Parasoft Recorder Tutorial - Generating API Tests

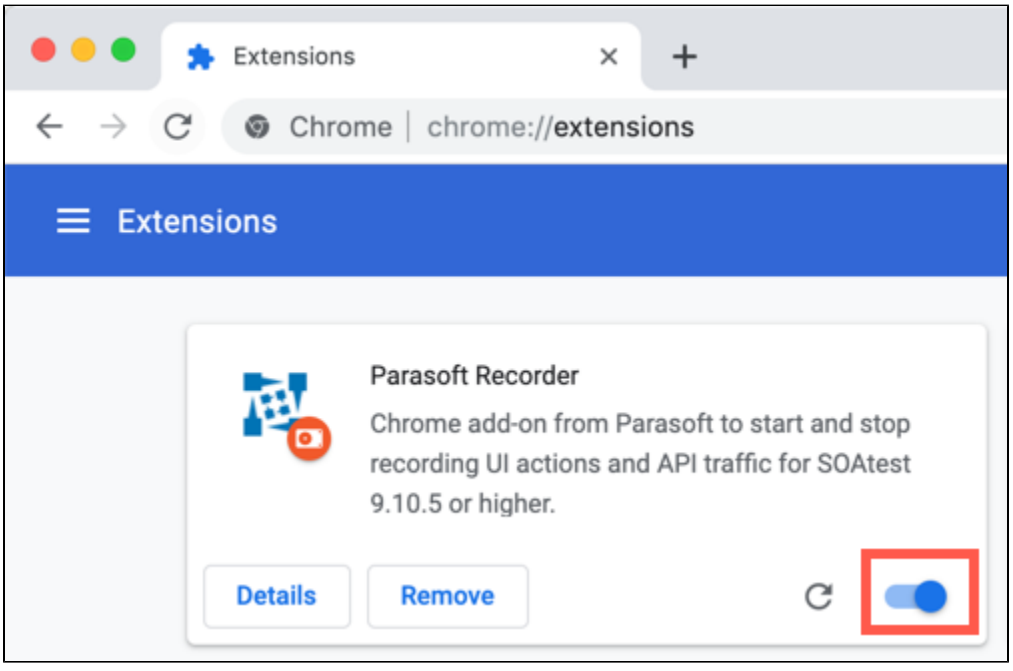
This tutorial describes how to deploy Parasoft Recorder, record API traffic, generate Smart API tests, and configure Smart API test generation. In this section:

Prerequisites

In order to complete this tutorial, you will need remote access to a SOAtest desktop or server, as well as an Advanced Test Generation feature activated for your license. See Getting Started with Parasoft Recorder for details.

Deploying the SOAtest Web Proxy

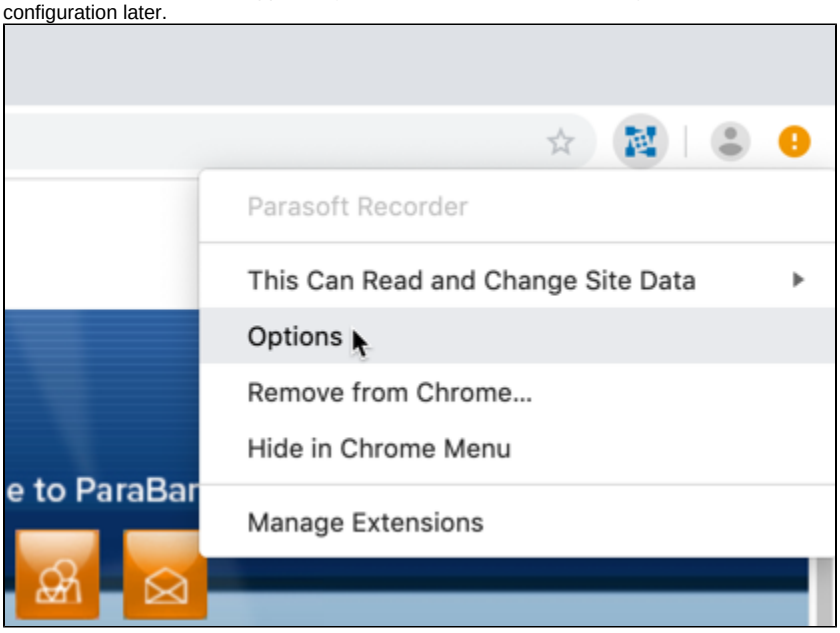
1. Run the Parasoft Recorder installer for your platform: parasoft_recorder_<version>_win32.exe for Windows or parasoft_recorder_<version>_macos.dmg for installation on Mac OS. For Mac OS, double-click the Parasoft Recorder.pkg once the installer is mounted.
2. Accept the license agreement when prompted.
3. Choose an installation location when prompted. Default is `C:\Program Files\Parasoft\Parasoft Recorder` on Windows and `/Applications/Parasoft` on Mac OS.
4. If you are on Windows, the installer will prompt you to install the SOAtest Web Proxy as a Windows service. This enables you to control the proxy using Windows services. The default port is 40090. Choose **Yes** and click **Next**.
5. Choose **Yes** and click **Next** when prompted to install the Parasoft Recorder Chrome Extension. An Internet connection with access to the Google Chrome store is required.
6. If you are on Mac OS, the installer will prompt you to confirm that you want to start the web proxy after installation, as well as start the proxy every time you log into the installation machine. The default port is 40090. Enable both options and click **Continue**. The web proxy must be running to generate Smart tests in SOAtest. You can disable automatic startup later.
7. Click **Install** to finish.
8. The installer will look for the Parasoft Web Root Certificate Authority (CA) and prompt you to install the CA if it cannot be found. The CA is normally installed as part of a normal SOAtest or Virtualize desktop installation and is required for recording traffic over SSL (see Installation). Confirm that you want to install the CA if prompted and continue.
9. Open your Chrome browser and enter `chrome://extensions` in the the address bar.
10. Click the slider to enable the plug-in.



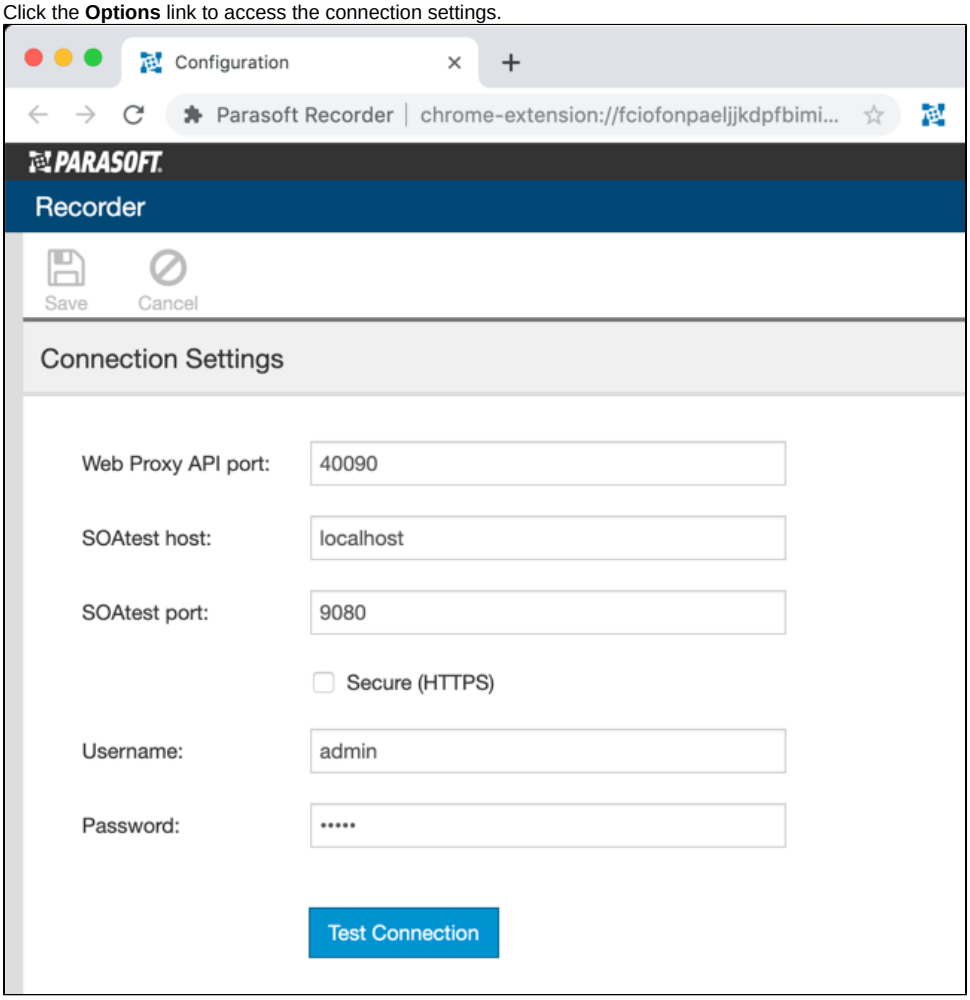
After enabling the plug-in, you will need to configure your connection to the SOAtest server.

