

メール送信時のトークンベース認証 (XOAUTH2) とその可能性



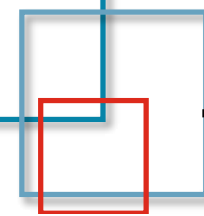
スピーカー紹介



加瀬 正樹（株式会社TwoFive）
2000年 ニフティ入社
2017年 TwoFive入社
2025年 JPAAWG プログラム委員



今村 侑輔（株式会社インターネットイニシアティブ）
2015年 IIJ 入社
2025年 JPAAWG プログラム委員



アジェンダ

- そもそもなぜ XOAUTH2 の話なのか、何を改善したいのか
- 過去の議論を振り返る
- 動作や実装
- メールサービス事業者 / 利用者がやるべきこと
- Open mic
 - 皆さんのご意見ください

お断り

- このセッションはスピーカー2名で仕様や動向などを調べながら作成した内容であり、必ずしも最新情報ではない場合もあります
- JPAAWG でもこのテーマについて今後議論を深めていくきっかけにしたいと思っていますので、忌憚のないご意見をいただきたいです
- 間違いや理解不足な点があればスピーカーに教えてください

なぜ XOAuth2 の話？

Gmail のメール送信方法が変更

- Google は2014年ごろから、メールソフト（MUA）からの接続方式として **XOAUTH2** を採用していて、SMTP/POP/IMAP で利用が可能
 - 安全性の低いアプリやクライアントからのアクセスを防ぐ目的
- 2025年1月以降、ユーザー名とパスワードのみでログインする安全性の低いアプリなどは**サポート対象外**

Gmail での XOAUTH2 の概説

- 1 Google Cloud Platform のプロジェクトを作成して、OAuth 同意画面を用意する



Gmail での XOAUTH2 の概説

2 OAuth クライアント ID を作成して（デスクトップアプリ） クライアント ID とクライアントシークレットを手元に記録しておく

Google Cloud google-api-proj1 スラッシュ (/) を使用してリソース、ドキュメント、...

Google Auth Platform / クライアント / クライアントの作成

概要
ブランディング
対象
クライアント
データアクセス
検証センター
設定

← OAuth クライアント ID の作成

クライアント ID は、Google の OAuth サーバーで個々のアプリを識別するために使用します。アプリが複数のプラットフォームで実行される場合、それぞれに独自のクライアント ID が必要になります。詳しくは、[OAuth 2.0 の設定](#) をご覧ください。OAuth クライアントの種類の[詳細](#)

