研究科単位での学内サービスのID統一化に向けたIdPの導入および運用についての検討

加藤 大弥[1,a)]　砂原 秀樹[1,b)]

概要：昨今のオンライン化の影響により本研究科の学生が学業で利用するサービス (以下, 学内サービス) が増加した. これらを活用するために認証認可機能が必要となったが, サービス毎に認証認可機能を導入しているため, 保持しているデジタルアイデンティティが増加した. これにより本研究科内のサービス毎においてアイデンティティマネジメントの運用コストおよびリスクが増えるという課題が発生した. そこで本研究では, 研究科内の学内サービスで利用するデジタルアイデンティティを統一化するために学内 IdP を導入し運用するための方法について検討する.

Feasibility of IdP implementation and operation for identity federation in Campus Services

1. はじめに

昨今の新型コロナウイルスの影響によって, 一昨年より講義や学生への業務作業を急速にオンライン対応しなければならないという現状があった. そのためコミュニケーションツール等の SaaS の活用や授業支援システム等のサービスを新たに学内に構築し, まずは早急にオンライン上で学生生活が可能なことを最優先に各種取り組みを行ってきた. これらの社会の変化はニューノーマルと呼ばれるようになり, ワクチン接種を行っている現段階および今後を考慮しても, 文部科学省の今後の方針 [1] にも挙げられるように, 継続的なオンラインでの学内サービスの提供を行っていく必要があると考えられる.

現在, 我々の学内に構築されているサービスは, 早急に学生生活の DX を進めるために各々のサービスがモノリシックに構築され, それぞれが独立して管理・運用されている状況である. これらを継続的に今後も運用していくという方針の下で我々は, サービス毎で保有し活用しているデジタルアイデンティティを継続的にマネジメントすることが運用上コストが高く課題であると考察した.

そこでこの課題を解決するためにサービス毎でデジタルアイデンティティを独立に管理することをやめ, 統一化した ID に対する管理を一元化させることでアイデンティティマネジメントを一括して行い, かつサービス毎に必要なときに認証認可の機能を提供する Identity Provider(以下,IdP) を学内に設置することを検討し, 現在導入を進めている. 本論文では IdP を学内に導入し運用していくための方法論およびポリシーについて得られたノウハウをもとに議論する.

[1] 慶應義塾大学メディアデザイン研究科
KMD, Yokohama, Kanagawa 223–8526, Japan
[a)] i.mas.trunk@kmd.keio.ac.jp
[b)] suna@kmd.keio.ac.jp

2. 学内サービスとデジタルアイデンティティ

2.1 学内サービス運用の課題

我々の研究科では, 早急な講義や学生への業務作業のオンライン化を行うために構築が完了した学内サービスから順次利用可能とする方針でシステム構築を行った. これにより早急なオンライン化が実現することが可能となった一方で図 1 に示すように, それぞれのシステムで認証機能を実装し認証情報が保存されている. しかしこれでは, 研究科内の新入生および卒業生の年度更新作業, サービス毎の個人情報の漏洩やパスワード変更等のセキュリティ対策やリスクマネジメントを行わなければならないという問題が発生している. またサービス毎に運用者がいるため, 認証方法, クレデンシャルの利用および保存方法, セキュリティポリシー等が異なっているという課題も抱えている.

今後も学内サービスが増加していくことを考えた場合,

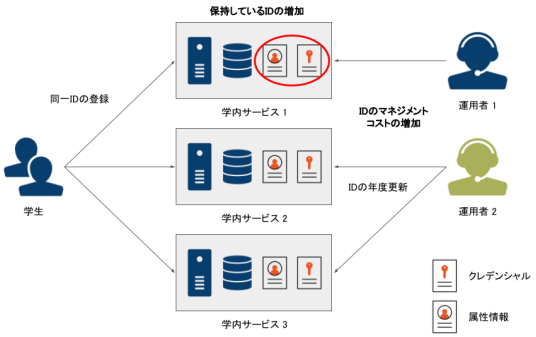

図 1 学内サービスの増加によるデジタルアイデンティティマネジメントの課題

現状のアイデンティティマネジメントでは継続的な運用が困難になると予想している. そこで, 学内に IdP を設置し各学内サービスで用いているデジタルアイデンティティを統一化し, 管理すべきデジタルアイデンティティを削減することで, これらの課題を解決することが可能となるのではと仮定し, 実際に導入するための方法および導入後の管理運用について検討し導入を進めている.

