Microsoft

SharePoint

Moodle and Office 365 Step-by-Step Guide: Federation using Active Directory Federation Services

Moodle and Office 365 Step-by-Step Guide: Federation using Active Directory Federation Services

Adam Bradley, Architect
Microsoft Corporation
June 2013

Applies to: Office 365 | AD FS 2.0 | Windows Azure Active Directory

Summary: This guide walks you through the setup of a basic lab deployment of Moodle, Active Directory Federation Services (AD FS) 2.0, and Windows Azure Active Directory to perform cross-product, browser-based identity federation. This setup supports a federated single sign-on (SSO) experience for Moodle and Office 365, in addition to user autoprovisioning, and user auto enrollment in Moodle through Office 365.

Contents

About this guide

This guide provides step-by-step instructions for configuring a basic identity federation deployment between Moodle and Office 365. This deployment uses federated identities and leverages the capabilities of Microsoft® Active Directory® Federation Services 2.0 (AD FS 2.0), the Security Assertion Markup Language (SAML) 2.0 protocol (http://go.microsoft.com/fwlink/?LinkId=193996), and SAML 2.0 HTTP POST binding (provided by the SimpleSAMLPHP authentication framework).

Objectives for this project:

- Provide an Office 365 single sign-on experience for a Federated namespace via the AD FS Secure Token Service (STS) where AD FS acts as the Identity Provider (IdP), and STS and Moodle (via SimpleSAMLPHP) both act as the Service Provider (SP)
- Use native Moodle extensions where possible
- Support User autoprovisioning in Moodle
- Support Course auto enrollment in Moodle

Terminology used in this guide

Throughout this document, there are numerous references to federation concepts that are called by different names in the Microsoft and Shibboleth products. The following table assists in drawing parallels between the two vendors' technologies.

Table 1. Terminology differences

AD FS 2.0 name	Shibboleth name	Concept
Security token	Assertion	An XML document that is created and sent during a federated access request that describes a user
Claims provider	Identity provider (IdP)	A partner in a federation that creates security tokens for users
Relying party	Service provider (SP)	A partner in a federation that consumes security tokens to provide access to applications
Claims	Assertion attributes	Data about users that is sent inside security tokens

In this deployment, each product performs both the claims provider/identity provider role and the relying party/service provider role.

About the author

Adam Bradley (abradley@microsoft.com) is an Office 365 Architect for Microsoft.

Prerequisites and other requirements

This lab assumes that you have an existing deployment of Office 365, with AD FS 2.0 configured to support federated authentication (single sign-on). Follow the guidance provide at http://technet.microsoft.com/en-us/library/jj151794.aspx.

AD FS 2.0

AD FS 2.0 assumes the role of Identity Provider and Security Token Service and will handle login requests that follow the WS-Fed (Active), WS-Trust (Passive), and SAMLP standards.

This deployment uses a wildcard SSO certificate to provide a cost effective way of securing a number of different services with a single certificate.

Moodle

Use the Microsoft Web Platform Installer to install Moodle along with the other prerequisites, including MySQL and PHP. Be sure that you have also made these modifications:

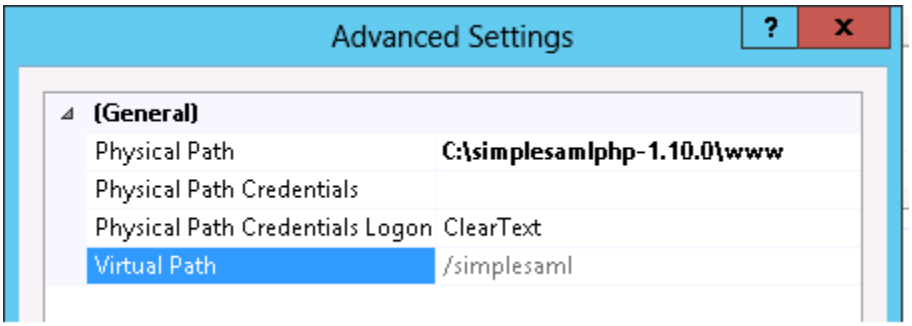
- Update the server URL and configure it to use HTTPS.
- Enable both the SAML Authentication and SAML Enrolment modules (covered later in this document).

