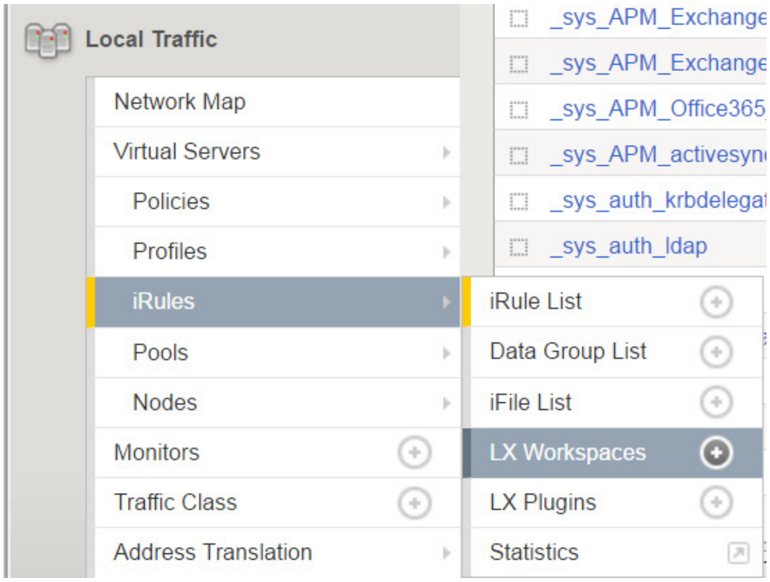
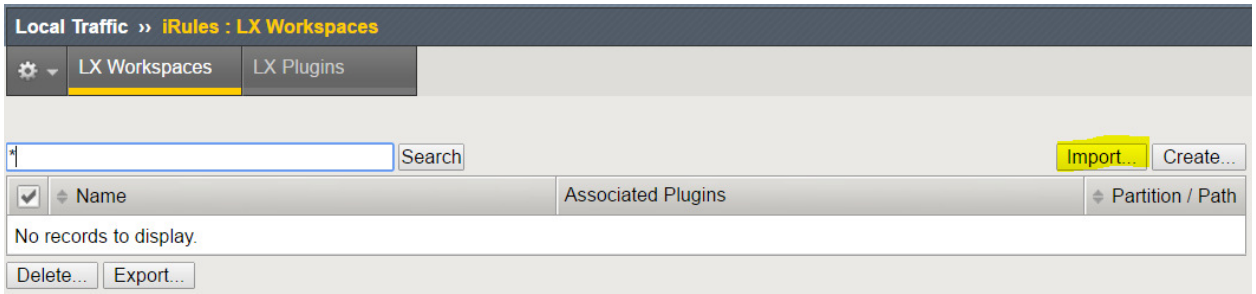
F5 ADFS iRulesLX Implementation

2/10/2017

1. First you will need to import the iRuleLX Workspace into the BIG-IP.

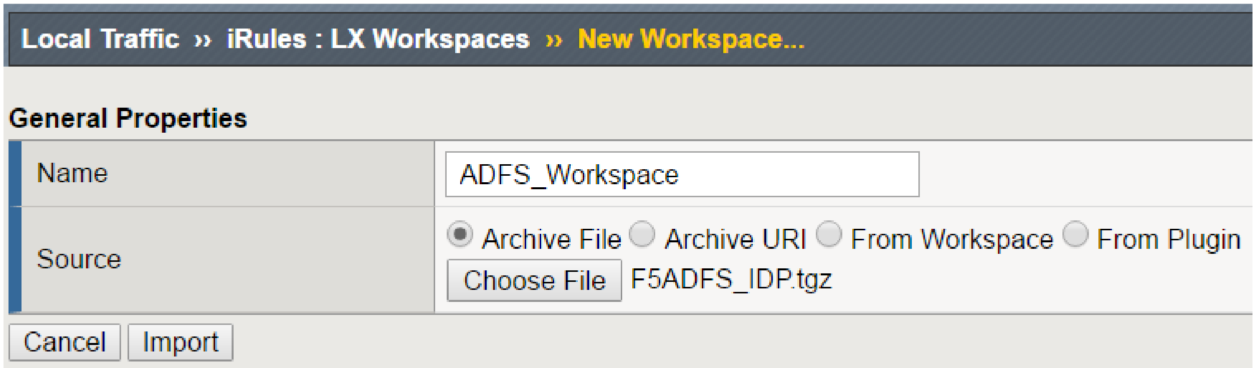




Search: *

No records to display.

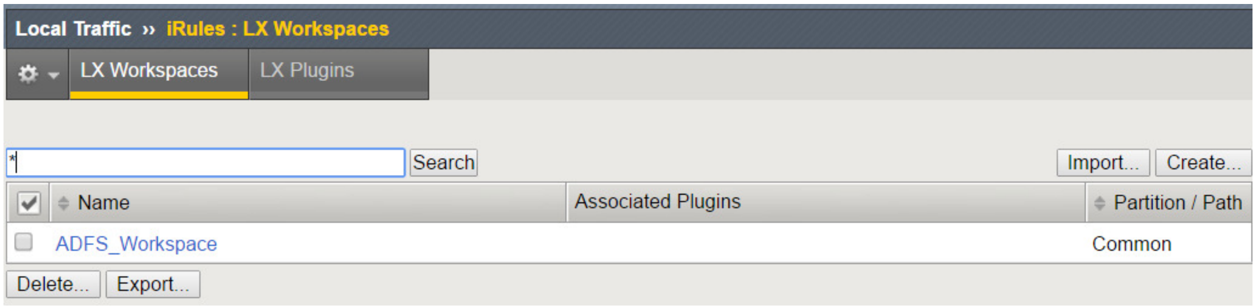
2. Give the workspace a name and chose the archive file to import and click on “Import”



