ixnet

Library scope: global
Named arguments: supported

Introduction

provides functions for IxNetwork

RENAT will connect to the App server and control the test ports. Test files and result will be insde the RENAT server.

In order to run RENAT test case with *IxLoad*, the *TCLServer* must be activated with *Administrator* privileges on the Ixia App server.

Notes: Ignore the *self* parameters when using those keywords.

Shortcuts

Add Port · **A**dd Quicktest · **A**pply Traffic · **C**hange Frame Rate · **C**hange Frame Rate Dynamic · **C**hange Frame Size · **C**lose · **C**ollect All Data · **C**ollect Data · **G**et All Test Result · **G**et Quicktest List · **G**et Quicktest Result · **G**et Quicktest Result Path · **G**et Test Report · **G**et Test Result · **L**oad And Start Traffic · **L**oad Config · **L**oad Traffic · **L**oss From File · **P**ing · **R**eset Config · **R**un Quicktest · **S**et All Traffic Item · **S**et Bgp Items · **S**et Bgp Neighbor · **S**et Capture Port · **S**et Traffic Item · **S**hould Be Pingable · **S**tart Capture · **S**tart Protocol · **S**tart Traffic · **S**top All Protocols · **S**top And Save Capture · **S**top Quicktest · **S**top Traffic · **W**ait Until Connected

Keywords

Keyword	Arguments	Documentation

Add Port	*self*, *force=True*, *time_out=2m*, *learn_time=2m*	Add ports using the information from active local config
		• `time_out` is the wait time until port is connected (default is 2m)
		• `learn_time` is the time waiting for arp to be learned (default is 2m)
		Sample of local config |tester:
		`tester:`
		`  device: ixnet03_8009`
		`  config: quicktest.ixncfg`
		`  real_port:`
		`  - chassis: 10.128.4.41`
		`  card: 4`
		`  port: 3`
		`  ip: 10.100.11.2`
		`  mask: 24`
		`  gw: 10.100.11.1`
		`  - chassis: 10.128.4.41`
		`  card: 4`
		port: 4
		`  ip: 10.100.14.2`
		`  mask: 24`
		`  gw: 10.100.14.1`

Add Quicktest	*self*, *name*, *test_type=rfc2544throughput*, *tx_mode=interleaved*, *clear_all=True*	Create a new Quicktest with default value Type could be one of following: `rfc2544throughput`,*rfc2544frameLoss,rfc2544back2back* `Use Tester.Load Config` to load a customized quicktest When `clear_all` is True, any existed quicktests will be cleared. Transmit mode `tx_mode` takes following values: `interleaved` (default) or `sequential`. The mode should be identical with the transmit mod of the ports. **Notes**: The keyword **does not** create necessary ports. It should be used with a existed configuration by Tester.*Load Config* or Tester.*Add Port* keyword.
Apply Traffic	*self*	Applies the current traffic configuration **Note:** This is a blocking command
Change Frame Rate	*self*, *value*, *pattern=.**	Changes the frame rate Parameter: • `value`: value to set. Depends on the current configuration, this could be `percent line rate` or `bit per second` etc. • `traffic_pattern`*: a regular expression to identify traffic item name, default is everything* `.*`
Change Frame Rate Dynamic	*self*, *value*, *pattern=.**	Changes the traffic flow rate on-fly No need to stop the running traffic to change the rate Parameter: • `value`: value to set. Depend on the current configuration, this could be `percent line rate` or `bit per second` etc. • `pattern`*: a regular expression to identify traffic item name, default is everything* `.*`
Change Frame Size	*self*, *type*, *value*, *pattern=.**	Changes the frame size Parameter:

		▪ `type` : could be `fixed size` , `increment_from`,`increment_step` or `increment_to` ▪ `value` : value to set ▪ traffic_pattern*: a regular expression to identify traffic item name, default is everything* `.*
Close	*self*	Disconnects the current tester client
Collect All Data	*self*, *prefix=stat_*	Deprecated. Use
Collect Data	*self*, *view*, *prefix=stat_*	Depricated. Use *Get Test Result*
Get All Test Result	*self*, *prefix=stat_*	Collects all Ixia traffic data after traffic is stopped. Results are CSV files that are stored in `result` folder. The prefix `prefix` is appended to the original view name
Get Quicktest List	*self*	Returns current loaded Quicktest list
Get Quicktest Result	*self*, *test_index=-1*, *prefix=*, *enable_all=True*	Get the result.csv file from the latest Quicktests `test_index` is a index of the current Quicktest. `-1` means that last one.
Get Quicktest Result Path	*self*, *test_index=-1*	Returns the path of the newest run of a Quicktest `test_index` is a index of the current Quicktest. `-1` means that last one.
Get Test Report	*self*, *local_name=ixnet_report.pdf*, *enable_all=True*	Generates and get report of the current active test in PDF format `local_name` : name of the report on local machine. Default is `ixnet_report.pdf`
Get Test Result	*self*, *view*, *prefix=stat_*	Collects traffic data of a `view` and export to a CSV file in `result` folder Currently, supported views are: `Port Statistics` , `Global Protocol Statistics` , `BGP Aggregated Statistics` , `BGP Aggregated State Counts` , `OSPF Aggregated Statistics` , `OSPF Aggregated State Counts` , `OSPFv3 Aggregated Statistics` , `OSPFv3 Aggregated State Counts` , `L2-L3 Test Summary Statistics` , `Flow Statistics` , `Flow Detective` , `Data Plane Port Statistics` , `User Defined Statistics` , `Traffic Item Statistics` Result were store as CSV files in `result` folder. If there is no valid data, view will be silently ignored The prefix `prefix` is appended to the view name for the CSV file.
Load And Start Traffic	*self*, *wait_time1=10s*, *wait_time2=10s*	Combines *Load Traffic* and *Start Traffic* to one keyword.
Load Config	*self*, *config_name=*, *wait_time=2m*, *wait_time2=2m*, *apply=True*, *protocol=True*, *force=True*, *tx_mode=interleaved*	loads traffic configuration, applies and start protocol if necessary. The config file name was defined in the *`local.yaml* which is a Ixia Network configuration file and located in the *config* folder of the test. The keyword remap the vports to real port when data is specified in the local configuration file. For some reasons, the txMode is cleared when remapping happens. Use `tx_mode` to set the TxMode of the remapped ports. Parameters: ▪ `apply` : applies traffic when `True` otherwise ▪ `protocol` : starts all protocols when `True` otherwise ▪ `force` : force to reclaim the ports when `True` otherwise ▪ `tx_mode` : *sequential* or *interleaved*(default) ▪ `wait_time` : wait time after applying protocols ▪ `wait_time2` : maximum wait time befor all ports become available. In common case, this is calculated automatically so user does not need to change this value. See Common for more details about the yaml configuration files.
Load Traffic	*self*, *wait_time=2m*, *wait_time2=2m*, *apply=True*, *protocol=True*, *force=True*, *tx_mode=interleaved*	
Loss From File	*self*, *file_name=Flow_Statistics.csv*, *tx_frame_i=3*, *frame_delta_i=5*, *time1_i=23*, *time2_i=24*	Returns `packet loss` by miliseconds and delta frame. The calculation should be performed when traffic is stopped. The calculation supposed traffic is configured by frame per second

Ping	*self*, *dst_ip*, *src_port_index=0*, *src_intf_index=0*	Ping from Ixia to `dst_ip`
		The keyword return the output string as it is. The return could be
		`Port <portName>: ping failed: port not assigned`
		`Response received from <sourceIp>/unknown . Sequence Number <sequenceNumber>`
		`Ping request to <destinationIp>/unknown ip failed: <GenericPingError>/<error>: <genericError>unknown reason`
		`Error: Couldn't find any source interface for Send Ping to <destinationIp> on <portName> Id <id>`
		Error: Couldn't find any source IP for Send Ping to <destinationIp> on <portName> Id <id>
		Parameters:
		▪ src_port_index: index of Ixia port (starts from 0)
		▪ src_intf_index: index of interface insides the port (starts from 0)
		Examples:
		`Tester.Ping 1.1.1.1 0 0`
		`Tester.Ping 1.1.1.1`

