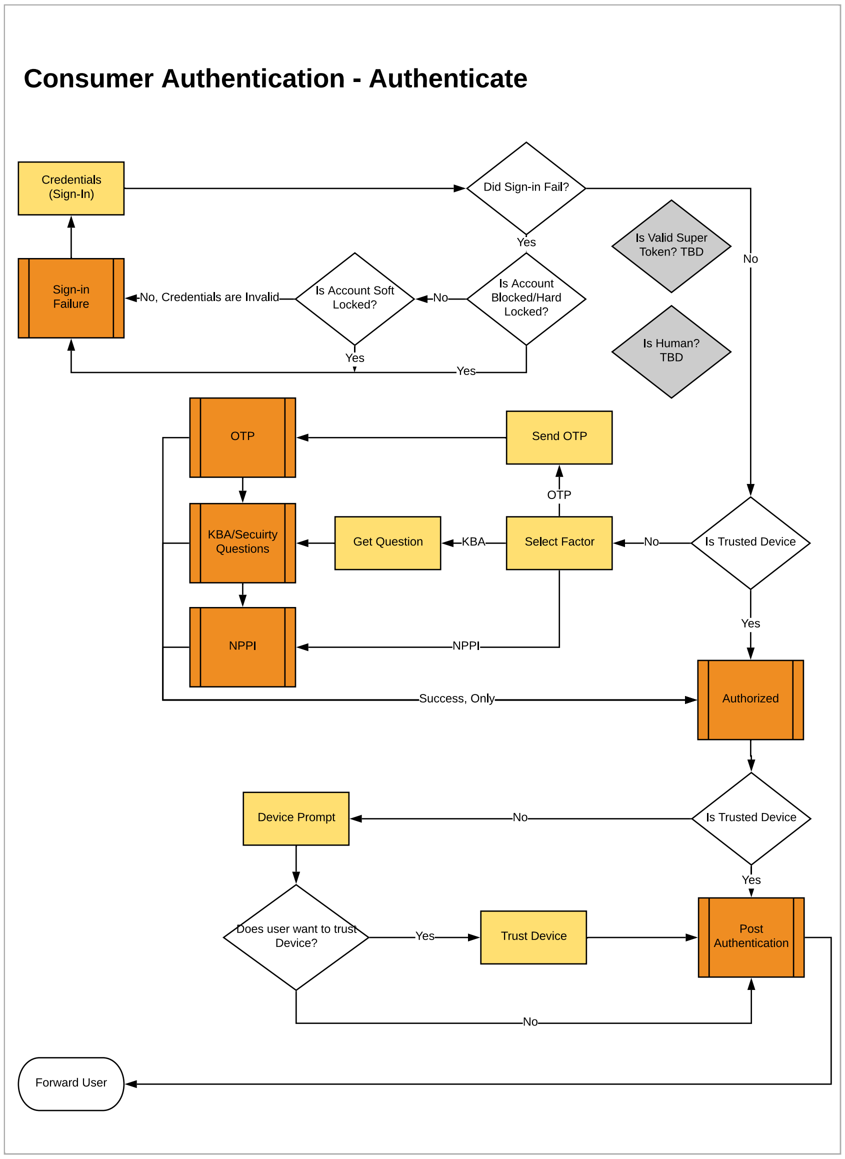
Authenticate - Consumer Authentication API V1

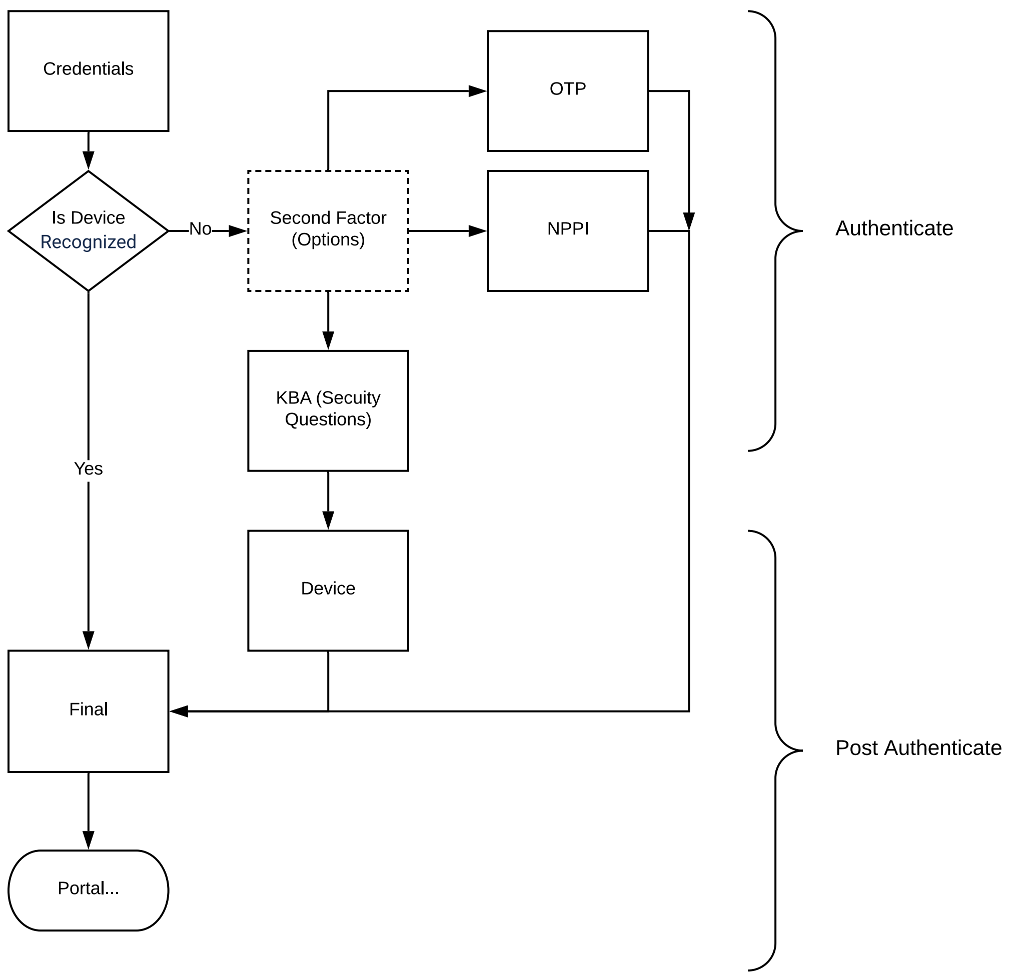
Created by Jason Gray, last modified by Suresh Chinta on Mar 21, 2020