Name: ADFS_Workspace

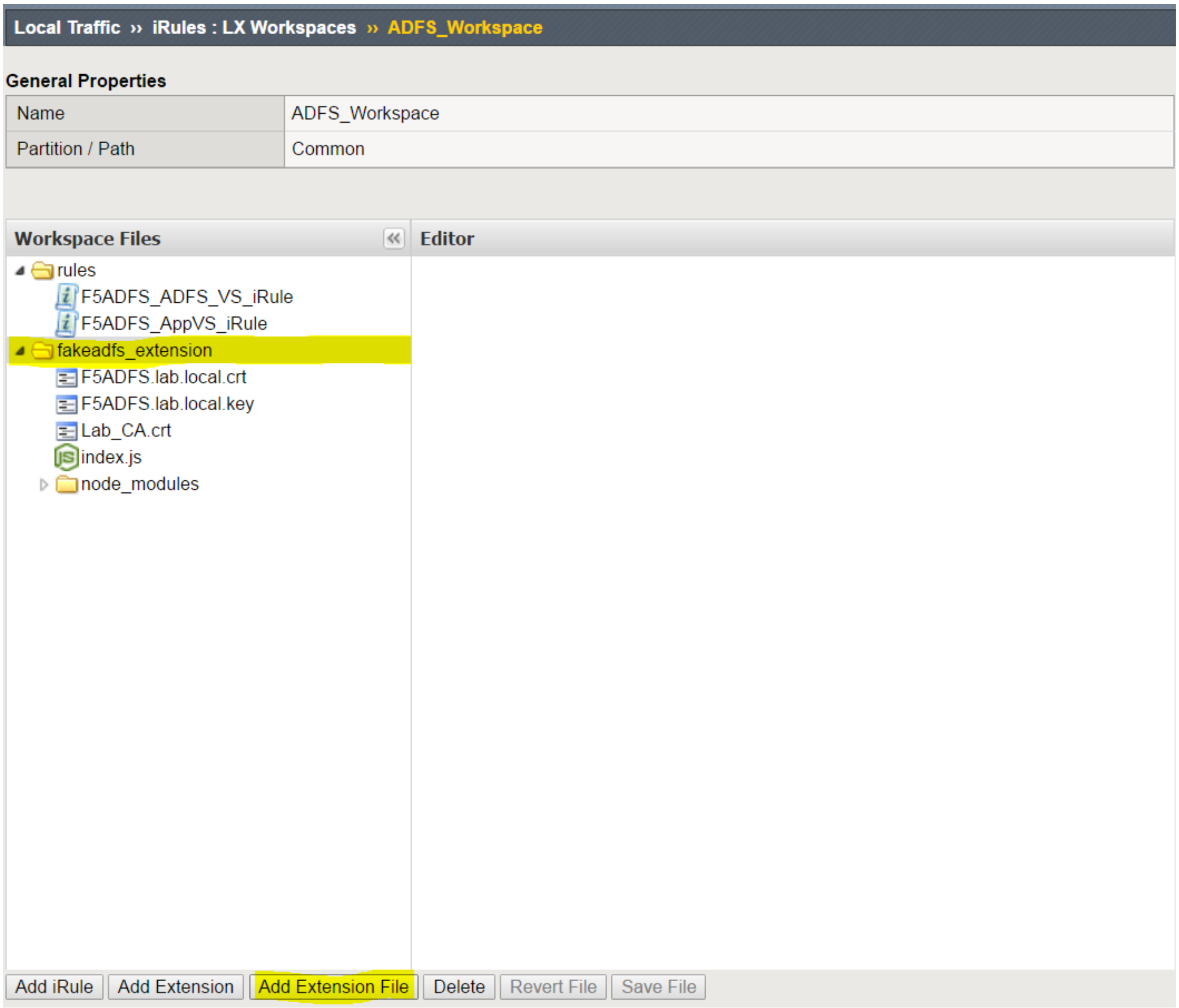
Source: ● Archive File ○ Archive URI ○ From Workspace ○ From Plugin

F5ADFS_IDP.tgz



☑	Name	Associated Plugins	Partition / Path
□	ADFS_Workspace		Common

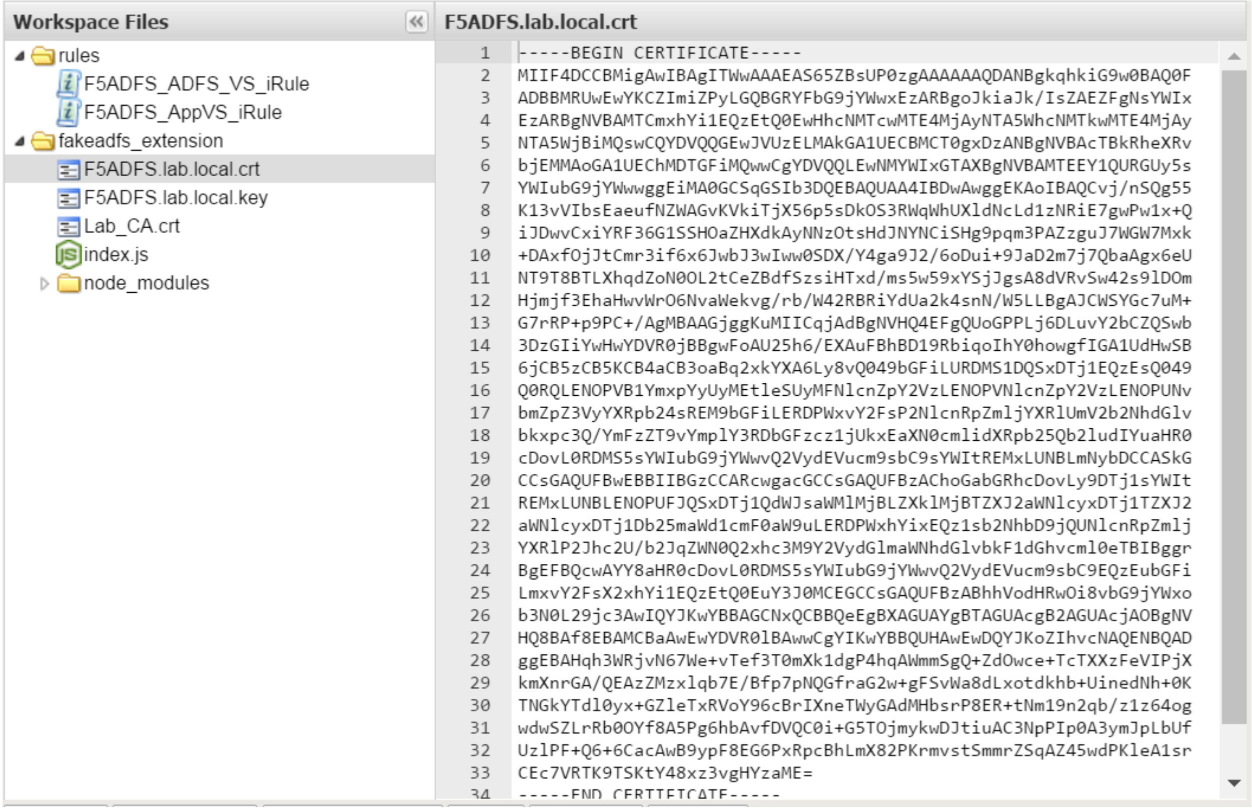
3. Now that you have the Workspace imported you will need to modify it for your environment. Click on the Workspace to open it.
4. Now you need to create new Certificate and Key files and paste your certificate and key into the respective files. Select the fakeadfs_extension in the left pane and then click “Add Extension File” at the bottom of the window to create the new file. You will need two new files, one for Certificate and one for Key. The name of the files are not critical as you will modify the reference in the iRulesLX code in a later step.



