# (no subject)

**Mitchell Lewars** <mlewars@pingidentity.com>
Draft

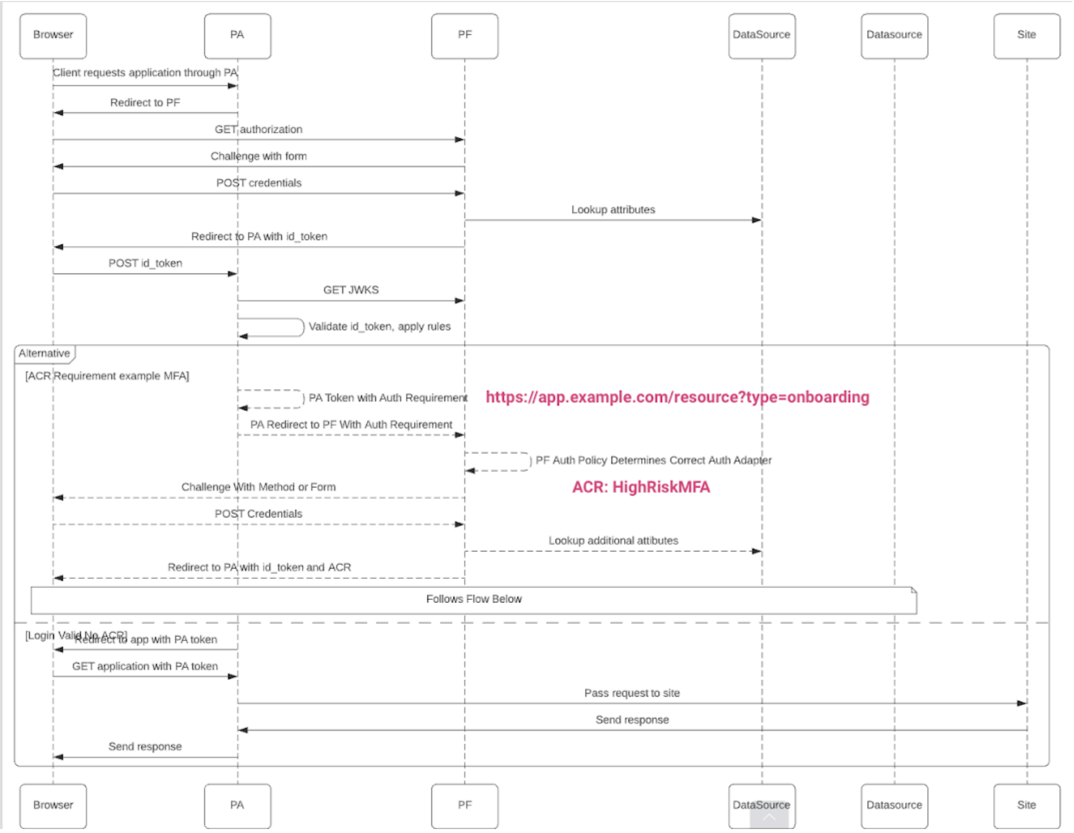
Fri, Sep 24, 2021 at 4:31 PM

Hello all,

In class we talked about different usage cases.  One was related to Access Control and placing requirements on a resource.  This ACR will then have to be included in the session from the OpenID Provider (Authorization Server) / PingFederate.  Technically, the client makes a request with a pre-specified value in the URI and this will tell PA to add the ACR; when the user is sent to login then PF will see the ACR and use the correct Auth Method to provide the required ACR.

In My lab MFA could be ANY adapter that can be used.

If this above is confusing consider this image.  Note the URL has the value.