Connecting to SOAtest Server

1. Click the Parasoft icon that appears by the address bar. You can also right-click the icon and choose **Options** if you need to change the configuration later.

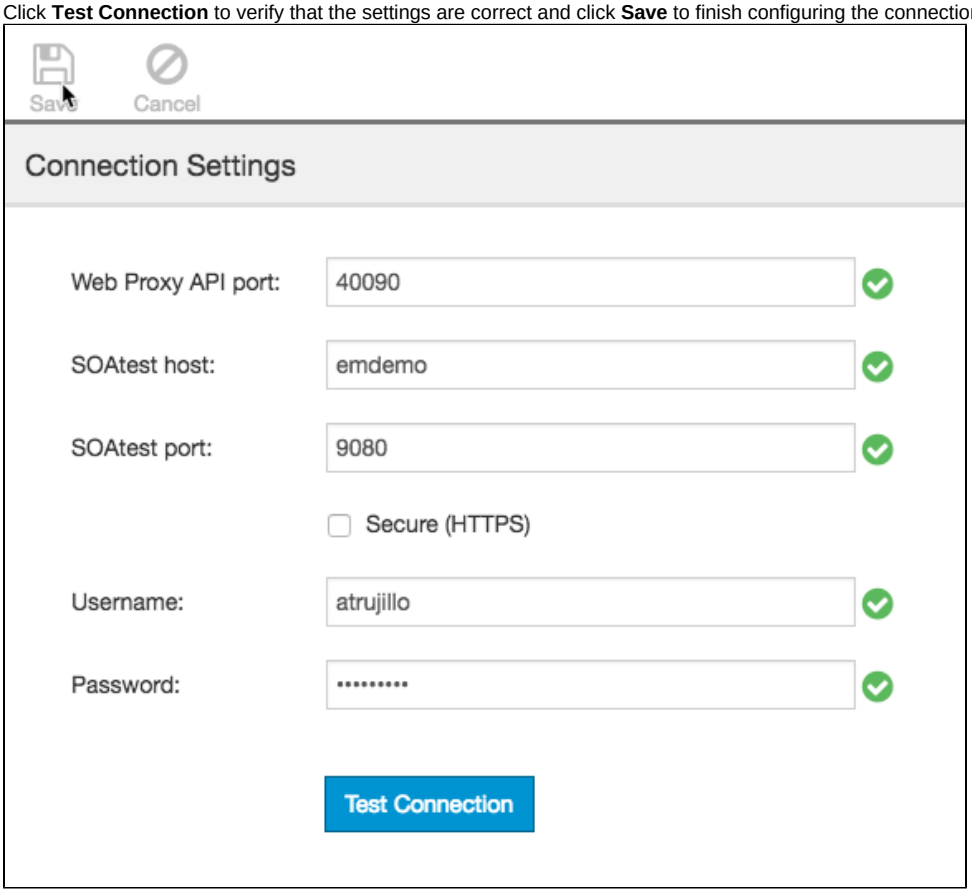


2. Click the **Options** link to access the connection settings.



3. Specify the following settings:
    a. **Web Proxy API Port**: 40090 (default).
    b. **SOAtest host**: machine name or IP address where SOAtest server is running.
    c. **SOAtest port**: port number for the SOAtest server host.
    d. If SOAtest server is configured to communicate over SSL, enable the **Secure (HTTPS)** option.
    e. **Username/Password**: your login credentials for SOAtest server.

4. Click **Test Connection** to verify that the settings are correct and click **Save** to finish configuring the connection settings.