General Properties

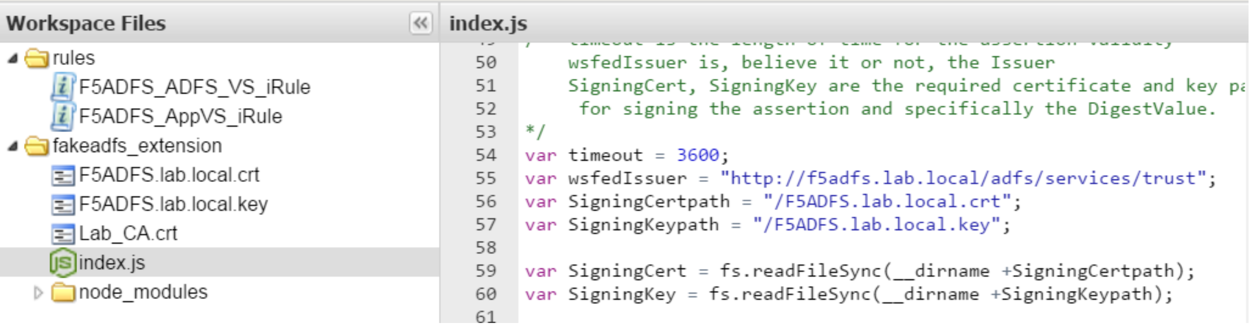
Name	ADFS_Workspace
Partition / Path	Common

5. Select the file that you created and paste in the text from your certificate and key into their respective file. (NOTE: If you are using Internet Explorer and cannot paste into the file, use a different browser. I have seen issues with locked down IE browsers not being able to paste here) After you have the text pasted into the file you will need to click the “Save File” button at the bottom of the window.



```
-----BEGIN CERTIFICATE-----
MIIF4DCCBMigAwIBAgITWwAAAEAS65ZBsUP0zgAAAAAAQDANBgkqhkiG9w0BAQ0F
ADBBMRUwEwYKCZImiZPyLGQBGRYFbG9jYWwxEzARBgoJkiaJk/IsZAEZFgNsYWIx
EzARBgNVBAMTCmxhYi1EQzEtQ0EwHhcNMTcwMTE4MjAyNTA5WhcNMTkwMTE4MjAy
NTA5WjBiMQswCQYDVQQGEwJVUzELMAkGA1UECBMCT0gxDzANBgNVBAcTBkRheXRv
bjEMMAoGA1UEChMDTGFiMQwwCgYDVQQLEwNMYWIxGTAXBgNVBAMTEEY1QURGUy5s
YWIubG9jYWwwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCvj/nSQg55
K13vVIbsEaeufNZWAGvKVkiTjX56p5sDkOS3RWqWhUXldNcLd1zNRiE7gwPw1x+Q
iJDwvCxiYRF36G1SSHOaZHXdkAyNNzOtsHdJNYNCiSHg9pqm3PAZzguJ7WGW7Mxk
+DAxfOjJtCmr3if6x6JwbJ3wIww0SDX/Y4ga9J2/6oDui+9JaD2m7j7QbaAgx6eU
NT9T8BTLXhqdZoN0OL2tCeZBdfSzsiHTxd/ms5w59xYSjJgsA8dVRvSw42s9lDOm
Hjmjf3EhaHwvWrO6NvaWekvg/rb/W42RBRiYdUa2k4snN/W5LLBgAJCWSYGc7uM+
G7rRP+p9PC+/AgMBAAGjggKuMIICqjAdBgNVHQ4EFgQUoGPPLj6DLuvY2bCZQSwb
3DzGIiYwHwYDVR0jBBgwFoAU25h6/EXAuFBhBD19RbiqoIhY0howgfIGA1UdHwSB
6jCB5zCB5KCB4aCB3oaBq2xkYXA6Ly8vQ049bGFiLURDMS1DQSxDTj1EQzEsQ049
Q0RQLENOPVB1YmxpYyUyMEtleSUyMFNlcnZpY2VzLENOPVNlcnZpY2VzLENOPUNv
bmZpZ3VyYXRpb24sREM9bGFiLERDPWxvY2FsP2NlcnRpZmljYXRlUmV2b2NhdGlv
bkxpc3Q/YmFzZT9vYmplY3RDbGFzcz1jUkxEaXN0cmlidXRpb25Qb2ludIYuaHR0
cDovL0RDMS5sYWIubG9jYWwvQ2VydEVucm9sbC9sYWItREMxLUNBLmNybDCCASkG
CCsGAQUFBwEBBIIBGzCCARcwgacGCCsGAQUFBzAChoGabGRhcDovLy9DTj1sYWIt
REMxLUNBLENOPUFJQSxDTj1QdWJsaWMlMjBLZXklMjBTZXJ2aWNlcyxDTj1TZXJ2
aWNlcyxDTj1Db25maWd1cmF0aW9uLERDPWxhYixEQz1sb2NhbD9jQUNlcnRpZmlj
YXRlP2Jhc2U/b2JqZWN0Q2xhc3M9Y2VydGlmaWNhdGlvbkF1dGhvcml0eTBIBggr
BgEFBQcwAYY8aHR0cDovL0RDMS5sYWIubG9jYWwvQ2VydEVucm9sbC9EQzEubGFi
LmxvY2FsX2xhYi1EQzEtQ0EuY3J0MCEGCCsGAQUFBzABhhVodHRwOi8vbG9jYWxo
b3N0L29jc3AwIQYJKwYBBAGCNxQCBBQeEgBXAGUAYgBTAGUAcgB2AGUAcjAOBgNV
HQ8BAf8EBAMCBaAwEwYDVR0lBAwwCgYIKwYBBQUHAwEwDQYJKoZIhvcNAQENBQAD
ggEBAHqh3WRjvN67We+vTef3T0mXk1dgP4hqAWmmSgQ+ZdOwce+TcTXXzFeVIPjX
kmXnrGA/QEAzZMzxlqb7E/Bfp7pNQGfraG2w+gFSvWa8dLxotdkhb+UinedNh+0K
TNGkYTdl0yx+GZleTxRVoY96cBrIXneTWyGAdMHbsrP8ER+tNm19n2qb/z1z64og
wdwSZLrRb0OYf8A5Pg6hbAvfDVQC0i+G5TOjmykwDJtiuAC3NpPIp0A3ymJpLbUf
UzlPF+Q6+6CacAwB9ypF8EG6PxRpcBhLmX82PKrmvstSmmrZSqAZ45wdPKleA1sr
CEc7VRTK9TSKtY48xz3vgHYzaME=
-----END CERTIFICATE-----
```