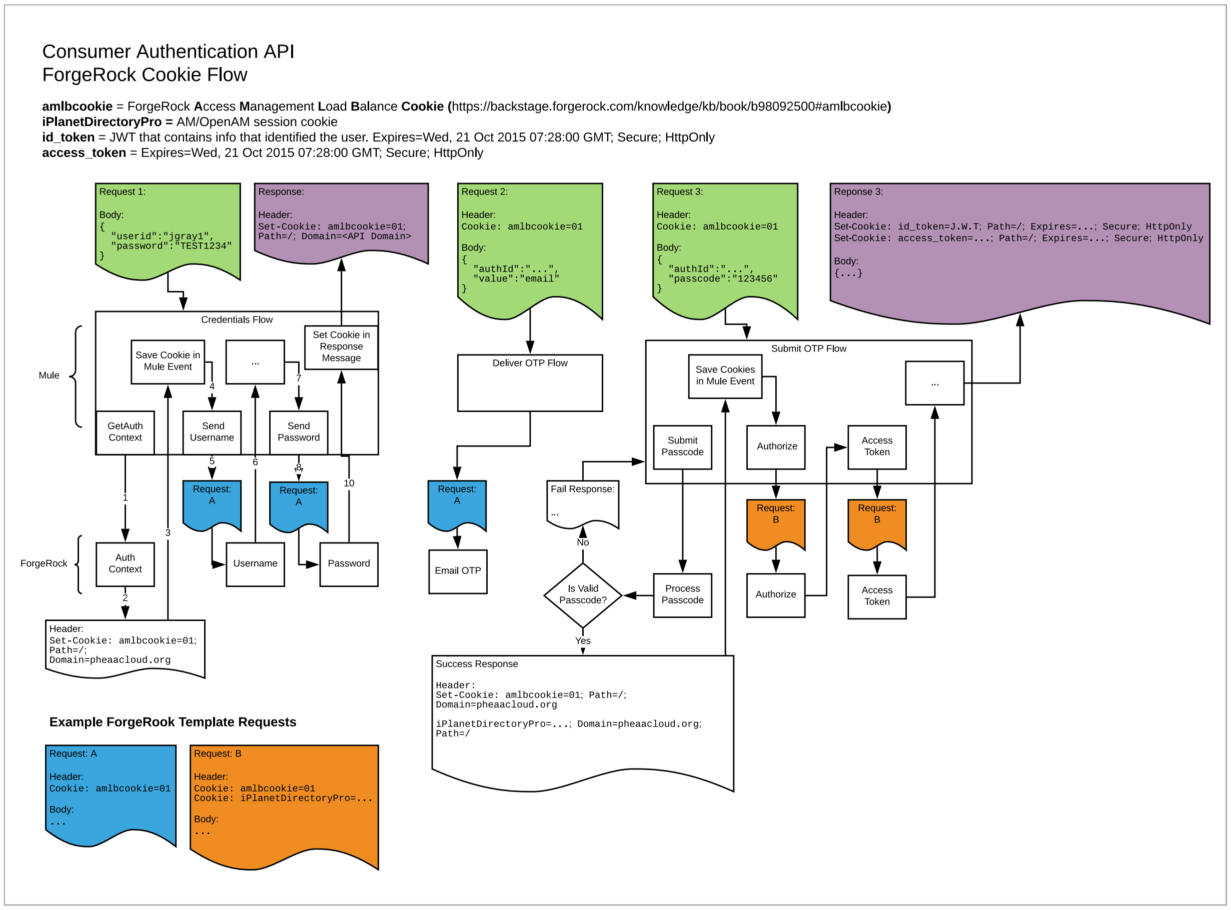
- Authenticate
  - POST /authenticate
    - Request Bodies for each Stage
      - Credentials
        - Minimum Request
        - Full Request
        - Attributes
      - Second Factor Selection
        - Select OTP Path/Deliver OTP
        - Attributes
        - Second Factor Request for Aggregator
        - Re-send OTP (Can be to the same or different contact in user's profile)
      - Submit OTP
      - Select KBA and Retrieve Question
      - Select NPPI Path and Submit NPPI
      - Resubmit NPPI
      - Submit KBA Answer
      - Submit Device Recognition Choice
      - Invalid Credentials Response
      - Not Human Response
      - Account Soft/Temporary Locked Response
      - Account Blocked Hard/Indefinitely Locked Response
      - Second Factor Options Response
      - Successful OTP Delivery Response
      - Maximum OTP (Re)send Threshold Response - User with KBA
      - Maximum OTP (Re)send Threshold Response - User without KBA
      - OTP Verify Failure - Invalid OTP Response
      - OTP Verify Failure - OTP Valid but Expired Response
      - OTP Verify Failure Limit Exceeded - Transition to KBA (if user has KBA)
      - OTP Verify Failure Limit Exceeded - Transition to NPPI (if user doesn't have KBA)
      - OTP Delivery Failure Response
      - Invalid Super Token
      - KBA Question
      - NPPI - When User Doesn't have KBA / Verified Contacts
      - KBA Fail and Return Next Question Response
      - KBA Fail and Transition to NPPI Response
      - Invalid NPPI (Non-Public Personal Information) Response
      - Device Prompt Response
      - Final Response

Authenticate


Consumer Authentication - Authenticate
Credentials (Sign-In)
Did Sign-in Fail?
Yes
No
Is Valid Super Token? TBD
Is Human? TBD
Sign-in Failure
No, Credentials are Invalid
Is Account Soft Locked?
No
Is Account Blocked/Hard Locked?
Yes
Yes
OTP
Send OTP
OTP
KBA/Secuirty Questions
Get Question
KBA
Select Factor
No
Is Trusted Device
NPPI
NPPI
Yes
Success, Only
Authorized
Device Prompt
No
Is Trusted Device
Yes
Does user want to trust Device?
Yes
Trust Device
Post Authentication
No
Forward User

Consumer Authentication - Authenticate Stages




Consumer Authentication API
ForgeRock Cookie Flow

amlbcookie = ForgeRock **A**ccess **M**anagement **L**oad **Bal**ance **Cookie (**https://backstage.forgerock.com/knowledge/kb/book/b98092500#amlbcookie**)**
iPlanetDirectoryPro = AM/OpenAM session cookie
id_token = JWT that contains info that identified the user. Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
access_token = Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly

```
Request 1:

Body:
{
  "userid":"jgray1",
  "password":"TEST1234"
}
```

```
Response:

Header:
Set-Cookie: amlbcookie=01;
Path=/; Domain=<API Domain>
```

```
Request 2:

Header:
Cookie: amlbcookie=01

Body:
{
  "authId":"...",
  "value":"email"
}
```

```
Request 3:

Header:
Cookie: amlbcookie=01

Body:
{
  "authId":"...",
  "passcode":"123456"
}
```

```
Reponse 3:

Header:
Set-Cookie: id_token=J.W.T; Path=/; Expires=...; Secure; HttpOnly
Set-Cookie: access_token=...; Path=/; Expires=...; Secure; HttpOnly

Body:
{...}
```

```
Header:
Set-Cookie: amlbcookie=01;
Path=/;
Domain=pheaacloud.org
```

```
Success Response

Header:
Set-Cookie: amlbcookie=01; Path=/;
Domain=pheaacloud.org

iPlanetDirectoryPro=...; Domain=pheaacloud.org;
Path=/
```

```
Fail Response:

...
```

Example ForgeRook Template Requests

```
Request: A

Header:
Cookie: amlbcookie=01

Body:
...
```

```
Request: B

Header:
Cookie: amlbcookie=01
Cookie: iPlanetDirectoryPro=...

Body:
...
```

POST /authenticate

Request Bodies for each Stage

Credentials

Minimum Request

```
{
    "userId": "myUser1",
    "password": "myPassw0rd"
}
```

Full Request

```
{
    "userId": "myUser1",
    "password": "myPassw0rd",
    "token": {
        "type": "recaptchav3",
        "value": "some-token"
    },
    "devicePrint": ""
}
```

Attributes

- **userId** = Consumers User ID/Username
- **password** = Consumers Password
- **token (optional) =** A multipurpose field used for different types of tokens. Can help identify what "type" of additional authentication is needed.
  - **reCAPTCHAv3** Token. This will send a backend request to the reCAPTCHA processor for a score.
  - **Super** Token. Use to bypass certain functionality,
    - Data Aggregator, such as Mint, that we have an agreement with to bypass **reCAPTCHAv3** and **OTP** if valid Token and IP range match.