アプリケーションの種類 *
デスクトップ アプリ

名前 *
デスクトップ クライアント: 2

OAuth 2.0 クライアントの名前。この名前はコンソールでクライアントを識別するためのみ使用され、エンドユーザーには表示されません。

注: 設定が有効になるまで 5 分から数時間かかることがあります

作成 キャンセル

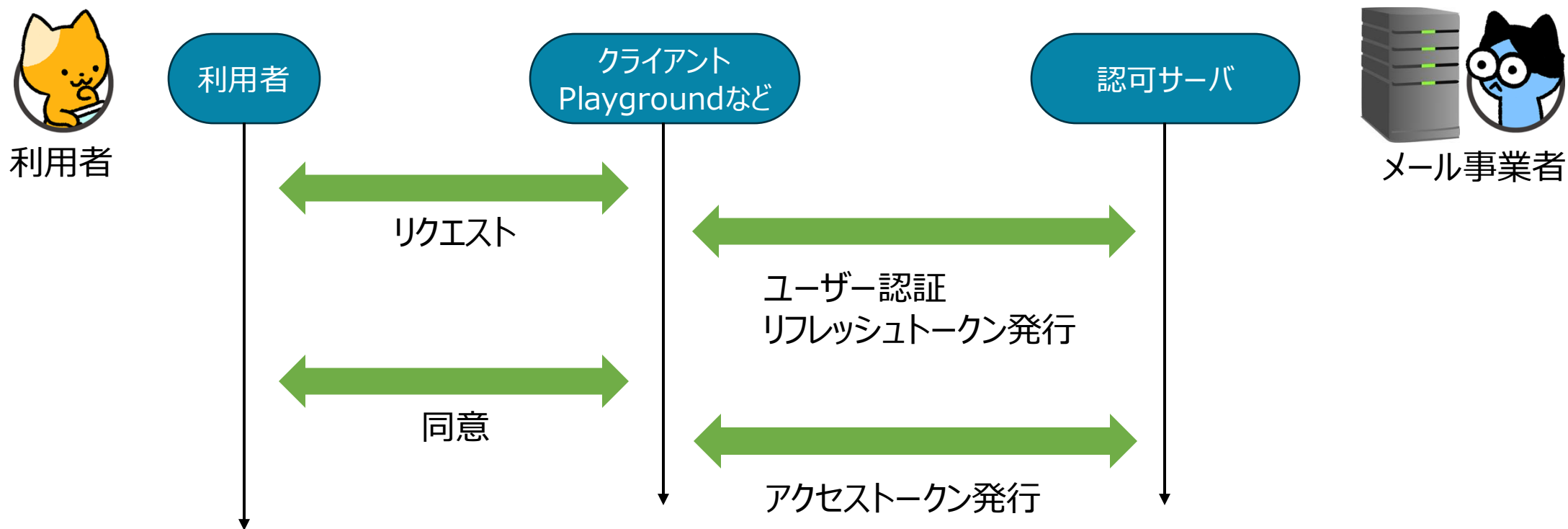
Additional information

クライアント ID	apps.googleusercontent.com
作成日	2025年9月11日 12:52:57 GMT+9
最終利用日	2025年9月10日 (Note: this data could be delayed by a day or more.)

i 6 か月間使用されていない OAuth クライアントは、削除の対象となります。非アクティブによる削除の通知が届きます。削除後 30 日間はクライアントを復元できます。 [Learn more](#)

Gmail での XOAUTH2 の概説

- 3 (OAuth Playground などを利用して) リフレッシュトークンを取得して、認可サーバからアクセストークンを取得させる



Gmail での XOAUTH2 の概説

- 4 SMTP 接続して、AUTH コマンドとして XOAUTH2 を指定して、
エンコードしたアクセストークンを送付して、メール送信の認可を得る

```
$ openssl s_client -connect smtp.gmail.com:587 -starttls smtp -crlf
. . .
250 SMTPUTF8
ehlo XXXXXXXXXXXX
250-smtp.gmail.com at your service, [2401:2500:102:3007:XXX:XXX:XXX:XXX]
250-SIZE 35882577
250-8BITMIME
250-AUTH LOGIN PLAIN XOAUTH2 PLAIN-CLIENTTOKEN OAUTHBEARER XOAUTH
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8
AUTH XOAUTH2 dXNlcj1rYXNIQHNvZnRlc3Q . . . 5TN2F2Sy1lbmpRMDIwNgEB
235 2.7.0 Accepted
```

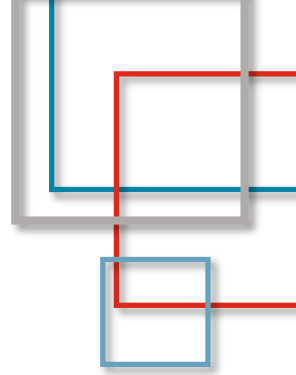
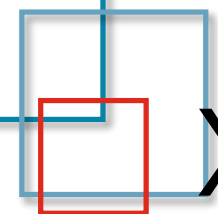
Gmail のメール送信方法が変更

- Google は2014年ごろから、メールソフト（MUA）からの接続方式として **XOAUTH2** を採用していて、SMTP/POP/IMAP で利用が可能
 - 安全性の低いアプリやクライアントからのアクセスを防ぐ目的
- 2025年1月以降、ユーザー名とパスワードのみでログインする安全性の低いアプリなどは**サポート対象外**
- Gmail を利用する場合は、**Webメールか専用アプリが選択肢**

[参考] XOAUTH2 vs OAUTHBEARER

	XOAUTH2	OAUTHBEARER
ドキュメント	Google Developers ドキュメント	RFC 7628
データ形式の特徴	user=xxxxx<¥a> auth=xxxxx<¥a><¥a>	n,user=xxxxx,<¥a> host=xxxxx,<¥a> port=xxxxx,<¥a>...
普及状況	✓	
標準化		✓
Google サポート	✓	
Microsoft サポート	✓	✓ ※
Dovecot		✓
Postfix		✓

※ <https://officeprotocoldoc.z19.web.core.windows.net/files/MS-XOAUTH/%5bMS-XOAUTH%5d-220215.pdf>



XOAUTH2 が求められる理由とは？

- 今村さんの見解
- 加瀬の見解



過去の議論

JPAAWG 3rd ORT での議論(2020年)

