# Hands on guide to help you with Passthrough 2.0 Migration

Contents

[Hands on guide to help you with Passthrough 2.0 Migration 1](#_Toc413918570)

[Introduction 1](#_Toc413918571)

[Old Passthrough 1](#_Toc413918572)

[Passthrough 2.0 3](#_Toc413918573)

[Prerequisites: what’s needed from you? 3](#_Toc413918574)

[Performing a Passthrough 2.0 request 4](#_Toc413918575)

[Step ONE 4](#_Toc413918576)

[Step TWO 5](#_Toc413918577)

[Display a ticket or Knowledge base article to the user after Passthrough 6](#_Toc413918578)

[Source Code: 6](#_Toc413918579)

[FAQ: 7](#_Toc413918580)

## Introduction

This document will help you get started with migrating to the new Passthrough. If you have any questions or need any clarifications please reach out to us with answers to these questions. This will help us better assist you.

1. Where is your current Passthrough page located and what is the URL to access it?
2. What programming language are you using to invoke Passthrough?
3. Do you use pass through for more than one Parature Instance or Department of yours? If yes, can you give us all the instance and department details you plan to use Passthrough?
4. Are you facing any issues while performing Passthrough2.0? Did you get any error message with any GUID while performing the new passthrough? Can you share the error message and a brief description of what’s happening with us?

## Old Passthrough

Before we get into more about the new Passthrough it will be a better idea to briefly brush up how the current passthrough works. If you are already aware of it, feel free to jump to the next section. For others I will try to keep this crisp.

Passthrough is a neat feature in Parature which lets your authenticated users access the Parature Portal, without showing them an additional login page. So this is in a way a SSO – Single Sign On experience.

Below is a sample html page, which outlines details needed to make a passthrough for a user on click of a submit button. Based on your system/programming language you can trigger a client form post on click of a Link to Parature portal in a similar fashion to perform passthrough.

<html>

<head>

<!--

This content type declaration is required in the head tag

by the browser to encode the contents properly.

-->

<meta http-equiv='content-type' content='text/html; charset=utf-8' />

</head>

<form name="Support"

target=\_blank

method=post

action="http://<yourInstance>.parature.com/ics/support/security.asp?deptID=<yourDepartmentID>"

>

<!-- Authentication, these values should not be changed once setup -->

<input type=hidden name="sessEmail" value="api@parature.com" >

<input type=hidden name="sessID" value="<MD5HashValueOfCustomerEmailAndAdminPassword>" >

<table>

<!-- Customer Information, these values should change based on the customer information for the specific request -->

<!-- Required -->

<tr><td>First Name </td><td><input type=text name="cFname" value="<firstName>" ></td></tr>

<tr><td>Last Name </td><td><input type=text name="cLname" value="<lastName>" ></td></tr>

<tr><td>Email </td><td><input type=text name="cEmail" value="<customerEmailID>" ></td></tr>

<!-- Required only for specific departments -->

<tr><td>UserName </td><td><input type=text name="cUname" value="<customerUserName>" ></td></tr>

<!-- Optional -->

<tr><td>password </td><td><input type=text name="cPassword" value="<password>"> </td></tr>

<tr><td>Status </td><td><input type=text name="cStatus" value="REGISTERED"> <!-- REGISTERED;PENDING;TRASHED --></td></tr>

<tr><td>tou? </td><td><input type=text name="cTou" value="1"> <!-- Blank for no; 1 for yes --> </td></tr>

<tr><td colspan=2>

<input type="submit" value="Passthrough for Support" /></a>

</td></tr>

</table>

</form>

</html>

## Passthrough 2.0

From now on for simplicity, let's call the new Secure Passthrough using SSO as Passthrough 2.0. The below flow chart explains Passthrough 2.0 in detail. Passthrough 2.0 will work seamlessly with the old passthrough. But after end of February 20th we will no longer support the old Passthrough.



## Prerequisites: what’s needed from you?

You will need some sort of server side logic to perform the Passthrough 2.0. The new SSO based Passthrough uses Certificates to establish mutual trust relationship between your server side and our Passthrough end point. To make the new workflow work you need a X.509 - *Trusted Certificate Authority issued certificate*. If you don’t have one then you need to acquire one from a trusted CA (ex. SSL Certificate issued by Verisign). Self-signed certificate issued by yourself will not work for Passthrough2.0.

To complete the configurations on Parature side, you need to pass us few details namely

1. The Subject and Issuer details about your signed certificate. We will use it to create your unique *Secure Pass through instance Id*, which is needed to perform the new passthrough.

Example: Here is the valid issuer and subject details for the certificate used by bing

Issuer details: CN = Microsoft IT SSL SHA2,OU = Microsoft IT,O = Microsoft Corporation,L = Redmond,ST = Washington,C = US

Subject: CN = www.bing.com

1. Instance id and department id where you are currently using old pass through. This is needed to update our back ends for your departments with the new secure passthrough details.