- Mobile App, to bypass **reCAPTCHAv3.**
- **devicePrint (optional) =** A device fingerprint object used when device is remembers.
- **channel** Medium in which API will be used. (web or mobile)

Second Factor Selection

Select OTP Path/Deliver OTP

```
{
  "authId": "${authId}",
  "secondFactor":"otp",
  "delivery": {
    "contactId":"1",
    "channel":"email"
  }
}
```

Attributes

- **authId** = Needed to chain all Auth requests together
- **secondFactor =** Needed to select the OTP path and trigger OTP Delivery
- **otpDeliveryInfo =** Information needed to deliver an OTP

Second Factor Request for Aggregator

- combination of `secondFactor: aggregator` and `superToken` will bypass user second factor and provide a session token with role as `aggregator`
- Aggregator second factor option is not published for consumer authentication.

```
{
    "authId": "{{authId}}",
    "secondFactor": "aggregator",
    "superToken": "{{guid}}"
}
```

Re-send OTP (Can be to the same or different contact in user's profile)

- Re-send of OTP can be made as the user chooses to - no matter whether the system says OTP delivery is successful or not.
- Total maximum OTPs allowed to be requested by a borrower in a given auth interaction are 3 (three). This count includes the initial OTP request

```
{
  "authId": "${authId}",
  "secondFactor":"otp",
  "resendOtp": true,
  "delivery": {
    "contactId":"1",
    "channel":"email"
  }
}
```

Submit OTP

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "passcode": "123456"
}
```

- Attributes
- **authId** = Needed to chain all Auth requests together
- **passcode**= Consumers one-time passcode

Select KBA and Retrieve Question

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "secondFactor":"kba"
}
```

- Attributes
- **authId** = Needed to chain all Auth requests together
- **secondFactor =** Needed to select the KBA path so a security can be returned.

Select NPPI Path and Submit NPPI

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "secondFactor":"nppi",
  "nppi": {
    "accountIdentifier": "1234567890",
    "lastName": "gray",
    "birthDate": "1980-12-25"
  }
}
```

- Attributes
- **authId** = Needed to chain all Auth requests together
- **secondFactor =** Needed to select the NPPI path
- **nppi** = Non-Public Personal Information (NPPI) is the information use to identified the user.

Resubmit NPPI

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "nppi": {
    "accountIdentifier": "1234567890",
    "lastName": "gray",
    "birthDate": "1980-12-24"
  }
}
```

- Only use when preceded by an NPPI match failure. The Second Factor needs to be selected on a previous request.
- Attributes
- **authId** = Needed to chain all Auth requests together
- **nppi** = Non-Public Personal Information (NPPI) is the information use to identified the user.
  - **accountIdentifier** = Account holder's **SSN** (9 Digit) or **Account Number** (10 Digit)
  - **lastName** = Account holders' **Last Name**
  - **birthDate** = Account holder's **Date of Birth**

Submit KBA Answer

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "kbaAnswer": "cheeseburger"
}
```

- Attributes
- **authId** = Needed to chain all Auth requests together
- **kbaAnswer** = Answer to the security question
- **destinationInfo (optional) =** Used for deep linking. A white listed code that maps to an endpoint for redirection after authentication. Note, white listed map could be stored in AEM as endpoint for map should be made configurable.

Submit Device Recognition Choice

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "saveDevice": true
}
```

- Attributes
- **authId** = Needed to chain all Auth requests together
- **saveDevice** = Boolean value on whether or not to save the device.
- **Common Response Bodies**

Invalid Credentials **Response**

```
{
    "status": "fail",
    "stage": "credentials",
    "failureReasons": [
        {
            "code": "icr",
            "description": "Invalid Credentials"
        }
    ]
}
```

Not Human Response

```
{
    "status": "fail",
    "stage": "credentials",
    "failureReasons": [
        {
            "code": "bot",
            "description": "Not Human"
        }
    ]
}
```

Account Soft/Temporary Locked Response

```
{
    "status": "fail",
    "stage": "credentials",
    "failureReasons": [
        {
            "code": "slk",
            "description": "Soft/Temporary Locked",
            "secondsRemaining": "90"
        }
    ]
}
```

Account Blocked Hard/Indefinitely Locked Response

```
{
    "status": "fail",
    "stage": "credentials",
    "failureReasons": [
        {
            "code": "hlk",
            "description": "Blocked Hard/Indefinitely Locked"
        }
    ]
}
```

Second Factor Options Response

```
{
    "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIs
    "status": "success",
    "stage": "second-factor",
    "otpInfo": {
        "contactOptions": [
            {
                "id": "f687d6e785a468a55e96eeae477a5011",
                "channels": [ "email" ],
                "description": "a*****@example.com"
            },
            {
                "id": "3ff16d0e461a2b68e4e1a255746a8665",
                "channels": [ "email" ],
                "description": "b*****@example.org"
            },
            {
                "id": "d05b743af75101806eab3bd4fffc56d4",
                "channels": [ "sms", "voice" ],
                "description": "(***) ****-1234"
            },
            {
                "id": "d05b743af75101806eab3bd4fffc56d4",
                "channels": [ "voice" ],
                "description": "(***) ****-1234"
            }
        ]
    },
    "hasKBA": true,
    "hasNPPI": true,
    "lastLoginOn": "2019-08-19T15:10:16Z"
}
```

Successful OTP Delivery Response

```
{
    "authId": "{{authId}}",
    "status": "success",
    "stage": "otp",
    "delivery": {
        "channel": "email",
        "contactId": "L5uqAQoXUTvTHLWcOAJ3wUYs22"
    }
}
```

Maximum OTP (Re)send Threshold Response - User with KBA

```
{
    "authId": "${authId}",
    "status": "fail",
    "stage": "kba",
    "kbaQuestion": "What was the name of your High School?",
    "failureReasons": [
        {
            "code": "mro",
            "description": "Maximum OTP send threshold reached"
        }
    ]
}
```

Maximum OTP (Re)send Threshold Response - User without KBA

```
{
    "authId": "${authId}",
    "status": "fail",
    "stage": "nppi",
    "failureReasons": [
        {
            "code": "mro",
            "description": "Maximum OTP send threshold reached"
        }
    ]
}
```

OTP Verify Failure - Invalid OTP Response

```
{
    "authId": "${authId}",
    "status": "fail",
    "stage": "otp",
    "failureReasons": [
        {
            "code": "iop",
            "description": "Invalid One Time Passcode"
        }
    ]
}
```

OTP Verify Failure - OTP Valid but Expired Response

```
{
    "authId": "${authId}",
    "status": "fail",
    "stage": "otp",
    "failureReasons": [
        {
            "code": "eop",
            "description": "Expired One Time Passcode"
        }
    ]
}
```

OTP Verify Failure Limit Exceeded - Transition to KBA (if user has KBA)

```
{
    "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIs
    "status": "fail",
    "stage": "kba",
    "kbaQuestion": "What was the name of your High School?",
    "failureReasons": [
        {
            "code": "ole",
            "description": "One Time Passcode Limit Exceeded"
        }
    ]
}
```

OTP Verify Failure Limit Exceeded - Transition to NPPI (if user doesn't have KBA)

```
{
    "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIs
    "status": "fail",
    "stage": "nppi",
    "failureReasons": [
        {
            "code": "ole",
            "description": "One Time Passcode Limit Exceeded"
        }
    ]
}
```

OTP Delivery Failure Response

```
{
    "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIs
    "status": "fail",
    "stage": "otp",
    "failureReasons": [
        {
            "code": "fod",
            "description": "One Time Passcode Delivery Failure"
        }
    ]
}
```