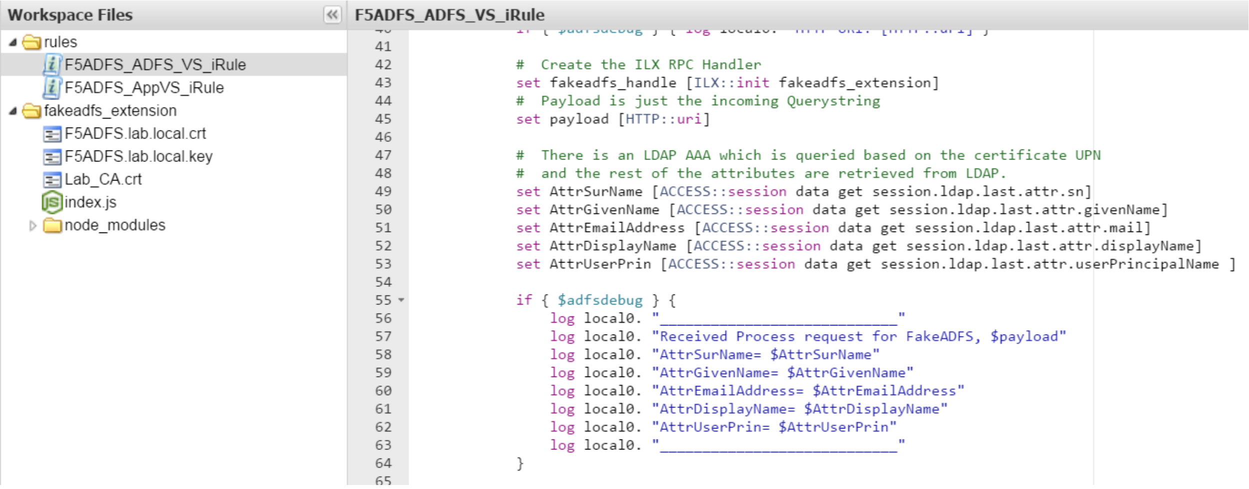
6. Now select the “index.js” file in the left pane. Scroll down in the right window until you find the variable assignments for wefedIssuer, SigningCertPath, and SigningKeyPath. Update the file names for the Certificate and Key and change the FQDN for the wsFedIssuer to match the FQDN you will use for the ADFS Virtual Server. Then click the “Save File” button.



```
/    timeout is the length of time for the assertion validity
     wsfedIssuer is, believe it or not, the Issuer
     SigningCert, SigningKey are the required certificate and key pa
      for signing the assertion and specifically the DigestValue.
*/
var timeout = 3600;
var wsfedIssuer = "http://f5adfs.lab.local/adfs/services/trust";
var SigningCertpath = "/F5ADFS.lab.local.crt";
var SigningKeypath = "/F5ADFS.lab.local.key";

var SigningCert = fs.readFileSync(__dirname +SigningCertpath);
var SigningKey = fs.readFileSync(__dirname +SigningKeypath);

```

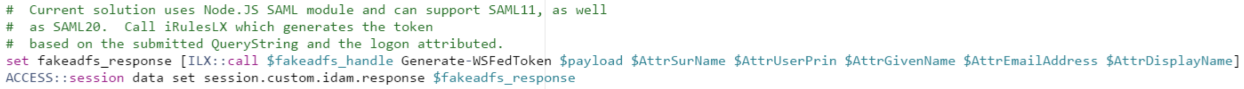
7. You can now delete the demo certificates that were included in the workspace by selecting the file in the left column and clicking the “Delete” button at the bottom.

8. If there are any changes needed to the fields included in the SMAL Claims, add variables here in the F5ADFS_ADFS_VS_iRule. These are pulling values from the LDAP query that is run in the APM Policy. You can also add the variable to the log statements that will be logged if the adfsdebug variable is set to 1 at the top of the iRule. Only use this for troubleshooting as it will generate quite a few log entries for each ADFS Virtual Server access.



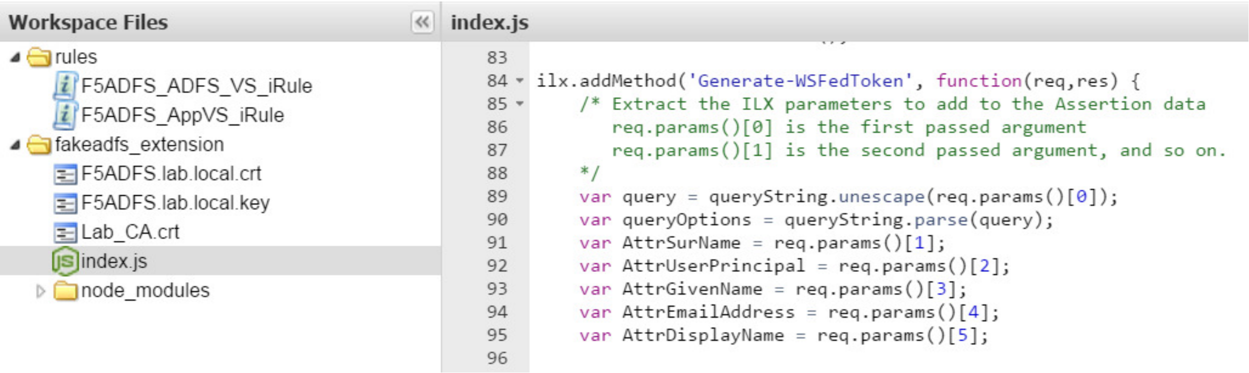
```
        #  Create the ILX RPC Handler
        set fakeadfs_handle [ILX::init fakeadfs_extension]
        #  Payload is just the incoming Querystring
        set payload [HTTP::uri]

        #  There is an LDAP AAA which is queried based on the certificate UPN
        #  and the rest of the attributes are retrieved from LDAP.
        set AttrSurName [ACCESS::session data get session.ldap.last.attr.sn]
        set AttrGivenName [ACCESS::session data get session.ldap.last.attr.givenName]
        set AttrEmailAddress [ACCESS::session data get session.ldap.last.attr.mail]
        set AttrDisplayName [ACCESS::session data get session.ldap.last.attr.displayName]
        set AttrUserPrin [ACCESS::session data get session.ldap.last.attr.userPrincipalName ]

        if { $adfsdebug } {
            log local0. "_____________________________"
            log local0. "Received Process request for FakeADFS, $payload"
            log local0. "AttrSurName= $AttrSurName"
            log local0. "AttrGivenName= $AttrGivenName"
            log local0. "AttrEmailAddress= $AttrEmailAddress"
            log local0. "AttrDisplayName= $AttrDisplayName"
            log local0. "AttrUserPrin= $AttrUserPrin"
            log local0. "_____________________________"
        }
```

9. If you make any changes to the variables, you will also have to update the RPC call for the iRulesLX to include those changes for passing the variables to iRulesLX.



