Shibbolethの運用とアプリケーションの設計・実装について

産業技術大学院大学

長尾雄行, 土屋陽介, 森本祥一, 中鉢欣秀,
市川本浩, 石島辰太郎

2009年3月13日

はじめに

- この報告書では、平成２０年度に産業技術大学院大学がUPKI SSO実証実験において実施した内容について報告する
- 本学では、平成18年度からShibbolethの普及及び導入に関して検討しており、これまでの研究成果も併用しながら実証実験を行った
- 本学での活動概要
  - UPKI提供の構築手順に従うとIdP及びSPが問題なく構築及び運用できることを確認した
  - SPとして複数ユーザのマウスカーソルをリアルタイムに可視化するアプリケーションを作成し、サービスとして公開した
  - 上記の他、運用上の工夫として、IdPの構築の半自動化を行い、実験用サーバプールを構築してアプリケーションの開発に利用した

運用系と学内実験系

- 本学では、運用向けシステム（運用系）とShibbolethアプリケーションの開発及びテストの為のシステム（学内実験系）を個別に用意して実験を行った
- 運用系
  - UPKIが稼働するフェデレーションに接続し、学外IdP経由のアクセスを許すもの
  - IdPとしてはOpenLDAPをバックエンドとするものとActiveDirectoryをバックエンドとするものの2系統を稼働させた
  - SPとしてマルチマウス・アプリケーションを開発し、非同期HTTPリクエストとShibbolethの基本的な相互運用性を確認した
- 学内実験系
  - 学内実験専用のShibbolethサーバプールを構築してSPのアプリケーション開発に利用した

主な運用系の機材

- サーバ1及びサーバ2
  - H/W: Pentium4 3GHz / メモリ1GB / GbE
  - O/S: CentOS 5.2 (32bit版)
- サーバ3
  - H/W: DualCore Opteron 1.8GHz / メモリ2GB /GbE
  - O/S: Windows Server 2003 R2 SP2 (32bit版)

運用系サーバの機能

- サーバ1(sv1.shib.aiit.ac.jp)
  - IdP (認証のバックエンドにOpenLDAPを利用)
  - SP (マルチマウスアプリケーション)
- サーバ2(sv2.shib.aiit.ac.jp)
  - IdP (認証のバックエンドにADを利用)
- サーバ3(aelad01.shib.aiit.ac.jp)
  - AD (サーバ2と接続して運用)

※サーバ証明書はUPKIのサーバ証明書発行プロジェクトより取得して利用した

運用系IdPの構築

- UPKI提供の手順書を利用するとIdPを問題なく構築し、運用できることを確認した
- 本学では、手順書をスクリプト化してIdPの構築を半自動化して運用を行うことにした
- 具体的には
  - CentOSのキックスタートでOSをインストール
  - スクリプトで必要パッケージを構築し、ノード固有のデータは管理用サーバ(非公開)からコピーして配備する
- ノード固有のデータ
  - (自動配備) OpenLDAPやShibbolethの設定ファイル
  - (手動配備) サーバ証明書と秘密鍵(秘密鍵の取り扱いを厳重にする為に手動配備とした)