Invalid Super Token

```
{
    "status": "fail",
    "stage": "aggregator",
    "failureReasons": [
        {
            "code": "ist",
            "description": "Invalid Super Token"
        }
    ]
}
```

KBA Question

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "status": "success",
  "stage": "kba",
  "kbaQuestion": "What is your favorite food?"
}
```

NPPI - When User Doesn't have KBA / Verified Contacts

```
{
  "authId": "${authId}",
  "status": "success",
  "stage": "nppi"
}
```

KBA Fail and Return Next Question Response

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "status": "fail",
  "stage": "kba",
  "kbaQuestion": "What is your favorite food?",
  "failureReasons": [
    {
      "code": "ian",
      "description": "Invalid KB Answer"
    }
  ]
}
```

KBA Fail and Transition to NPPI Response

```
{
    "authId": ${authId},
    "status": "fail",
    "stage": "nppi",
    "failureReasons": [{
        "code": "ian",
        "description": "Invalid KB Answer"
    }]
}
```

Invalid NPPI (Non-Public Personal Information) Response

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "status": "fail",
  "stage": "nppi",
  "failureReasons": [
    {
      "code": "inp",
      "description": "Invalid NPPI"
    }
  ]
}
```

Device Prompt Response

```
{
  "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im90cENNoYWluIiwib3RrIjoidnA2ZnBuZmJsdHVkdHFscG00NGtjaTBjYSIsIm
  "status": "success",
  "stage": "device"
}
```

Final Response

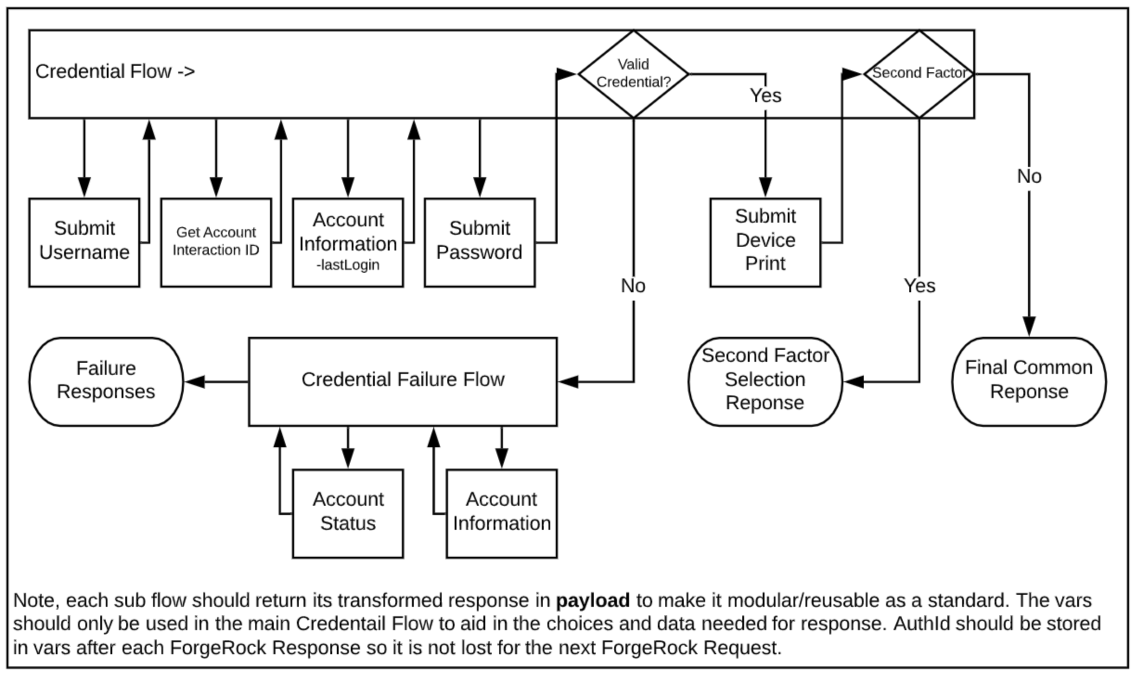
header

```
Set-Cookie: access_token=...; Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
Set-Cookie: id_token=...; Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
Set-Cookie: refresh_token=...; Expires=Wed, 21 Oct 2015 07:28:00 GMT; Secure; HttpOnly
```

body

```
{
  "status": "success",
  "stage": "final",
  "userId": "jgray1",
  "now": "2019-08-19T15:10:16Z",
  "accessExpiresIn": 599,
  "lastLoginOn": "2019-08-01T15:10:16Z",
  "aii": "0",
  "roles": [
    "Consumer"
  ]
}
```

- Properties
- <property>: <example-value>
- **realm**: "dev1-aes"
  authIndexType: "service"
  authIndexValue: "myMfaTree"
  authURL: "https://devforgeaes.pheaacloud.org/auth/json/realms/root/realms/${realm}/authenticate?authIndexType=${authIndexType}&authIndexValue=${authIndexValue}"
  authorizeURL "https://devforgeaes.pheaacloud.org/auth/oauth2/realms/root/realms/${realm}/authorize"
  accessTokenURL: "https://devforgeaes.pheaacloud.org/auth/oauth2/realms/root/realms/${realm}/access_token"
  amAdminUsername: "delegated-am-admin"
  amAdminPassword: "TEST1234"
  amAdminAuthURL: "https://devforgeaes.pheaacloud.org/auth/json/realms/root/realms/${realm}/authenticate"
  amAdminUserInfoURL: "https://devforgeaes.pheaacloud.org/auth/json/realms/root/realms/${realm}/users/{{username}}?_fields=username,pwdAccountLockedTime,pwdFailureTime,inetUserStatus,mail,kbaUser,lastLoginTime,personReferenceId"
  softlockExpirationSeconds: 600
  useKBA: true
  useNPPI: true
- Flows
- IF the **userId** exists and **password** exist in the **Request Payload**
- THEN Continue to **Credentials Stage**
- ELSE
- Some other flow
- **Credentials Stage**



Note, each sub flow should return its transformed response in **payload** to make it modular/reusable as a standard. The vars should only be used in the main Credentail Flow to aid in the choices and data needed for response. AuthId should be stored in vars after each ForgeRock Response so it is not lost for the next ForgeRock Request.

-
- SET **credentials** = Request Payload (See **Request Bodies for each Stage)**
- Use the following information to communicate with **ForgeRock**
- **1. Create Authentication Context**
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

- **Body**
- None/Empty
- **Response from ForgeRock**

Header

```
Set-Cookie amlbcookie=01; Path=/; Domain=pheaacloud.org
```

Body

```
{
    "authId": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhdXRoSW5kZXhWYWx1ZSI6Im15TWZhVHJlZSIsIm90ayI6ImVoMnZscnM4MGIyNDZ0bm01MWEyZW5yYzU2
    "callbacks": [
        {
            "type": "NameCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "User Name"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": ""
                }
            ]
        }
    ]
}
```

- **Response**
- Save the **authId**
- Save the **amlbcookie.** This is found in the **Authentication Context Response** header as it will need to be sent along with all future requests in this flow.
- IF **callbacks** exists in the **Response** and **callbacks[0].type = "NameCallback"**
- THEN continue on to **Step 2**
- ELSE return the **Invalid Credentials Response** as defined in the **Common Response Bodies**
- **2. Submit Username**
- Build the next Request to ForgeRock by setting the **callbacks[0].input[0].value = credentials.userId.** This field should be in the **callback** object where type = "**NameCallback**" and input[0].name = "**IDToken1**". See [Username Here] below.
- **Request to ForgeRock**
- Endpoint: POST**${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "NameCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "User Name"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": "[Username Here]"
                }
            ]
        }
    ]
}
```