SimpleSAMLPHP

This deployment uses the SimpleSAMLPHP framework to extend the authentication protocol abilities of Moodle and provide SAML2 protocol support. SimpleSAMLPHP acts as a Service Provider and is configured to send authentication requests to a remote SAMLP Identity Provider. Follow these steps to install and configure SimpleSAMLPHP.

1. Download the latest version of SimpleSAMLPHP. At the time of writing, this was simplesamlphp-1.10.0.tar.gz from http://code.google.com/p/simplesamlphp/downloads. Unzip the download to a secure location.

2. In Internet Information Services (IIS) Manager, map a Virtual Directory "/simplesaml" to the "www" directory inside the unzipped download. Ensure IIS has rights to this directory.



3. Update the SimpleSAMLPHP Service Provider configuration in the *config/authsources.php* file. The updated configuration should look like the one in this example.

```
'default-sp' => array(
'saml:SP',
// The entity ID of this SP. Can be NULL/unset, in which case an entity
ID is generated based on the metadata URL.
'entityID' => NULL,
// The entity ID of the IdP this should SP should contact.
'idp' => 'http://idp.contoso.com/adfs/services/trust',
// The URL to the discovery service.
'discoURL' => NULL,
// NameIDPolicy must be unspecified for ADFS
'NameIDPolicy' => 'urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified',
'simplesaml.nameidattribute' => 'email',
```

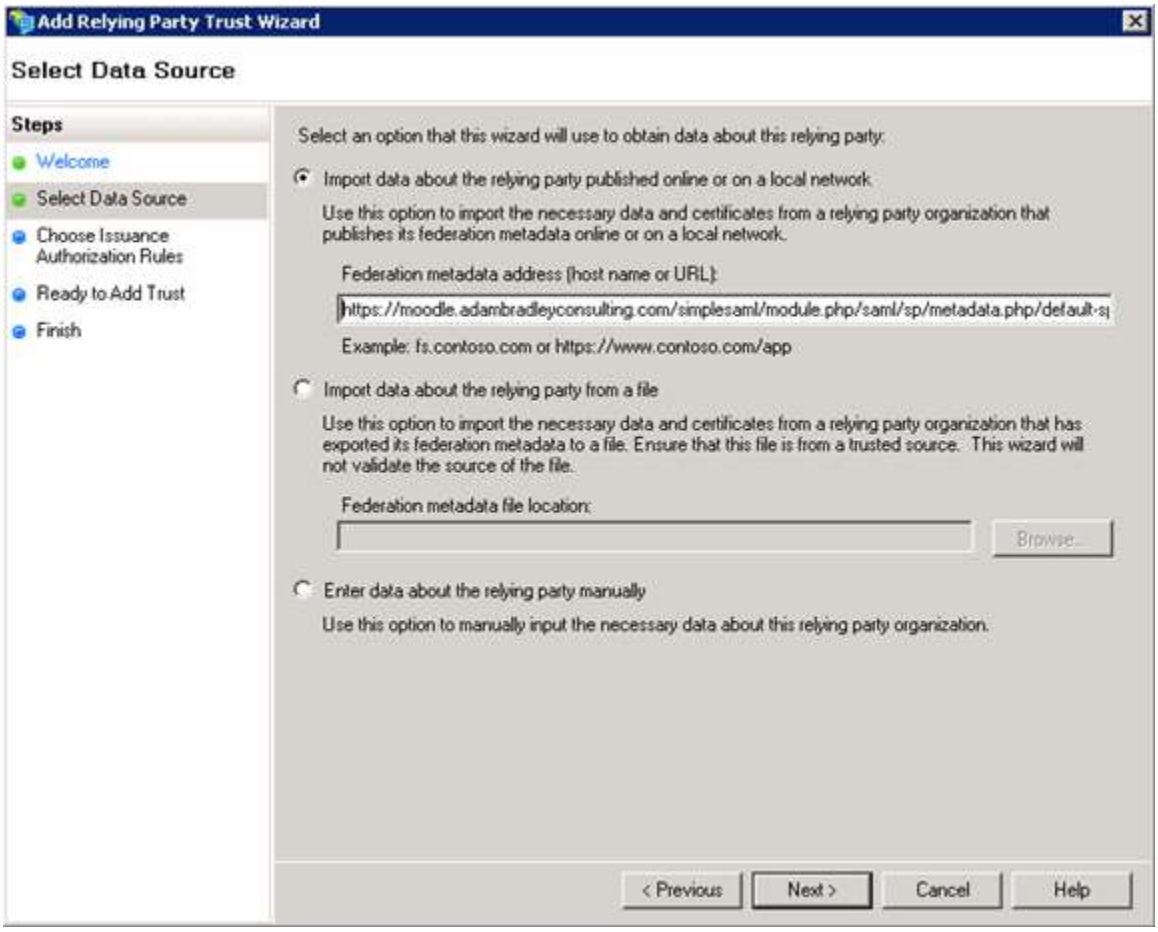
   For more information on how to configure SimpleSAMLPHP, see http://simplesamlphp.org/docs/stable/saml:sp.