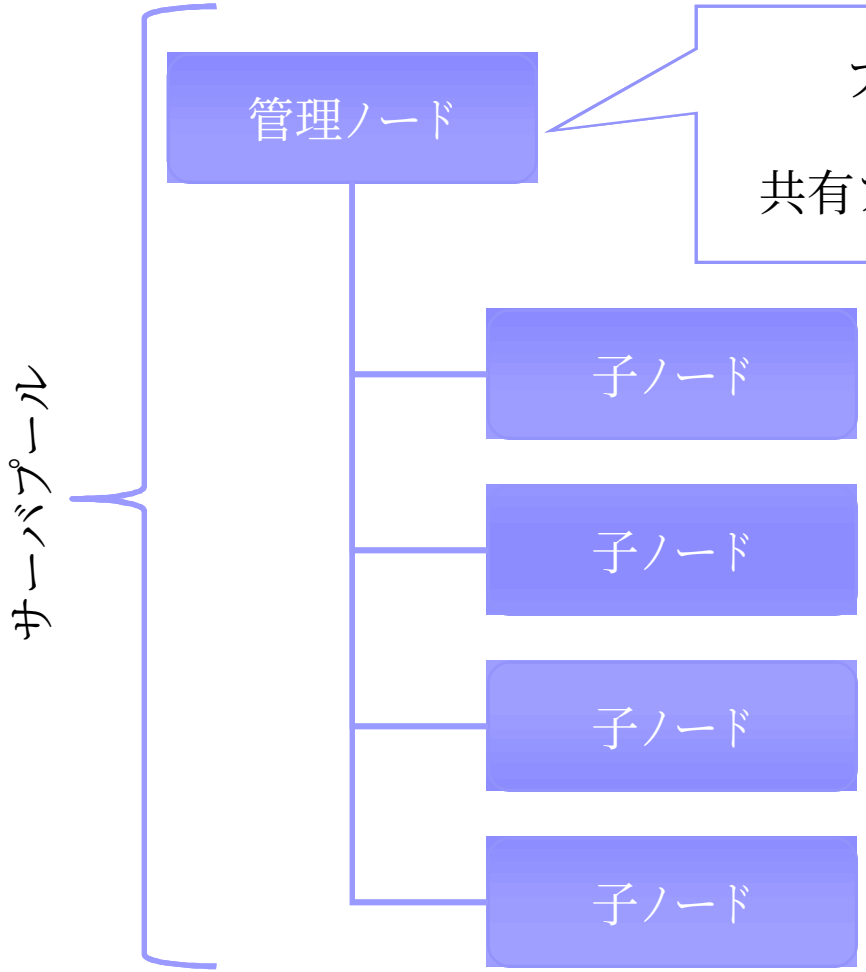
運用系のIdPでの確認事項等

- NIIのDS経由でSPの利用が可能
- メタデータの自動更新が可能
- 日本語を含む属性の送信が可能
  - Active Directory 経由
  - OpenLDAP 経由
- 未解決の課題
  - アプリケーション側でセッションを30分以上持続させる方法が不明(Ajaxを利用する場合に問題となる)
    - IdPのrelying-party.xmlの中でassertionLifetimeの設定を長くするとセッションの持続時間を長くできるが、本質的な解決ではない

学内実験系サーバプール

- ShibbolethではIdP, SP, DSというエンティティを連携させて「フェデレーション」を形成して運用を行う
- 運用の開始前に学内で仮想的なフェデレーションを作って IdP, SP 等の理解を深めることにした。
- この目的のために、以下に述べる学内実験系サーバプールを構築した

学内実験系サーバプールの概要



■ メインメモリが2GB以上のPCを子ノードとして利用することができる

■ PXEが利用できないセグメントで子ノードを起動する際にはUSBメモリあるいはCD-ROMを利用

■ 子ノード間で共有するデータはNFSもしくはNASに保存する

管理ノードと子ノード

- 学内実験系サーバプールは管理ノードと子ノードから成っている
- 管理ノードは子ノードのパッケージ構成や実験系のメタデータを一元管理する。ブートサーバとしても機能する
- 子ノードはブート時に管理ノードからHTTPでパッケージをダウンロードし、メインメモリに配備する
  - OSはSLAXをカスタマイズした
  - HDDを利用しない
  - ブートにはPXE,CD-ROM,USBメモリが利用できる
  - ブート完了後は CD-ROM や USB メモリを抜いても稼働する
  - 一枚のCD-ROMで複数の子ノードを起動できる

サーバプール･デザインシート

- 学内実験系では複数のサーバの構成を一台の管理サーバで管理しているが、メタデータはXMLのため手動で編集するのが難しく、サーバ数が増加すると管理が困難であった。
- そこで、テキストエディタで編集が可能な「サーバプール･デザインシート」を設計及び実装して実験の補助とした
- 特徴
  - 変数の展開が可能（ダミーのデータの生成に利用できる）
  - 外部コマンドの出力を取り込むことができる（証明書の内容を取り込む際に利用できる。証明書を切り替えることが容易に成った）

サーバプール・デザインシート(例)