Once we configure our backend with your details, we will update you. You can use then use the rest of the documentation to help you perform Passthrough 2.0. The examples code snippets below are in .NET, but you can perform the equivalent operation to invoke our end points in any language of your choice.

## Performing a Passthrough 2.0 request

Earlier when a user clicks the link for Parature Portal, you would perform a form post with all the necessary information required to authenticate the user from the client’s browser to Security.asp. In the new workflow, it is a two-step activity.

### Step ONE

As part of the first step you will deposit all the details needed to perform the passthrough for the user into a Parature secure drop off location (<https://sso-mutual-auth.parature.com/ext/ref/dropoff>). This drop off should happen from your server side and the authenticity of your request is validated using pre agreed SSL Certificates.

* Make sure to include the certificate you have agreed to use in the request you make to sso-mutual-auth.parature.com. If not your request will be rejected as an unauthorized request.
* Also you need to add an http header with key "ping.instanceId" and pass your secure pass through instance id, which is unique to you and we have shared with you post configuration, as value.

On making a successful drop off of the user information you will get a unique reference id called “refID” for the details you shared with us.

For reference, below are the sample JSON object you will deposit at our drop off location while performing Passthrough 2.0. The "subject" should be the secure passthrough instance id unique to you given by us. The Payload object contains the details about the user.

NOTE: Find all the values passed in the old passthrough to security.asp. To retain your functionality/use case as is, you need to pass all those values in the new passthrough as part of the “payload” object key value pairs.

Below is the sample JSON with user information used to create/update user details and his account mapping.

{

"subject": "yourSecurePassthroughInstanceID",

"payload": {

"sessEmail": "**activeCSREmailID@abc.com**",

*"cFname": "userFirstName",*

*"cLname": "userLastName",*

*"cEmail": "parature\_tester@parature.com",*

*"cPassword": "password",*

*"cStatus": "REGISTERED",*

*"cUname": "userName",*

"cTou": "1",

"amName": "Users Associated Account Name",

"deptID": "45014",

}

}

NOTE: Another sample JSON with necessary user identifier information to perform pure authentication alone *when you have the Flag to let passthrough update/create new customers turned off*. If you use passthrough to create or update customers, please send all the values you sent in the past in the JSON key value pairs.

{

"subject": "yourSecurePassthroughInstanceID",

"payload": {

"sessEmail": "**activeCSREmailID@abc.com**",

*"cEmail": "parature\_tester@parature.com",*

*"cUname": "userName",*

"deptID": "45014",

}

}

For security reasons, the refID returned to you after the drop off is short lived for just 15 seconds.

NOTE: There is no validation of the data at the time you drop off to us. Even for depositing an invalid data you will get a refID. But Security2.asp will error if you have passed any invalid data.

### Step TWO

As part of the second step, you will then perform a form post of this RefID and secure pass through instance ID assigned to you from the clients browser to https://<yourinstance>.parature.com/ics/support/security2.asp. This is similar to the way you have triggered the client form post in the previous passthrough, except for the details you are passing now. Below is the sample code in *C#* used to trigger a form post on client side with necessary data.

//Secure PassThrough URL

var **portalURL** = "https://<yourinstance>.parature.com/ics/support/security2.asp";

//this page adds details to its DOM to do a client side POST to security2.asp, which would get the user authenticated to the support portal based on the refID and instance ID.

//refID is short lived. so you cannot cache it.

//assigngin the URL to which the client form post should be triggered

**this.Form.Action = portalURL;**

//creating the html that will be appened to the div.

var sb = new StringBuilder();

**sb.Append(string.Format(@"<input type=""hidden"" name=""refID"" value=""{0}"" />", refId));**

**sb.Append(string.Format(@"<input type=""hidden"" name=""instanceID"" value=""{0}"" />",** **securePassThroughPingFedInstanceID));**

//injecting the script which will trigger a form post once the page is loaded with necessary details

sb.Append("<script type='text/javascript'>**document.forms[0].submit();**</script>");

// appending the html constructed above into a div so it renders in the clients browser

securePassThroughDetails.InnerHtml = sb.ToString();

The rest of the flow from your perspective is similar to the old Passthrough. Security2.asp securely retrieves the information deposited by you using the refID, to authenticate the user and displays the portal.

## Display a ticket or Knowledge base article to the user after Passthrough

Sometimes you would want to display a particular ticket or a knowledge base article to the user after performing a Passthrough. Knowing this is a common use case, Parature supports this workflow. This is achieved by passing the right query parameters to security2.asp.