Reset Config	*self*	Clears current config and creates new blank config
Run Quicktest	*self*, *test_index=0*, *wait_until_finish=True*	Runs the Quicktest and wait until it finishes **Warning**: it could take a long time to finish a quicktest
Set All Traffic Item	*self*, *enabled=True*	Enables/Disables all traffic items at once
Set Bgp Items	*self*, *port_index*, *neighbor_index*, *route_range_index*, *is_enable*	Enables/Disables BGP entry by a set of port,neighbor,route_range index Parameters: ▪ port_index: index of the port ▪ neighbor_index: index of the neighbor or * ▪ route_range_index: index of the route range or `*' ▪ is_enable: ${TRUE} or ${FALSE} Note Examples:
Set Bgp Neighbor	*self*, **indexes*, ***kwargs*	Enables/Disables BGP entry by neighbor index kwargs contains following parameters: ▪ indexes: is a list of index of BGP neighbor (index is started from zero) ▪ vport_index: is the target vport index ▪ enabled: TRUE or FALSE Examples: Tester.*Set BGP Item* | 0 | 1 | vport_index=0 | enabled=${FALSE} Tester.*Set BGP Item* \| 0 \| 1 \| vport_index=1 \| enabled=${TRUE}
Set Capture Port	*self*, *data_mode=True*, *control_mode=True*, *port_index=0*	Capture packets for follow port port_index: is a index of current test port (start from 0) data_mode: capture data packets and save in <intf>_HW.cap file control_mode: capture controls packets and save in <intf>_SW.cap file **Note**: control_mode saves all control packets and data_mode only saves data packets. **Note**: control_mode saves all control packets and data_mode only saves data packet Examples: Tester.*Set Capture* | 0 Tester.*Set Capture* | control_mode=${TRUE} | 0 | 1
Set Traffic Item	*self*, **items*, ***kwargs*	Enables/Disables some traffic items items Parameters: ▪ items: a list of Ixia traffic item name ▪ enabled: False or True ,the mode to set traffic item to, default is True (enabled) **Note**: traffic item could be specified by ::<num> format. In this case the num is the order of traffic item count from zero. Returns True if all items are set coordinately or otherwise Examples: Set Traffic Item | Traffic Item 1 | Traffic Item 2 Set Traffic Item | @{item_list} Set Traffic Item | Traffic Item 1 | enabled = ${FALSE}
Should Be Pingable	*self*, *dst_ip*, *src_port_index=0*, *src_intf_index=0*	Ping from Ixia and raise an error if ping fails The keyword return *True* if succeeds
Start Capture	*self*, *wait_time=30s*	Start packet capture Target ports are set by the configuration file or by [Set Capture] keyword
Start Protocol	*self*, *wait_time=1m*	Starts all protocols and wait for wait_time Default wait_time is 1 minute. Make sure wait_time is big engouh to start all protocols.
Start Traffic	*self*, *wait_time=30s*	Starts the current traffic settiing and wait for wait_time. **Note:** This is a asynchronus action. After called, the keyword finishes immediatly but it will take a while before traffic starts By default the keyword will wait for 30 seconds.
Stop All Protocols	*self*, *wait_time=30s*	Stop all running protocols
Stop And Save Capture	*self*, *prefix=*, *wait_until_finish=True*, *monitor_interval=5s*	Stop current capture and save the resuls to folder specified by path Captured files will be saved in current result folder with prefix appended in their names. Examples: Tester.*Start Capture* Sleep | 10s Tester.*Stop And Save Capture* | ${RESULT_FOLDER}/capture.zip
Stop Quicktest	*self*, *test_index=0*	Stops a running test

Stop Traffic	*self*, *stop_protocol=False*, *wait_time=10s*	Stops the current traffic and wait for `wait_time` Parameters: ▪ stop_protocol: if `True` also stops all running protocols ▪ wait_time: time to wait after apply the command
Wait Until Connected	*self*, *timeout_str=5m*	Waits until ports become enabled and connected

Altogether 36 keywords.
Generated by Libdoc on 2018-03-20 02:58:13.