XENA NETWORKS

Automating Xena Test Suites

VALKYRIE 2544

VALKYRIE 2889

VALKYRIE 3918

Rev. 2

Preface

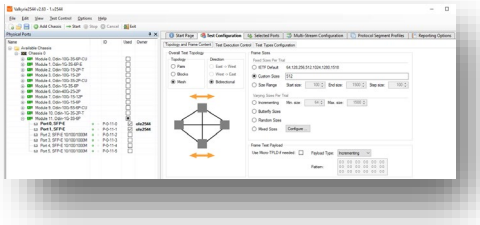
Xena Networks provides several Benchmarking Test Suites.

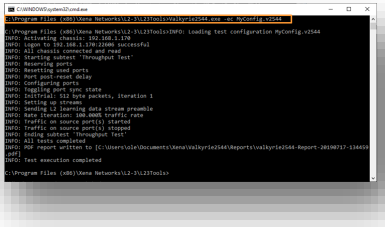
VALKYRIE 2544 VALKYRIE 2889 VALKYRIE 3918

Each Test Suite has an application Executable.

Valkyrie2544.exe
Valkyrie2889.exe
Valkyrie3918.exe

These Test Suites can be executed from GUI and also Automated.



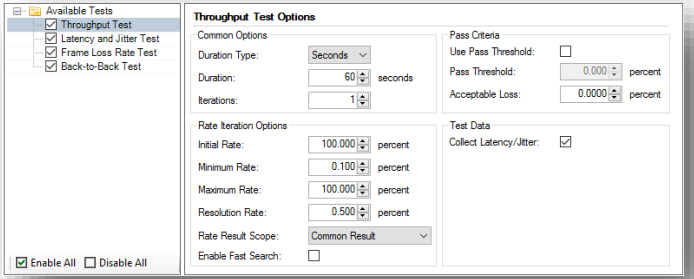


Preface

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

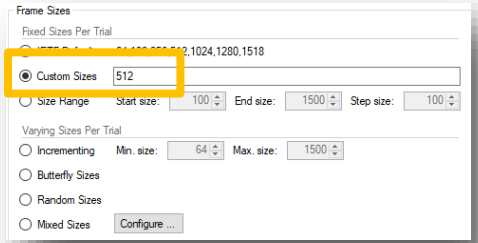
A standards definitions rules *(e.g. RFC2544 - Throughput).*

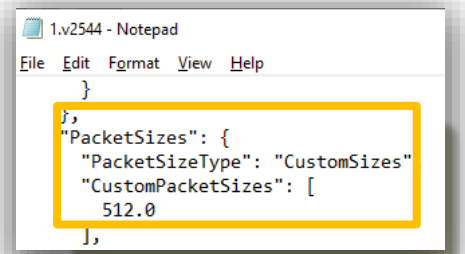
Operations such as Start/Stop/Set Traffic are determined by presets and also changed on the fly based on live results and conditions.



A set of configurable parameters.

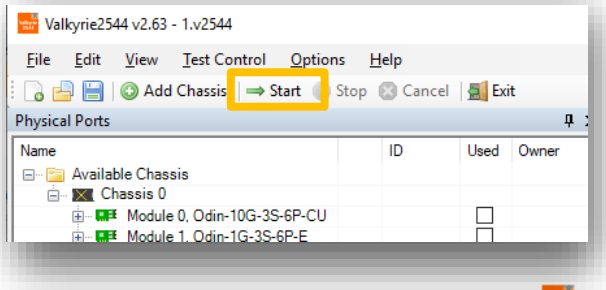
Parameters are stored in a JSON formatted file:





Preface

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

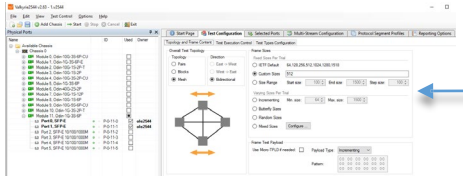


Can also be initiated from a command line (Valkyrie2544.exe / Valkyrie2889.exe / Valkyrie3918.exe + CONFIG)

Executed either manually or by a script/program.

