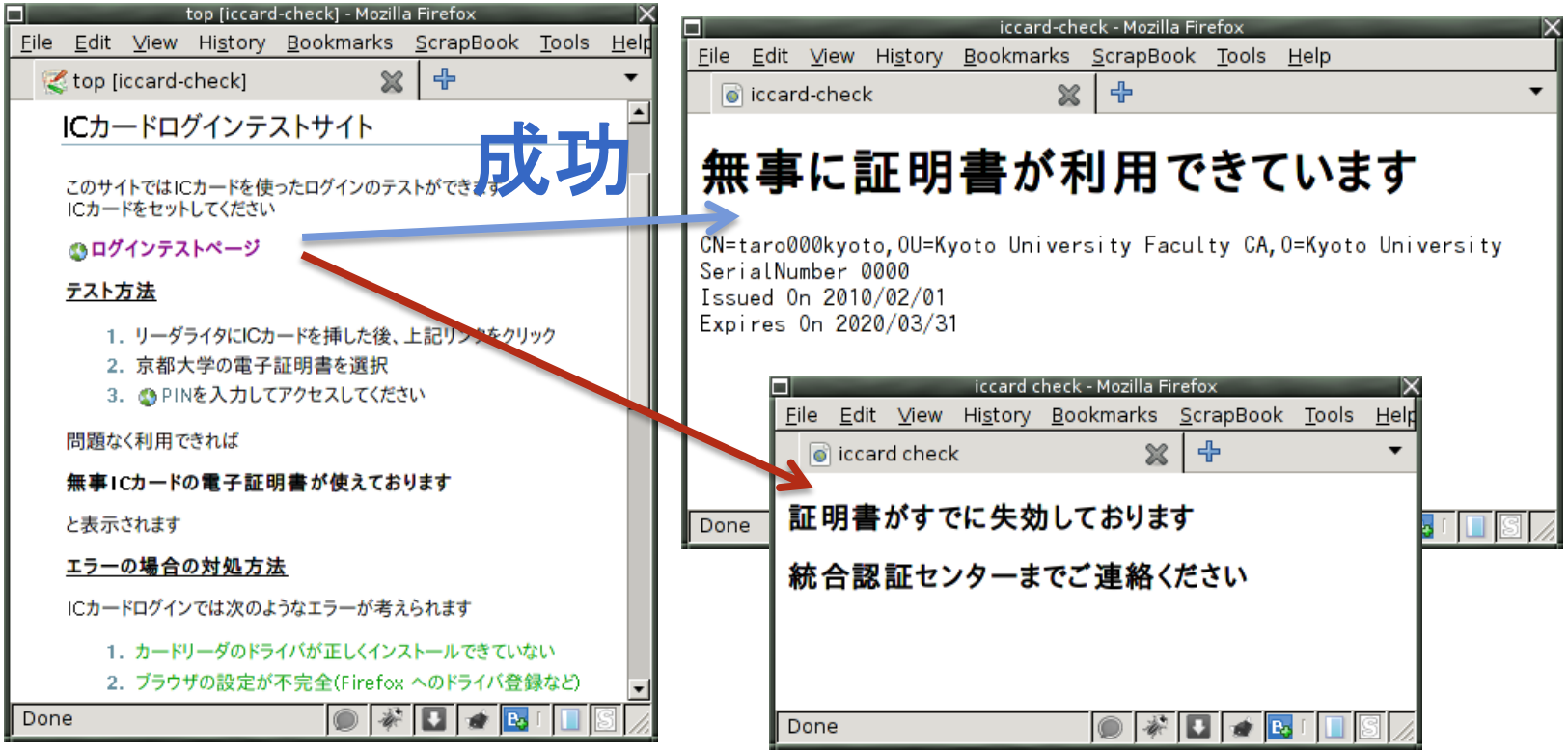
2. グループウェアの
   ポータルにログイン
   - 多くのサービスは
     この状態で利用可能

