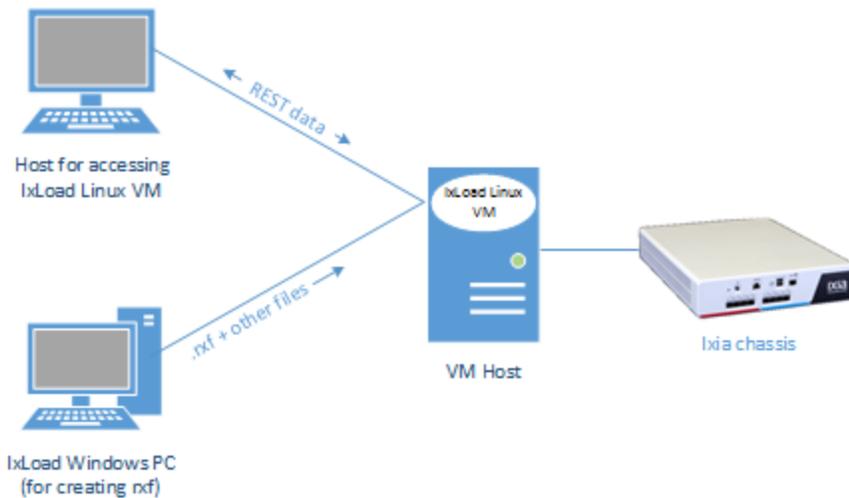


This guide provides instructions for using IxLoad on a Linux platform.

IxLoad can be used on Linux through the IxLoad REST API. Most, but not all, of the features supported from the REST API on Windows are also supported on Linux. See [IxLoad Features not Supported from REST on Linux on page 4](#) for details.

To use IxLoad on Linux, Ixia supplies an OVA with the following software pre-installed:

- IxLoad middleware (infrastructure software only, no UI support)
- IxLoad REST gateway service



For the IxLoad 8.20 release, you can only run existing repository (.rxf) files, you cannot create them from Linux.

The workflow for running repository files is:

1. Create a test in the IxLoad GUI, and save the .rxf file.
2. Deploy the OVA and start the VM.
3. Connect to the IxLoad REST gateway service on the VM.
4. Upload the rxf and supporting files to the VM.
5. Use the REST API to issue commands to start and run the test.

This document describes this workflow.

Create the Test

1. On a Windows PC with the IxLoad client installed, create the test that you want to run.
2. Save the test repository (.rxf) file, along with any supporting files (sample audio files, traffic capture files, etc.) that the test requires in a location that can be accessed from the VM.

Deploy the OVA

1. Use the hypervisor of your choice to deploy the OVA, and start the VM.
2. Use the hypervisor's console to find the IP address of the VM.

Connect to the VM and Upload the Test

1. Using a browser extension or application (such as PuTTY), establish an SCP connection to the VM. When you connect to the VM, the default path is automatically set to a shared folder intended for transferring test files to and from the VM.
2. Upload the .rxfile and any supporting files to the VM. The maximum size of a file you can upload is 1GB.

There are two ways to upload files:

- Through a script
- Through a remote file browser

Script method

To upload files from a script, the IxLoad REST API includes the `uploadFile` operation. You must create a script that includes `uploadFile`. You can use any scripting language that has libraries capable of HTTP file upload requests (such as `httplib` on Python). You cannot use a GUI REST client to upload files.

IxLoad includes a sample Python REST script, `IxLoadUtils.py`, that demonstrates the use of `uploadFile`. `IxLoadUtils.py` and other sample REST scripts are stored on the IxLoad client installation path, in a subfolder named `RestScripts`.

To upload a file, the script executes a POST request on the following URL:

```
http://127.0.0.1:8080/api/v0/resources
```

The `uploadFile` operation in the script takes three parameters:

<code>filename</code>	represents the path of the local file to be uploaded to remote location
<code>uploadPath</code>	<p>represents the path relative to the shared folder (<code>/mnt/ixload_share</code>) on the Linux VM.</p> <p>For example, if the upload path is: <code>uploads/SimpleRun.rxf</code> the file will be uploaded to: <code>/mnt/ixload_share/uploads/SimpleRun.rxf</code></p>
<code>overwrite</code>	determines whether or not an existing file on the remote location with the same name as the uploaded file is overwritten. The default value is <code>true</code> .

The image below shows an example of a script that uses the uploadFile operation.

```
import os
import requests

url = 'http://127.0.0.1:8080/api/v0/resources/'
headers = {'Content-Type': 'multipart/form-data'}

def uploadFile(fileName, uploadPath, overwrite=True):
    params = {'overwrite': overwrite, 'uploadPath': uploadPath}

    print 'Uploading...'
    try:
        with open(fileName, 'rb') as f:
            resp = requests.post(url, data=f, params=params, headers=headers)
    except requests.exceptions.ConnectionError as e:
        print (
            'Upload file failed. Received connection error. One common cause for this error is the size of the file to be uploaded.'
            'The web server sets a limit of 1GB for the uploaded file size. Received the following error: %s' % str(e)
        )
    except IOError as e:
        print 'Upload file failed. Received IO error: %s' % str(e)
    except Exception:
        print 'Upload file failed. Received the following error: %s' % str(e)
    else:
        print 'Upload file finished.'
        print 'Response status code %s' % resp.status_code
        print 'Response text %s' % resp.text

fileNamePath = 'SimpleRun.rxf'
relativeUploadPath = 'uploads/' + os.path.split(fileNamePath)[1]
overwrite = True

uploadFile(fileNamePath, relativeUploadPath, overwrite)
```

For full information on uploadFile and the other IxLoad REST commands, see the *IxLoad REST API Guide* (available from the Ixia website: <https://support.ixiacom.com/user-guide>).

Remote file browser method

You can upload files using a remote file browser. To use this method, connect to /mnt/ixload_share and pass the following credentials:

Username:	ixload
Password:	ixia123

Run the Test

1. Start the REST client you want to use for the test.
2. Issue the REST commands to start and run the test. For information on the REST commands, see the *IxLoad REST API Guide* (available from the Ixia website: <https://support.ixiacom.com/user-guide>).

When used on Windows, REST API calls that operate on files (such as the .rxf file) require the full path to the file. The same is true on Linux -- you must use the full path to the file. The file path on the VM always begins with /mnt/ixload-share. Uploading programatically (i.e., from a script) or from a remote file browser must be done from this location.

For example, the loadTest operation loads the rxf file.

On Windows, the path might be:

```
{"fullPath": "C:\\path\\to\\files\\myTest.rxf"}
```

On Linux, the path might be:

```
{"fullPath": "/mnt/ixload-share/myTest.rxf"}
```

If the .rxf file includes references to any other files, make sure these files are also uploaded in the shared location on the VM. If the files referenced from the .rxf have absolute paths, you must modify them (for example, by doing a PATCH request) to point to their new location on /mnt/ixload-share.

IxLoad Features not Supported from REST on Linux

- AppLibrary protocols
- Resource Manager
- Profiles (e.g. Real files)
- IxReporter

Notices

Copyright Notice

© Keysight Technologies 2016–2018

No part of this document may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies, Inc. as governed by United States and international copyright laws.

Warranty

The material contained in this document is provided “as is,” and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Keysight disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Keysight shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Keysight and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement (“DFARS”) 227.7202, the U.S. government

acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at

<http://www.keysight.com/find/sweula> or <https://support.ixiacom.com/support-services/warranty-license-agreements>.

The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Key-sight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data. 52.227-14 (June 1987) or DFAR 252.227-7015 (b) (2) (November 1995), as applicable in any technical data.