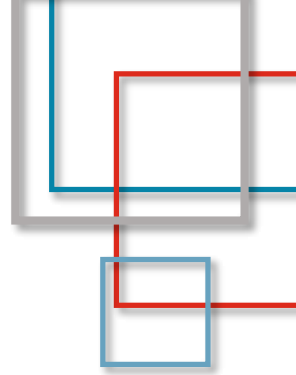
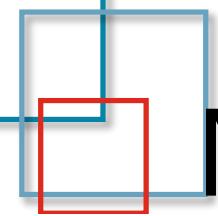
- メール送受信におけるパスワードレス認証をテーマにした議論
- 利便性と安全性のバランス (fool-proof, abuse-proof)
- 認証方式としての代替手段: FIDO, 生体認証, OAuth
- 送信方式としての代替手段: メールアプリ
- 緩和方法: パスワード認証へのペナルティ強化
- JPAAWG で継続して検討するテーマにする

11:00-12:00 A2-RT2

パスワードレス認証の進展と普及の課題について

Open Round Table セッションは、一つのテーマについて参加者が主体的に議論に参加し、セッションオーナーが議論内容をまとめます。このテーブルでは、メールを含む ID 認証としてのパスワードレス認証の課題について議論します。

<https://meetings.jpaaawg.org/3rd2020/program/>



M3AAWG での議論

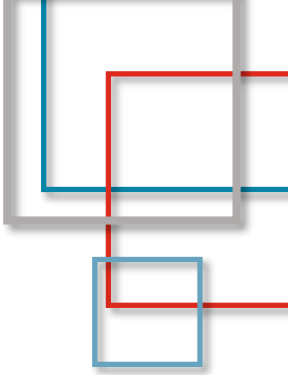
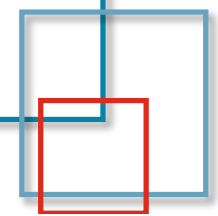
- 2023 Dublin: Modern User Authentication for Email



<https://www.m3aawg.org/blog/ModernUserAuthenticationEmail>



実際にやってみた



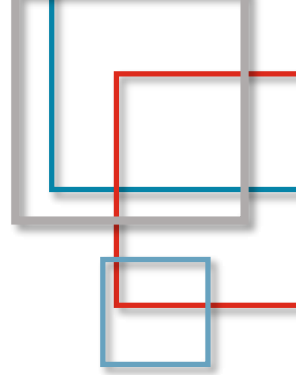
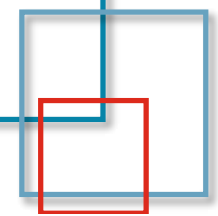
XOAUTH2 実装してみた

構成情報:

OS: RockyLinux9.6(Blue Onyx)

MTA: Postfix 3.5.25

POP/IMAP, 認証: Dovecot CE 2.3.16



XOAUTH2 実装してみた

やったこと:

- postfix, dovecot のインストール (dnf install)
- keycloak のインストール
- dovecot config 追加
- postfix config 追加
- Keycloak の設定

XOAUTH2 実装してみた

dovecot config

- 既存の認証方式に OAUTHBEARER と XOAUTH2 を追加
- args に指定する OAUTH2 用の IDP設定は使うサービスによって指定が変わります

```
auth_mechanisms = $auth_mechanisms oauthbearer xoauth2

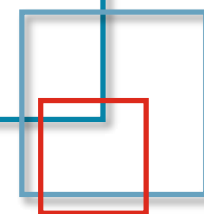
passdb {
    driver = oauth2
    mechanisms = xoauth2 oauthbearer
    args = /etc/dovecot/dovecot-oauth2.conf.ext
}
```

XOAUTH2 実装してみた

dovecot config

- postfix 用に dovecot の認証を利用するための socket を追加

```
auth_mechanisms = plain login
passdb {
  driver = pam
}
userdb {
  driver = passwd
}
service auth {
  unix_listener /var/spool/postfix/private/auth {
    mode = 0660
    user = postfix
    group = postfix
  }
}
```

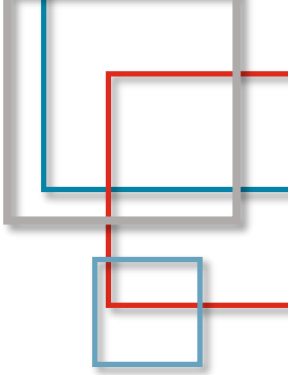
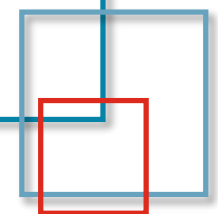


XOAUTH2 実装してみた

postfix config

- postfix の認証に dovecot 側で用意したソケットを使うように設定

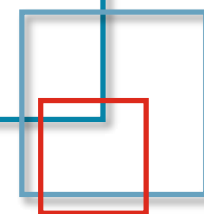
```
smtpd_relay_restrictions = permit_sasl_authenticated, reject
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
```