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

Preface



GUI

Valkyrie2544.exe
Valkyrie2889.exe
Valkyrie3918.exe

Microsoft .NET Based

Command line

LINUX

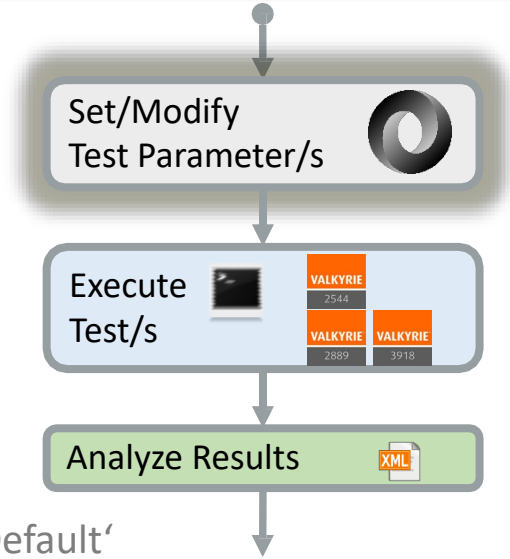
```
C:\WINDOWS\system32\cmd.exe
C:\Program Files (x86)\Xena Networks\L2-3\L23Tools>Valkyrie2544.exe -ec MyConfig.v2544

C:\Program Files (x86)\Xena Networks\L2-3\L23Tools>INFO: Loading test configuration MyConfig.v2544
INFO: Activating chassis: 192.168.1.170
INFO: Logon to 192.168.1.170:22606 successful
INFO: All chassis connected and read
INFO: Starting subtest 'Throughput Test'
INFO: Reserving ports
INFO: Resetting used ports
INFO: Port post-reset delay
INFO: Configuring ports
INFO: Toggling port sync state
INFO: InitTrial: 512 byte packets, iteration 1
INFO: Setting up streams
INFO: Sending L2 learning data stream preamble
INFO: Rate iteration: 100.000% traffic rate
INFO: Traffic on source port(s) started
INFO: Traffic on source port(s) stopped
INFO: Ending subtest 'Throughput Test'
INFO: All tests completed
INFO: PDF report written to [C:\Users\ole\Documents\Xena\Valkyrie2544\Reports\valkyrie2544-Report-20190717-134459
.pdf]
INFO: Test execution completed

C:\Program Files (x86)\Xena Networks\L2-3\L23Tools>
```

At prompt type:
mono ./Valkyrie2544.exe –c monotest01.v2544 –e
You then get:

```
INFO: Logon to 192.168.1.170:22606 successful
INFO: All chassis connected and read
INFO: Starting subtest 'Throughput Test'
INFO: Reserving ports
INFO: Resetting used ports
INFO: Configuring ports
INFO: Toggling port sync state
INFO: InitTrial: 512 byte packets, iteration 1
INFO: Setting up streams
INFO: Sending MAC learning packets
INFO: Rate iteration: 100.000% traffic rate
INFO: Traffic on source port(s) started
INFO: Traffic on source port(s) stopped
INFO: Ending subtest 'Throughput Test'
INFO: All tests completed
INFO: PDF report written to [/home/mikkel/Xena/Xena2544-2G/Reports/xena2544-repo
rt-20150904-141323.pdf]
INFO: Test execution completed
[mikkel@localhost xenabin]$
```

Test Stages

Pre-Test – Configuration Stage:

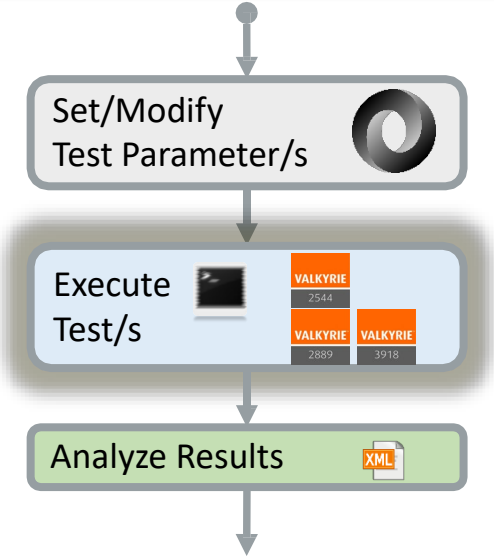



```
# Import JSON Library
import json

# Read configuration file to variable
with open('Configuration.v2544' ,'r',encoding='utf-8') as data_file:
    v2544_Configuration = json.loads(data_file.read())

# Modify Value of variable (e.g. TestOptions->PacketSizes->PacketSizeType)
v2544_Configuration['TestOptions']['PacketSizes']['PacketSizeType'] = 'ITUTDefault'

# Write modified to the file that is expeted To(2) Be(b) Used.
with open('2bUsed.v2544', 'w',encoding='utf-8') as f:
    json.dump(v2544_Configuration, f, indent = 2,sort_keys = True ,ensure_ascii=True)
```