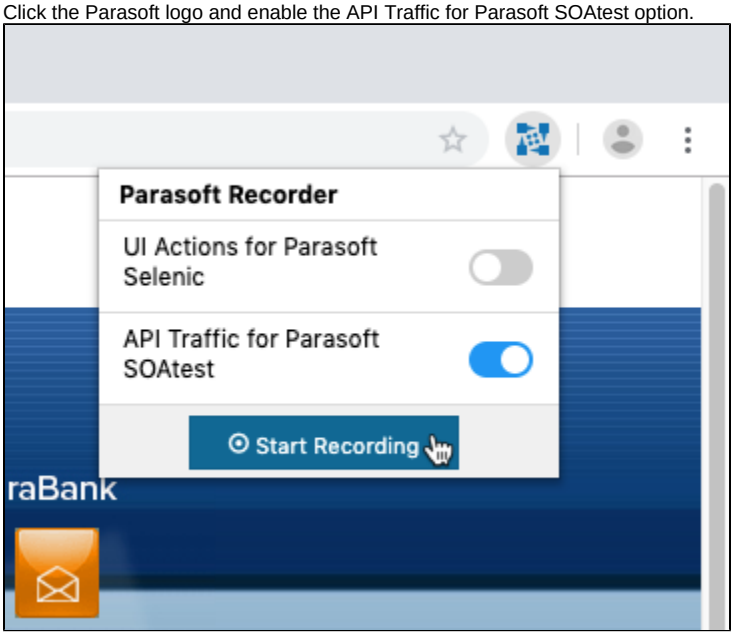


Creating Tests

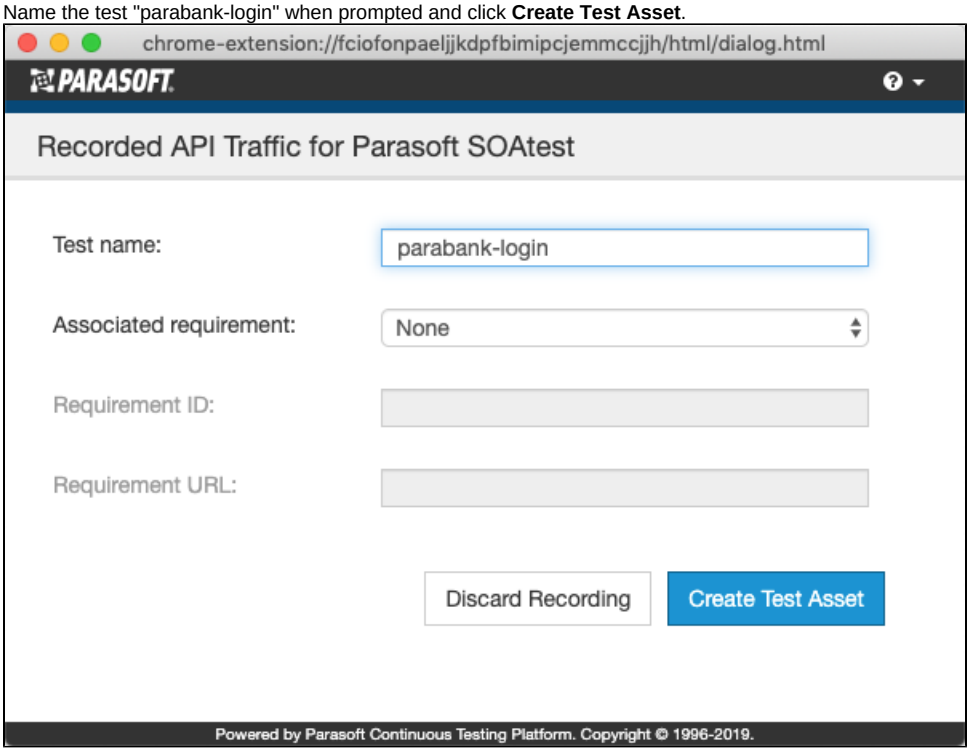
In this section, we'll create a set of test scenarios for the ParaBank demo application. We'll also demonstrate how to modify the test generation settings and how to teach the Smart API Test Generator to apply HTTP authentication, HTTP headers, and JSON assertion settings from Resource Templates. Start the ParaBank server before beginning this tutorial (see Setting Up ParaBank).

Scenario 1 - Logging In

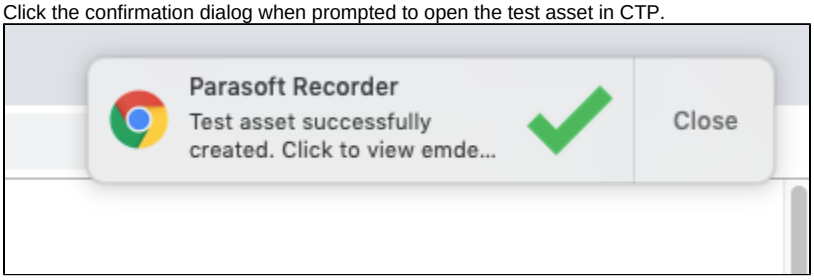
1. Open ParaBank in your Chrome browser.
2. Click the Parasoft logo and enable the API Traffic for Parasoft SOAtest option.



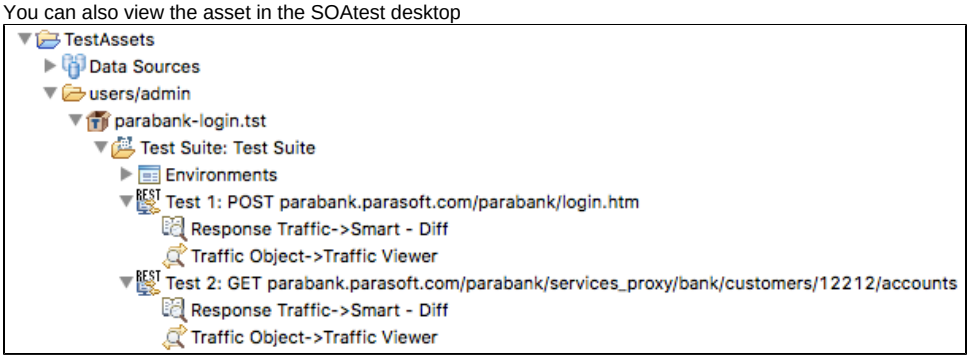
3. Choose **Start Recording** and log into the application (username: john, password: demo).
4. Click the Parasoft logo and choose **Stop Recording**.
5. Name the test "parabank-login" when prompted and click **Create Test Asset**.



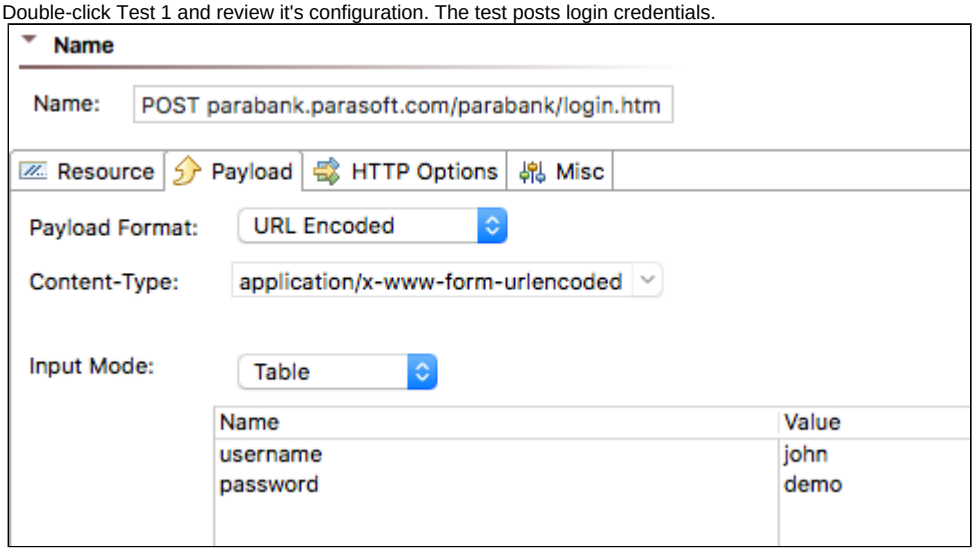
6. Click the confirmation dialog when prompted to open the test asset in CTP.



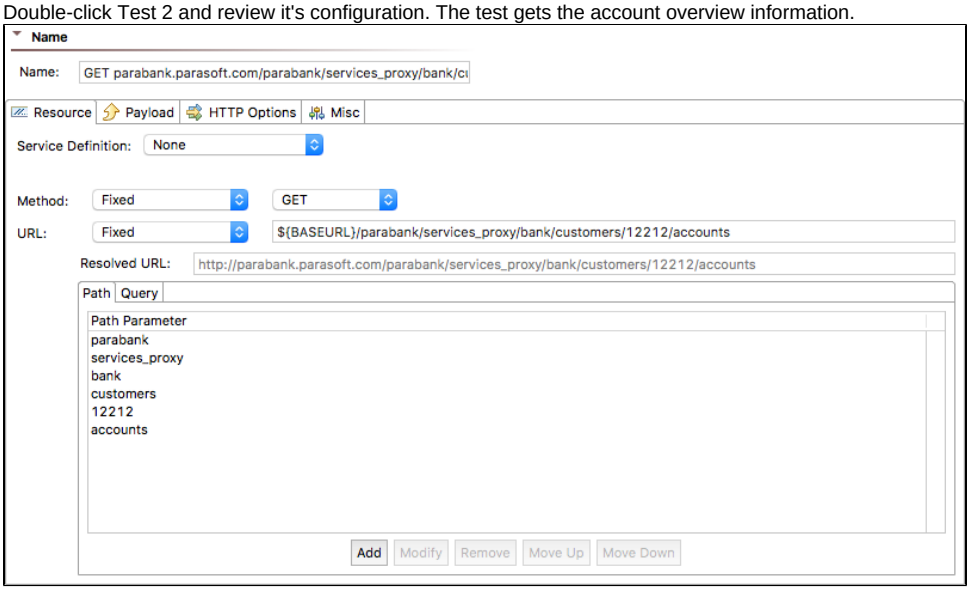
You can also view the asset in the SOAtest desktop