3. よりセキュアなサービス
   利用時はICカード認証が
   要求される

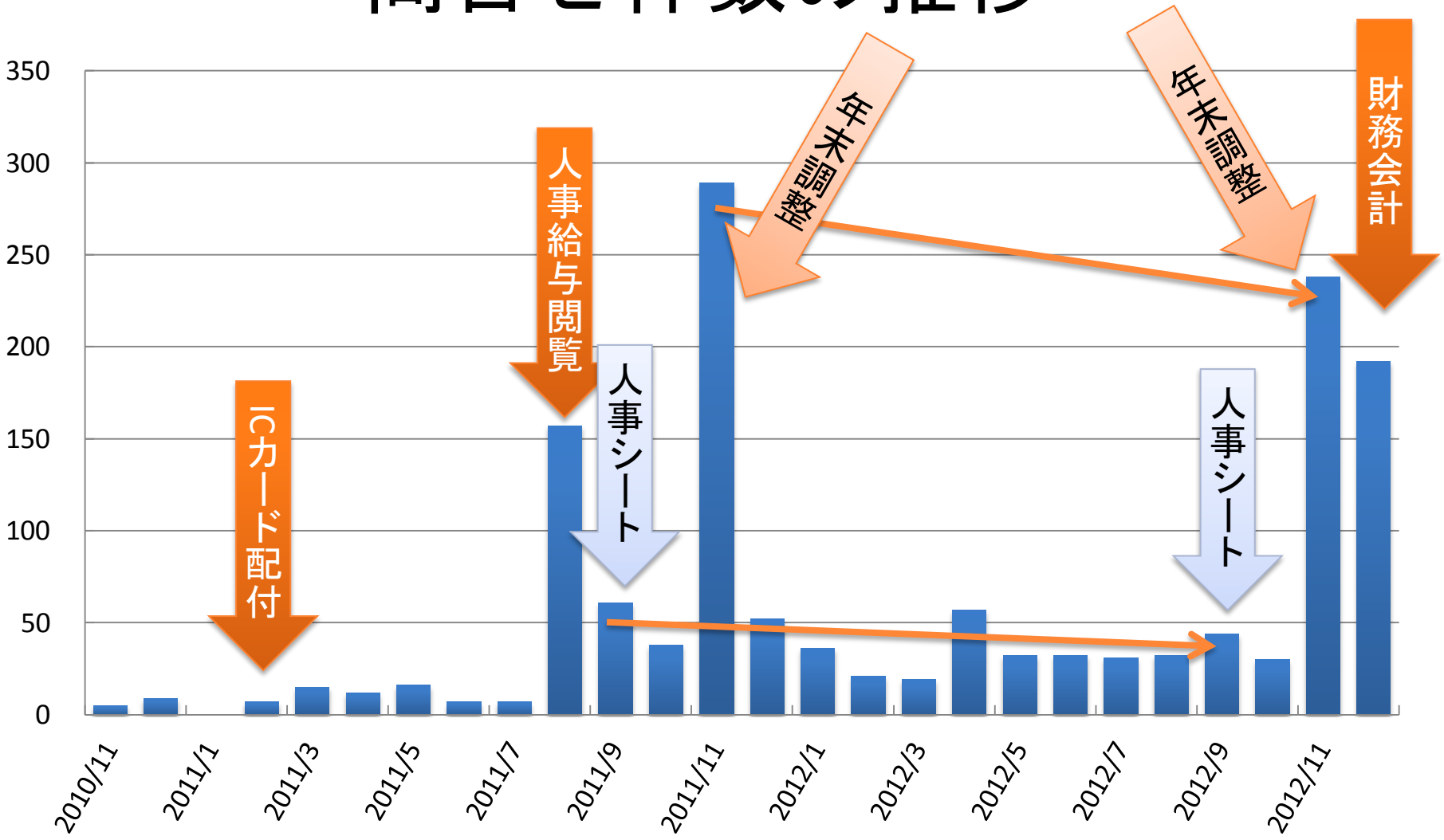
   例：給与閲覧、人事シート記入、
   年末調整、財務会計

電子証明書の確認サイト設置

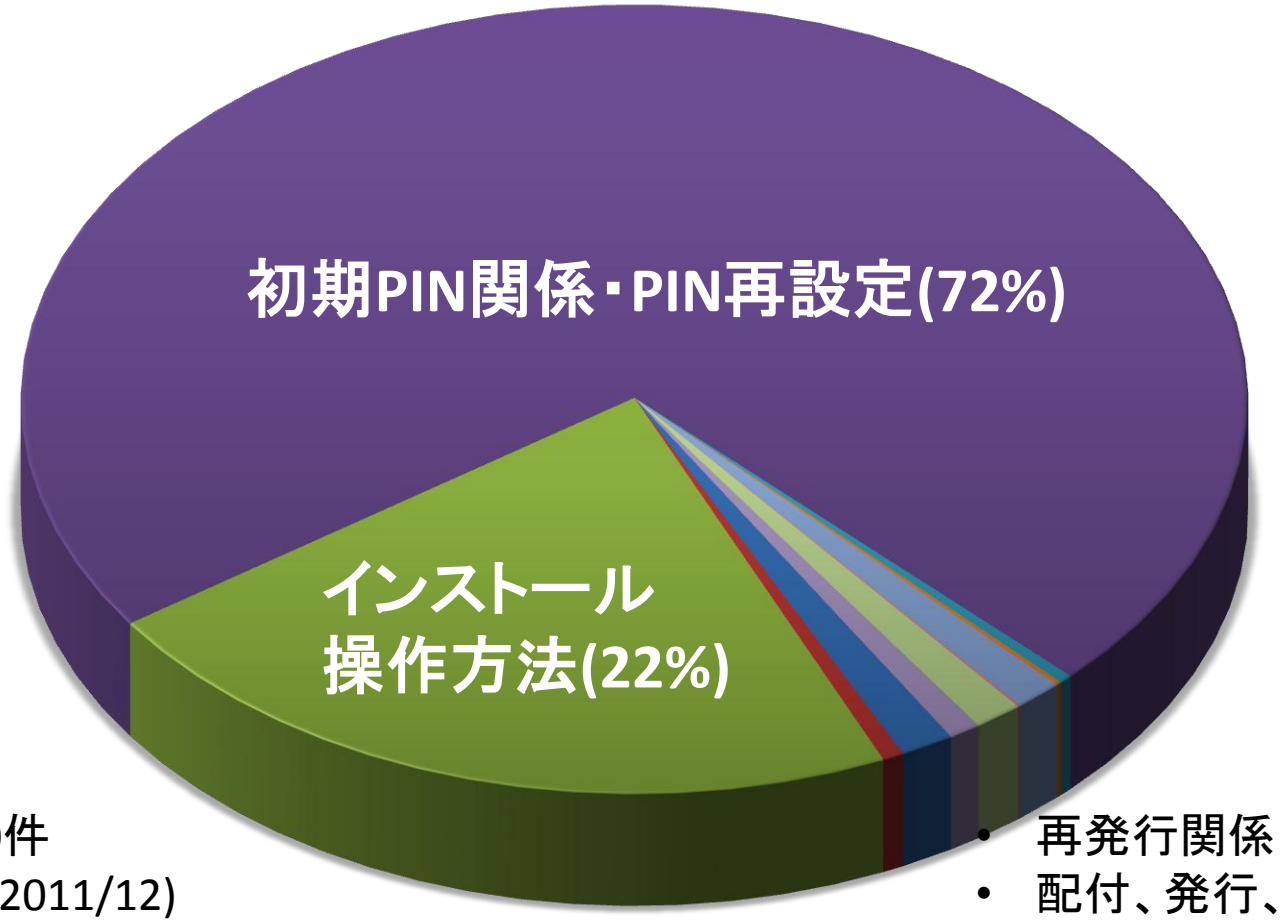
- IC身分省内の電子証明書にアクセスできるか確認するサイトを準備　→ 状況把握に有効



ICカード認証に関する問合せ件数の推移



ICカード認証に関する問合せの内訳



総数約1400件
（2009/11～2011/12)

- 再発行関係
- 配付、発行、取得資格など
- カード利活用関係
- :

トラブル事例

- PIN忘れ多発

  →リモートでの再設定機能が有効

- ブラウザやOSのアップデートに伴うトラブル
  - Google Chrome 8 → 9 で利用不可(原因未確認)
  - IE9/Win7 で利用不可多発 (原因未確認)

    (SSL re-negatiation 問題に関連か?)
  - Windows7 / Mac 64bit化 (ドライバが未対応)

Shibbolethでの証明書認証に向けて

1. IdPに証明書認証用の認証メソッドを追加
2. IdP側のデフォルト認証方式を設定
3. SP側がどの認証メソッドを要求するかを指示

- 基本動作は確認済み。調査・動作試験中。

参考：https://www.gakunin.jp/docs/fed/technical/idp/customize/certificate-auth

IdPに認証メソッドを追加

- handler.xml

```
<LoginHandler xsi:type="RemoteUser">
  <AuthenticationMethod>urn:oasis:names:tc:SAML:2.0:ac:classes:TLSClient
  </AuthenticationMethod>
</LoginHandler>
```

証明書認証用の認証メソッドを追加

- httpd/*/ssl.conf など

```
<Location /idp/Authn/RemoteUser>