2.2 学内 IdP の役割

まず学内で IdP を運用するためにその役割を明確にする必要がある. 我々の学内における IdP に期待する役割は大きく分けて

- デジタルアイデンティティの統一および一括管理
- 学内サービスへの認証認可機能の提供
- デジタルアイデンティティマネジメントのコスト削減

であり, 図 2 のように機能することを想定している. この学内 IdP の利用者は我々の研究科の在校生, 卒業生, 教職員, 他大学所属の授業履修者等の多岐にわたり, またおそらく長期の運用において例外となる新たな属性の利用者が増えることが十分に考えられる. そのため学内の IdP は柔軟なオペレーションに対応することを念頭に置いたものである必要がある.

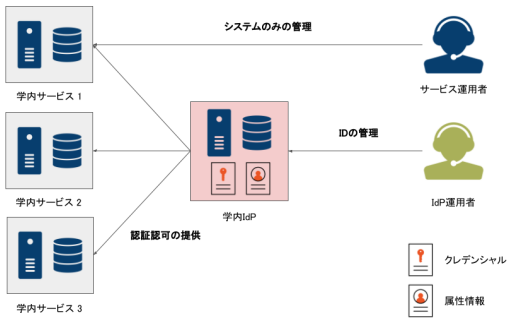

図 2 想定している IdP の役割

また我々は自身で IdP を設置せず, 各大学において構築されている共通認証システム (我々の所属組織においては, 慶應義塾共通認証システム keio.jp[2]) と呼ばれるような全学を対象とする IdP を活用することについても検討を行った. 具体的には, 学認への SP 登録を利用した認証連携を行う方法である. この方法は各大学での導入事例も多くあり [3], 導入方法についても詳細に示されている. しかし,Shibboleth を用いた認証連携を実現する際の問題点としては, 活用までの初期設定の煩雑さが挙げれる [4][5]. この作業をサービス毎に都度行う必要があるということ, また SAML および XML の仕様が大部を占めることになっており, 正しく利用し始めるまで知識および技能の習得が難しく連携を開始するまでに膨大な時間を要すること,OWASP2019[6] から現在まで問題視されている XXE 攻撃への対処として json 形式の認証情報を利用することを推奨しているという点から, 我々の研究科では扱いきれないという判断をした.

また, 本研究科では, 大学組織外の講師や研究員, 他研究科や他大学に対しても開講している講義へ参加している人々に対してアカウントを発行する必要があるため, 学認に挙げられるような大学に所属していることを証明することが基本的な役割である IdP を活用することが適していないおらず, 役割を分けるべきであるとした.

しかしこれらの例外として大学組織の統合認証基盤が後述する IDaaS を用いている場合については, 所属組織のポリシーによって共通認証システムを利用することも可能であると考えられる.

2.3 IDaaS の活用

IDaaS とは,Identity as a Service の略称でありアイデンティティ管理をクラウドにて管理するサービスであり,2.2 で挙げた役割をクラウドサービスとして提供しているものである. 代表的なものとして,Auth0 や Okta が挙げられ様々なサービスで本論文と同様の役割を持たせるために用いられている. また,Google Workspace によって組織のアカウントを統合している場合においても同様に 2.2 の役割を提供しているほか, 学内のサービスをソーシャルログイン等を用いて統合することが可能である.

組織内において資金面やデジタルアイデンティティの運用ポリシーを満たすことができれば利用を検討し, これらの IDaaS を活用することが可能である場合, 本論文の目的であるデジタルアイデンティティを継続的にマネジメントする運用コストを抑えるという目的を達成することができ, 尚且つ後述する様々な検討事項, 運用ポリシー, システムの管理運用についてもサービスに一任することが可能であるため十分に検討する必要がある.

3. IdP 導入の検討

3.1 IdP 構築のポリシー

学内に IdP 構築するための検討したポリシーを表 1 に示す. この中でも重要視した検討事項については, 後述する IdP の選定, クレデンシャルの管理, 属性情報の管理である.

また構築時に検討する項目としては, システムをどこまで物理的に分割するか, 冗長性およびダウンタイムの SLA をどう設定するかについては組織ごとに異なるため十分検討する必要がある.

我々の検討した項目について一部示す. システムをデプロイするマシンの冗長性については, マシン自体が起動しなくなった場合に備えてホットスタンバイを用意している. ホットスタンバイをしているマシンには今回 IdP として採用した keycloak の機能である, スタンドアローンクラスターモードを適応しており, これにより現在起動しているサーバーとデータソースやセッション情報のリアルタイム共有, メインのサーバーがダウンした際の自動フェイルオーバーを実現している. またデータベースについても冗長構成をとっており, 上記のようにリアルタイムにユーザーのデータソースやセッション情報をバックアップする必要があることから, ストリーミングレプリケーションを行う想定としている. そのためにデータベースは PostgreSQL を採用している. またこれらのシステムは, ホットスタンバイは用意しているものの障害発生時にできるだけ早く普及する必要がある. そこで我々は Docker[7] を活用し, システム構成をコードとして管理しておき, そのコードを実行するだけで元のシステム構成を手間なく高速な構築を可能にしている.

3.2 IdP の選定

IdP については,Keycloak を採用している. 選定項目としては,

- 開発コミュニティの活性度
- ライセンス
- 提供可能な認証認可のプロトコル
- ロールの詳細設定
- IGA への拡張性

を重視している. まず開発コミュニティが活発であり継続的な開発が行われていることが非常に二つの側面にいて重要である. 一つ目はソフトウェアを利用していくなかでセキュリティ対策が行われていくことであり, 二つ目は利用していく中でバグやプロトコルの実装不備を発見した場合, 即座にコミュニティへ貢献し反映することが可能である点にある. 提供可能な認証認可プロトコルについては, 学内サービスに対して認証機能を提供するために,OpenID Connect を利用する.OpenID Connect を採用する理由としては学内サービスに認証機能を導入する際に, クライアントライブラリが様々な言語, ソフトウェアで提供されていることが挙げれれる. 本研究科において提供可能な学内サービスの条件として,OpenID Foundation Certified Relying Party Libraries List[8] にある,OpenID Connect のクライアントライブラリを利用可能であることとした. また,OAuth2.0 implicit grant flow の認可結果からユーザー識別子を取り出し認証を行う通称 OAuth 認証というものが存在しているが, これはセキュリティリスク [9] から利用しないものとする. ロールの詳細設定については, 授業支援システム等で学生, 教員, 職員のように区別する必要がある場合にロールベースアクセスコントロールを行う必要があるためロールを細かく設定可能であることが必要となる. 詳細は 3.4 に記述する.

3.3 クレデンシャルの管理

クレデンシャルの管理については, セキュリティの観点からストレージ用のプライベートなネットワークを構築しストレージ用のマシン上のデータベースで管理することとしている. またクレデンシャル情報を失うことが IdP 運用において一番重要なインシデントとなるため,RAID によるデータの冗長化だけではなく, 複数マシン上のデータベースに対してレプリケーションを行うこととしている. この時用いることが可能なデータベースは数多く存在しており, 各組織のストレージ状況に合わせたものを使用することが可能である. 我々の場合においては IdP 上のデータ更新を即材に各データベースに反映させること, データベースが動作しない場合即座に切り替わることを重要視し,PostgreSQL のストリーミングレプリケーションと複数台のマシンを利用している.

3.4 属性情報の管理

属性情報についても 3.3 と同様であり, 外部のストレージに保存している. 収集する属性情報に関しては学内サービスのユースケースを十分に検討し最低限度のものを収集し管理することとしている. 主な属性情報としては, 氏名, メールアドレス, 学籍番号, 性別, 電話番号, 住所等が挙げられるが, 本当に学内サービスにおいてこれらの情報を収集する必要があるのかを精査する必要がある. 具体的には, 学部, 研究科の内部情報ページへのアクセスするというユースケースの場合, ユーザーを判別するために少なくとも性別, 電話番号, 住所の情報は必要ない. この場合本研究科では,kmd.keio.ac.jp という研究科で配布しているメールアドレスを配布していることからこの情報を用いることが適切だと考察できる.