7. Double-click Test 1 and review it's configuration. The test posts login credentials.



8. Double-click Test 2 and review it's configuration. The test gets the account overview information.

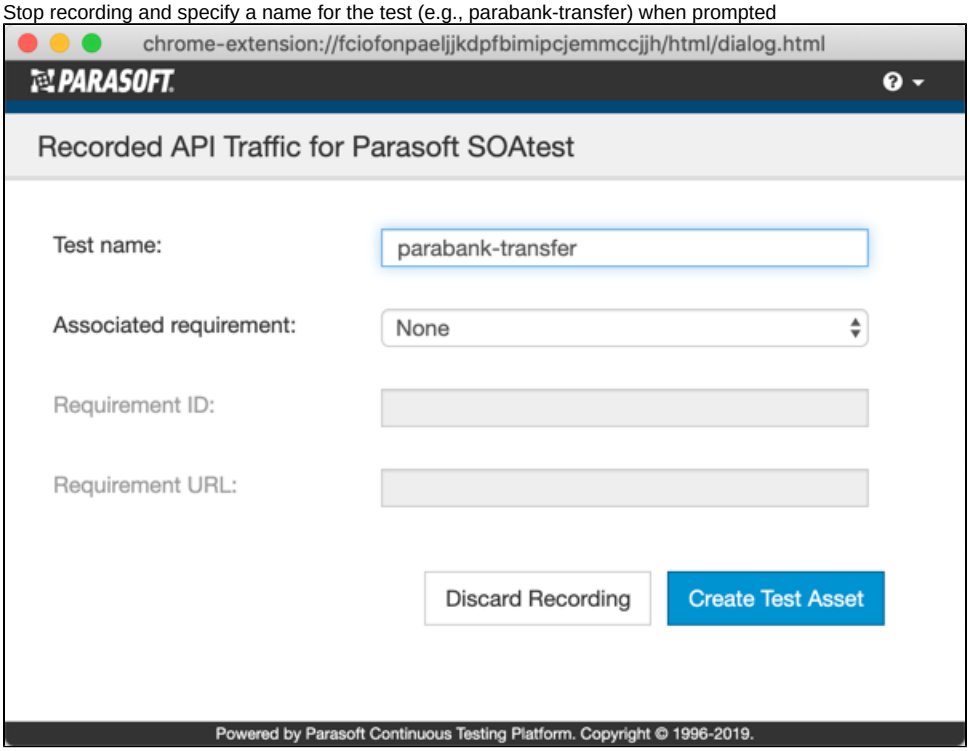


9. As long as the values remain constant, the tests will pass on consecutive executions.

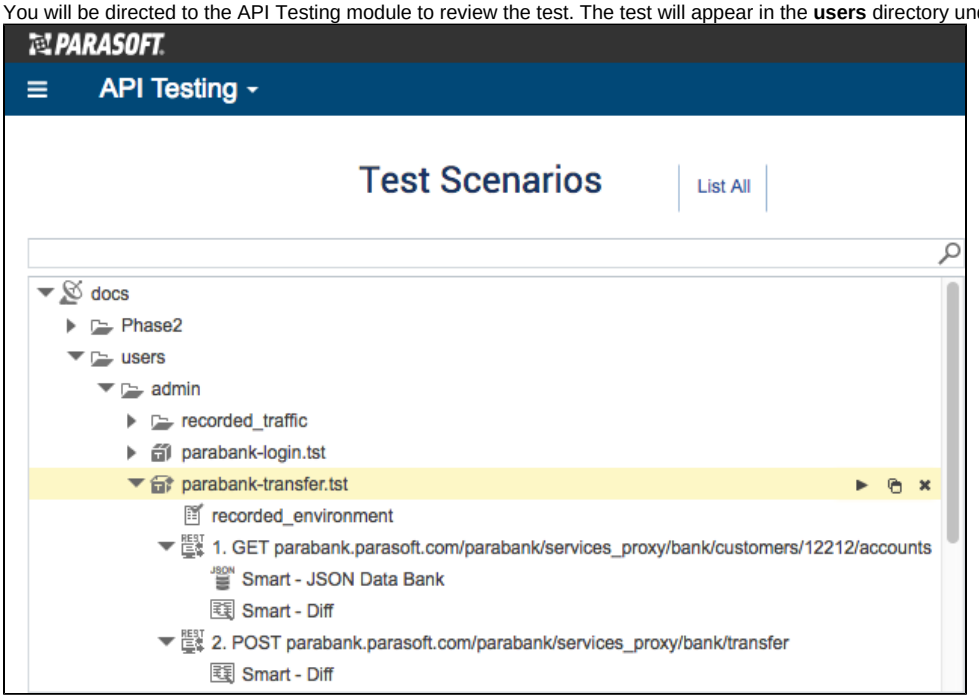
Scenario 2 - Transfer Funds

1. Log into Parabank and start recording a new scenario.
2. Click the **Transfer Funds** link and enter 100 in the Amount field.
3. Set the default From and To account fields to account number 12345 and click **Transfer**.

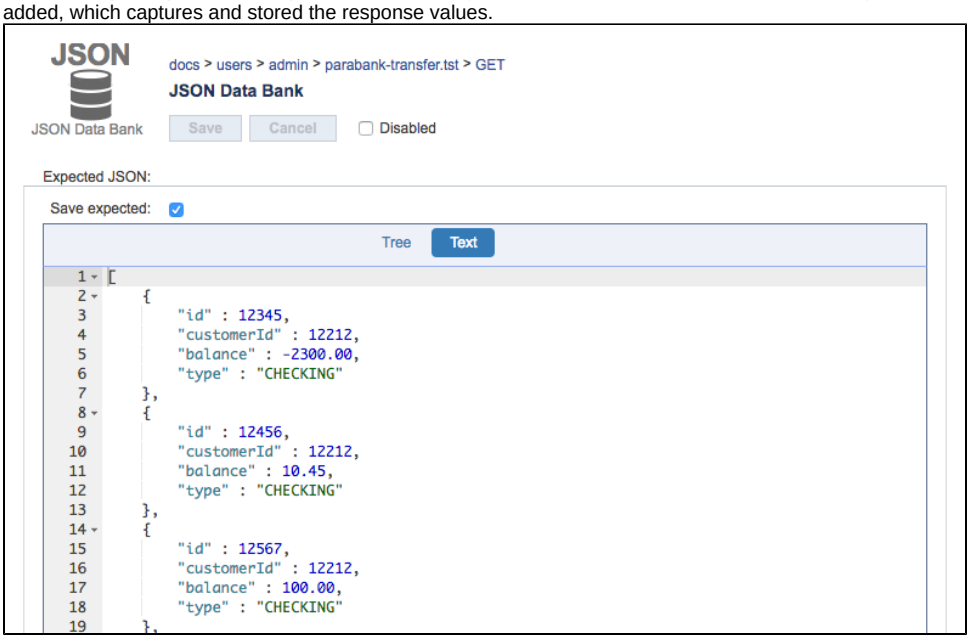
4. Stop recording and specify a name for the test (e.g., parabank-transfer) when prompted



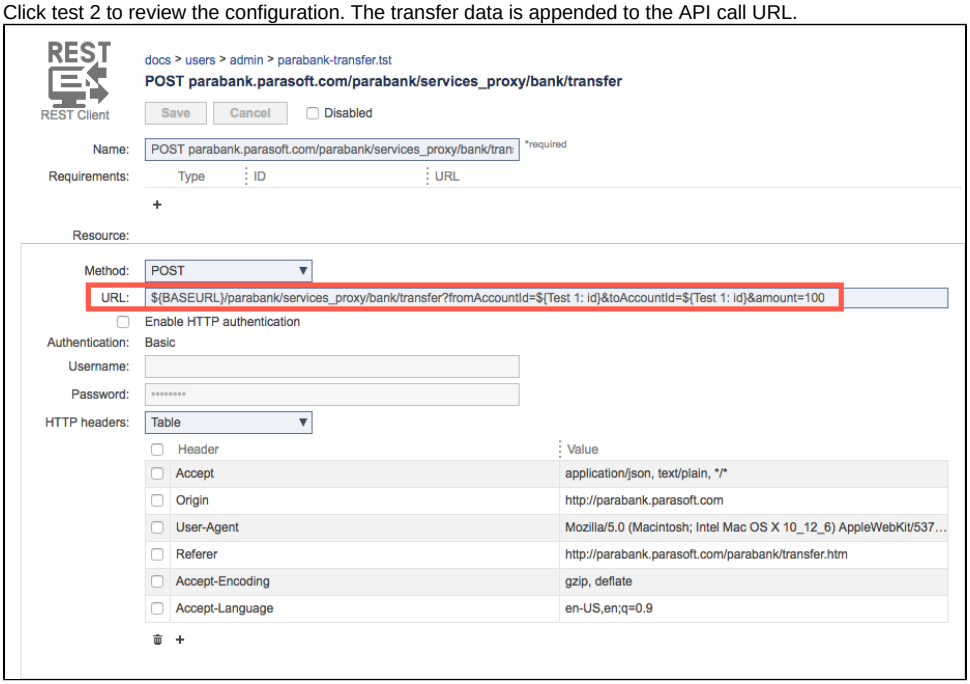
5. **Create Test Asset** and click the notification to log to view the test in CTP. You can also review the test in the SOAtest desktop. We'll use CTP for this scenario.
6. You will be directed to the API Testing module to review the test. The test will appear in the **users** directory under your user name.



7. Click test 1 ("GET parabank . . .") to review the configuration. The test is similar to Test 2 in the log in scenario, but a JSON Data Bank was added, which captures and stored the response values.

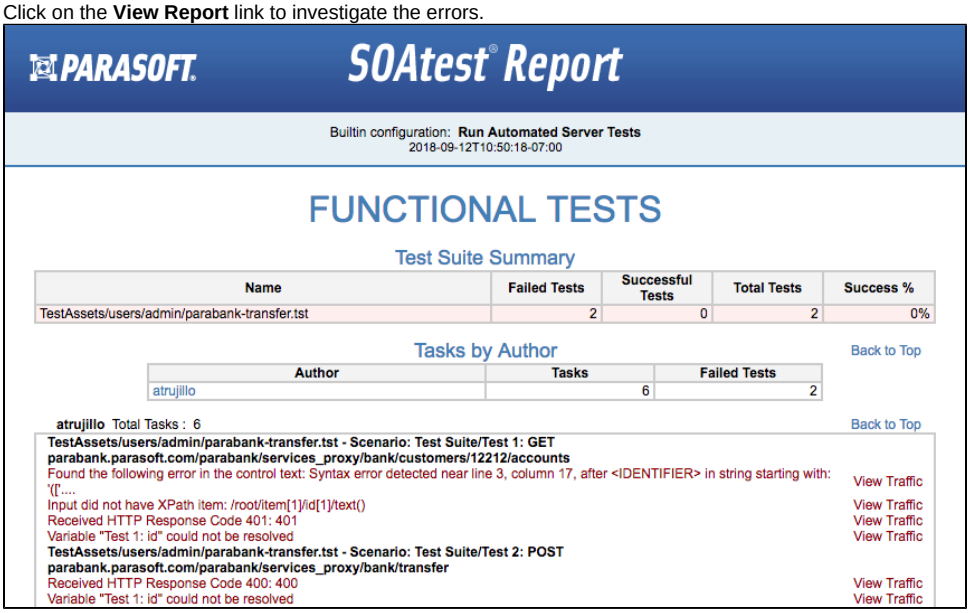


8. Click test 2 to review the configuration. The transfer data is appended to the API call URL.



9. Rerun the tests. The tests will fail.

10. Click on the **View Report** link to investigate the errors.



PARASOFT SOAtest® Report

Builtin configuration: **Run Automated Server Tests**
2018-09-12T10:50:18-07:00

FUNCTIONAL TESTS

Test Suite Summary

Name	Failed Tests	Successful Tests	Total Tests	Success %
TestAssets/users/admin/parabank-transfer.tst	2	0	2	0%

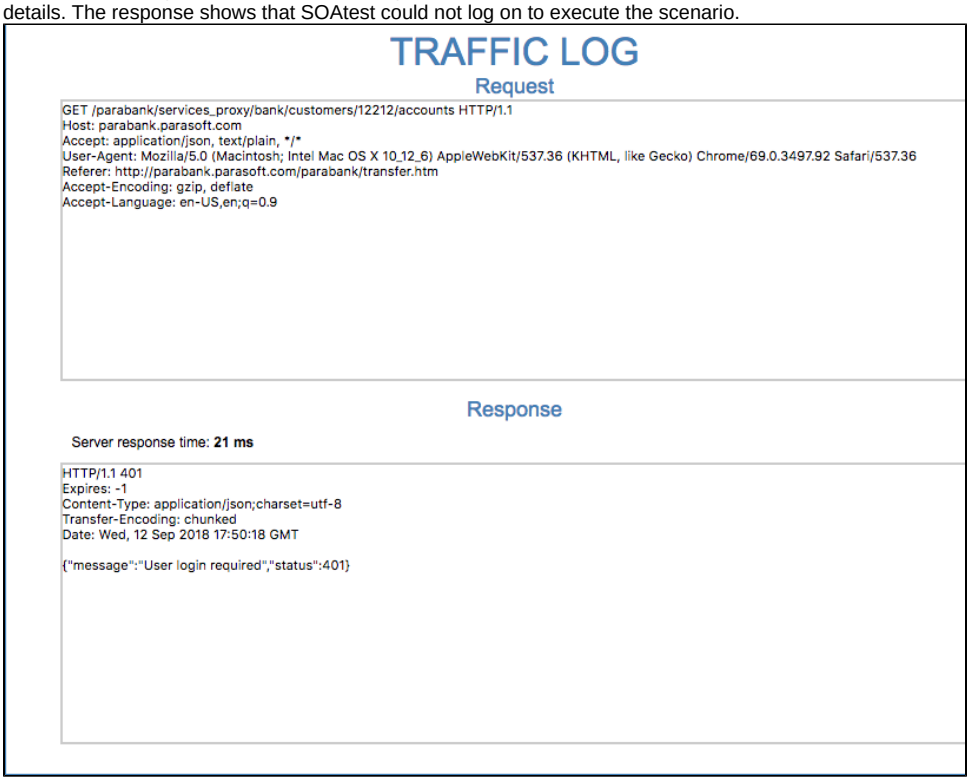
Tasks by Author — Back to Top

Author	Tasks	Failed Tests
atrujillo	6	2

atrujillo Total Tasks : 6 — Back to Top

TestAssets/users/admin/parabank-transfer.tst - Scenario: Test Suite/Test 1: GET
parabank.parasoft.com/parabank/services_proxy/bank/customers/12212/accounts
Found the following error in the control text: Syntax error detected near line 3, column 17, after <IDENTIFIER> in string starting with: '{"....' — View Traffic
Input did not have XPath item: /root/item[1]/id[1]/text() — View Traffic
Received HTTP Response Code 401: 401 — View Traffic
Variable "Test 1: id" could not be resolved — View Traffic
TestAssets/users/admin/parabank-transfer.tst - Scenario: Test Suite/Test 2: POST
parabank.parasoft.com/parabank/services_proxy/bank/transfer
Received HTTP Response Code 400: 400 — View Traffic
Variable "Test 1: id" could not be resolved — View Traffic

11. There are several problems, but the most important is the 401 error, which refers to unauthorized access. Click the **View Traffic** link for details. The response shows that SOAtest could not log on to execute the scenario.