        SSLCACertificateFile /opt/shibboleth-idp/credentials/Camp-CA.crt
        SSLVerifyClient require
        SSLVerifyDepth 3
        SSLRequireSSL
        SSLOptions +ExportCertData +StdEnvVars
        SSLUserName SSL_CLIENT_S_DN_CN
        SSLRequire %{SSL_CLIENT_S_DN_O} eq "Test_University_A"
</Location>
```

証明書認証の処理は apache に依存

- attribute-resolver.xml (詳細略)

IdP側デフォルト認証方式の設定

- relying-party.xml

```
<rp:DefaultRelyingParty provider="https://idp.example.jp/idp/shibboleth"
    defaultAuthenticationMethod=
     "urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport"
    defaultSigningCredentialRef="IdPCredential">
```

IdP側のデフォルトの認証方式としてパスワード認証を指定
SP側から特に指定がなければこの方式を利用

この設定を書かずにパスワード認証と証明書認証を併用しようとすると証明書認証がデフォルトになってしまう

SPから認証方式指定

- shibboleth2.xml

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

パスワードの場合

証明書の場合

IdPのどの認証方式を利用するかを指示
特に指定しなければIdP側のデフォルトを利用

まとめ

- Shibboleth認証対応のフォワードプロキシ
  - SPとしての利用例
- Windowsログオン・Shibboleth連携
  - IdPとしての利用例
- ICカード・証明書認証
  - グループウェア（リバースプロキシ型SSO）での事例紹介
  - Shibbolethでの利用に向けてテスト中