# Microsoft 2-factor authenticatie als middel voor SURFsecureID

Datum: 19 augustus 2019
Versie: 1.1
Auteur: Beat Nideröst
Comments: Finale versie na review door SURFnet

# Inleiding

Veel onderwijs- en onderzoeksinstellingen in Nederland maken gebruik van federatieve authenticatie via SURFconext voor het authentiseren van gebruikers bij applicaties. Daarbij is het gebruik van enkel wachtwoordauthenticatie voor veel toepassingen onvoldoende. Daarom biedt SURF 2-factor authenticatie (2FA) aan via SURFsecureID (SSID).

Veel instellingen maken daarnaast gebruik van Microsoft Active Directory Federation Services (AD FS) in combinatie met Azure Active Directory (AAD) voor de authenticatie van gebruikers naar o.a. de Microsoft cloud (Office 365, Azure, etc.) Zij beschikken vaak over licenties voor het gebruik van 2FA van Microsoft en een aantal instellingen zet deze functionaliteit ook in.

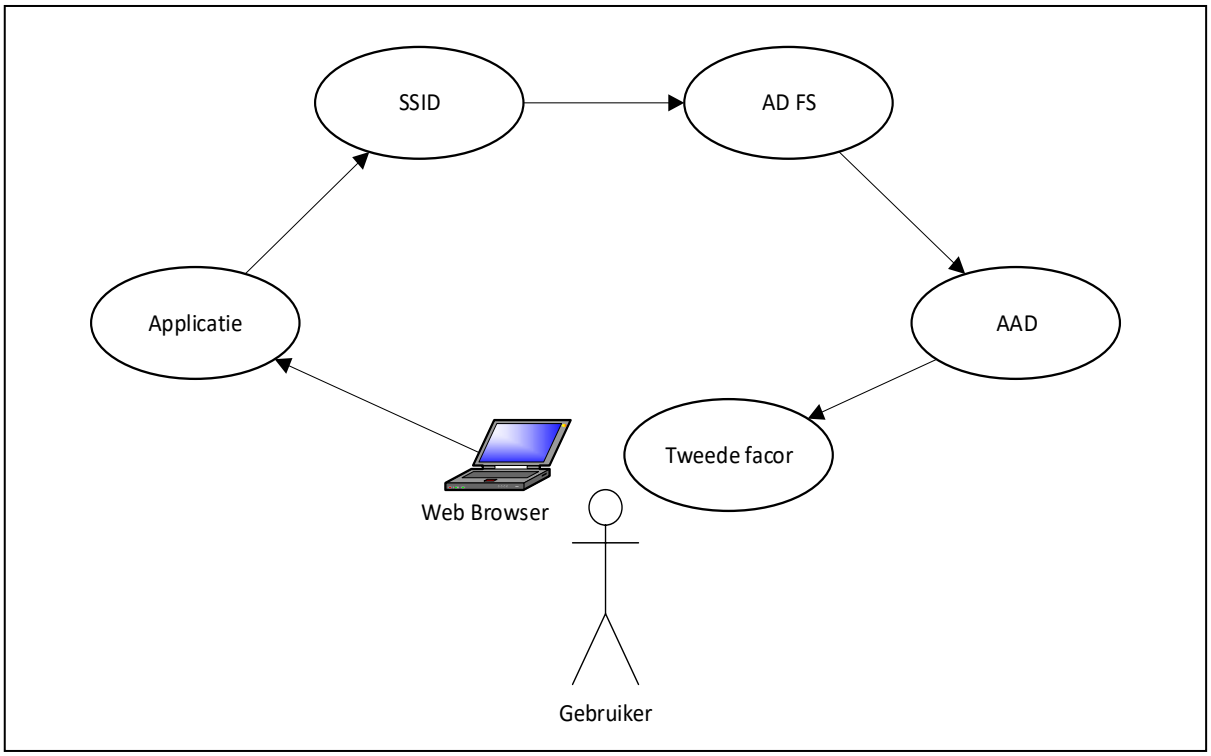
Voor gebruikers is het onhandig en verwarrend als zij meerdere 2FA-authenticatiemiddelen naast elkaar moeten inzetten, afhankelijk van de applicatie die zij willen gebruiken. Daarom ligt het voor de hand dat SURF in SSID naast de huidige middelen ook 2FA van Microsoft als middel beschikbaar stelt.