また必要な属性情報であってもすべてを IdP で管理する必要があるかを精査する必要もある. 属性情報が増加すると,IdP にユーザーを登録する際にパスワード管理者の負担が増えることになる. また属性情報が増えるほどデータの関連性が強くなり IdP からデータが漏洩した際のリスクが高くなるという問題があるためどの属性情報をどこで管理するのかが大きな課題となる.

近年では, 属性情報についてもまとめて管理することで属性情報を統一化してサービスとして提供することが可能な Identity Governance & Administration(IGA) を実装す

表 1 本学内で検討した IdP 構築のポリシー

検討項目	検討する内容	検討結果
物理マシンの選定	CPU,RAM, 電源の冗長性	要検討
マシンの冗長性	ディスク,RAID	HDD,RAID 1
システムの冗長性	マシン構成, ネットワーク構成	ロードバランスなし, ホットスタンバイ
データベースの冗長性	マシン構成, レプリケーション	PostgreSQL, ストリーミングレプリケーション
IdP の選定	ソフトウェアの選定,License	Keycloak
クレデンシャルの管理	保存場所	Keycloak
属性情報の管理	保存場所	Keycloak(or Midpoint)
ソフトウェアのデプロイ	デプロイ方法	Docker
ダウンタイム SLA	ダウン時の対応, 推定復旧時間	要検討, 法定停電時は対象外等

る手法もあり, これにより学内サービスをデジタルアイデンティティについての管理を全く行わずに実装することが可能となる. 我々は次のステップとして導入を検討する予定である.

4. IdP 運用の検討

4.1 IdP 運用のポリシー

次に構築した IdP を運用するために検討したポリシーの一部を表 2 に示す. 基本的な運用方針としては,2.2 でも述べたように必ず発生しうる例外な処理に対して柔軟に都度対応することが前提になっている. これは, ポリシーを定めないという考えではなく, 想像できない例外的な対応を処理することが可能な構造になっているかということを発生する対応の都度確認する必要があると検討したためである. しかし, ユーザー登録等の IdP としての基本的な機能をどこまで学生に操作させるべきか, ソフトウェアの更新及びバージョン管理をどのように行うべきかといった基本的な運用ポリシーは管理者内で共通認識を持つことが可能な程度に定めている. 本論文では電源の冗長性, ホットスタンバイ, 記憶媒体のストック等についてのハードウェアに関する運用についての検討は割愛する.

4.2 ユーザー登録, およびアカウントリカバリ

ユーザー登録, およびアカウントリカバリは IdP を運用するために最も重要であり例外的な運用が発生する項目である. これについては現在においても検討しきれていない部分があるが, 基本方針としては, ユーザー名をメールアドレス, シークレットをパスワードとして登録し, 利用することをファーストステップとした. パスワードの作成ポリシーは数多く存在しているが我々の基本方針として, ある程度長いランダムな文字列を管理者側で自動発行し基本的にそのパスワードを学生が利用するというポリシーとしている. これはユーザビリティが著しく低下するものブルートフォース攻撃およびリスト型攻撃の突破可能性を極力低下させるためである. この時のパスワードは, 視認性を考慮して o01l+/を除いた数字および小大英字 58 文字から,12 文字を生成することとした. また運用の基本方針としては,NIST SP800-63-3[10] のパスワードの項目に準拠し, 学生がパスワードを適切に管理しコピーアンドペーストで入力することを想定している. ただし今後の展開としてスマートフォンでのソフトウェア OTP や FIDO を活用した 2FA の導入を検討しており, パスワードのみでの認証を行わない (PIN 等によるその場にいることを証明する行為は必要) ことを想定している.