XOAUTH2 実装してみた

dovecot に対して localhost から imap 接続してみた結果

```
[root@default ~]# telnet 0 143
Trying 0.0.0.0...
Connected to 0.
Escape character is '^'.
* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE
LITERAL+ STARTTLS AUTH=PLAIN AUTH=OAUTHBEARER AUTH=XOAUTH2]
Dovecot ready.
```

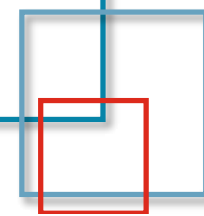


XOAUTH2 実装してみた

postfix に対して localhost から smtp 接続してみた結果

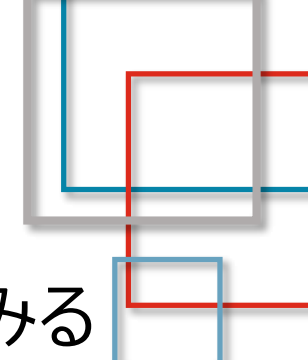
- smtp

```
[root@default ~]# telnet 0 25
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.
220 hoge.hoge.local ESMTP Postfix
ehlo hoge
250-hoge.hoge.local
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-AUTH PLAIN LOGIN OAUTHBEARER XOAUTH2
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-DSN
250-SMTPUTF8
250 CHUNKING
```

XOAUTH2 実装してみた

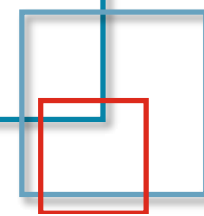
dovecot が問い合わせする IDP は....今回は keycloak を使ってみる



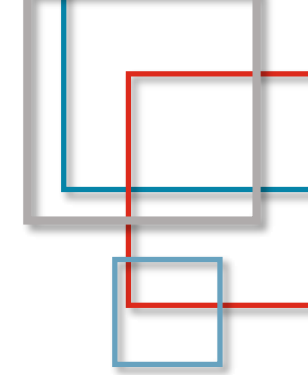
dovecot の設定

/etc/dovecot/dovecot-oauth2.conf.ext

```
introspection_url = http://localhost:8080/realms/mail/protocol/openid-connect/token/introspect/  
introspection_mode = post  
client_id = dovecot  
client_secret = ${client_secret}  
username_attribute = username  
#tls_ca_cert_file = /etc/ssl/certs/ca-certificates.crt  
active_attribute = active  
active_value = true
```

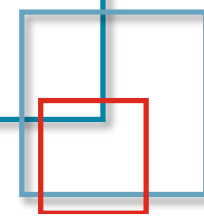


XOAUTH2 実装してみた



keycloak の設定

- openjdk (java) のインストール
- keycloak のインストール
 - keycloak 起動
 - 起動の前に bootstrap user / pass を要指定
 - # ./bin/kc.sh start --hostname http://\${host}:\${port}/auth --**http-enabled true** で横着できる
 - realm 作成
 - client 作成
 - client secret を控えておく
 - user 作成
 - role 作成



XOAUTH2 実装してみた

実際に XOAUTH2 で接続してみる

- アクセストークンを取得

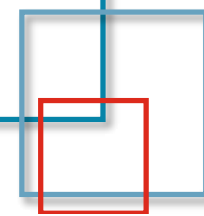
```
curl --request POST -s
--url http://localhost:8080/realms/mail/protocol/openid-connect/token ¥
--header 'content-type: application/x-www-form-urlencoded' ¥
--data client_id=dovecot ¥
--data client_secret=${client_secret} ¥
--data grant_type=password ¥
--data username=${username} ¥
--data password=${password}
```

XOAUTH2 実装してみた

実際に XOAUTH2 で接続してみる

- 取得したアクセストークンをエンコードして XOAUTH2 アクセス

```
[root@default ~]# telnet 0 143
Trying 0.0.0.0...
Connected to 0.
Escape character is '^'.
* OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+
STARTTLS AUTH=PLAIN AUTH=OAUTHBEARER AUTH=XOAUTH2] Dovecot ready.
01 authenticate xoauth2 dXNlc.....
01 OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE SORT
SORT=DISPLAY THREAD=REFERENCES THREAD=REFS THREAD=ORDEREDSUBJECT
MULTIAPPEND URL-PARTIAL CATENATE UNSELECT CHILDREN NAMESPACE UIDPLUS LIST-
EXTENDED I18NLEVEL=1 CONDSTORE QRESYNC ESEARCH ESORT SEARCHRES WITHIN
CONTEXT=SEARCH LIST-STATUS BINARY MOVE SNIPPET=FUZZY PREVIEW=FUZZY PREVIEW
STATUS=SIZE SAVEDATE LITERAL+ NOTIFY SPECIAL-USE] Logged in
```