Browser
PA
PF
DataSource
Datasource
Site
Client requests application through PA
Redirect to PF
GET authorization
Challenge with form
POST credentials
Lookup attributes
Redirect to PA with id_token
POST id_token
GET JWKS
Validate id_token, apply rules
Alternative
[ACR Requirement example MFA]
PA Token with Auth Requirement
https://app.example.com/resource?type=onboarding
PA Redirect to PF With Auth Requirement
PF Auth Policy Determines Correct Auth Adapter
ACR: HighRiskMFA
Challenge With Method or Form
POST Credentials
Lookup additional attibutes
Redirect to PA with id_token and ACR
Follows Flow Below
[Login Valid No ACR]
Redirect to app with PA token
GET application with PA token
Pass request to site
Send response
Send response
Browser
PA
PF
DataSource
Datasource
Site

Here is a lab you can try, either in your Class machine or elsewhere if you have PA and PF.  Its considered a more advanced usage case but I think most of the class would get some good experience if you can do it.

=====---------------

> This solution brief documents a method for protecting HTTP query string parameters for PingAccess resources, using Groovy Script and Authentication Requirements rules combined in a Rule Set. For example, a protected resource path will only allow a specific query parameter name/value pair if a condition is met, such as a requirement for MFA.

Overview:

- The Groovy Script rule is responsible for evaluating the name and value of the HTTP parameter.
- The Authentication Requirement rule defines a minimum authentication context class reference (ACR) that must be met to access the protected resource.
- The Rule Set combines both rules above with a success criteria of Any, meaning either of the rules can evaluate true for the Rule Set to pass.
- PingFederate must also be configured with a Requested AuthN Context Selector in a policy tree to handle the acr_values parameter from the authorization request. The policy contract must also manually fulfill the reserved SAML_AUTHN_CTX with the minimum requirement text value based on the PA requirement rule.

Requirements:

- PingAccess 5.2+
- PingFederate 8.x+ (the Requested AuthN Context Authentication Selector)
- An application in PingAccess will be configured for type Web, with multiple resources defined. The rule set can be applied to individual resources' Web policy, or to the entire application's Web policy. In the examples published here, the rule set is applied to the application's Web policy.
- An authentication policy in PingFederate, beginning with Requested AuthN selector and ending with APC

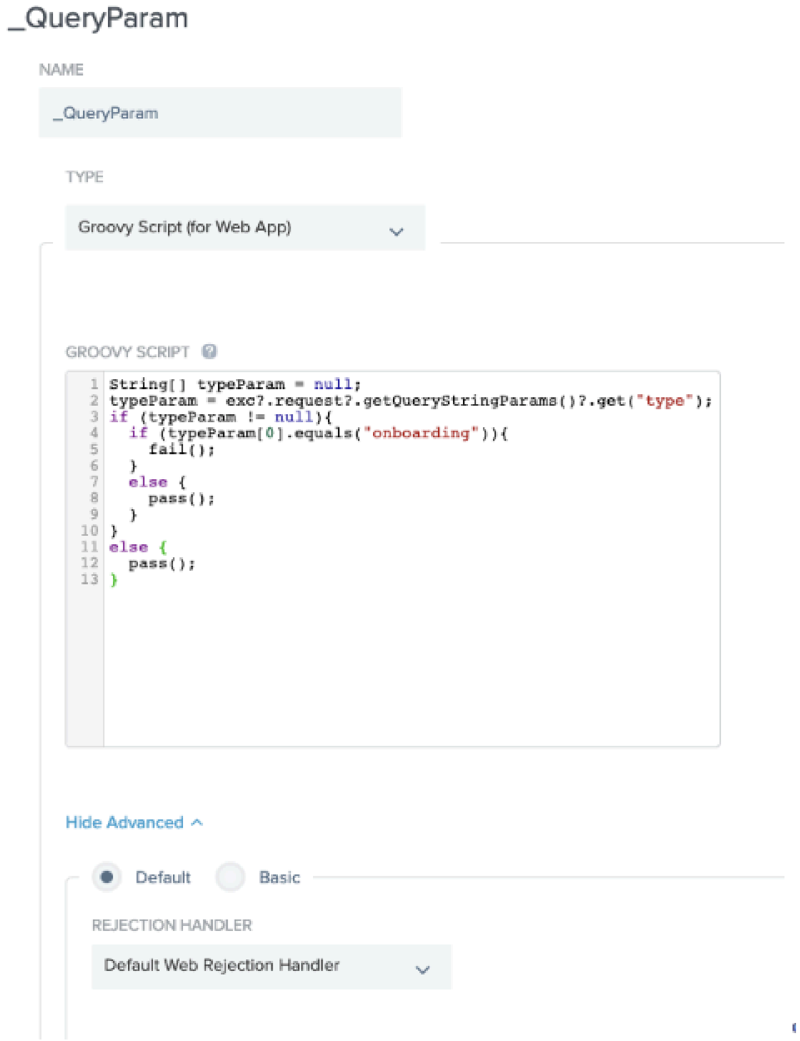
Assumptions:

If a request is made to https://app.example.com/resource?type=onboarding, the request must be protected with MFA. A request to https://app.example.com/resource?${*anything except type=onboarding*} will only require first-factor authentication.

In this example, the protected HTTP query parameter name is **type** and must have a value of **onboarding** for the Groovy rule to fail. The authentication requirement rule is configured with a minimum requirement of **HighRiskMFA**. When the Groovy rule fails, it will force the authentication requirement rule to succeed in order to pass, because the success criteria is set to Any. The key/value pair **type=onboarding** can exist anywhere in the query string, regardless of position.

PingAccess Groovy Script:

```
String[] typeParam = null;

typeParam = exc?.request?.getQueryStringParams()?.get("type");

if (typeParam != null){

  if (typeParam[0].equals("onboarding")){

    fail();

  }

  else {

    pass();

  }

}

else {

  pass();

}
```



_QueryParam

NAME

_QueryParam

TYPE

Groovy Script (for Web App)

GROOVY SCRIPT

```
String[] typeParam = null;
typeParam = exc?.request?.getQueryStringParams()?.get("type");
if (typeParam != null){
  if (typeParam[0].equals("onboarding")){
    fail();
  }
  else {
    pass();
  }
}
else {
  pass();
}
```

● Default ○ Basic

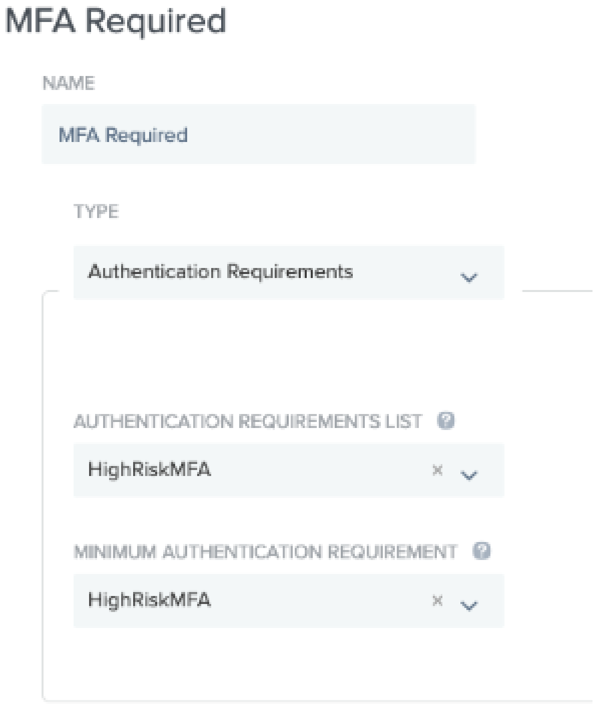
REJECTION HANDLER

Default Web Rejection Handler

One thing to note is that the query parameter value must be single-valued, hence the **typeParam[0]** variable indexing the first element of the query parameter key name. Also notice that the getQueryStringParam()?.get(String keyName) Groovy script method will return a Java string array (String[]), so the typeParam variable must be initialized as such.