アカウントリカバリについては, 現状の原則として管理者がランダムパスワードを再設定し学生にメールで通知する方法を検討している. 本研究科では, 研究科独自のメールアドレスを学内サービスとは別に入学時に発行しているためこのメールアドレスを必ず利用することとし, 入学時点で利用者が本人であるという認識のもとで行う. しかし, 学内サービスにメール機能も含まれている場合もしくは, 学内以外の人が利用する場合について何をもとに受け取りてが本人であるかを確認するのかという例外的な処理については検討しきれておらず今後も課題となっている.

4.3 システムアップデート

IdP として活用しているシステムをアップデートする場合について述べる. 結論から述べると, 手順書に記載されているコマンドの入力等の手順自体を Dockerfile 等のコードに落とし込んでおき, 管理者がアップデートしたいバージョンを

```
# .env This is a version control file.
KEYCLOAK_VERSION=15.0.2
MIDPOINT_VERSION=4.3.1
```

のように指定し, その Dockerfile を実行するだけでシステムアップデートを行えるようにしている. これは手順書を読んですべてのコマンドを手入力して随時実行していくことによる更新作業時間の短縮, 打ち間違いや実行されたことを確認してからコマンドを入力するなどの手順書からは読み取りにくい文脈によるヒューマンエラーを減らすことを目的としている. また, アップデートした際にうまく動作しなかった場合のダウングレードおよびロールバックについても同様であり, 現在動作しているシステムのバー

表 2 本学内で検討した IdP 運用のポリシー

検討項目	備考	参考
ユーザー登録	時期, 登録者	メールアドレス作成時, 管理者が一括登録
ユーザー情報通知	書類, メール,PGP 等	入学の書類に同梱
アカウントリカバリ	IdP の機能, メール, 管理者	管理者がリカバリ, 再配布については検討中
提供する認証認可プロトコル	SAML,OpenID Connect 等	OpenID Connect
ソフトウェアの運用	マシン,VM,Docker	Docker
システムアップデート		ダウンタイム, メンテナンスは学内に通知, カナリアアップデートも検討

ジョンさえ指定すれば任意のシステムを構築することが可能となっている.

しかし, コードとして表現しているアップデート手順を変更する必要がある場合も考えられ,Dockerfile 自体のメンテナンスについても都度していく必要がある. これにより手順書以外に管理運用しなければならない項目が増え, 管理者の負担になるのではないかという問題が考えられるが, 本方式の目的はあくまでも先述べたようにシステムアップデート作業の時間短縮と作業時のヒューマンエラーを減らすことであるためそのためには必要な負担であると考えている.

また, 実際のアップデートを検討する際にテスト環境として本番環境と同様の環境でダミーデータを用いて動作検証を行うための環境を用意しており, そこでバージョンアップが正しく行えているのか, 影響の有無, 依存関係の有無を確認した後に本番環境の更新を行っている. 現在この確認作業は管理者が認証機能が動作しているか, 認証前後の RP の http レスポンスが正しいかを curl やウェブブラウザを用いて手作業で行っている. これについては, アップデート手順と同様にテスト手順, 判定をコードして落とし込み,CICD ツールを用いて今後自動化していくことを検討している.

5. 運用状況の可視化

ここまで定めた運用ポリシーが正しく反映されているのか, 構築された IdP が正しく動作しているのか, また, 動作しているシステムの状況, アクセスログや監査ログによる動作およびセキュリティチェックを行うために運用状況を可視化する必要がある. これらは各システム毎, ソフトウェア毎, マシン毎にそれぞれアクセスしログ情報を確認することやマシンの CPU,RAM, 各種 I/O をコマンドで確認することでも可能である. しかしこの手法では, 知識, 運用方法, マジカルナンバー等の特定の運用者個人に大きく依存してしまい, その運用者がいないと継続して運用ができない状況を作り出してしまう. そこですべてのシステムの状況, ログ等を集約し可視化することで問題が発生した際に原因を特定しやすい管理基盤を構築することが重要であると考えた.