2at heeft onderzocht of dit technisch mogelijk is. Dat blijkt zo te zijn. Dit document beschrijft de oplossingsrichting en exacte configuratie van AD FS en AAD waarmee het mogelijk is om 2FA van Microsoft als middel in te zetten voor SSID.

# Oplossingsrichting

Figuur 1 toont de technische componenten die samen de inzet van 2FA van Microsoft als authenticatiemiddel in SSID mogelijk maken. Daarbij is de applicatie gekoppeld aan SURFsecureID. SURFconext bepaalt aan de hand van de ingelogde gebruiker en het authenticatieverzoek van de applicatie waar de gebruiker op inlogt of een tweede factor nodig is. Wanneer dit het geval is wordt SSID aangeroepen om de gebruiker via een tweede factor te authentiseren.

SSID kiest op basis van de ingelogde gebruiker en het gewenste authenticatieniveau een geschikt middel. In geval van 2FA van Microsoft stuurt SSID de gebruiker via de AD FS server naar AAD waar de



gebruiker zich met de tweede factor authentiseert. Vervolgens stuurt AAD de gebruiker via AD FS, en SSID terug naar de applicatie, die de gebruiker toegang verleent.

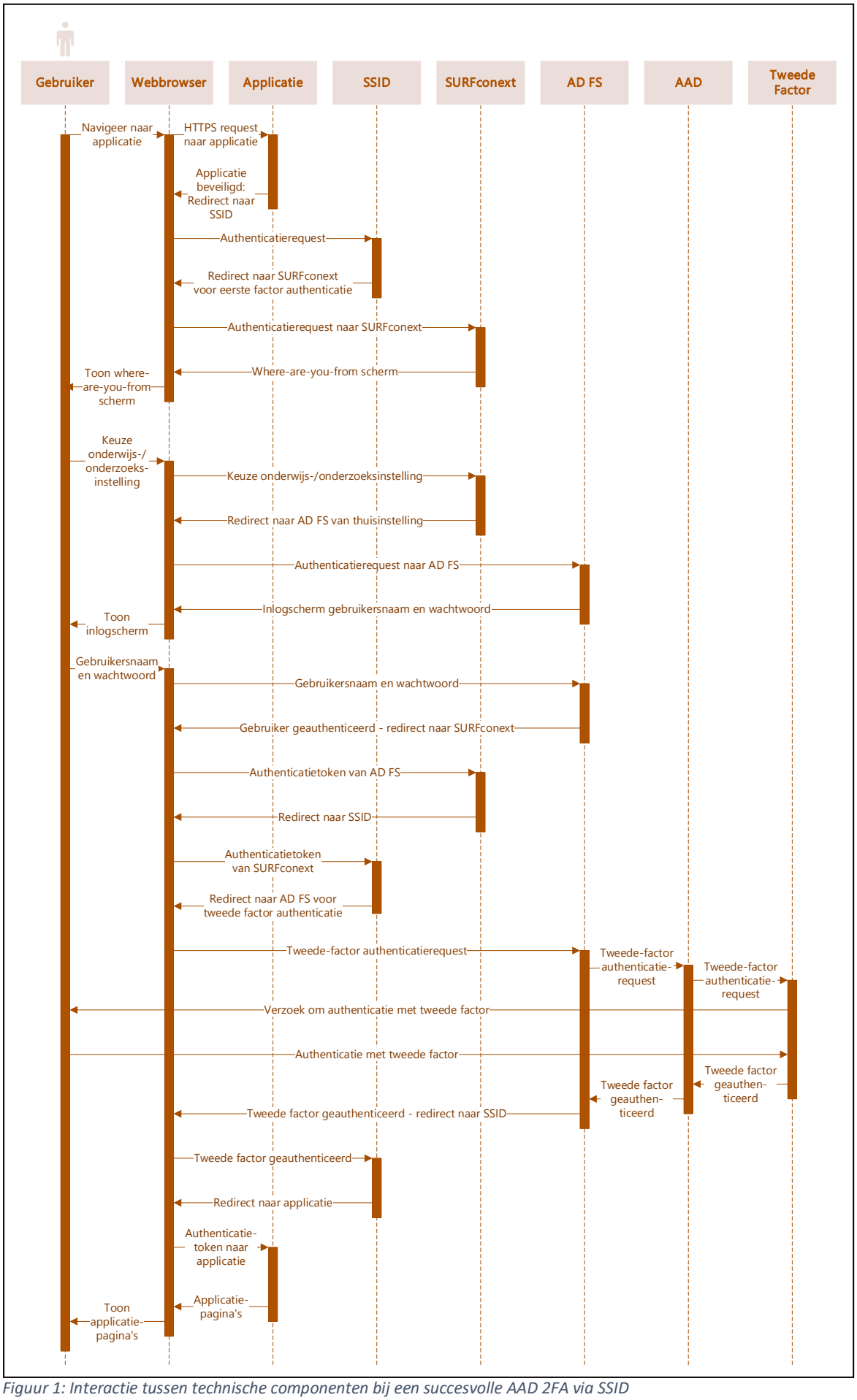
Figuur 2 op de volgende pagina toont de volledige interactie tussen de technische componenten bij een succesvolle AAD 2FA via SSID. Een belangrijk detail in deze interactie is het authenticatierequest dat SSID via de webbrowser van de gebruiker naar de AD FS server stuurt om 2FA te triggeren. Dit request is hieronder weergegeven (codesnippet 1). Het lijkt sterk op de requests die SSID naar andere middelen – zoals TIQR of SMS – stuurt. Een belangrijk verschil is echter de waarde in het AuthnContextClassRef element in de RequestedAuthnContext. Deze moet in het geval van AD FS "http://schemas.microsoft.com/claims/multipleauthn" zijn in plaats van bijvoorbeeld "http://test.surfconext.nl/assurance/loa2".

```
<samlp:AuthnRequest
  xmlns:samlp='urn:oasis:names:tc:SAML:2.0:protocol'
  xmlns:saml='urn:oasis:names:tc:SAML:2.0:assertion'
  ID='_b52444dc3d1955da3268d27b913970c36f7a457828'
  Version='2.0'
  IssueInstant='2019-03-08T17:41:10Z'
  Destination='https://adfs.hartingcollege.nl/adfs/ls/'
  AssertionConsumerServiceURL='https://beat.moonshot.utr.surfcloud.nl/sp/acs.php'
  ProtocolBinding='urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST'>
  <saml:Issuer>https://beat.moonshot.utr.surfcloud.nl/sp/metadata.php</saml:Issuer>
  <saml:Subject>
                          <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">student05@student-hartingcollege.nl</saml:NameID>
  </saml:Subject>
  <samlp:RequestedAuthnContext>
    <saml:AuthnContextClassRef>http://schemas.microsoft.com/claims/multipleauthn</s
aml:AuthnContextClassRef>
  </samlp:RequestedAuthnContext>
```