4. Configure the Identity Provider to be used with this Service Provider. Update the *metadata/saml20-idp-remote.php* file. The certFingerprint value is the certificate thumbprint of the AD FS Token-signing certificate. The updated configuration should look like the one in this example:

```
$metadata['http://idp.contoso.com/adfs/services/trust'] = array(
'name' => array(
'en' => 'ADFS IdP',
),
'description'          => 'Here you can login with your account on the
Active Directory network.',
'SingleSignOnService'  => 'https://idp.contoso.com/adfs/ls',
'SingleLogoutService'  =>
'https://idp.consoso.com/adfs/ls/?wa=wsignout1.0&wreply=https://idp.con
toso.com/adfs/ls/?wa=wsignoutcleanup1.0',
'certFingerprint' => '571f9d649a950280de1b25f7c1259bf84ff7501d',
);
```

5. When the Service Provider is configured, import the Service Provider Metadata. The URL for this is available from the Administrative Console “Federation” tab.

6. Add SimpleSAMLPHP as a relying party in AD FS 2.0 using the standard SAML Metadata.



7. Finalize the configuration of AD FS 2.0 to communicate with SimpleSAMLPHP:

    a. In the Relying Party Trust relationship, set the advanced setting to use the SHA-1 secure hash algorithm.

    b. Switch off claims encryption in AD FS by using these PowerShell cmdlets:
    Add-PSSnapin Microsoft.Adfs.PowerShell
    Set-ADFSRelyingPartyTrust -TargetName "SimpleSAMLPHP SP" -EncryptClaims $False

    c. Add the necessary claim mappings by importing from the file containing the claim mappings. See **Error! Reference source not found.** at the end of this document for the omplete list of claims.