ログを管理するための管理基盤の役割としては, ログの収集, 保存, 検索, 可視化である. ログ基盤については, 難しいことはせず, 一般的に用いられているソフトウェアを利用することで導入コストおよび運用コストを抑え, リアルタイムなシステム状況の把握, ログの検索およびデータのパースを行うことを重要視することとした. そこで我々は,ElasticSearch によるログの検索,Fluentd によるデータの収集,Kibana による可視化を用いることで管理基盤を構築し, リアルタイムなシステム状況の把握については, 各システムに Metricbeat を導入することでシステム状況を一括管理を実現した.

表 3 収集しているシステム状況

取得する項目	取得したデータの内容
cpu	CPU usage
load	CPU load averages
memory	Memory usage
network	Network IO
process	Per process metrics
process_summary	Process summary
uptime	System Uptime
socket_summary	Socket summary
core	Per CPU core usage
diskio	Disk IO
filesystem	File system usage for each mountpoint
fsstat	File system summary metrics
raid	Raid
socket	Sockets and connection info (linux only)

5.1 システム状況

システム状況については, 各マシンごとではなく動作しているソフトウェアの Docker コンテナ単位で可視化を行っている. 収集している情報については細かく検討することができていないが, 表 3 に示している取得可能なほとんど情報を保存している. 現状の我々のシステムから得られるシステムのデータ量は 1 日あたり 450MByte 程度となっており, 年間 165GByte となっている. 今後システムを冗長化していく場合, データ量が増加していくため取得する情報を細かく取捨選択する必要がある. 実際のシステム状況の可視化について図 3 に示す. 現状の画面では, 右上に Docker の各種稼働状況 (CPU,MEM,DISK_IO), 右側に稼働コンテナ数, 下部に稼働状況のリアルタイムなグラフを表示している.

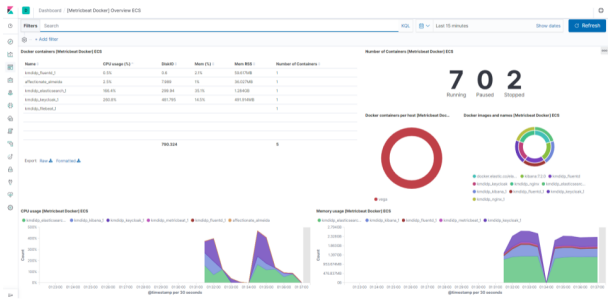

図 3 実際のシステム状況の可視化画面

5.2 アクセスログ・監査ログ

IdP に対する認証認可をした際のアクセスログおよび監査ログについても同様にログ管理基盤に収集している. 我々は収集した大量のログの検索およびログ内の各種項目のソート高速に行うこと, インシデント発生時に然るべき組織に対して早急に該当ログを提出を実現するために, アクセスログおよび監査ログを json 形式で出力することとした. また, アクセスログおよび監査ログは政府機関における情報システムのログ取得・管理の在り方の検討に係る調査報告書 [11] を参考として,1 年間分を直ちに検索可能な形式で保存している. 実際に出力されるアクセスログは,

keycloak_1 {
”timestamp”:”2021-11-01T12:59:27.412Z”,
”sequence”:132,
”loggerClassName”:”org.jboss.logging.Logger”,
”loggerName”:”org.keycloak.events”,
”level”:”WARN”,
”message”:”type=LOGIN_ERROR,
realmId=master, clientId=security-admin-console,
userId=null,
ipAddress=192.168.100.202,
error=user_not_found,
auth_method=openid-connect,
auth_type=code,
redirect_uri=http://192.168.100.202:8080/auth/admin/master/console/,
code_id=67e3d559-7bcd-4d46-8b70-25220a879400,
username=IOT,
authSessionParentId=67e3d559-7bcd-4d46-8b70-
25220a879400,
authSessionTabId=C1M6msV0w_c”,
”threadName”:”default task-1”,
”threadId”:208,
”mdc”:,
”ndc”:””,
”hostName”:”d576d6160b96”,
”processName”:”jboss-modules.jar”,
”processId”:437
}

となっている. 示したのは、テスト用ネットワークにおける管理コンソールへのログイン失敗時のアクセスログである。実環境では,ipAddress が実際の接続元 IP、redirect_uri が学内サービス URI となる。現状としてはログ全体の検索機能およびソート機能のみを実装しているが, 今後は json をパースしアクセス元 IP アドレスや単位時間における可視化を行う予定である.