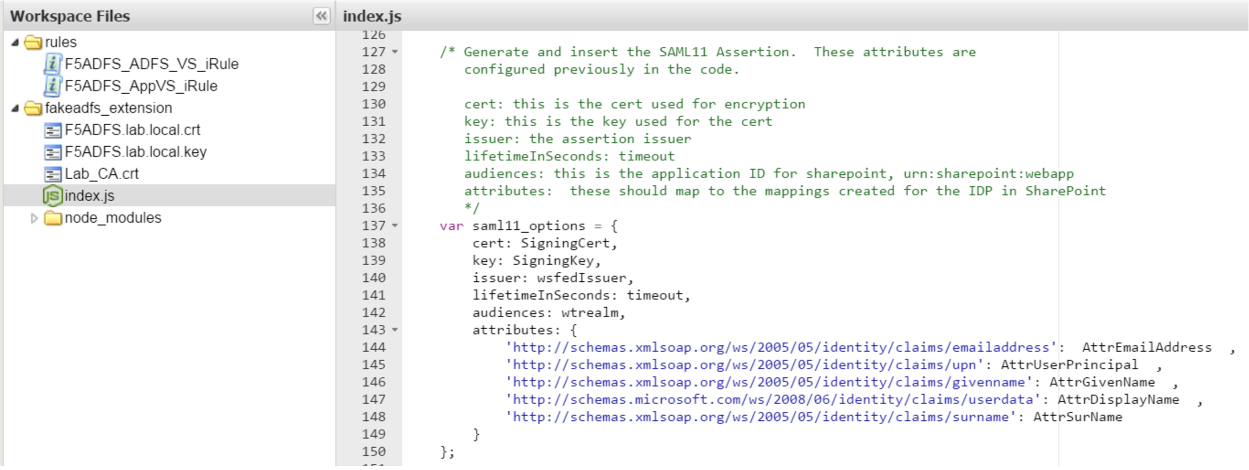
```
#  Current solution uses Node.JS SAML module and can support SAML11, as well
#  as SAML20.  Call iRulesLX which generates the token
#  based on the submitted QueryString and the logon attributed.
set fakeadfs_response [ILX::call $fakeadfs_handle Generate-WSFedToken $payload $AttrSurName $AttrUserPrin $AttrGivenName $AttrEmailAddress $AttrDisplayName]
ACCESS::session data set session.custom.idam.response $fakeadfs_response
```

10. (iRulesLX Code changes to index.js) The parameters passed start with a zero index as denoted by the [0] through [5] below indicating 6 parameters are being passed by the iRule call above. If you added/removed variables you will need to modify the variables in index.js to match the RPC call above.



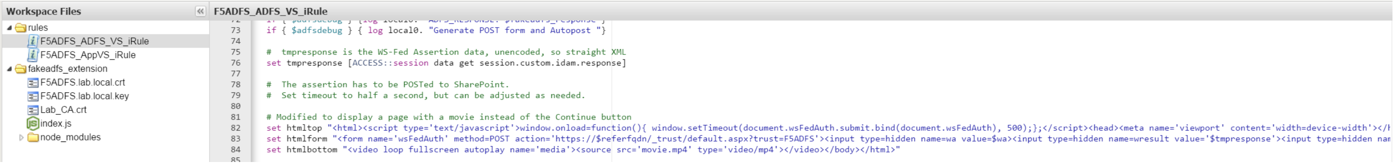
```
ilx.addMethod('Generate-WSFedToken', function(req,res) {
    /* Extract the ILX parameters to add to the Assertion data
       req.params()[0] is the first passed argument
       req.params()[1] is the second passed argument, and so on.
    */
    var query = queryString.unescape(req.params()[0]);
    var queryOptions = queryString.parse(query);
    var AttrSurName = req.params()[1];
    var AttrUserPrincipal = req.params()[2];
    var AttrGivenName = req.params()[3];
    var AttrEmailAddress = req.params()[4];
    var AttrDisplayName = req.params()[5];
```

11. Now you have to add the claim schema and then the iRulesLX variable to the list so it will be added to the SAML/wsfed claim.



```
/* Generate and insert the SAML11 Assertion.  These attributes are
   configured previously in the code.

   cert: this is the cert used for encryption
   key: this is the key used for the cert
   issuer: the assertion issuer
   lifetimeInSeconds: timeout
   audiences: this is the application ID for sharepoint, urn:sharepoint:webapp
   attributes:  these should map to the mappings created for the IDP in SharePoint
   */
var saml11_options = {
    cert: SigningCert,
    key: SigningKey,
    issuer: wsfedIssuer,
    lifetimeInSeconds: timeout,
    audiences: wtrealm,
    attributes: {
        'http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress':  AttrEmailAddress  ,
        'http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn': AttrUserPrincipal  ,
        'http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname': AttrGivenName  ,
        'http://schemas.microsoft.com/ws/2008/06/identity/claims/userdata': AttrDisplayName  ,
        'http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname': AttrSurName
    }
};
```

12. The redirect back to the SAML SP with the token is handled with this code. If you wish to make changes to what is displayed in the web browser while the SAMP SP/Site consumes and processes the ws-fed token, you can change this HTML Code in the F5ADFS_ADFS_VS_iRule.



```
if { $adfsdebug } { log local0. "Generate POST form and Autopost "}

#  tmpresponse is the WS-Fed Assertion data, unencoded, so straight XML
set tmpresponse [ACCESS::session data get session.custom.idam.response]

#  The assertion has to be POSTed to SharePoint.
#  Set timeout to half a second, but can be adjusted as needed.

# Modified to display a page with a movie instead of the Continue button
set htmltop "<html><script type='text/javascript'>window.onload=function(){ window.setTimeout(document.wsFedAuth.submit.bind(document.wsFedAuth), 500);};</script><head><meta name='viewport' content='width=device-width'></h
set htmlform "<form name='wsFedAuth' method=POST action='https://$referfqdn/_trust/default.aspx?trust=F5ADFS'><input type=hidden name=wa value=$wa><input type=hidden name=wresult value='$tmpresponse'><input type=hidden nam
set htmlbottom "<video loop fullscreen autoplay name='media'><source src='movie.mp4' type='video/mp4'></video></body></html>"
```

Example of modified HTML code that removed the Continue Button and now displays a video in the browser while waiting on SAML SP. Removed the following code from the end of the htmlform variable: <input type='submit' value='Continue'>

The name of the trusted provider returned in the redirect is highlighted in green below. If you wish to use a different name change it there.

This example has HTML code added for loading a video file called movie.mp4 from the BIG-IP and playing it on the page while waiting on the website to load. You could add style sheets, java scripts, images, etc. to display whatever you would like to see while waiting.