TRAFFIC LOG

Request

```
GET /parabank/services_proxy/bank/customers/12212/accounts HTTP/1.1
Host: parabank.parasoft.com
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.92 Safari/537.36
Referer: http://parabank.parasoft.com/parabank/transfer.htm
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Response

Server response time: **21 ms**

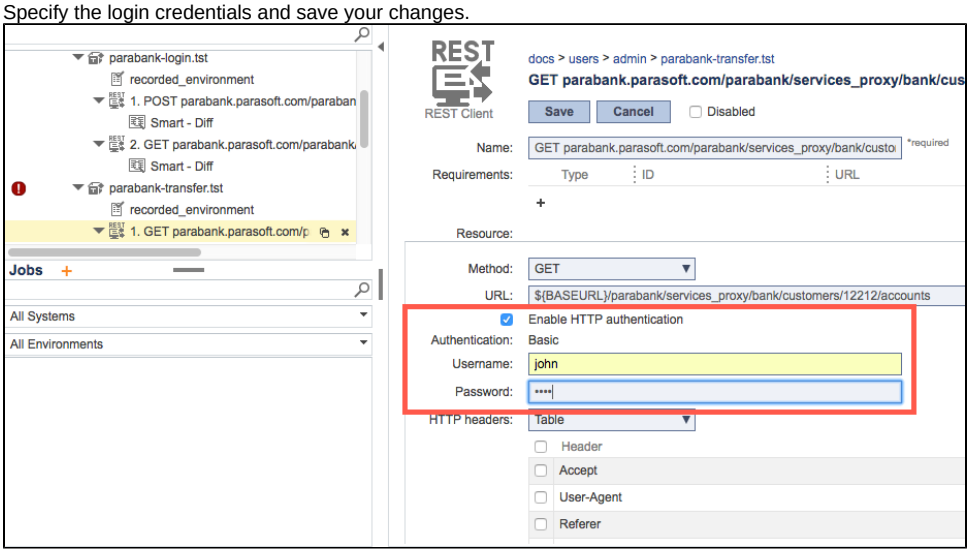
```
HTTP/1.1 401
Expires: -1
Content-Type: application/json;charset=utf-8
Transfer-Encoding: chunked
Date: Wed, 12 Sep 2018 17:50:18 GMT