*Codesnippet 1: SAML2.0 authenticatierequest dat 2FA triggert in AD FS*

## Testopstelling

Om de werking van de oplossingsrichting te testen heeft 2at een testopstelling ingericht bestaande uit een AD FS server die gekoppeld is aan AAD, SURFconext en SSID. De inrichting van deze testopstelling is in de basis gelijk aan de inrichting die voor andere testen voor SURFnet is gebruikt. De gebruikte Window versie is Windows Server 2016 EN US 64-bit version 1607 OS Build 14393.2879. De AD FS server versie is 4.0 met farm behavior level 3.




*Figuur 1: Interactie tussen technische componenten bij een succesvolle AAD 2FA via SSID*

## Configuratie

De AD FS server is geconfigureerd met Forms Based Authentication als primary authentication method en met Azure MFA als multi-factor authentication method.

De 2FA in AD FS wordt door SSID geïnitieerd door het versturen van een authenticatierequest met daarin de waarde "http://schemas.microsoft.com/claims/multipleauthn" in de RequestedAuthnContext (zie codesnippet 1). Dit is de tweede fase in het authenticatieproces, nadat de eerste fase – authenticeren via gebruikersnaam en wachtwoord – al via SSID en SURFconext is afgerond.

De koppeling naar AAD is opgezet met een Azure MFA authentication provider als "additional authentication provider"[1]. De koppeling met SURFconext is de relying party trust zoals deze standaard tussen SURFconext en een AD FS server wordt gerealiseerd.[2]

De koppeling met SSID gebruikt aanvullende relying party trust naast de relying party trust die een instelling al met SURFconext heeft. Dit terwijl een applicatie die SSID wenst in te zetten, expliciet aan SSID gekoppeld wordt in plaats van aan SURFconext (zie figuur 2). Het initieren van de tweede factor authenticatie vanuit SSID terwijl de eerste factor via SURFconext is geauthenticeerd werkt – zelfs als AD FS vanuit SSID andere metadata, identifiers en certificaten ontvangt als vanuit SURFconext. Dit aangezien AD FS het ingelogd zijn van de gebruiker via SURFconext via een web browser cookie onthoudt.