NOTE: It is also worth pointing out that in testing in both my lab and customer environment that the time for the iRulesLX SAML token processing consistently took less than a second, but the redirect and consumption of the token by SharePoint could take a while and varied dramatically.

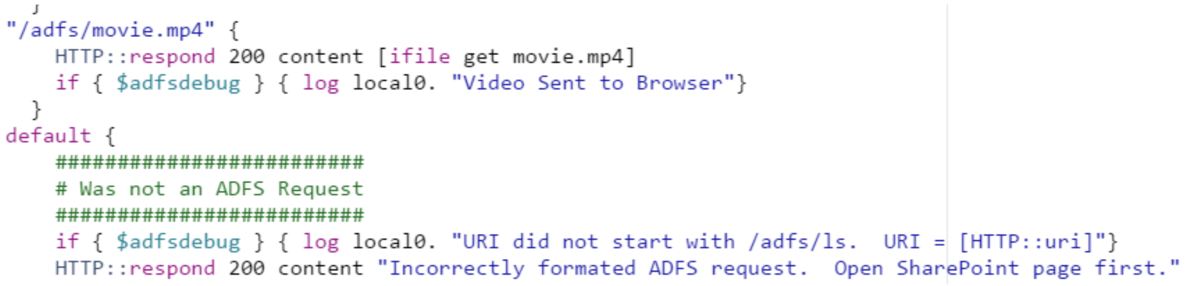
set htmltop "<html><script type='text/javascript'>window.onload=function(){ window.setTimeout(document.wsFedAuth.submit.bind(document.wsFedAuth), 500);};</script><head><meta name='viewport' content='width=device-width'></head><body style='margin: 0px;'>"

set htmlform "<form name='wsFedAuth' method=POST action='https://$referfqdn/_trust/default.aspx?trust=F5ADFS'><input type=hidden name=wa value=$wa><input type=hidden name=wresult value='$tmpresponse'><input type=hidden name=wctx value=$wctx></form/>"

set htmlbottom "<video loop fullscreen autoplay name='media'><source src='movie.mp4' type='video/mp4'></video></body></html>"

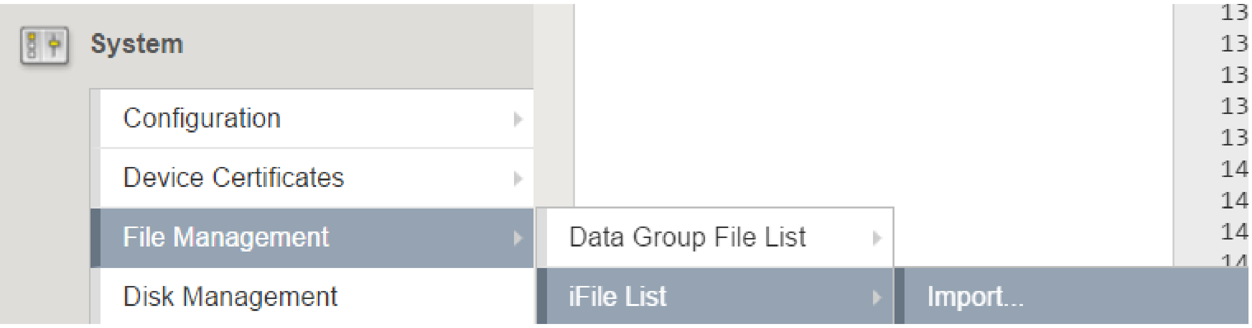
13. Import any supporting files as iFiles into the BIG-IP and then add them to the 'switch' statement in the F5ADFS_ADFS_VS_iRule so the browser can request and load them. You can use this to load CSS style sheets, JS scripts, images, etc.

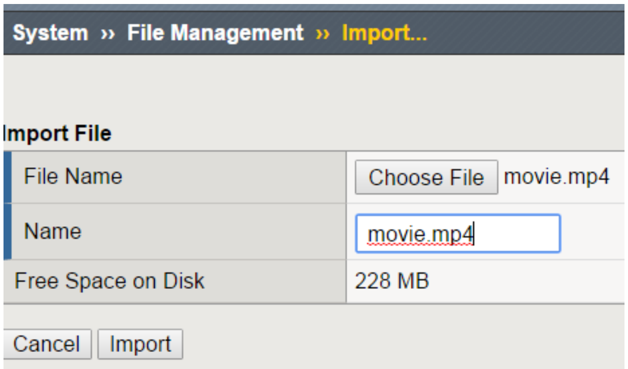
    NOTE: The path for the file is preceded with /adfs/



```
  }
"/adfs/movie.mp4" {
    HTTP::respond 200 content [ifile get movie.mp4]
    if { $adfsdebug } { log local0. "Video Sent to Browser"}
  }
default {
    #########################
    # Was not an ADFS Request
    #########################
    if { $adfsdebug } { log local0. "URI did not start with /adfs/ls.  URI = [HTTP::uri]"}
    HTTP::respond 200 content "Incorrectly formated ADFS request.  Open SharePoint page first."
```

14. Importing a file as an iFile and making it available to iRules:



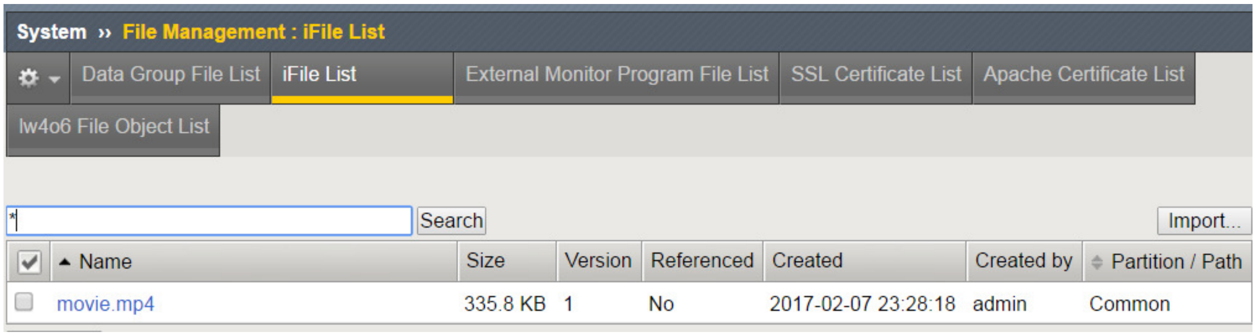


System » File Management » Import...

Import File

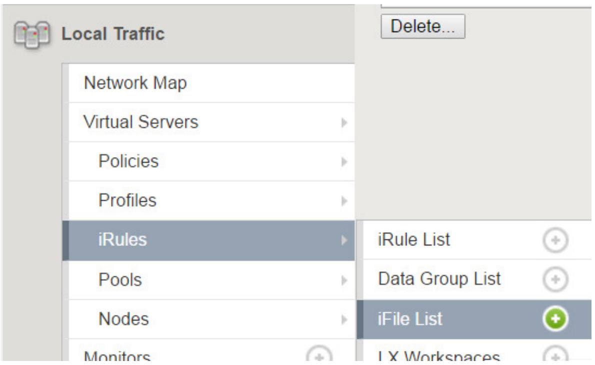
File Name	Choose File movie.mp4
Name	movie.mp4
Free Space on Disk	228 MB

Cancel Import

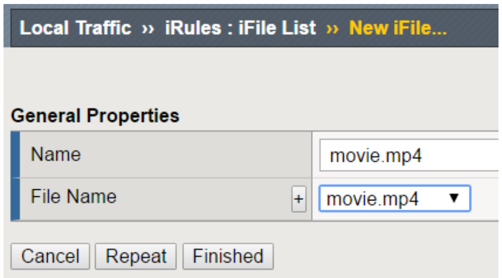


☑	Name	Size	Version	Referenced	Created	Created by	Partition / Path
□	movie.mp4	335.8 KB	1	No	2017-02-07 23:28:18	admin	Common

15. Now you have to make the iFile available to iRules by adding it to the iRules iFile List.

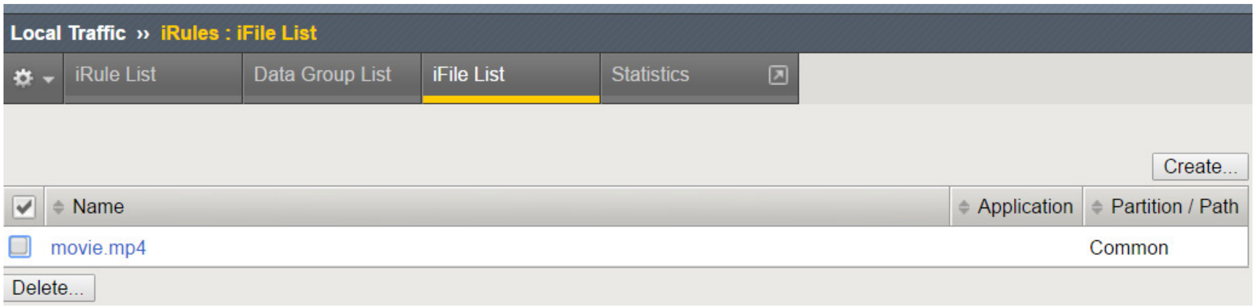


16. The name entered here is what has to be referenced in the HTML code, the switch command, and the ifile get



Name: movie.mp4

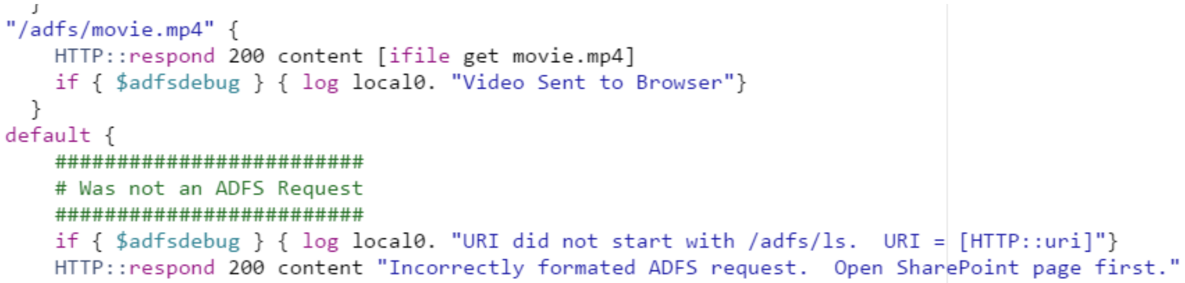
File Name: movie.mp4



☑	Name	Application	Partition / Path
□	movie.mp4		Common

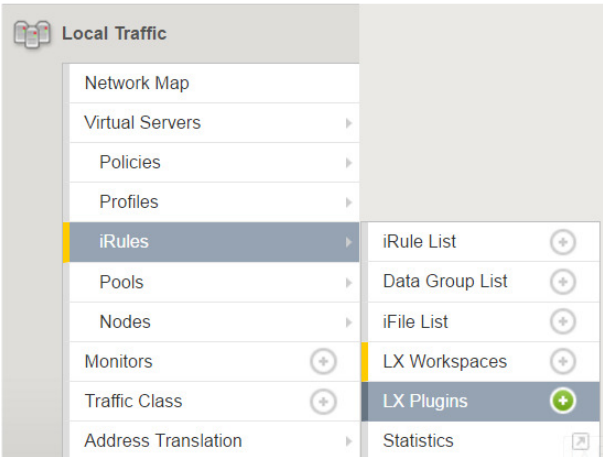
17. The "default" option in the F5ADFS_ADFS_VS_iRule switch statement will display a message indicating that the user needs to access the SharePoint server page first. This check only determines that the HTTP request did not have the path /adfs/ls and was not one of the iFiles that was added to the switch. If the parameters passed to the ADFS server are not correct, but

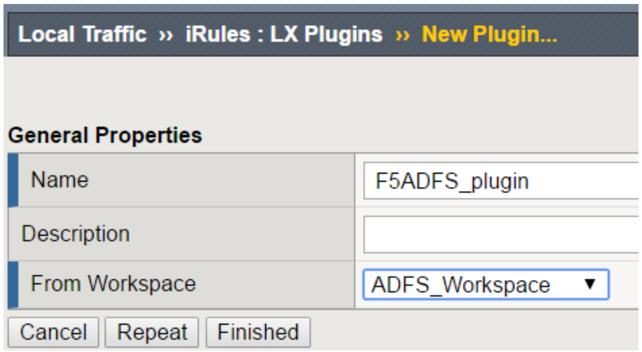
it does have /adfs/ls as the path, it will not trigger this message.



```
    "/adfs/movie.mp4" {
        HTTP::respond 200 content [ifile get movie.mp4]
        if { $adfsdebug } { log local0. "Video Sent to Browser"}
    }
    default {
        ##########################
        # Was not an ADFS Request
        ##########################
        if { $adfsdebug } { log local0. "URI did not start with /adfs/ls.  URI = [HTTP::uri]"}
        HTTP::respond 200 content "Incorrectly formated ADFS request.  Open SharePoint page first."
```

18. Now that the modifications to the iRules and iRulesLX code have been made, we need to create an iRulesLX Plugin from the workspace. This exposes the iRules to LTM so they can be assigned to Virtual Servers, etc.