Moodle Plugins

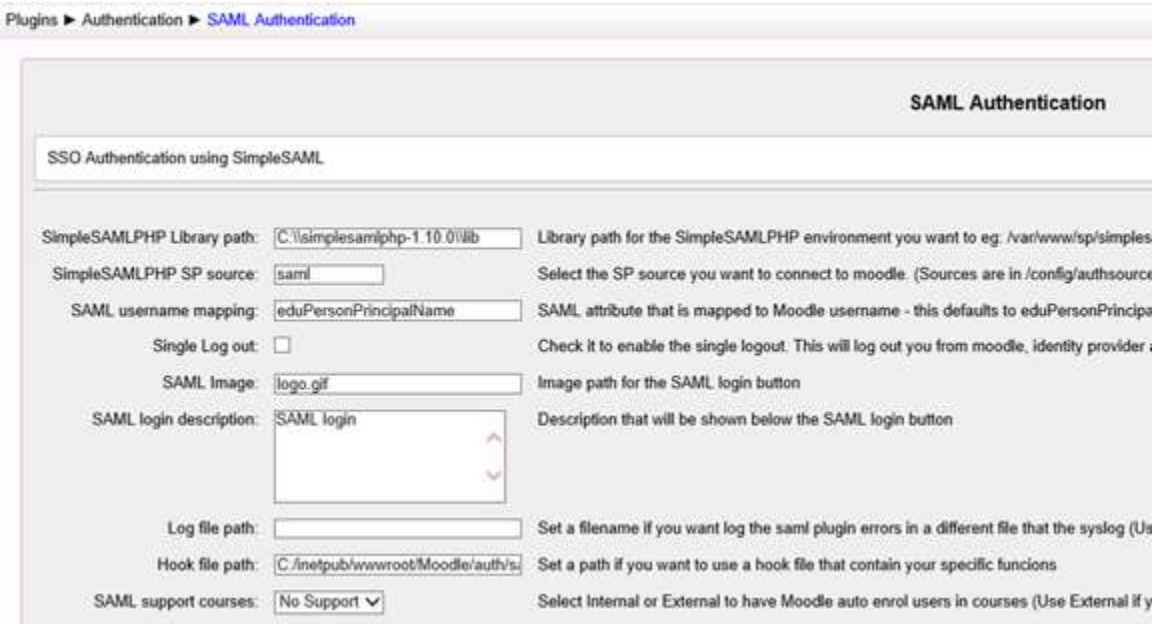
The Moodle community provides integration and product enhancements for all aspects of the product, including the areas of authentication and enrollment.

To provide support for single sign-on and automated account provisioning in Moodle, use the SAMLP Authentication plugin. If you're interested in automatic course enrollment based on your Active Directory group membership, use the SAMLP Enrollment plugin.

Both of these plugins use the SimpleSAMLPHP framework as the underlying mechanism to handle the SAMLP login flows.

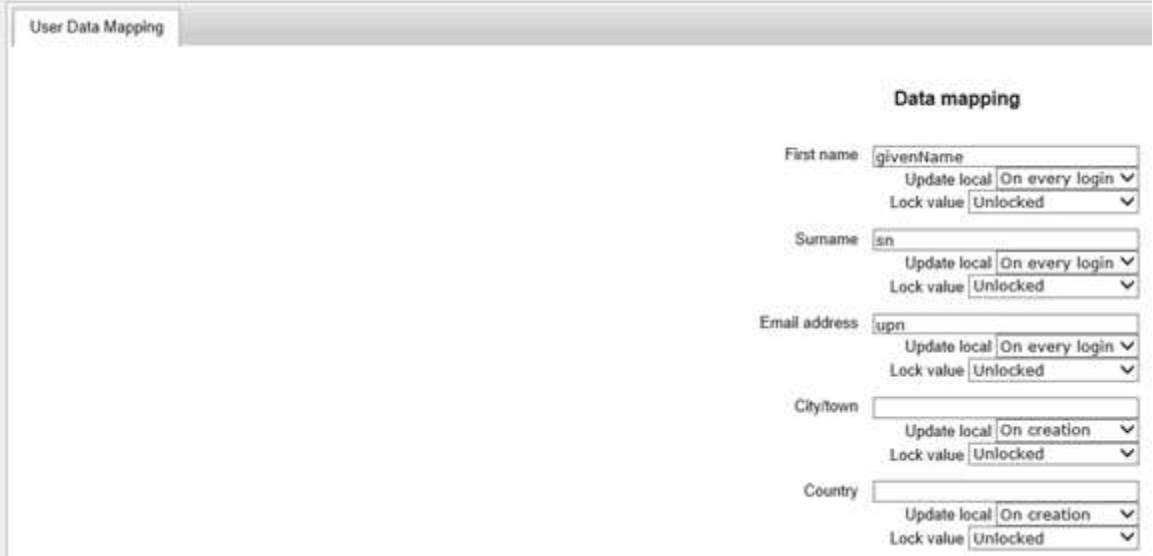
SAML Authentication Plugin

The SAML Authentication Plugin can be downloaded from the Moodle Wiki and installed by following the instructions available at https://moodle.org/plugins/view.php?plugin=auth_saml. This plugin allows for the automated creation of accounts in Moodle when users are authenticated with a valid SAMLP claim.



In the Site Administration interface, enable and configure the SAML Authentication Plugin. On a Windows platform, the path to the SimpleSAMLPHP library must include the double backslashes (\\) as shown in the **SimpleSAMLPHP Library Path** entry.

You can also configure User Data Mapping (which populates the Moodle User profile with values from the SAML Attributes in the incoming claim). Ensure that all of the attributes you want to include in the Moodle User profile are added to the incoming claim (as configured in AD FS 2.0).

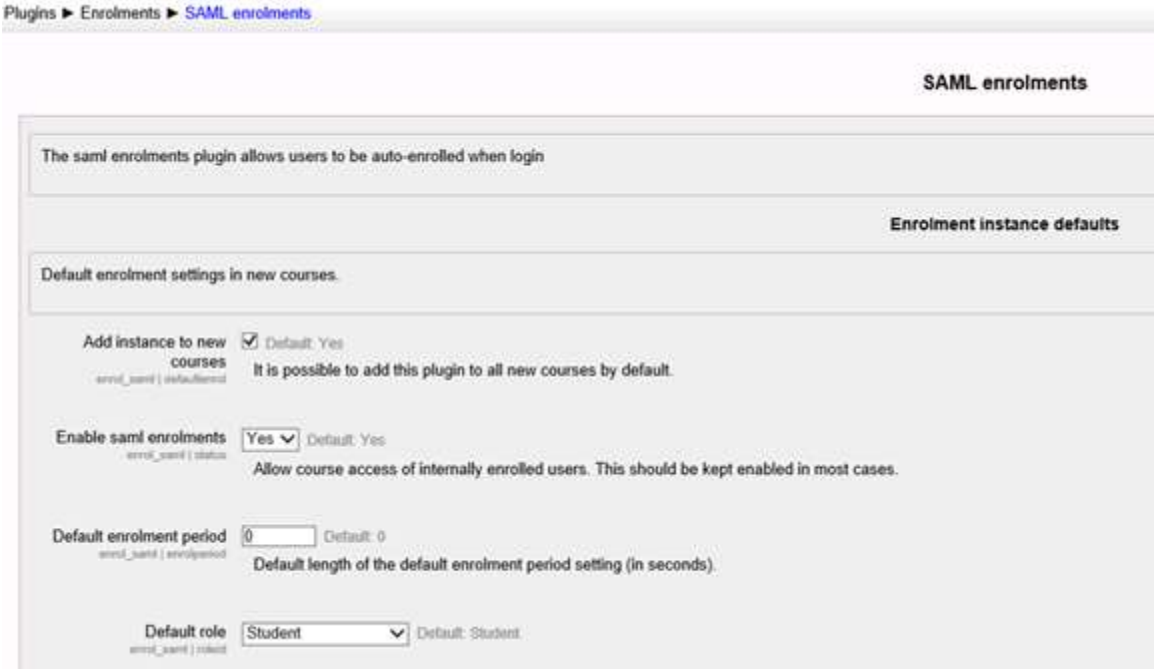


SAML Enrolment Plugin

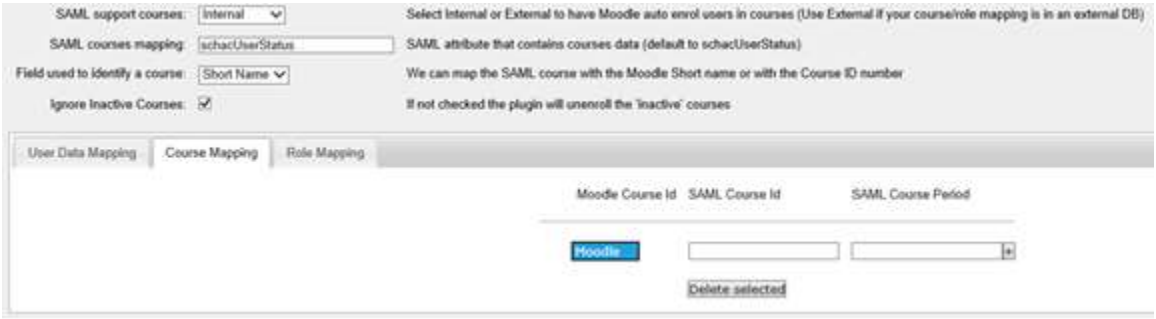
The SAML Enrolment Plugin can be downloaded from the Moodle Wiki and installed by following the instructions available at https://moodle.org/plugins/view.php?plugin=enrol_saml. This plugin depends on the implementation of the SAML Authentication Plugin, and enables the user to be automatically enrolled in Moodle courses based on the SAML `schacUserStatus` attribute.

Full documentation for this setup, which does the bulk of the enrollment work, is available in the SAML Authentication Plugin Documentation (https://github.com/pitbulk/moodle_saml/blob/master/auth/saml/moodle_auth_saml.txt).

You configure this component in the SAML Authentication Plugin. Only a small amount of configuration is needed after you enable the plugin.



The user's Active Directory Group membership must be sent in the SAML schacUserStatus attribute for the Plugin to process SAML enrollments automatically. All course and role mappings must be added to the SAML Authentication Plugin.



Appendix A: Using AD FS 2.0 with SimpleSAMLPHP – Claim mapping rules

To apply the following claim mapping rules to the SimpleSAMLPHP SP, copy the list at the end of this document to a file, and then use the Set-AdfsRelyingPartyTrust Powershell command to import, as in this example.

```
Add-PSSnapin Microsoft.Adfs.PowerShell
Set-AdfsRelyingPartyTrust -TargetName "SimpleSAMLPHP SP" -
IssuanceTransformRulesFile "c:\SimpleSAMLPHP-issuance-transformation-
rules.txt"

---snip---
@RuleTemplate = "LdapClaims"