- **Response**
- Save the **authId**
- IF **callbacks** exists in the **Response** and **callbacks[0].type = "PasswordCallback"**
- THEN continue on to **Step 3**
- ELSE return the **Invalid Credentials Response** as defined in the **Common Response Bodies**
- **3. Submit Metadata**
- Save **aii (Account Interaction Id)** default to 1. **Note, we will add a service call here in a future Sprint.** TODO.
- Save **channel** as "web".
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "${authId}",
    "callbacks": [
        {
            "type": "TextInputCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "authMetadata"
                },
                {
                    "name": "defaultText",
                    "value": "{\"aii\":\"some value\",\"channel\":\"alexa | web | mobile-web | mobile-app\"}"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": "{\"aii\":\"${aii}\",\"channel\":\"${channel}\"}"
                }
            ]
        }
    ]
}
```

- **Response**
- Save the **authId**
- **4. Get Last Login Date Time**
- Retrieve Last Sign-in using Account Info Flow
- Save **lastLoginTime** as **lastLoginOn** from **Account Info Response**. If **lastLoginTime** exists read **lastLoginTime[0] else** default to null. This needs to be saved before the password request because if the password is successful the **lastLoginTime** will be updated to the current login.
- Continue on to **Step 4**
- **5. Submit Password**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the **callbacks[0].input[0].value = credentials.password.** This field should be in the **callback** object where type = "**PasswordCallback**" and input[0].name = "**IDToken1**". See [Password Here] below.
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "PasswordCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "Password"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": "[Password Here]"
                }
            ]
        }
    ]
}
```

- IF **callbacks** exists in the **Response** and **callbacks[0].type = "TextOutputCallback"**
- THEN continue on to the **Submit Device Print**
- ELSE IF **callbacks** exists in the **Response** and **callbacks[0].type = "NameCallback"**
- THEN follow **Credentials Failure Flow**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- **5. Submit Device Print**
- SET **authId** = payload.**authId**
- IF **credentials.devicePrint** does not exist, SET **credentials.devicePrint = "{}"**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[1].type = "HiddenValueCallback"
    - callbacks[1].input[0].name = "IDToken2"
    - callbacks[1].input[0].value = **credentials.devicePrint**
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [{
        "type": "TextOutputCallback",
        "output": [{
            "name": "message",
            "value": ""
        }, {
            "name": "messageType",
            "value": "4"
        }]
    }, {
        "type": "HiddenValueCallback",
        "output": [{
            "name": "value",
            "value": ""
        }, {
            "name": "id",
            "value": "devicePrint"
        }],
        "input": [{
            "name": "IDToken2",
            "value": "[Device Print Here]"
        }]
    }]
}
```

- **Response**
- Save the **authId**
- IF **callbacks** exists in the **Response** and **callbacks[0].type = "ChoiceCallback"**
- THEN continue on to the **Second Factor Stage**
- ELSE IF **callbacks** exists in the **Response** and **callbacks[0].type = "NameCallback"**
- THEN follow **Credentials Failure Flow**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- **Credentials Failure Flow**
- To determine Account Status a separate call to AM ForgeRock is needed this is done by using the **Account Info Flow**
- Using the **Response Payload** from the **Account Info Flow**
- IF **pwdAccountLockedTime?** and **sizeOf(pwdAccountLockedTime)** == 1 in **Response**
- THEN set **softLockDateTime** = **pwdAccountLockedTime[0] as DateTime**

Mule Hint
payload.**pwdAccountLockedTime[0]** is a string and will need to be convert to a DateTime so a DateTime comparison can be made. To do this in MuleSoft DataWeave please see the following.

```
payload.pwdAccountLockedTime[0] as LocalDateTime {format: "yyyyMMddHHmmss.SSS'Z'"} as DateTime
```

- **softLockExpirationDateTime = softLockDateTime + SoftLock Expiration Limit (Number of Seconds)**

Mule Hint
// Hard Coded 600 seconds

```
softLockExpirationDateTime = softLockDateTime + ("PT600S" as Period)
```

// Dynamic using a API Config Property

```
softLockExpirationDateTime = softLockDateTime + (("PT$(p('softlockExpirationSeconds') default 0)S") as Period)
```

- IF **inetUserStatus** exists and **inetUserStatus[0]** is not **"Active"** in the **Response**
- THEN **failureReason = { "code": "hlk", "description": "Hard/Indefinitely Locked" }**
- ELSE IF **softLockExpirationDateTime** greater than **Now()**
- THEN
- **secondsRemaining = softLockExpirationDateTime as Number - Now() as Number**
  failureReason = { "code": "slk", "description": "Temporarily Locked", "secondsRemaining": secondsRemaining}
- ELSE
- **failureReason = { "code": "icr", "description": "Invalid Credentials" }**
- Return the Invalid Credentials Response as defined in the Common Response Bodies
- **Account Info Flow**
- 1. AM Authenticate
- **Request to ForgeRock**
- Endpoint: POST **${amAdminAuthURL}**

Header

```
Content-Type: application/json
Accept-API-Version: resource=2.1
X-OpenAM-Username: {{amAdminUsername}}
X-OpenAM-Password: {{amAdminPassword}}
```

- IF **tokenId** exists in the **Response**
- THEN continue on to **AM Authorize**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- 2. AM Authorize
- **Request to ForgeRock**

- Endpoint: GET **${amAdminDataURL} replace "{{username}}" with Request userID**

Header

```
Content-Type: application/json
Cookie: iPlanetDirectoryPro={{CookieValue}}
Accept-API-Version: resource=2.1, protocol=1.0
```

- **Response from ForgeRock (Example)**

Body

```
{
  "username": "sukh08142019",
  "pwdAccountLockedTime": [
    "20190816161724.516Z"
  ],
  "mail": [
    "jgray1@aessuccess.org"
  ],
  "pwdFailureTime": [
    "20190816161724.513Z",
    "20190816161724.514Z",
    "20190816161724.515Z",
    "20190816161724.516Z"
  ],
  "inetUserStatus": [
    "Active"
  ],
  "lastLoginTime": [
    "20190823234210Z"
  ]
}
```

- **Second Factor Stage**
- Retrieve **Account Info** using **Account Info Flow**
- Save **hasKBA** Boolean from **Account Info Response.** If **kbaUser** exists read **kbaUser[0] else** default to false
- Use **emails** (mail[]) and **phones** (phone[]) from **Account Info Response** to build **contactOptions**
- The **description** is a masked version of the email or phone.

Email Masking Rules
An asterix (*) is the masking character
The @ symbol is anyway shown.
Email is defined as local-part@domain
Domain is defined as name.tld (tld = Top level domain, which is the value after the last "dot" example com)

For local-part
If 1 character, then mask it.
If 2 characters long, then mask last the last character.
If 3 or greater characters long, then mask characters in between first and last character, but mask length should not display more than 3 characters

For domain
Always show TLD (Top level domain), which is the value after the last "dot" (ex, com for example.com)
If name is 1 character, then mask it.
If name is 2 or greater characters, then mask from 2nd up to the last, but mask length should not display more than 3 characters.

Examples
a@example.com = *@e***.com (very extreme edge case)
ab@example.com = a*@e***.com (extreme edge case)
abc@example.com = a*c@e***.com (edge case)
abcd@example.com = a**d@e***.com (edge case)
abcde@example.com = a***d@e***.com
abcdef@example.com = a***f@e***.com
abcdefghijklmnopqrstuvwxyz@example.com = a***z@e***.com

- The **value** is an MD5 hash of the email or phone