For example to display a particular knowledge base article with question ID 1173 you would perform a client form post as described in “Step TWO” of “Performing a Passthrough2.0 request” to Security2.asp with “task=knowledge” and “questionID=1173” query parameters as shown below.

var **portalURL** = "https://<yourinstance>.parature.com/ics/support/security2.asp?**task=knowledge&questionID=1173**";

To display a particular ticket you should pass the query parameters “task=ticket” and “ticket\_id=<ticketIDYouLikeToDisplay>” as shown below

var **portalURL** = "https://<yourinstance>.parature.com/ics/support/security2.asp?**task=ticket&ticket\_id=1173**";

If you don’t pass the ticket id or question ID, home page of ticket and knowledge modules will be shown to the user.

## Source Code:

We have a sample implementation written in asp.NET and C# posted to github available here <https://github.com/Parature/Passthrough2.0> . The logic to perform passthrough is implemented in the page load event of ParatureNewPassthrough.aspx. Enough comments has been added, if you have any issues you can reach out to us using regular means or through github.

## FAQ:

 Here are few tips to help you in a FAQ fashion.

1. We are unable to even open the drop off URL in Internet Explorer on our Windows 2003 server

We have seen few clients having issues on their Windows server 2003 boxes. We believe its due to the cryptographic limitations on the server and possibly due to the security patches on the box. Clients who had this issue have been able to use their 2008 or 2012 Windows Servers to alleviate the problem. Few were able to deploy a web service in the cloud to mitigate the issue as well.

1. I am unable to post the JSON data to sso-mutual-auth.parature.com. I get unauthorized message. What to do?

If you face issues with dropping off user information then it can be primarily due to the fact there is no mutual trust established through Certificates. Have you already shared the necessary details as mentioned in the Perquisite section as described above If yes, then are you passing the agreed certificate details in your request to sso-mutual-auth.parature.com? IF you are doing everything right and still facing an issue, please contact us and we can verify what the exact issue is.

Also check if you have fiddler running. Fiddler can cause trouble while intercepting the messages with a man in the middle certificate.

1. I am getting unauthorized messaged while dropping off information to sso-mutual-auth.parature.com in production but it works fine on the developer box. Or I have issues making it work in the website but the Windows form “Passthrough Playground” app in github works fine. What could be wrong?

This could certainly be related to your certificate.

* 1. Possibly your certificate store is corrupted. We would suggest you find all instances of your certificate installed in the machine, delete and reinstall it. Now export the .CER file from the freshly installed certificate, to be used in your program.
  2. Or the Application pool doesn’t have enough permission to retrieve the certificate.

You need to give access to the Account the current app pool for your server side logic is configured for. Please find below information helpful.

To get it working without having to run the Virtual Directory with elevated permissions (these should be valid for Server 2003):

* When importing the certificate from the .p7b /pfx file, use the **MMC** Snap-in ‘**Certificates’** selecting ‘**Computer account’**, import into ‘**Personal’** to avoid it ending up in the ‘Current User’ store.
* Grant access to the private key to the account the AppPool and Site run under using winhttpcertcfg.exe e.g:
  + winhttpcertcfg -g -c LOCAL\_MACHINE\My -s [<certSubject>](http://www.aviatcare.com) -a <appPoolOrSiteAccount>

Relevant information:

WinHttpCertCfg.exe Guide - <https://msdn.microsoft.com/en-us/library/windows/desktop/aa384088(v=vs.85).aspx>

WinHttpCertCfg.exe Download - <http://www.microsoft.com/en-gb/download/confirmation.aspx?id=19801>

Similar issue with Server 2008 - <http://stackoverflow.com/questions/2609859/how-to-give-asp-net-access-to-a-private-key-in-a-certificate-in-the-certificate>

1. I am able to post the User information and retrieve the refID but I am getting error while transferring the user to security2.asp. What could be wrong?

There are few things that could be wrong at this time. Few of the reasons are listed below

1. The user information you provided is in valid. Or you are not passing all the required fields as explained above during the client form post to security2.asp. Please validate the data you are passing to Security2.asp and give us a sample of the data you are passing and we can take a look at that.

Most of the times you will get a GUID along with the error. Please send us the GUID along with the time stamp when it happened so that we could look at our logs to understand more about your issue.

1. Newly created users are unable to access the portal. What could be wrong?

Are you using API to create the new user on Parature side? Are you passing the registration status as “REGISTERED” either while creating the user or while performing a passthrough? Get in touch with us with the user account details who is having issues and we will be able to help you with this.

1. Get random errors in our production servers that we are unable to establish SSL/TLS secure channel.

We only support TLS, we don’t support SSL V3. To see the ciphers and protocols we support on our end point you can refer to the ssl labs analysis report. Our servers should have a common SSL protocol and cipher suite to make communication over https.

Navigate to <https://www.ssllabs.com/ssltest/> enter our url sso-mutual-auth.parature.com and Analyze to find the details.

In .NET to restrict calls only in TLS you can use this below line, before making the request.

*ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls*