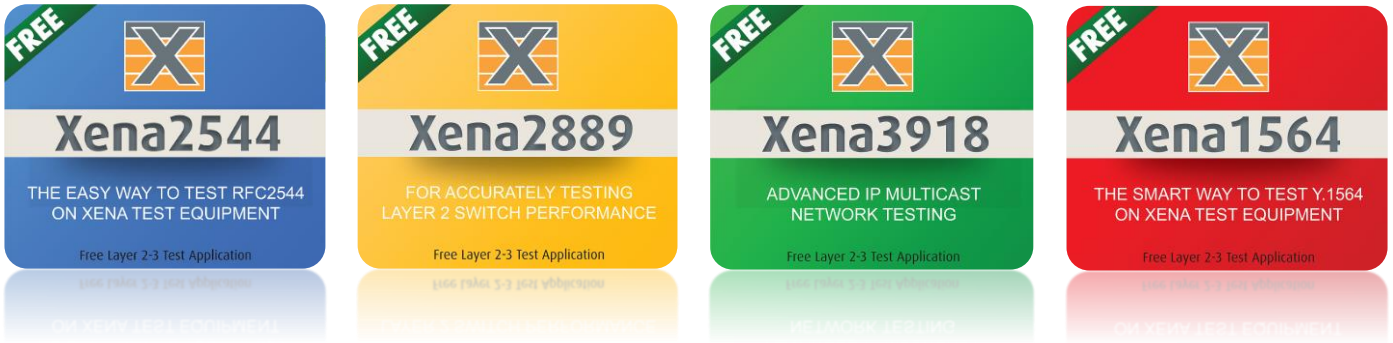
Automating Xena Test Suites



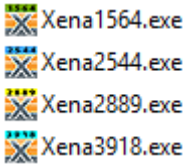
python

Rev. 1

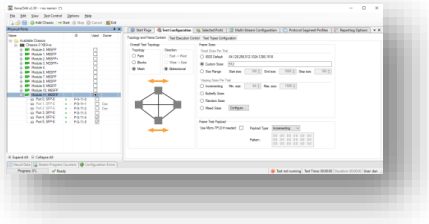
Preface

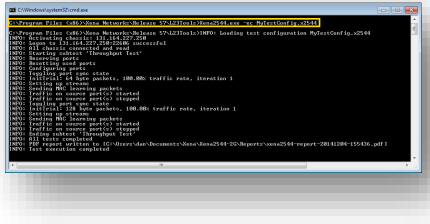
Xena Networks provides several Benchmarking Test Suites.

Each Test Suite has an application Executable.



Test Suites can be executed from GUI and also Automated.



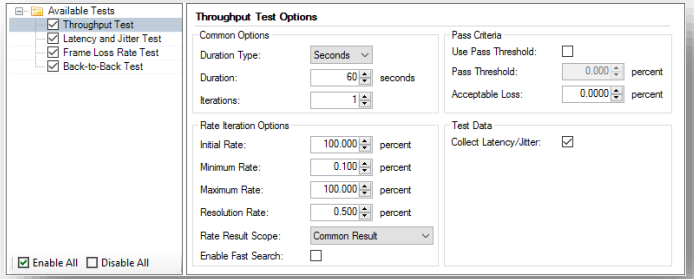


Preface

The Application has **embedded test logic** based on:

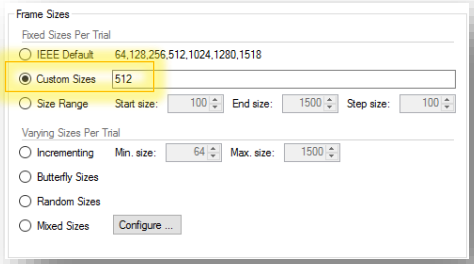
A standards definitions rules (e.g. *RFC2544* - Throughput).

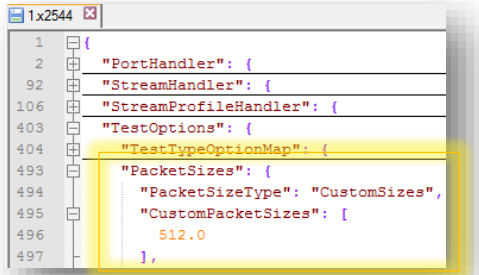
Operations such as Start/Stop/Set Traffic are determined by presets and also changed on the fly based on live results and conditions.



A set of configurable parameters.

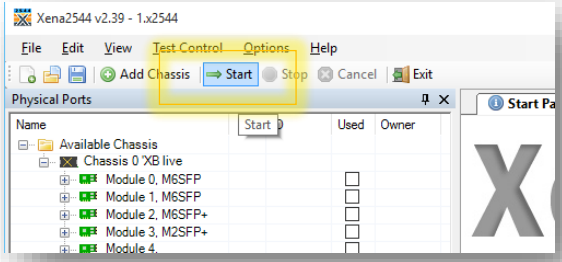
Parameters are stored in a JSON formatted file:





Preface

A Test can be initiated from within the Test Suite GUI:

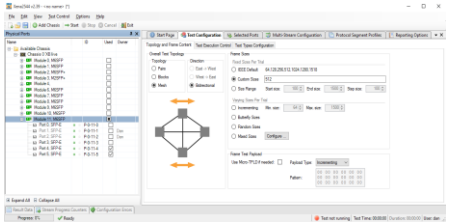


Can also be initiated from a command line (Xena1564.exe Xena2544.exe Xena2889.exe Xena3918.exe + CONFIG)

Executed either manually or by a script/program.

Supported both on Windows and Linux shell (mono powered).

Preface



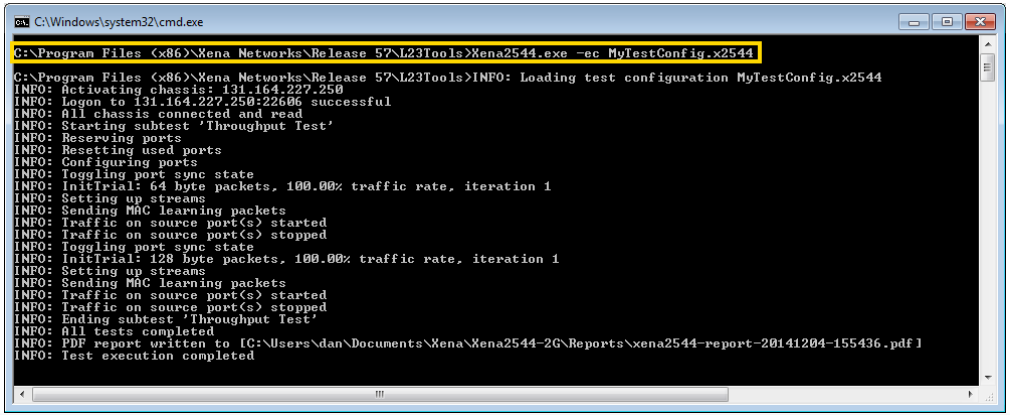
Microsoft .NET Based

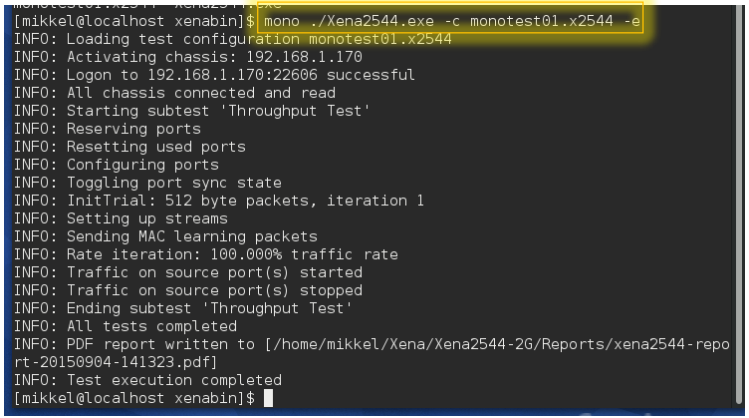
GUI

Xena1564.exe
Xena2544.exe
Xena2889.exe
Xena3918.exe

Command line

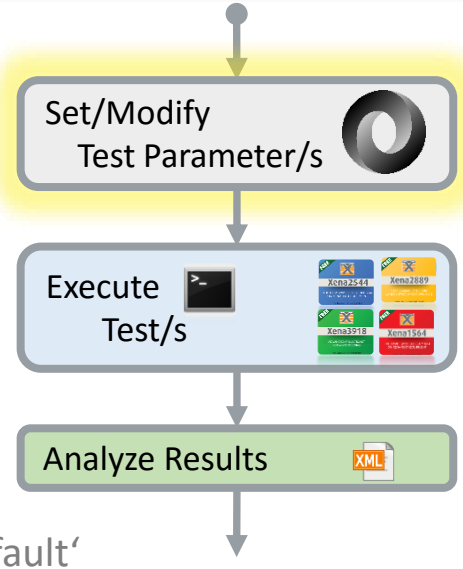
LINUX





Test Stages

Pre-Test – Configuration Stage:

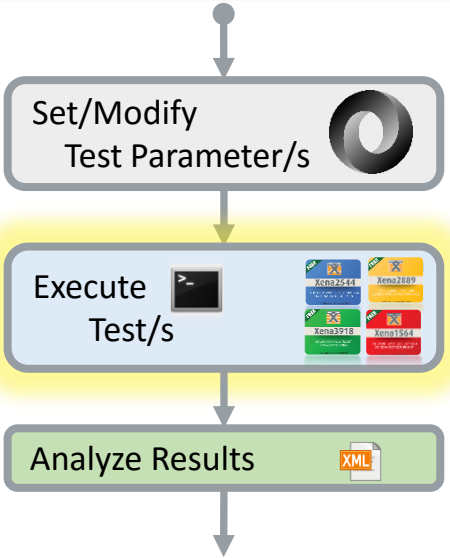



```
# Import JSON Library
import json

# Read configuration file to variable
with open('Configuration.x2544' ,'r',encoding='utf-8') as data_file:
    x2544_Configuration = json.loads(data_file.read())

# Modify Value of variable (e.g. TestOptions->PacketSizes->PacketSizeType)
x2544_Configuration['TestOptions']['PacketSizes']['PacketSizeType'] = 'IEEEDefault'

# Write modified to the file that is expeted To(2) Be(b) Used.
with open('2bUsed.x2544', 'w',encoding='utf-8') as f:
    json.dump(x2544_Configuration, f, indent = 2,sort_keys = True ,ensure_ascii=True)
```

Test Stages

Live Test – Execute(/Monitor) Stage:

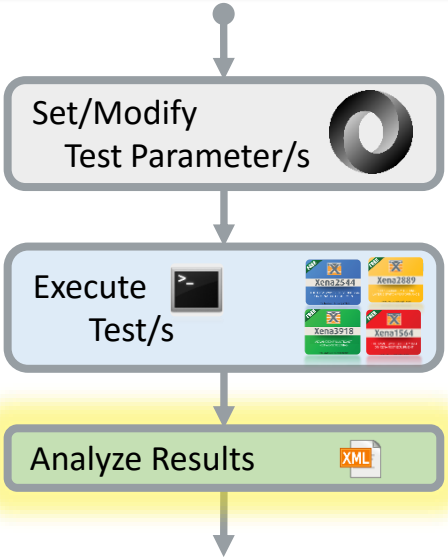
```
# Import subprocess Library
import subprocess

# Call the exe using “-ec” and the “2bUsed” Configuration file
subprocess.call(["C:\Program Files (x86)\Xena Networks\Release
62.3\L23Tools\Xena2544.exe","-ec","2bUsed.x2544"])

# Linux example
args = ["mono", "./tools/pkt_gen/xena/Xena2544.exe", "-ec",
        "./tools/pkt_gen/xena/profiles/2bUsed.x2544"]

subprocess.call([args, stdout=sys.stdout])
```



Test Stages

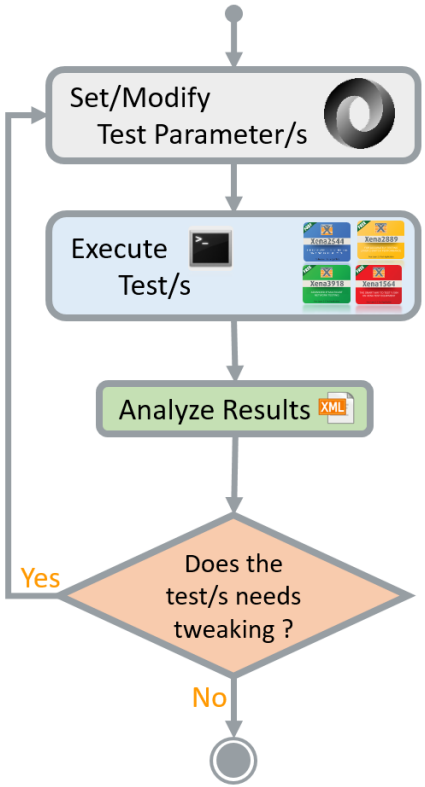
Post Test – Analysis Stage:

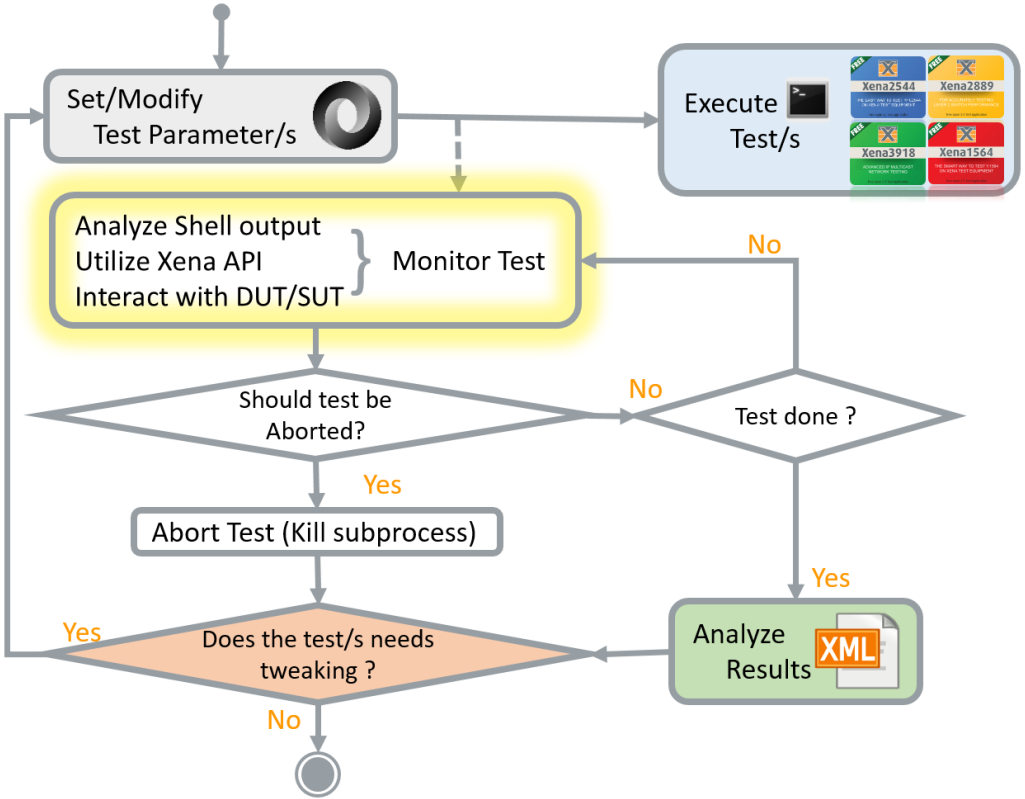
```
# Import ElementTree Library for easy XML parsing
import xml.etree.ElementTree as ET

# Parse the XML report and get root handle
root = ET.parse(r'xena2544-report.xml').getroot()

# Address specific results counters by name/location
TotalTxFrames = root[0][1][0].get('TotalTxFrames')
```



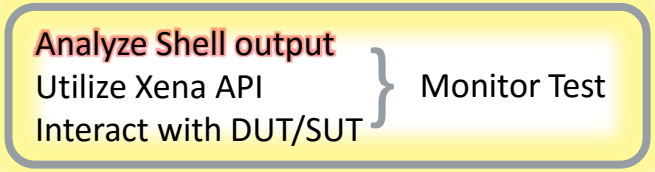

```
<?xml version="1.0" encoding="utf-8"?>
<!--Xena2544 Test Report-->
<Xena2544>
  <TestResults>
    <Summary>
      <Identification TestCompany="Xena Networks ApS" Customer="Xena Networks" CustomerAccessID="" CustomerServiceID="" />
      <Metrics TestDateTime="20151006-122916" TestDuration="21" GeneratedBy="Xena2544 v2.28" />
      <Comment><![CDATA[]]></Comment>
    </Summary>
    <TestResult TestType="Throughput">
      <Result FrameSize="100" TestState="PASS" TotalTxRatePcnt="1" TotalTxFrames="20832" TotalTxRateBpsL1="19998720" TotalTxRa
        <Port ID="P-0-11-0" PortTxFrames="10416" PortTxBpsL1="9999360" PortTxBpsL2="8332800" PortTxPps="10416" PortRxFrames="1
        <Port ID="P-0-11-1" PortTxFrames="10416" PortTxBpsL1="9999360" PortTxBpsL2="8332800" PortTxPps="10416" PortRxFrames="1
      </Result>
    </TestResult>
  </TestResults>
  <TestConfiguration>
</Xena2544>
```

Blocking Vs Non Blocking





Blocking Vs Non Blocking



In order to execute a Test Suite and monitor test one must first spawn a subprocess

Use Popen to create a subprocess:

```
ps = subprocess.Popen(args, stdout=sys.stdout)
```

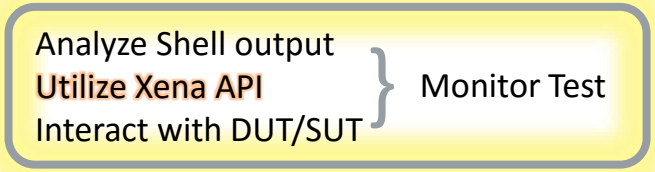
Use Communicate to retrieve Shell output:

```
ps.communicate()
```

Use Terminate or Kill to Abort or Stop test:

```
ps.kill()
```

Blocking Vs Non Blocking



In order to also monitor the Xena test ports counters one must Utilize Xena RAW API.

Opening a TCP socket to the chassis IP (port *22611*) and retrieving counters via the API commands.

In order to monitor test ports no Reservation is required.

Overriding Test Ports reservation will abort the test.

More information on how to use Xena RAW API @ **Xena Automation Introduction** pdf.