@RuleName = "Default"
c:[Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/windowsaccountname",
Issuer == "AD AUTHORITY"]
 => issue(store = "Active Directory", types =
("http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn",
"http://schemas.xmlsoap.org/claims/Group",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname",
"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname"), query =
";userPrincipalName,userPrincipalName,tokenGroups,mail,givenName,sn;{0}",
param = c.Value);

@RuleName = "Transform UPN to epPN"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
 => issue(Type = "urn:oid:1.3.6.1.4.1.5923.1.1.1.6", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "Transform Group to epSA"
c:[Type == "http://schemas.xmlsoap.org/claims/Group", Value == "Domain
Users"]
 => issue(Type = "urn:oid:1.3.6.1.4.1.5923.1.1.1.9", Value =
"member@contoso.com",
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "CN"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
 => issue(Type = "cn", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "mail"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/email"]
 => issue(Type = "mail", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");
```

```
@RuleName = "eduPersonPrincipalName"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
 => issue(Type = "eduPersonPrincipalName", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "UID"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
 => issue(Type = "uid", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "UPN"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname"]
 => issue(Type = "UPN", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "GivenName"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname"]
 => issue(Type = "givenName", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");

@RuleName = "Surname"
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname"]
 => issue(Type = "sn", Value = c.Value,
Properties["http://schemas.xmlsoap.org/ws/2005/05/identity/claimproperties/at
tributename"] = "urn:oasis:names:tc:SAML:2.0:attrname-format:uri");
---snip---
```