This rule will be combined with the below Authentication Requirement Rule in a Rule Set:

And the Rule Set:



MFA Required

NAME

MFA Required

TYPE

Authentication Requirements

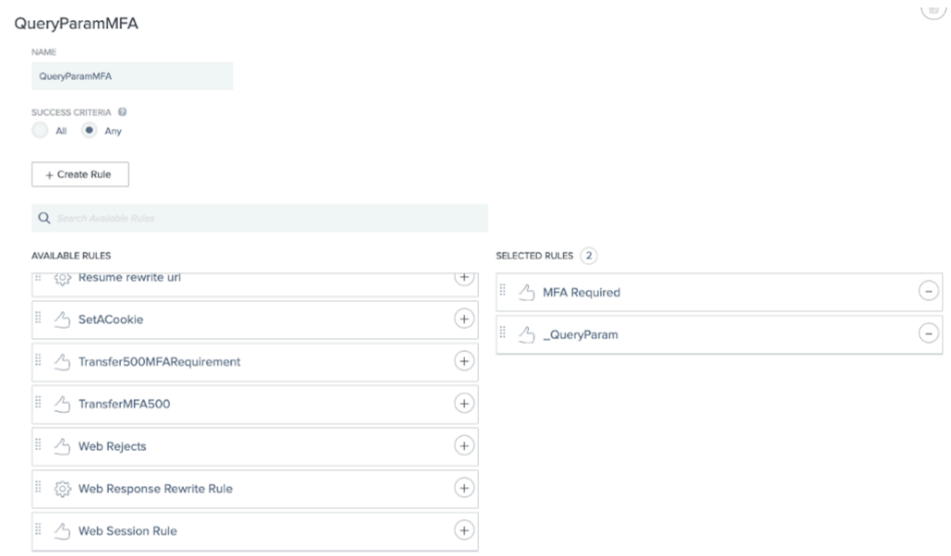
AUTHENTICATION REQUIREMENTS LIST

HighRiskMFA

MINIMUM AUTHENTICATION REQUIREMENT

HighRiskMFA

And the Rule Set:



QueryParamMFA

NAME

QueryParamMFA

SUCCESS CRITERIA

○ All ● Any

+ Create Rule

Search Available Rules

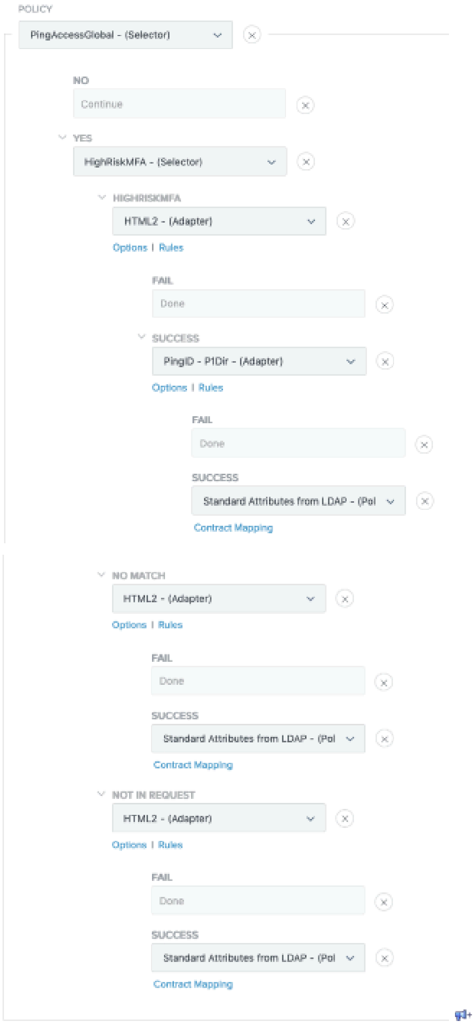
AVAILABLE RULES

- Resume rewrite url
- SetACookie
- Transfer500MFARequirement
- TransferMFA500
- Web Rejects
- Web Response Rewrite Rule
- Web Session Rule