```
%dw 2.0
import dw::Crypto
output application/json
---
[
    {
        "name": "email",
        "value": Crypto::MD5("aaaaaa@example.com" as Binary),
        "description": "a*****@example.com"
    },
    {
        "name": "email",
        "value": Crypto::MD5("bbbbbb@example.com" as Binary),
        "description": "b*****@example.org"
    },
    {
        "name": "sms",
        "value": Crypto::MD5("7177201234" as Binary),
        "description": "(***) ****-1234"
    },
    {
        "name": "voice",
        "value": Crypto::MD5("7177201111" as Binary),
        "description": "(***) ****-1111"
    }
]
```

- Return the **Second Factor Options Response** as defined in the **Common Response Bodies**
- **OTP Choice**
- This choice requires the out of band delivery (currently only by email) of an OTP (One-Time Passcode) and then for it to be submitted on a subsequent request.
- **Deliver OTP**
- SET **authId** = payload.**authId**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "ChoiceCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = 1
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "ChoiceCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "Default value is 0, KBA. OTP trigger value is 1. NPPI is 2"
                },
                {
                    "name": "choices",
                    "value": [
                        "KBA",
                        "OTP",
                        "NPPI"
                    ]
                },
                {
                    "name": "defaultChoice",
                    "value": 0
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": 1
                }
            ]
        }
    ]
}
```

- IF **callbacks** exists in the **Response** and **callbacks[0].type = "PasswordCallback"**
- THEN continue on to the **Submit OTP (One-Time Passcode) Stage**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- **Submit OTP**

- This stage requires the delivery of a One-Time Passcode
- **Request to ForgeRock**
- SET **authId** = payload.**authId**
- SET **passcode** = payload.**passcode**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "PasswordCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = **passcode**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "PasswordCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "One Time Password"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": "[Passcode Here]"
                }
            ]
        }
    ]
}
```

- IF **tokenId** exists in the **Response**
- THEN continue on to the **Final Stage**
- ELSE IF **callbacks** exists in the **Response** and **callbacks[0].type = "ChoiceCallback"**
- THEN continue on to the **Remember Device Stage**
- ELSE IF status **code = "401"** and **reason = "Unauthorized"** and **message = "Login failure"** in the **Response**
- THEN return the **Invalid Credentials Response** as defined in the **Common Response Bodies**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- Bypass Remember Device (Temporary)
- This step is to temporary bypass the choice to remember a device which will be done in a later sprint.
- **Request to ForgeRock**
- SET **authId** = payload.**authId**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "ChoiceCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = 0
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "ChoiceCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "0 = NO. 1 = YES"
                },
                {
                    "name": "choices",
                    "value": [
                        "NO",
                        "YES"
                    ]
                },
                {
                    "name": "defaultChoice",
                    "value": 0
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": 0
                }
            ]
        }
    ]
}
```

- **KBA Choice**
- Knowledge-based authentication (KBA) is a scheme in which the user is asked to answer at least one "secret" question.
- **Path**
- POST **${authURL}**
- **Get Question**
- **Request to ForgeRock**
- SET **authId** = payload.**authId**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "ChoiceCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = 0
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "ChoiceCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "Default value is 0, KBA. OTP trigger value is 1. NPPI is 2"
                },
                {
                    "name": "choices",
                    "value": [
                        "KBA",
                        "OTP",
                        "NPPI"
                    ]
                },
                {
                    "name": "defaultChoice",
                    "value": 0
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": 0
                }
            ]
        }
    ]
}
```

- **Sample Response from ForgeRock**

Body

```
{
    "authId": "...",
    "callbacks": [
        {
            "type": "TextInputCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "userAnswer"
                },
                {
                    "name": "defaultText",
                    "value": "{\"qid\":\"33\",\"qtext\":\"What was the nickname of your grandfather?\"}"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": ""
                }
            ]
        }
    ]
}
```

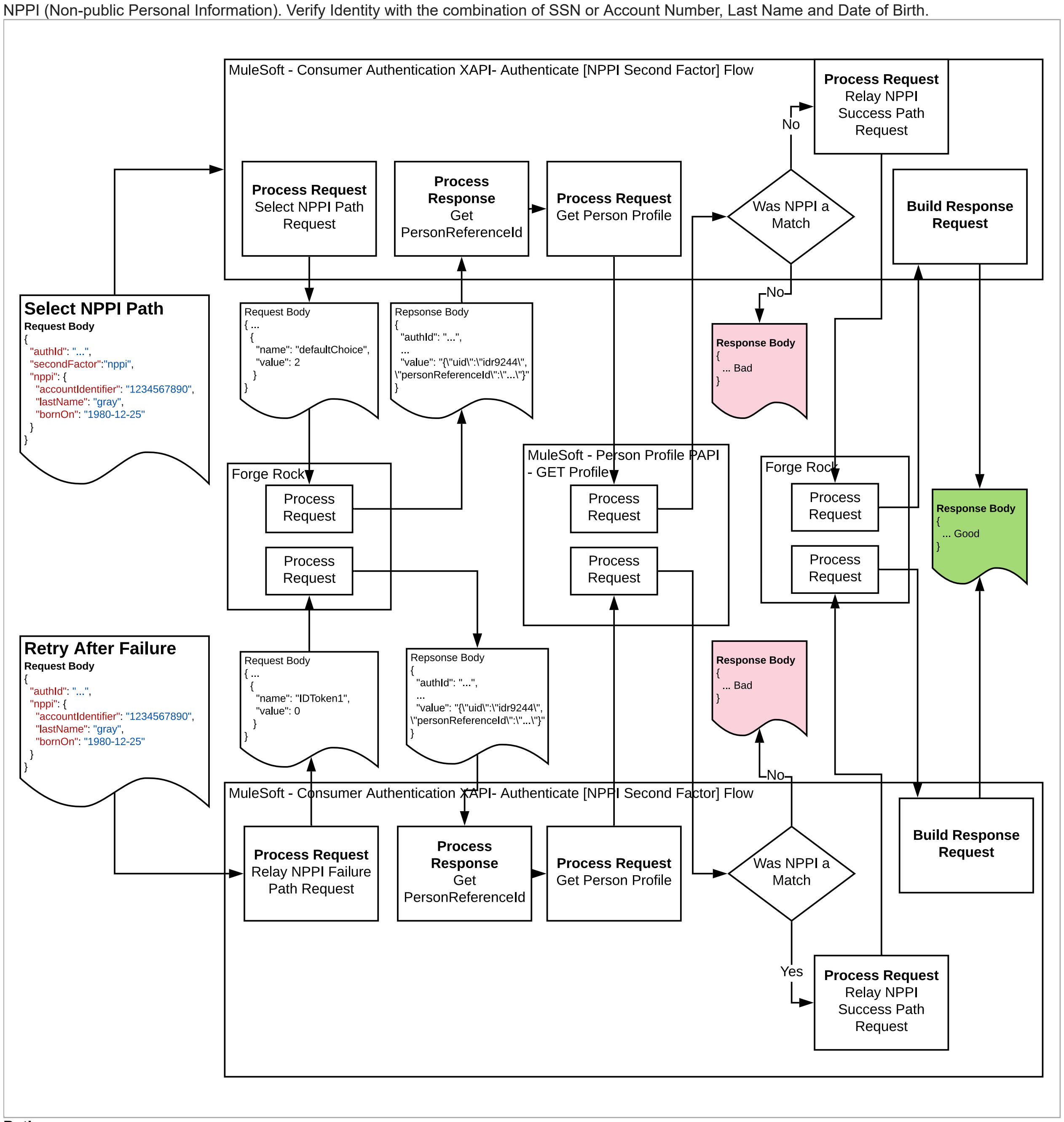
- IF **callbacks[0].output[1].value** when **callbacks[0].type = "TextInputCallback"**
- THEN IF **callbacks[0].output[0].value = "nppiDecision"**
- THEN jump to the **NPPI Choice**
- ELSE IF **callbacks[0].output[0].value = "userAnswer"**
- THEN Build the flow **Response** by capturing the **questionText (qtext)** from **callbacks[0].output[1].value**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- **Answer Question**
- Used to answer the preceding question.
- **Request to ForgeRock**
- SET **authId** = payload.**authId**
- Continue to sent the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "TextInputCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = payload.**kbaAnswer**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "TextInputCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "userAnswer"
                },
                {
                    "name": "defaultText",
                    "value": ""
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": "{{kbAnswer}}"
                }
            ]
        }
    ]
}
```