{"message":"User login required","status":401}
```

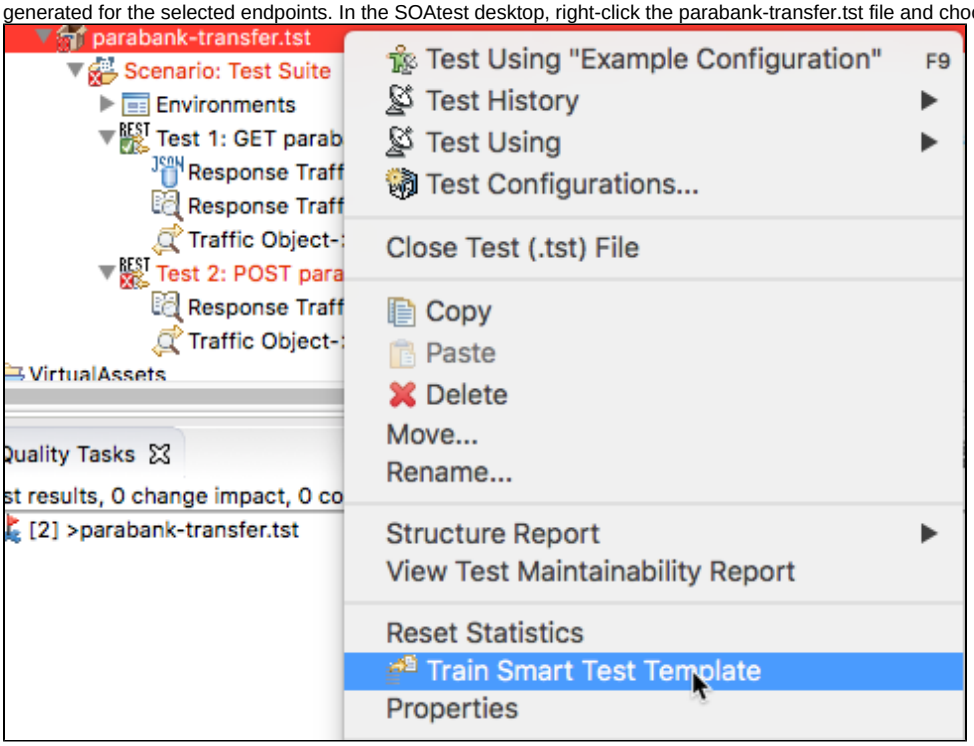
Enabling Authentication

1. Click test 1 in the transfer scenario and enable the Enable HTTP authentication option.

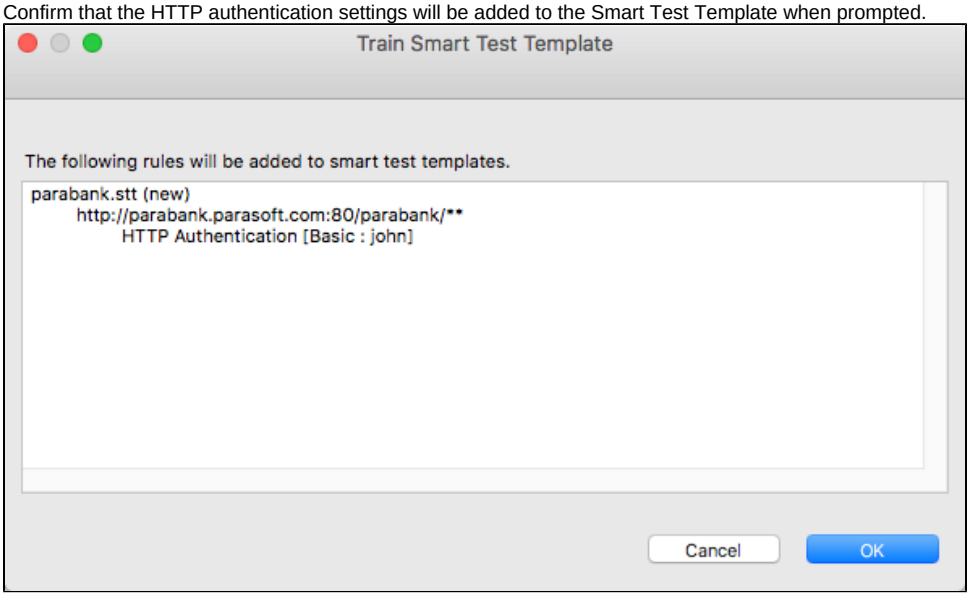
2. Specify the login credentials and save your changes.



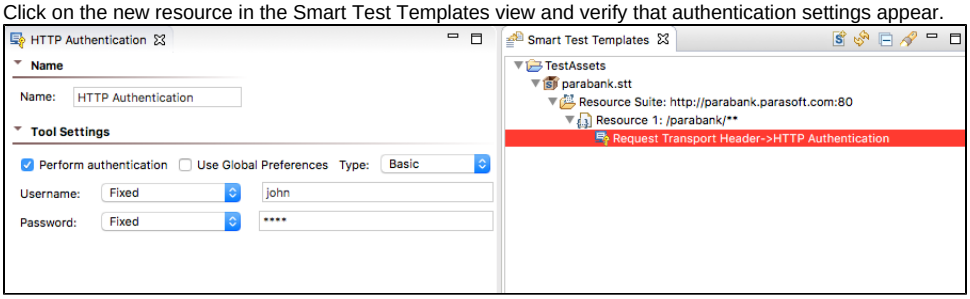
3. Rerun the test. Test 1 passes, but test 2 still fails because the authentication settings have not been configured for that particular test. Specify the authentication settings as you did for test 1.
4. We want a more scalable way to enable authentication for future tests, so we'll train SOAtest to enable authentication settings for all tests generated for the selected endpoints. In the SOAtest desktop, right-click the parabank-transfer.tst file and choose **Train Smart Test Template.**



5. Confirm that the HTTP authentication settings will be added to the Smart Test Template when prompted.



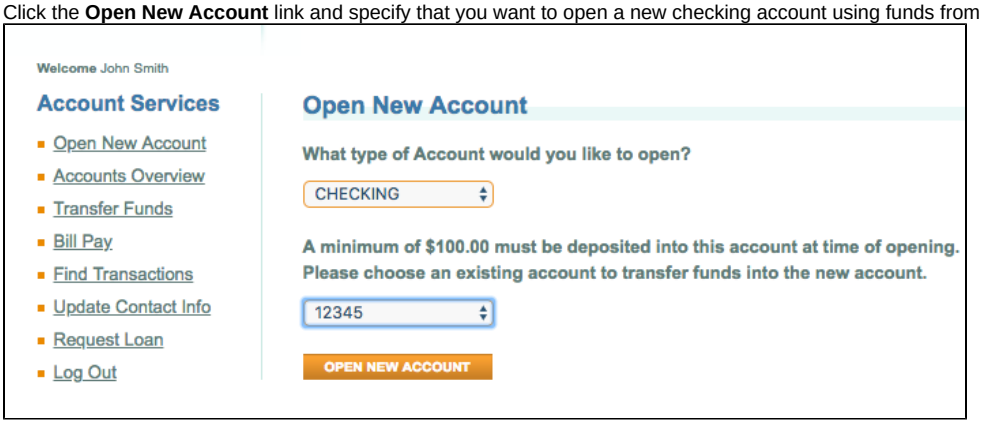
6. Click on the new resource in the Smart Test Templates view and verify that authentication settings appear.



We'll record another scenario to verify that the authentication settings are used.

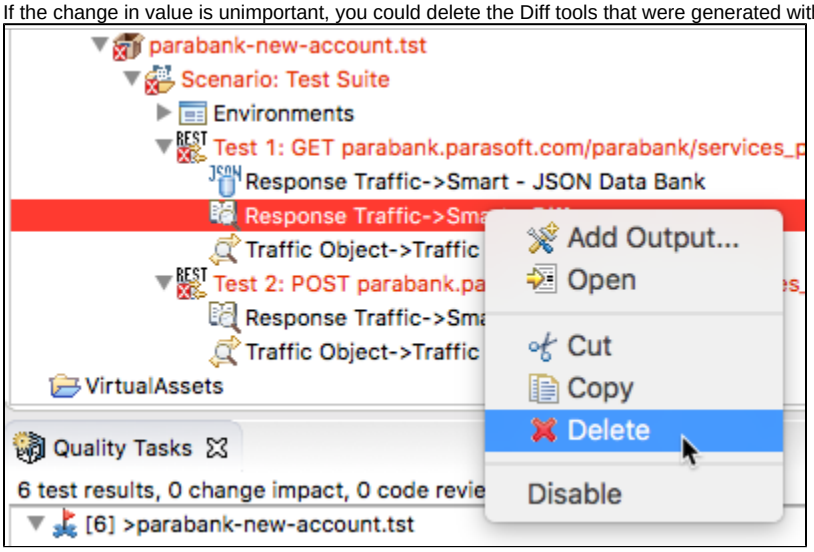
Scenario 3 - Open New Account

1. Log into ParaBank and start recording a new scenario.
2. Click the **Open New Account** link and specify that you want to open a new checking account using funds from account 12345.



3. Click **Open New Account** and stop recording.
4. Specify parabank-new-account as the name for the new asset when prompted and click Create Test Asset.
5. Log out of ParaBank and view the new test asset in SOAtest desktop or in CTP. You'll see that the HTTP authentication settings have been included in the tests, even though we logged in before recording the scenario.
6. Execute the tests, which fail because the balances associated with the accounts change each time the test executes.

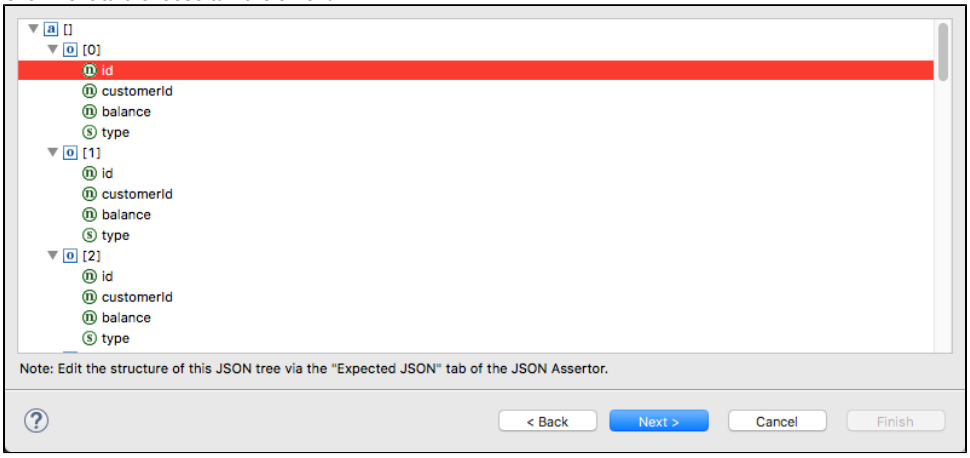
7. If the change in value is unimportant, you could delete the Diff tools that were generated with the test so that the initial values aren't considered.



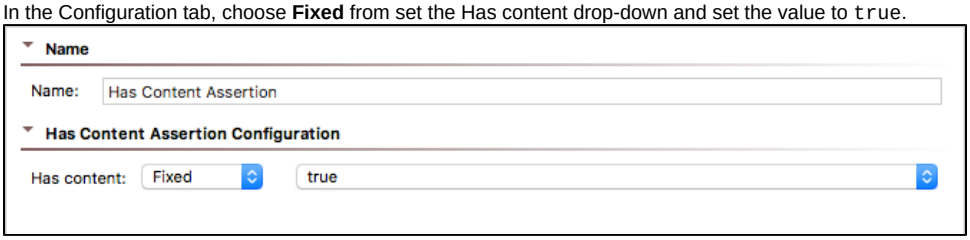
8. Re-run the test and it will now pass. See Configuring Test Creation for details on how to configure the Diff and JSON Assertor tools are handled during test generation.

Adding Assertions

1. Rerun the parabank-login test. Test 2 will fail because the values returned in the response have changed.
2. Delete the Diff tool and rerun the test. It will pass.