In bijlage 1 staat de volledige configuratie van de gebruikte Relying Party Trust.

## Keuzescherm bij meerdere 2FA methoden

AD FS bepaalt aan de hand van o.a. het type authenticatierequest, de persoon die probeert in te loggen, de attributen, claims en groepslidmaatschap van die persoon, het apparaat en het netwerkadres waarvan hij probeert in te loggen, de app waarop hij probeert in te loggen, de gevoeligheid van de gegevens die hij probeert te benaderen, het type bewerking dat hij wil uitvoeren en het beleid dat in AD FS is geconfigureerd, welke authenticatieniveau vereist is en welke (2FA) methoden beschikbaar zijn.

Wanneer er in een specifieke situatie meerdere authenticatiemogelijkheden beschikbaar zijn voor 2FA, kan AD FS worden geconfiguurd om de gebruiker te vragen welke authenticatiemogelijkheid hij wil gebruiken. Concreet kan de situatie zich voordoen dat een onderwijs-/onderzoeksinstelling vanuit AD FS zowel Azure MFA als SSID i.c.m. andere middelen zoals TIQR, SMS of Yubikey beschikbaar wil stellen. Dit laatste kan via de SURFsecureID plugin voor AD FS (ADFS.SCSA). Figuur 3 toont het keuzescherm dat een gebruiker in dit geval krijgt.

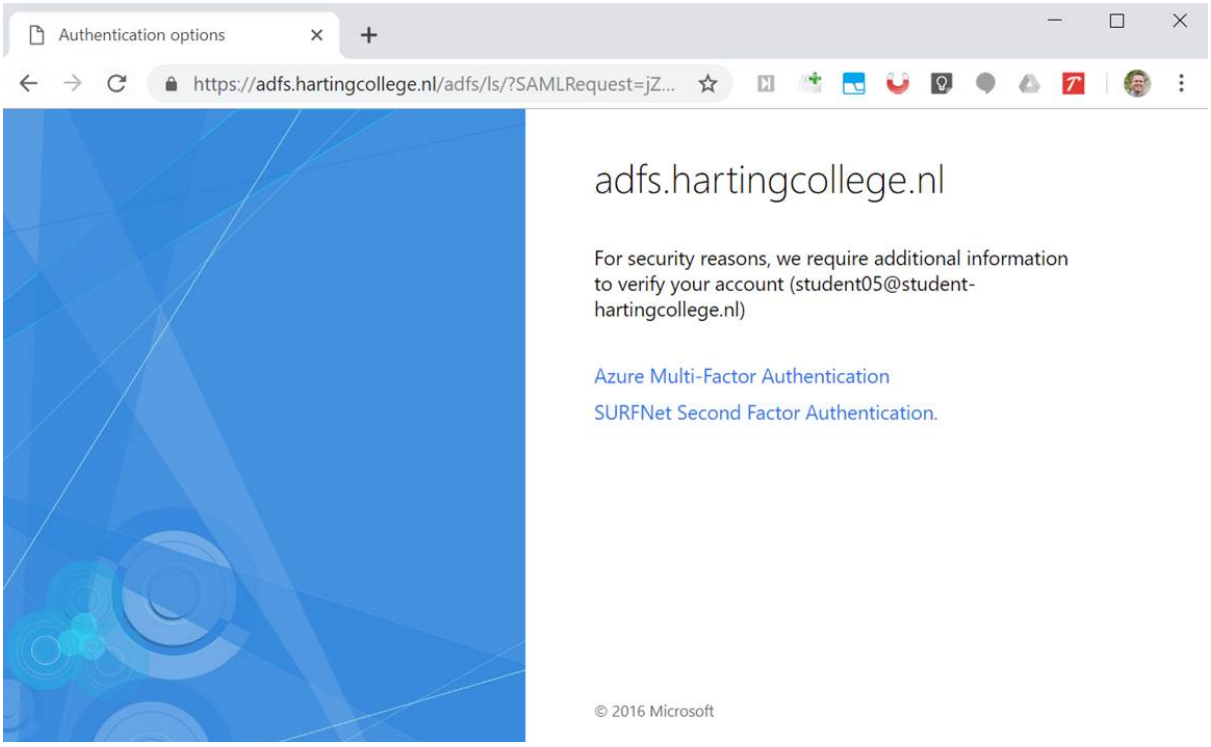
## Conclusie

Koppelen van AAD 2FA als middel aan SSID is technisch mogelijk met AD FS 4.0 als mediator. De configuratie waarmee dit mogelijk wordt is nagenoeg identiek aan de configuratie die wordt gebruikt voor een koppeling van AAD 2FA aan AD FS volgens de Microsoft standaard plus de standaard koppeling van SURFconext als relying party trust aan AD FS. Afwijkend is het SAMLP auhtenticatierequest dat SSID naar AD FS stuurt t.b.v. het aanroepen van de tweede factor plus de hiervoor benodigde extra koppeling van SSID als relying party trust aan AD FS. In tegenstelling tot bij

[1] Zie https://docs.microsoft.com/en-us/windows-server/identity/ad-fs/operations/configure-ad-fs-and-azure-mfa

[2] Zie https://docs.microsoft.com/en-us/windows-server/identity/ad-fs/operations/create-a-relying-party-trust

andere middelen dient SSID bij authenticatierequests richting AD FS een AuthnContextClassRef element met de waarde "http://schemas.microsoft.com/claims/multipleauthn" mee te geven.




*Figuur 2 Scherm t.b.v. keuze van gewenste 2FA methode*

## Bijlage 1: Relying Party Trust

Via PowerShell "get-adfsrelyingpartytrust":

```
AllowedAuthenticationClassReferences : {}
EncryptionCertificateRevocationCheck : CheckChainExcludeRoot
PublishedThroughProxy                : False
SigningCertificateRevocationCheck    : CheckChainExcludeRoot
WSFedEndpoint                        :
AdditionalWSFedEndpoint              : {}
ClaimsProviderName                   : {}
ClaimsAccepted                       : {}
EncryptClaims                        : True
Enabled                              : True
EncryptionCertificate                :
Identifier                           :
            {https://engine.test.surfconext.nl/authentication/sp/metadata}
NotBeforeSkew                        : 0
EnableJWT                            : False
AlwaysRequireAuthentication          : False
Notes                                :
OrganizationInfo                     :
                                       Technical Contact:
                                       Name:   Support SURFconext
                                       Emails:
                                         help@surfconext.nl
                                       Telephones:

                                       Support Contact:
                                       Name:   Support SURFconext
                                       Emails:
                                         help@surfconext.nl
                                       Telephones:

                                       Administrative Contact:
                                       Name:   Support SURFconext
                                       Emails:
                                         help@surfconext.nl
                                       Telephones:

ObjectIdentifier                     :
                                        1dbc6396-f85b-e711-a949-000d3a2a9bfa
ProxyEndpointMappings                : {}
ProxyTrustedEndpoints                : {}
ProtocolProfile                      : WsFed-SAML
RequestSigningCertificate            : {[Subject]
                                   C=NL, O=OpenConext, OU=Services, CN=Engine

                                       [Issuer]
                                   C=NL, O=OpenConext, OU=Services, CN=Engine

                                       [Serial Number]
                                         00EFD2451321D6ABB9
```

```
                                    [Not Before]
                                      23-10-2014 10:02:02

                                    [Not After]
                                      22-10-2024 10:02:02

                                    [Thumbprint]

                                  25728566C9942298368411E188C7AC4098F9E782
                                    }
EncryptedNameIdRequired             : False
SignedSamlRequestsRequired          : False
SamlEndpoints                       :
              {Microsoft.IdentityServer.Management.Resources.SamlEndpoint}
SamlResponseSignature               : AssertionOnly
SignatureAlgorithm                  :
                         http://www.w3.org/2001/04/xmldsig-more#rsa-sha256
TokenLifetime                       : 0
AllowedClientTypes                  : Public, Confidential
IssueOAuthRefreshTokensTo           : AllDevices
RefreshTokenProtectionEnabled       : True
RequestMFAFromClaimsProviders       : False
ScopeGroupId                        :
Name                                : SURFconext
AutoUpdateEnabled                   : True
MonitoringEnabled                   : True
MetadataUrl                         :
              https://engine.test.surfconext.nl/authentication/sp/metadata
ConflictWithPublishedPolicy         : False
IssuanceAuthorizationRules          :
IssuanceTransformRules              : @RuleTemplate = "LdapClaims"
                                      @RuleName = "AD attributen"
                                      c:[Type ==
     "http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccou
                                      ntname", Issuer == "AD AUTHORITY"]
                                       => issue(store = "Active
                    Directory", types = ("urn:mace:dir:attribute-def:uid",
                                  "urn:mace:dir:attributedef:displayName",
                                    "urn:mace:dir:attribute-def:mail",
   "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier",
                               "urn:mace:dir:attribute-def:employeeNumber"),
                                                                query =
  ";sAMAccountName,displayName,mail,userPrincipalName,employeeNumber;{0}",
                                                          param = c.Value);
                                      @RuleTemplate = "EmitGroupClaims"
                                      @RuleName = "schacHomeOrganization"
                                                               c:[Type ==
       "http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid",
                    Value == "S-1-5-21-940114010-4055878564-832165062-513",
                                             Issuer == "AD AUTHORITY"]
                                                      => issue(Type =
              "urn:mace:terena.org:attribute-def:schacHomeOrganization",
                                          Value = "hartingcollege.nl",
                   Issuer = c.Issuer, OriginalIssuer = c.OriginalIssuer,
                                             ValueType = c.ValueType);
```

```
                                        @RuleTemplate = "EmitGroupClaims"
                                        @RuleName = "eduPersonAffiliation
                                                              (employee)"
                                                               c:[Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid", Value
         == "S-1-5-21-940114010-4055878564-832165062-3601", Issuer == "AD
                                                               AUTHORITY"
                                                                      ]
                                                    => issue(Type =
   "urn:mace:dir:attribute-def:eduPersonAffiliation", Value = "employee",
        Issuer = c.Issuer, OriginalIssuer = c.OriginalIssuer, ValueType =
                                                          c.ValueType);

                                        @RuleTemplate = "EmitGroupClaims"
                                                               @RuleName =
                                         "eduPersonAffiliation (student)"
                                                               c:[Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid", Value
                          == "S-1-5-21-940114010-4055878564-832165062-3602",
                                                  Issuer == "AD AUTHORITY"
                                                                      ]
                                                    => issue(Type =
    "urn:mace:dir:attribute-def:eduPersonAffiliation", Value = "student",
Issuer = c.Issuer, OriginalIssuer = c.OriginalIssuer, ValueType = c.Value
                                                                  Type);
                                        @RuleTemplate = "EmitGroupClaims"
                                                               @RuleName =
                                       "eduPersonEntitlement (SURFdrive)"
                                                               c:[Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid", Value
                           == "S-1-5-21-940114010-4055878564-832165062-513",
                                                 Issuer == "AD AUTHORITY"]
                                                                => issue(
                  Type = "urn:mace:dir:attribute-def:eduPersonEntitlement",
                       Value = "urn:x-surfnet:surf.nl:surfdrive:quota:100",
        Issuer = c.Issuer, OriginalIssuer = c.OriginalIssuer, ValueType =
                                                          c.ValueType);

                                                               @RuleName =
                                                  "schacPersonalUniqueCode"
                                                               c:[Type ==
                                    "urn:mace:dir:attribute-def:employeeNumber"]
                                                                => issue(
       Type = "urn:schac:attribute-def:schacPersonalUniqueCode", Value =
  "urn:schac:personalUniqueCode:nl:local:hartingcollege.nl:employeeid:" +
                                                               c.Value);

                                                  @RuleName = "ImmutableID"
                                                               c:[Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountnam
                                                                      e"]
                                                                => issue(
                                               store = "Active Directory",
```

```
                        types = ("http://schemas.xmlsoap.org/claims/UPN",
    "http://schemas.microsoft.com/LiveID/Federation/2008/05/ImmutableID"),
               query = "samAccountName={0};userPrincipalName,objectGUID;{1}",
              param = regexreplace(c.Value, "(?<domain>[^\\]+)\\(?<user>.+)",
                                            "${user}"), param = c.Value);


DelegationAuthorizationRules         :
LastPublishedPolicyCheckSuccessful   : True
LastUpdateTime                       : 20-11-2018 11:47:39
LastMonitoredTime                    : 1-4-2019 12:11:01
ImpersonationAuthorizationRules      :
AdditionalAuthenticationRules        :
AccessControlPolicyName              : Permit everyone and require MFA for
specific group
AccessControlPolicyParameters        : {GroupParameter}
ResultantPolicy                      : RequireFreshAuthentication:False
                                       IssuanceAuthorizationRules:
                                       {
                                         Permit everyone
                                         except
                                          from 'HARTINGCOLLEGE\MFA' group;

                                         Permit users
                                           from 'HARTINGCOLLEGE\MFA' group
                                                    and when authentication
                                                                includes MFA
                                       }
```