- **NPPI Choice**
- NPPI (Non-public Personal Information). Verify Identity with the combination of SSN or Account Number, Last Name and Date of Birth.
- 
MuleSoft - Consumer Authentication XAPI- Authenticate [NPPI Second Factor] Flow
Process Request Select NPPI Path Request
Process Response Get PersonReferenceId
Process Request Get Person Profile
Was NPPI a Match
No
Process Request Relay NPPI Success Path Request
Build Response Request
No
Select NPPI Path
Forge Rock
Process Request
Process Request
MuleSoft - Person Profile PAPI - GET Profile
Process Request
Process Request
Forge Rock
Process Request
Process Request
Retry After Failure
No
MuleSoft - Consumer Authentication XAPI- Authenticate [NPPI Second Factor] Flow
Process Request Relay NPPI Failure Path Request
Process Response Get PersonReferenceId
Process Request Get Person Profile
Was NPPI a Match
Yes
Process Request Relay NPPI Success Path Request
Build Response Request

Select NPPI Path
Request Body
```
{
  "authId": "...",
  "secondFactor":"nppi",
  "nppi": {
    "accountIdentifier": "1234567890",
    "lastName": "gray",
    "bornOn": "1980-12-25"
  }
}
```

Request Body
```
{ ...
  {
    "name": "defaultChoice",
    "value": 2
  }
}
```

Repsonse Body
```
{
  "authId": "...",
  ...
  "value": "{\"uid\":\"idr9244\",
\"personReferenceId\":\"...\"}"
}
```

Response Body
```
{
  ... Bad
}
```

Response Body
```
{
  ... Good
}
```

Retry After Failure
Request Body
```
{
  "authId": "...",
  "nppi": {
    "accountIdentifier": "1234567890",
    "lastName": "gray",
    "bornOn": "1980-12-25"
  }
}
```

Request Body
```
{ ...
  {
    "name": "IDToken1",
    "value": 0
  }
}
```

Repsonse Body
```
{
  "authId": "...",
  ...
  "value": "{\"uid\":\"idr9244\",
\"personReferenceId\":\"...\"}"
}
```

Response Body
```
{
  ... Bad
}
```

- **Path**
- POST **${authURL}**
- **Select NPPI**
- SET **authId** = payload.**authId**
- SET **accountIdentifier** = payload.nppi.accountIdentifier
- SET **lastName**= payload.nppi.lastName
- SET **birthDate** = payload.nppi.birthDate
- Continue to send the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
    - callbacks[0].type = "ChoiceCallback"
    - callbacks[0].input[0].name = "IDToken1"
    - callbacks[0].input[0].value = 2
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
```

```
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "ChoiceCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "Default value is 0, KBA. OTP trigger value is 1. NPPI is 2"
                },
                {
                    "name": "choices",
                    "value": [
                        "KBA",
                        "OTP",
                        "NPPI"
                    ]
                },
                {
                    "name": "defaultChoice",
                    "value": 0
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": 2
                }
            ]
        }
    ]
}
```

- **Sample Response from ForgeRock**

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "TextInputCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "nppiDecision"
                },
                {
                    "name": "defaultText",
                    "value": "{\"uid\":\"idr9244\",\"personReferenceId\":\"531626125\"}"
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": ""
                }
            ]
        }
    ]
}
```

- SET **personRefId** from the previous **Response** by capturing the **personReferenceId** from **callbacks[0].output[1].value**
- SET **userId** from the previous **Response** by capturing the **uid** from **callbacks[0].output[1].value**
- **Call Person Identifier API**
- GET/person-identifier/v1/person-identifiers/{personRefId}

Body

```
{
    "referenceId": "5d2601f6a16f5e0001d756a5",
    "ssn": "*********",
    "accountNumber": "**********"
}
```

- IF sizeOf(vars.accountIdentifier) == 9 and vars.accountIdentifier == payload.ssn
- THEN identifierMatchFound= true
- ELSE IF sizeOf(vars.accountIdentifier) == 10 and vars.accountIdentifier == payload.accountNumber
- THEN identifierMatchFound= true
- ELSE
- THEN identifierMatchFound = false
- IF identifierMatchFound
- THEN **Call Person Profile API**
- GET /person-profile/v1/profiles/{personRefId}

Body

```
{
    "name": {
        "first": "John",
        "middle": "Quincy",
        "last": "Adams",
        "suffix": "Sr"
    },
    "birthDate": "2017-12-31"
}
```

- IF **payload.name.last?** and **payload.birthDate?** and **payload.name.last** == **vars.lastName** and **payload.birthDate** == **vars.lastName**
- THEN completeMatchFound = true
- ELSE
- completeMatchFound = false
- ELSE
- completeMatchFound = false
- **Device Stage**
- **Save Device Or Not Save Device**
- SET **authId** = payload.**authId**
- Continue to send the **amlbcookie** cookie.
- Build the next Request to ForgeRock by setting the following attributes.
  - callbacks[0].type = "ChoiceCallback"
  - callbacks[0].input[0].name = "IDToken1"
  - callbacks[0].input[0].value = **0** or **1 (0 = NO. 1 = YES)**
- **Request to ForgeRock**
- Endpoint: POST **${authURL}**

Header

```
Content-Type:application/json
Accept-API-Version:resource=2.0, protocol=1.0
```

Body

```
{
    "authId": "{{authId}}",
    "callbacks": [
        {
            "type": "ChoiceCallback",
            "output": [
                {
                    "name": "prompt",
                    "value": "0 = NO. 1 = YES"
                },
                {
                    "name": "choices",
                    "value": [
                        "NO",
                        "YES"
                    ]
                },
                {
                    "name": "defaultChoice",
                    "value": 0
                }
            ],
            "input": [
                {
                    "name": "IDToken1",
                    "value": [Save Answer Here]
                }
            ]
        }
    ]
}
```

- IF **tokenId** exists in the **Response**
- THEN continue on to the **Final Stage**
- ELSE
- Return the **Exception Response** as defined in the **Common Response Bodies**
- **Final Stage**
- If device is recognized, valid one-time passcode (OTP), valid security question answered, or valid NPPI/PII is entered then the final authenticate **Response** from ForgeRock should be given.
- **1. Authenticate Response**
- **Response from ForgeRock**
- The iPlanetDirectoryPro cookie is a very important part of this header.