XOAUTH2 実装してみた

実際に XOAUTH2 で接続してみる

- 取得したアクセストークンをエンコードして XOAUTH2 アクセス

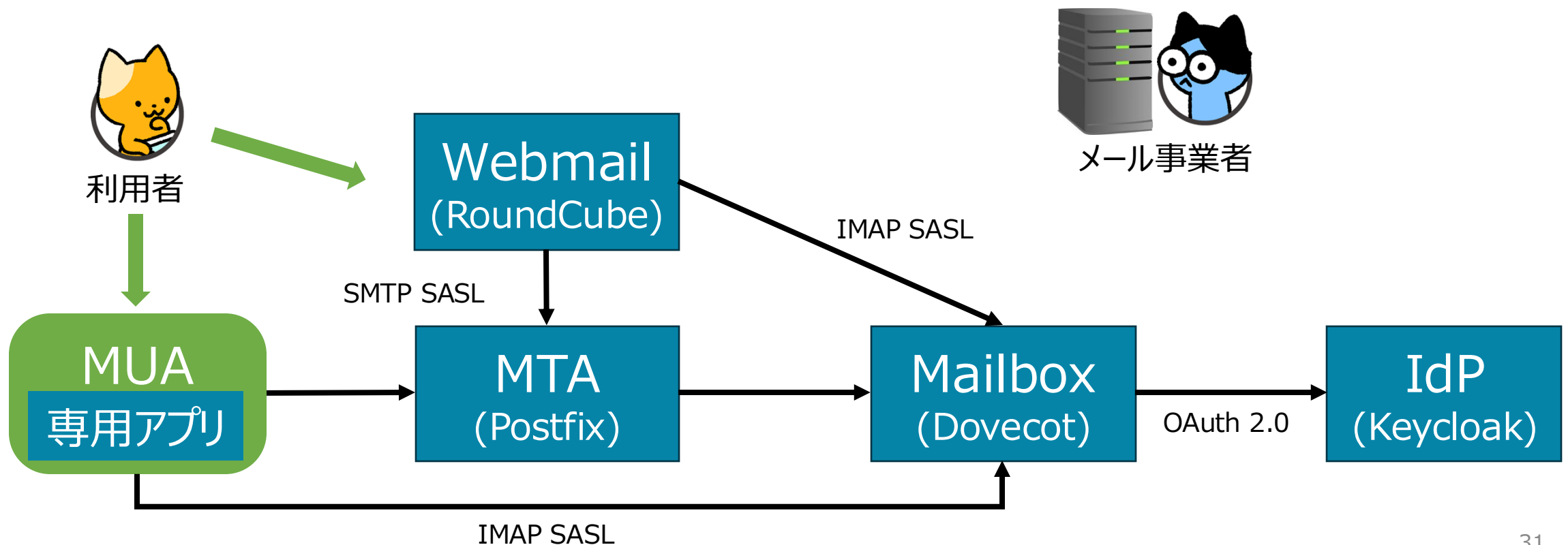
```
[root@default ~]# telnet 0 25
Trying 0.0.0.0...
Connected to 0.
Escape character is '^]'.
220 default.local ESMTP Postfix
ehlo hoge
250-default.local
250-PIPELINING
250-SIZE 10240000
250-VERFY
250-ETRN
250-STARTTLS
250-AUTH PLAIN OAUTHBEARER XOAUTH2
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-DSN
250-SMTPUTF8
250 CHUNKING
auth xoauth2
334
DXNlc.....
235 2.7.0 Authentication successful
```



対応するためには

メールサービス事業者としてどうしたらいい？

- XOAUTH2 または OAUTHBEARER を実装する
- Webメールを提供する
- 専用メールアプリを提供する



メール利用者としてどうしたらいい？

- XOAUTH2 に対応したメールソフトで Gmail を利用する
- Webメールを利用する
- 専用メールアプリを利用する
- クライアント証明書を利用する

Unfortunately, most mainstream mail clients, including Outlook, Thunderbird, and Apple Mail, do not natively support the `OAUTHBEARER` / `XOAUTH2` SASL mechanisms with third-party OAuth providers like Stalwart. As a result, even though OAuth is a secure and modern authentication method, it cannot be directly used with these clients to authenticate with Stalwart through standard mail protocols like IMAP or SMTP.

<https://stalw.art/docs/auth/oauth/interoperability/>



Open mic

Thank you