3. Although we are not concerned about the values returned by the GET call at this phase, we want to verify that the call returns data by adding a JSON Assertor. Right-click Test 2 and choose **Add Output...** in the SOAtest desktop.
4. Choose **Response> Traffic** in the left panel and choose **JSON Assertor** in the tool selection panel.
5. Click **Finish** and click the Configuration tab.
6. Click **Add** and choose **Has Content Assertion** from the Structure Assertions folder.
7. Click **Next** and choose an **id** element.

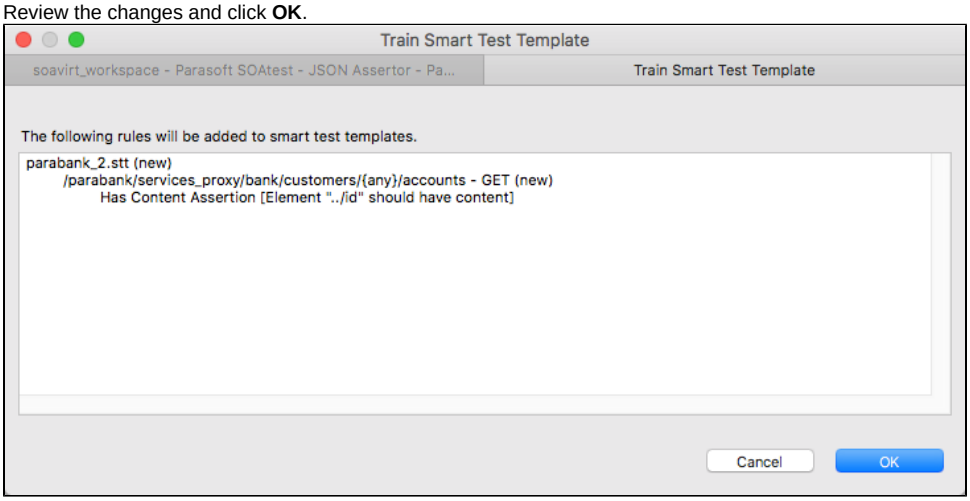


8. Click **Next** and choose the **"Apply to All "item[*]"** option so that the assertor is applied to all siblings in the response.
9. In the Configuration tab, choose **Fixed** from set the Has content drop-down and set the value to `true`.

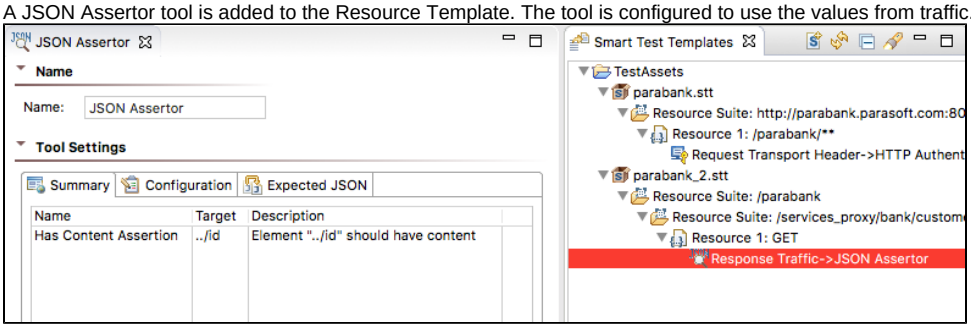


10. Save your changes and run the test.
11. The test passes, but we want to ensure that all generated tests assert that on `id` elements in the same way. Right-click on the test and choose **Train Smart Test Template**.

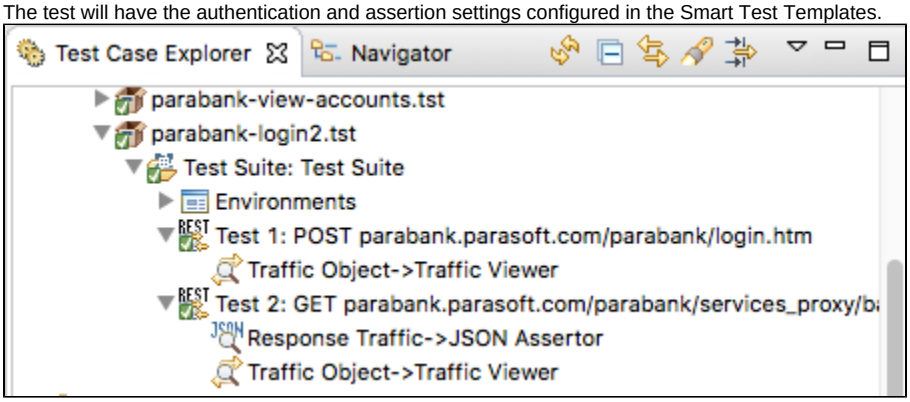
12. Review the changes and click **OK**.



13. A JSON Assertor tool is added to the Resource Template. The tool is configured to use the values from traffic.



14. Create a new test called `parabank-login2` following the process described in Scenario 1 -Logging In.
15. The test will have the authentication and assertion settings configured in the Smart Test Templates.



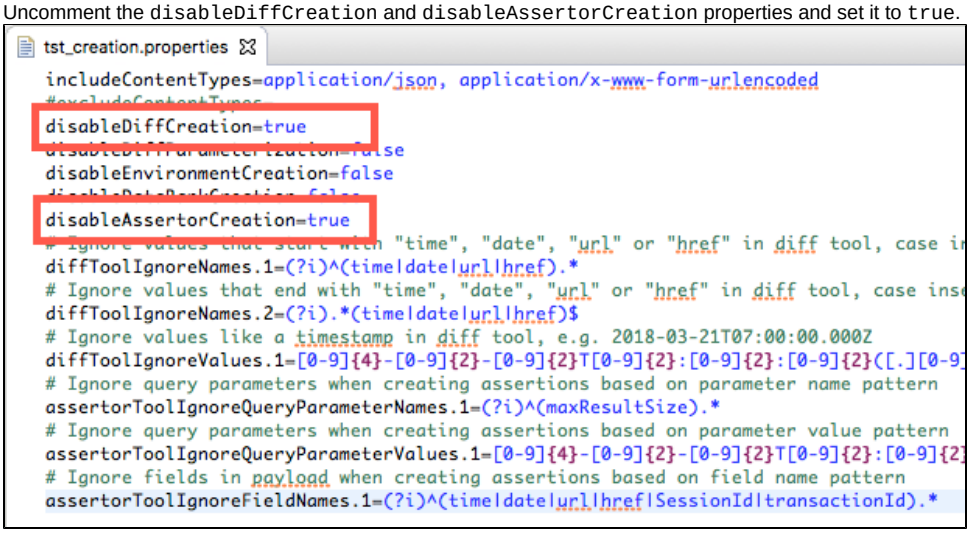
16. Run the test to verify that it passes.

Configuring Test Creation

Tests that have Diffs and JSON Assertor tools expect specific values during test execution. If your test scenario does not require specific values, you can prevent the web proxy from creating these tools automatically by disabling them in the tst_creation.properties configuration file. See Advanced Parasoft Recorder Configuration for details.

1. Open the configuration file. The file is locate TestAssets directory of the SOAtest desktop workspace. If you use CTP instead of the SOAtest desktop to create and execute API tests, you can configure the tst_creation.properties file shipped with the Parasoft Recorder in its installation directory. For Mac OS, you will need to right-click the SOAtest Web Proxy application icon and choose **Show Package Contents**.

2. Uncomment the `disableDiffCreation` and `disableAssertorCreation` properties and set it to `true`.



3. Save the file
4. You can verify the test creation settings by creating a new test.