6. 考察

昨今の急速な DX 化を進めるために学内に様々なサービスが構築され活用されてきた. しかしこれらは早急にサービスを学生に提供することを目的としているため, それぞれがモノリシックに独立して管理・運用されている状況である. 我々は本論文で示したように, 今後もこれらを継続的に運用していくためには, サービス毎で保有し活用しているデジタルアイデンティティを継続的に管理運用することが必要であると考察した.

そこでこの課題を解決するためにサービス毎でデジタルアイデンティティを保有することをやめ,ID に対する個別の管理を統一化させることでアイデンティティマネジメントを一括して行い, かつサービス毎に必要なときに認証認可の機能を提供する IdP を学内に設置することを検討し, システム構築ポリシーおよび運用ポリシーについて議論, 検討を行った. また検討結果をもと IdP の導入を進めている.

またここまでの成果物として, 実際に作成した学内に IdP を構築するためソフトウェア群, ログ管理基盤, バージョン管理機能を有している検証用の Dockerfile およびパッケージがあり. 今後はこのパッケージを簡易的な IdP 検証基盤として一般公開する予定である.

7. まとめ

本論文では, 昨今の DX 化の影響によりデジタルアイデンティティを保持しているモノリシックな学内サービスが増加したことで, アイデンティティマネジメントの運用コストおよびリスクが増えるという課題を, 学内 IdP を導入しデジタルアイデンティティを統一化することで解決できると考え, 実際に IdP を学内に導入するため実現可能性の検討および実現のためのポリシーを検討した. 最終的に学内へ IdP,IdP を運用するために各種データおよびログを可視化するための管理基盤を導入した. 今後は学内サービスに順次認証認可機能を提供し ID の統一化を行っていく予定である. また, 実際の運用を通して得られた知見を知識としてではなく,CICD および作業パイプラインとして表現することによるオペレーションの自動化についても検討を行う.

謝辞 本研究は慶應義塾大学と日立製作所との共同研究

のもと実施している. 多大なご支援に対して心より感謝いたします.

参考文献

[1] 文部科学省：コロナ対応の現状, 課題, 今後の方向性について，入手先 ⟨https://www.mext.go.jp/content/20200924-mxt_keikaku-000010097_3.pdf⟩ (2021.08.30).
[2] 慶應義塾 ITC：keio.jp マニュアル，入手先 ⟨https://www.itc.keio.ac.jp/ja/keiojp_manual.html⟩ (2021.08.30).
[3] 学認活用事例集（ケーススタディ）(2011/05/12 更新), 入手先 ⟨https://www.gakunin.jp/document/category/61⟩ (2021.11.01).
[4] 学認認証連携導入の諸問題 Shibboleth, 入手先 ⟨https://gakunin.jp/sites/default/files/2019-10/SSO_report_2009keio.pdf⟩ (2021.11.01).
[5] 学認 参加情報，入手先 ⟨https://www.gakunin.jp/join⟩ (2021.11.01).
[6] OWASP Top 10 Web Application Security Risks, 入手先 ⟨https://owasp.org/www-project-top-ten/⟩ (2021.11.01).
[7] Docker: Empowering App Development for Developers, 入手先 ⟨https://www.docker.com/⟩ (2021.11.01).
[8] Certified OpenID Connect Implementations, 入手先 ⟨https://openid.net/developers/certified/⟩ (2021.11.01).
[9] The problem with OAuth for Authentication., 入手先 ⟨http://www.thread-safe.com/2012/01/problem-with-oauth-for-authentication.html⟩ (2021.08.30).
[10] NIST Special Publication 800-63B, 入手先 ⟨https://openid-foundation-japan.github.io/800-63-3-final/sp800-63b.ja.html⟩ (2021.08.30).
[11] 平成 24 年 3 月内閣官房情報セキュリティセンター政府機関における情報システムのログ取得・管理の在り方の検討に係る調査報告書，入手先 ⟨https://www.nisc.go.jp/inquiry/pdf/log_shutoku.pdf⟩ (2021.11.01).