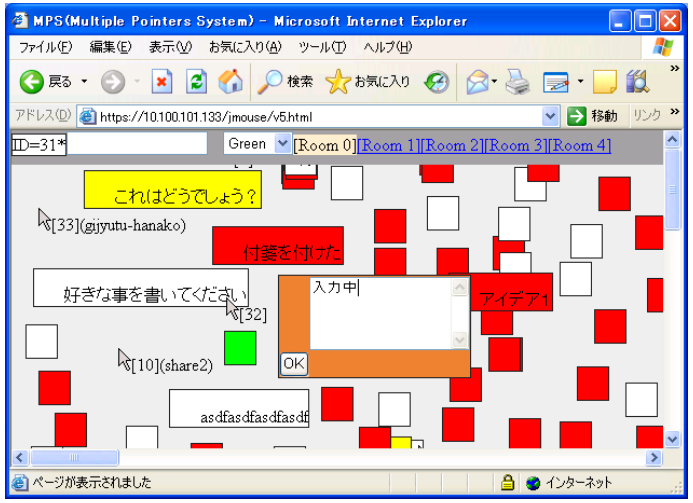
```
designsheet.default.{
 /* すべてのIdPとSPを列挙する */
 idp."https://XXX.YYY.ZZZ.WWW/idp/shibboleth". {
    hostname              = "XXX.YYY.ZZZ.WWW";
    pki.cert              =| "cat /path/to/cert | filter";
    organization.name     = "aiit${hostname}";
    organization.dispname = "${organization.name}(disp)";
    organization.url      = "http://${hostname}/";
    techsupport.surname   = "test";
    techsupport.givenname = "tarou";
    techsupport.email     = "test-tarou@${hostname}";
  }
  sp."https://10.100.101.163/sp/shibboleth". { /* 省略 */ }
}
```

Multiple Pointers System(MPS)

- Webアプリケーション上で複数ユーザのマウスカーソルをリアルタイムに共有するディスカッション用アプリケーション
- JavaScript と DHTML で実装している
- 同じ場に参加している他のユーザの活動の様子(アウェアネス)をリアルタイムに可視化するのがねらい
- Shibboleth と非同期HTTPリクエストの相互運用性を確認することができた

MPSの概要

- Web上でグループディスカッション
- 複数ユーザで同時利用
- 付箋が付けられる
- 特徴
  - マウスの動きもリアルタイムに可視化
  - ユーザの属性をカーソルに表示
  - 非同期HTTPリクエストを利用
- 実装の工夫
  - マウスの生成、移動や、オブジェクトの生成、削除、移動、ロック等を表現するバイトコードをHTTP上に流すことで通信の効率を上げている



■URL= http://sv1.shib.aiit.ac.jp/

■対応ブラウザは IE6, IE7

※ Firefox, Chrome, Safari では一部の機能が使えません

MPSで利用した属性

- MPSは下記の属性をIdPから受領して、アプリケーション固有のDBに保存する
  - uid
  - eduPersonPrincipalName
  - sn (姓)
  - givenName (名)
  - organizationName (o, 組織名)
  - organizationalUnitName (ou, 部署名)
  - eduPersonAffiliation

Shibbolethの属性情報では表現が困難な属性

- 更新を伴う属性
  - MPSでは下記の属性が該当する
    - ユーザのカーソル位置
    - ユーザのカーソルにつけるコメント
    - 付箋のテキスト
  - ユーザの役割等の動的に切り替えが必要な属性
    - 例：会議の司会者をボタンで切り替える場合など
- ユーザに紐づいていない情報
  - MPSでは、付箋の位置、テキストやロック状態が該当する

属性の取り扱い

- 前述のような属性を利用する必要がある場合、SPは下記のような設計をとる必要がある
  - アプリケーション固有のストレージ(LDAP, DB, メモリ等)を持ち、そこに個人情報のコピーを保存する。
  - 初回のアクセス時にIdPからの情報を元にユーザ情報を初期化し紐づける
- フェデレーション参加中のSP(IdP)における個人情報とそのコピーの取り扱いについてのルールが必要と思われる

ストリーミングサーバ用プラグイン

- 動画ストリーミング用のSPを構築中
  - Webページ上で動画のストリーミング配信を行う
  - ストリーミングにWindows Media Services(WMS)を使う
- WMSを直接Shibboleth化できなかった
  - プロトコルがHTTP(S)ではなくRTSPのため
- 部分的な解決策
  - Webページ上で一時IDを生成して，ビデオストリームへのURLに埋め込む
  - WMS用のプラグインを作成して一時IDにより承認を行う

まとめ

- 本実証実験では、IdPの運用とSPの開発を通じて、フェデレーション配下でShibbolethの運用が可能であることを確認した
- 今後も、現在開発中のアプリケーションや、組織間の連携を活用した教育用SPの開発等を通して活動を継続して行きたい

今後の展望

- MPSの機能を拡張して下記の機能を追加
  - PDFファイルをアップロードする機能
  - 手書き入力(タブレットPC等でWebページに手書きする機能)
  - Mac 及び複数ブラウザ対応
- 動画共有サービスの構築と公開
  - ビデオ教材をアップロードできるサービス
  - 属性に基づいてアクセス制御 (教員ならアップロード可にする等)