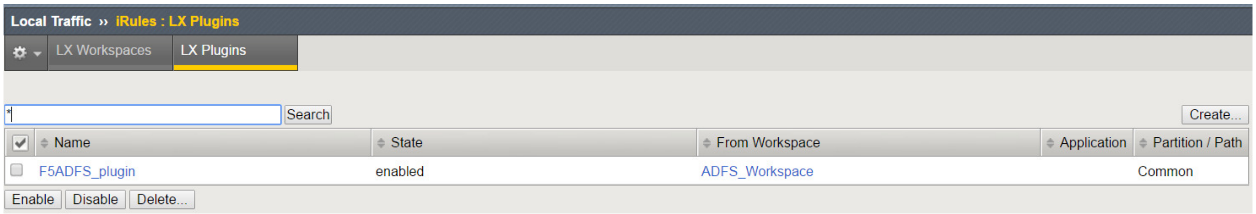




Local Traffic » iRules : LX Plugins » New Plugin...

General Properties

Name	F5ADFS_plugin
Description	
From Workspace	ADFS_Workspace ▼



Local Traffic » iRules : LX Plugins

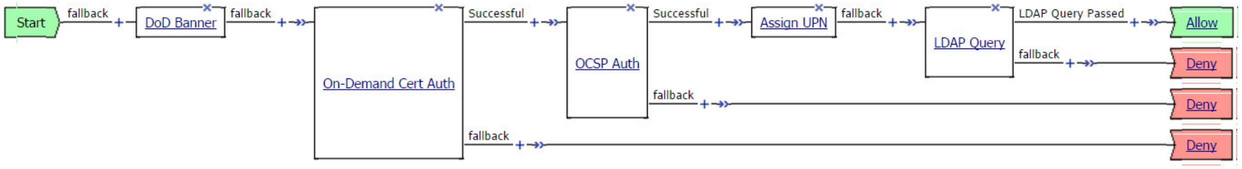
☑	Name	State	From Workspace	Application	Partition / Path
□	F5ADFS_plugin	enabled	ADFS_Workspace		Common

19. On the Application VS (i.e. SharePoint or website), apply the Access Profile that you have created which has an LDAP query in it.

NOTE: If you have any devices that need to be able to access this Virtual Server without authenticating through the APM Profile, you will need to apply an iRule to the Virtual Server that does ACCESS::disable for clients with those IP Addresses/subnets.

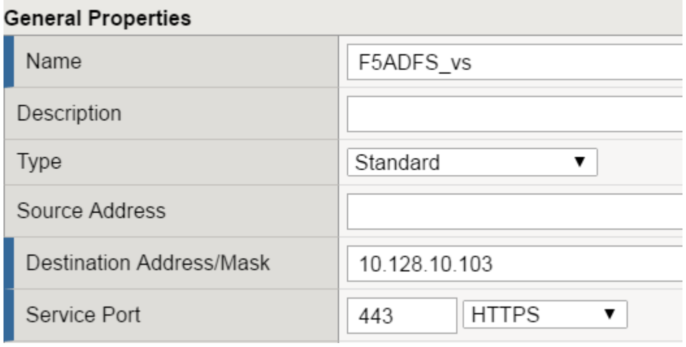
NOTE: The pool has to be 443 and you must have an SSL Server profile on the application Virtual Server.

For example:



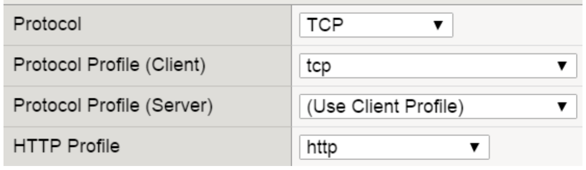
20. Now you need to create the ADFS Virtual Server. End users must be able to reach this virtual server and it will not have a pool assigned to it. It must be HTTPS/443 and will have an http profile, ssl client profile, APM Profile (same one used on the application VS), and the F5ADFS_ADFS_VS_iRule from the iRulesLX plugin.

(Example from my lab)

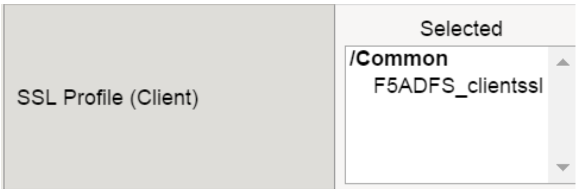


General Properties

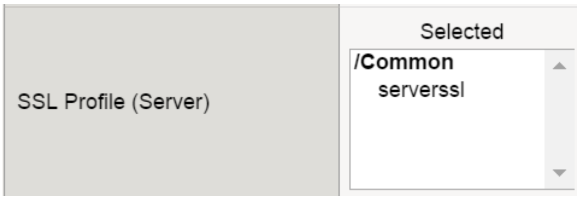
Name	F5ADFS_vs
Description	
Type	Standard ▼
Source Address	
Destination Address/Mask	10.128.10.103
Service Port	443 HTTPS ▼



Protocol	TCP ▼
Protocol Profile (Client)	tcp ▼
Protocol Profile (Server)	(Use Client Profile) ▼
HTTP Profile	http ▼



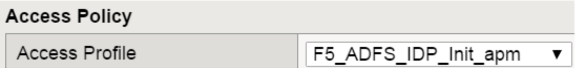
SSL Profile (Client)	Selected **/Common** F5ADFS_clientssl



SSL Profile (Server)	Selected **/Common** serverssl

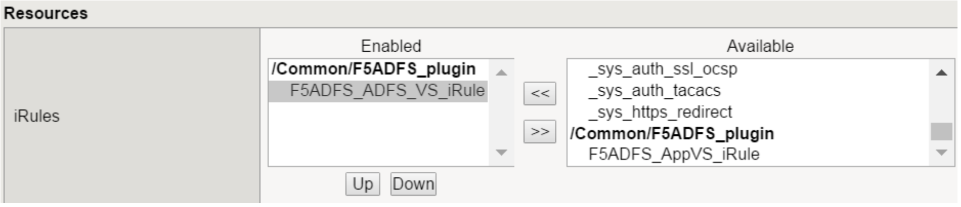
(NOTE: If you are using an Access Policy that is using On-Demand Cert Auth, you will need to include the client auth configuration in the SSL - Client profile used on the ADFS Virtual Server as well.)

Use the **SAME** Access Profile that you used on the Application Virtual Server. This is critical as the iRule is using APM session variables to get the information that was gathered on login in the APM policy to build the SAML token.



Access Policy

Access Profile: F5_ADFS_IDP_Init_apm



Resources

iRules

Enabled:
/Common/F5ADFS_plugin
F5ADFS_ADFS_VS_iRule

Available:
_sys_auth_ssl_ocsp
_sys_auth_tacacs
_sys_https_redirect
/Common/F5ADFS_plugin
F5ADFS_AppVS_iRule