SELECTED RULES 2

- MFA Required
- _QueryParam

The steps to setup an authentication policy in PingFederate are assumed. The below authentication policy will handle the HighRiskMFA ACR value:

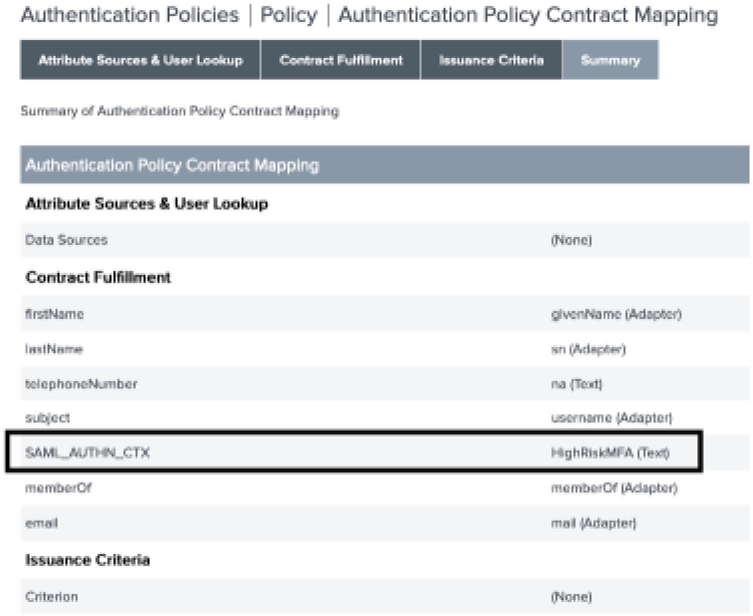


POLICY

PingAccessGlobal - (Selector)

- NO
  - Continue
- YES
  - HighRiskMFA - (Selector)
  - HIGHRISKMFA
    - HTML2 - (Adapter)
    - Options | Rules
    - FAIL
      - Done
    - SUCCESS
      - PingID - P1Dir - (Adapter)
      - Options | Rules
      - FAIL
        - Done
      - SUCCESS
        - Standard Attributes from LDAP - (Pol
        - Contract Mapping
  - NO MATCH
    - HTML2 - (Adapter)
    - Options | Rules
    - FAIL
      - Done
    - SUCCESS
      - Standard Attributes from LDAP - (Pol
      - Contract Mapping
  - NOT IN REQUEST
    - HTML2 - (Adapter)
    - Options | Rules
    - FAIL
      - Done
    - SUCCESS
      - Standard Attributes from LDAP - (Pol
      - Contract Mapping

On the HighRiskMFA branch that matches the requested ACR value, the contact mapping fulfills the SAML_AUTHN_CTX attribute:



Authentication Policies | Policy | Authentication Policy Contract Mapping

Summary of Authentication Policy Contract Mapping

| Authentication Policy Contract Mapping | |
|---|---|
| **Attribute Sources & User Lookup** | |
| Data Sources | (None) |
| **Contract Fulfillment** | |
| firstName | givenName (Adapter) |
| lastName | sn (Adapter) |
| telephoneNumber | na (Text) |
| subject | username (Adapter) |
| SAML_AUTHN_CTX | HighRiskMFA (Text) |
| memberOf | memberOf (Adapter) |
| email | mail (Adapter) |
| **Issuance Criteria** | |
| Criterion | (None) |

The resultant OIDC id_token will now contain an acr claim that matches the mapped SAML_AUTHN_CTX value (HighRiskMFA). This fulfills the requirement rule on PingAccess, so the request will be permitted.

=====-------=======

Best regards,

Ping Identity

**Mitchell Lewars**
Senior Professional Services Consultant
mitchelllewars@pingidentity.com
North America: +1 855.355.PING
Worldwide: +1 303.468.2857

**Connect with us:** Glassdoor logo LinkedIn logo twitter logo facebook logo youtube logo Blog logo