Test Stages

Live Test – Execute(/Monitor) Stage:

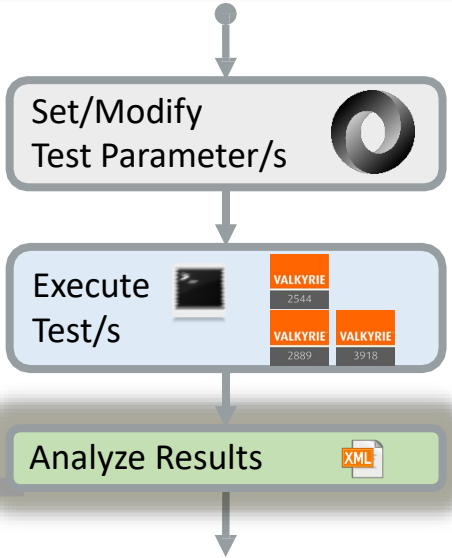
```
# Import subprocess Library
import subprocess

# Call the exe using “-ec” and the “2bUsed” Configuration file
subprocess.call(["C:\Program Files (x86)\Xena Networks\L2-3\L23Tools\Valkyrie2544.exe","-ec","2bUsed.v2544"])

# Linux example
args = ["mono", "./tools/pkt_gen/xena/Valkyrie2544.exe", "-ec",
        "./tools/pkt_gen/xena/profiles/2bUsed.v2544"]

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



Test Stages

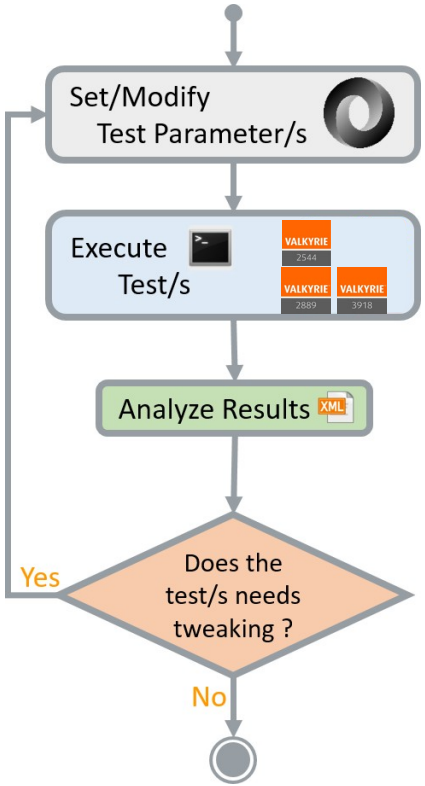
Post Test – Analysis Stage:

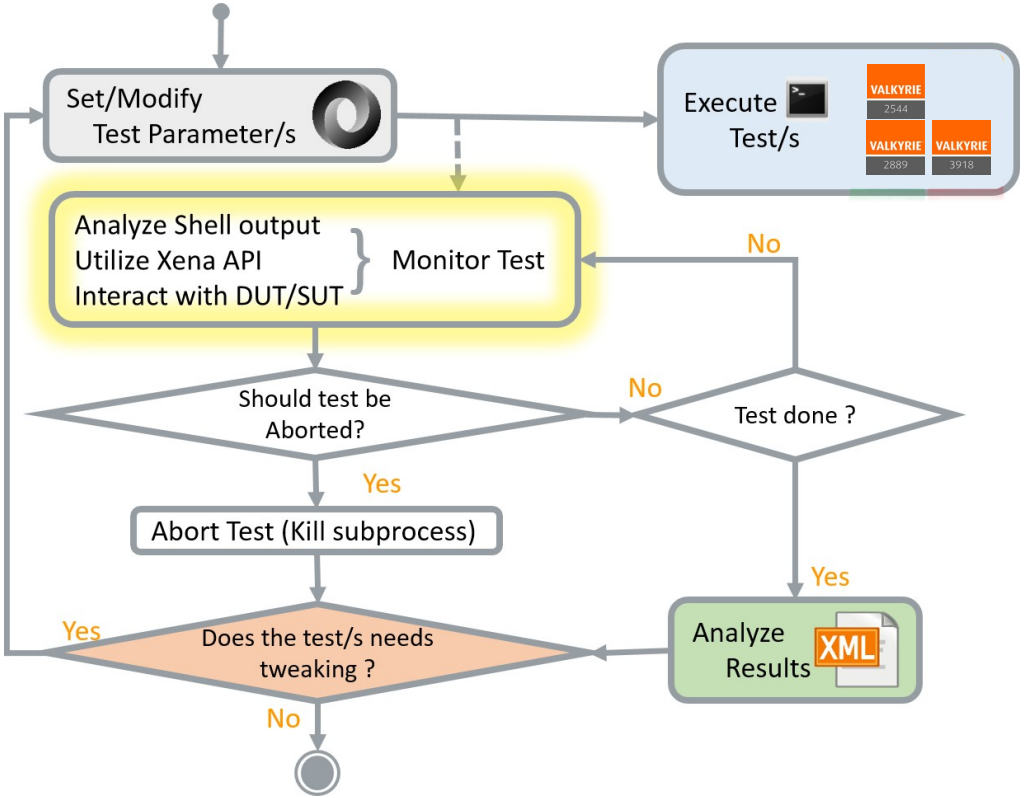
```
# Import ElementTree Library for easy XML parsing
import xml.etree.ElementTree as ET

# 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')
```



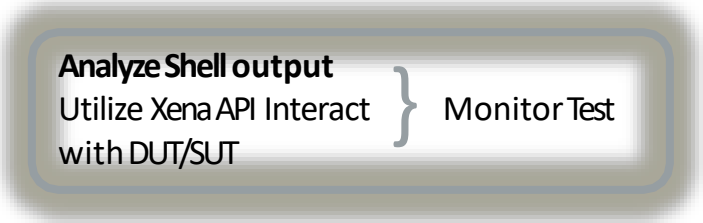

```
<?xml version="1.0" encoding="utf-8"?>
<!--Valkyrie2544 Test Report-->
<Valkyrie2544>
  <TestResults>
    <Summary>
      <Identification TestCompany="Xena Networks" Customer="Xena Networks" CustomerAccessID="" CustomerServiceID="" />
      <Metrics TestDateTime="20190923-095244" TestDuration="25" GeneratedBy="Valkyrie2544 v2.63" />
      <Comment><![CDATA[]]></Comment>
    </Summary>
    <TestResult TestType="Throughput">
      <Result FrameSize="512" TestState="PASS" TotalTxRatePcnt="100" TotalTxFrames="469924" TotalTxRateBpsL1="1999996544"
        <Port ID="P-0-11-0" PortTxFrames="234962" PortTxBpsL1="999998272" PortTxBpsL2="962404352" PortTxPps="234962" Port
        <Port ID="P-0-11-1" PortTxFrames="234962" PortTxBpsL1="999998272" PortTxBpsL2="962404352" PortTxPps="234962" Port
      </Result>
    </TestResult>
  </TestResults>
  <TestConfiguration>
</Valkyrie2544>
```

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)

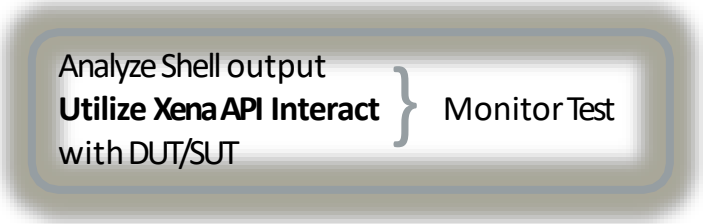
Use **Communicate** to retrieve Shell output:

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.**