Header

```
X-Frame-Options: SAMEORIGIN
Set-Cookie: iPlanetDirectoryPro=4AjW_2_BgzvNpLzUIbnyorEoUOg.*AAJTSQACMDIAAlNLABw5WFJZNWkxcUJIUGRMMGtMZjRFVmhCN2Q0dkk9AAR0eXBlAANDVFMAAlM
Set-Cookie: amlbcookie=01; Path=/; Domain=pheaacloud.org
Cache-Control: no-cache, no-store, must-revalidate
Content-API-Version: resource=2.1
Expires: 0
Pragma: no-cache
Content-Type: application/json
Content-Length: 177
Date: Tue, 02 Jul 2019 19:13:28 GMT
```

Body

```
{
    "tokenId": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhZXNzdWNjZXNzLm9yZyIsImlhdCI6MTUxOTY4OTYwMCwic3ViIjoiNWMwZmRhZjI4NDAwYWE
    "successUrl": "/auth/console",
    "realm": "/dev1-testrealm"
}
```

- **2. Authorize**
- **Request to ForgeRock**
- Endpoint: POST **${authorizeURL}**

Header

```
Content-Type:application/x-www-form-urlencoded
```

Body

```
scope:openid profile email phone
response_type:code
client_id:oauthclient
csrf:{{tokenId}}
redirect_uri:https://devforgeaes.pheaacloud.org:443/callback
state:{{$guid}}
decision:allow
nonce:{{$guid}}
```

- **Response from ForgeRock**
- **Header**
- Extract the code from the location header
- Example
- Location: https://devforgeaes.pheaacloud.org/callback?**code**=**jW5boRUWluqWqhSZ_yotZvkRh2o**&**iss**=https%3A%2F%2Fdevforgeamsrva.pheaacloud.org%3A8443%2Fauth%2Foauth2%2Frealms%2Froot%2Frealms%2Fdev1-aes&state=83157a17-87cb-4c9f-bdc9-d39d2327e8c6&**client_id**=oauthclient
- **Body**
- None
- **3. Obtain Tokens (Access Token)**
- **Request to ForgeRock**
- Endpoint: POST **${accessTokenURL}**

Header

```
Content-Type:application/x-www-form-urlencoded
```

Body

```
grant_type:authorization_code
code:jW5boRUWIuqWqhSZ_yotZvkRh2o
client_id:oauthclient
client_secret:password
redirect_uri:https://devforgeaes.pheaacloud.org:443/callback
```

- **Response from ForgeRock**

Body

```
{
    "access_token": "pSnaY_kz7mQcZcqqdSQhdXt8AuI",
    "refresh_token": "rq4d7fOmsw438SojxMutUj6fuKw",
    "scope": "phone openid profile email",
    "id_token": "eyJ0eXAiOiJKV1QiLCJraWQiOiJ3VTNpZklJYUxPVUFSZVJCL0ZHNmVNMVAxUU09IiwiYWxnIjoiUlMyNTYifQ.eyJhdF9oYXNoIjoiLVlyNUQ4MXVYd0xW
    "token_type": "Bearer",
    "expires_in": 179,
    "nonce": "3a0d286d-c2c4-4ef5-9783-f44c8ccfd675"
}
```

- **Build Response**
- Header
- Take the value of "**access_token**" from the **Response Body** and set a cookie called "**access_token**" to the value. Make sure the cookie is set to **Secure** and **HttpOnly**. This will ensure it only works on **HTTPS** and can not be accessed by **JavaScript**.
- Take the value of "**refresh_token**" from the **Response Body** and set a cookie called "**refresh_token**" to the value. Make sure the cookie is set to **Secure** and **HttpOnly**. This will ensure it only works on **HTTPS** and can not be accessed by **JavaScript**.
- Take the value of "**id_token**" from the **Response Body** and set a cookie called "**id_token**" to the value. Make sure the cookie is set to **Secure** and **HttpOnly**. This will ensure it only works on **HTTPS** and can not be accessed by **JavaScript**.

header

```
Set-Cookie: access_token=pSnaY_kz7mQcZcqqdSQhdXt8AuI; Expires=<date>; Path=/; Secure; HttpOnly;
Set-Cookie: refresh_token=pSnaY_kz7mQcZcqqdSQhdXt8AuI; Expires=<date>; Path=/; Secure; HttpOnly
Set-Cookie: id_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhZXNzdWNjZXNzLm9yZyIsImlhdCI6MTUxOTY4OTYwMCwic3ViIjoiNWMwZmRhZjI4ND
```

- Body
- Set **authId** to match value returned in the last **Authenticate Response**
- Set **status** to "success"
- Set **stage** to "final"
- Set **destinationInfo code.** To do this the flow will need a **Destination Map Endpoint** configuration created. If the **Destination Map Endpoint** can not be found the location should default to "/". Use the **destinationInfo code** from the **Request Body** to find the **destinationLocation**. If the **destinationInfo code** does not exist the value should default to "**home**".
  Example
  https://www.aessuccess.org/portal/deep-link.json

  Destination Map Endpoint File Content

  ```
  {
    "home":"https://www.aessuccess.org/portal",
    "payments":"https://www.aessuccess.org/portal/payments",
    "tax":"https://www.aessuccess.org/portal/inbox#tax"
  }
  ```

- Set **isDeviceTrusted** to false, Logic for true will be added in a future story.
- Set **accessInfo access_token** to match value returned in the last **Access Token Response**
- Set **accessInfo refresh_token** to match value returned in the last **Access Token Response**
- Set **accessInfo scope** to match value returned in the last **Access Token Response**
- Set **accessInfo expires_in** to match value returned in the last **Access Token Response**
- Set **accessInfo nonce** to match value returned in the last **Access Token Response**
- Set **authorizationClaims** to match the values/claims returned in the "**id_token**" from the **Response Body.** The value should be a **JWT (JSON Web Tokens)** which is a token with the format of **Header.Payload.Signature** take the **Payload** of the token and base64 decode it (could use a JWT Library).
- See **Final Response** as defined in the **Common Response Bodies**

No labels

6 Comments

Jason Gray

@Suresh Chinta @Srinivasa reddy Bathula

We should change the Set-Cookie to use Max-Age instead of Expires. Max-Age is not recommend and takes precedence over Expires. It's also easier to understand and less to format.

We also need to make sure the Secure and HttpOnly directives are being set.

https://mrcoles.com/blog/cookies-max-age-vs-expires/

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie

Suresh Chinta

@Jason Gray , Blog from 2009 🙂

Anyways, this will be backlogged for subsequent implementation.

Jason Gray

> "userChannel": "web | mobile-app"

I think just calling it channel is enough. I would also suggest it also be optional and using an opinionated approach default to "web".

Suresh Chinta

We're using channel attribute name for second-factor response. To avoid name collision (and confusion) was proposing userChannel.

Attribute is optional and defaults to 'web'.

Jason Gray

We never formal had an information exchange when the API switched teams, but the previous plan was to use the X-Client-Id header. All of our down stream XAPIs already have X-Client-Id build into them and if it where in the JWT it might be even better.

What we where planning on doing was creating a map of whitelisted ids (X-Client-Id) . Then, mapping the client id to channel. This also would allows us to know which client is having issues.

```
{
"clients":[
{
"id": "AEMAESCONSMR",
"channel":"web",
"description": "Adobe Experience Manager AES Consumer"
},
{
"id": "IOSAESCONSMR",
"channel":"mobile",
"description": "iOS AES Consumer"
},
{
"id": "ANDAESCONSMR",
"channel":"mobile",
"description": "Android AES Consumer"
}
]
}
```

Suresh Chinta

I don't believe I'm following the trail here. A meeting invite is forth-coming to discuss.