

RENAT

Library version: RENAT 0.1.11
Library scope: global
Named arguments: supported

Introduction

Document for RENAT framework

All in one pdf [renat.pdf](#)

Libraries

RENAT includes following libraries:

Common:

Common library of RENAT

VChannel:

Library controls connection to targets (servers, routers, ...)

Logger:

Library provides enhanced logging keywords

Optical:

Library provides keywords to control L1 switches, includes [mod_calient](#) mod, [mod_ntm](#) mod

Router:

Library provides keywords to control routers, includes [mod_juniper](#) mod , [mod_cisco](#) mod and [mod_gr](#) mod

Tester:

Library provides keywords to control testers, includes [mod_ixnet](#) , [mod_ixload](#) , [mod_ixbps](#) and [mod_avaproxy](#)

WebApp:

Common library for web application, includes 2 child libraries: [Samurai](#) and [Arbor](#)

Hypervisor:

Library provides keywords to control Hypervisor, included [mod_vmware](#)

LabKeyword:

Common lab keywords

Others

Changes:

Changes information

Choose each libraries for detail information and samples about keywords.

Shortcuts

Keywords

Keyword	Arguments	Documentation
---------	-----------	---------------

Altogether 0 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:14.



Common

Library version: RENAT 0.1.11
Library scope: global
Named arguments: supported

Introduction

Common library for RENAT

It loads config files and create necessary variables. The file should be the 1st library included from any test case.

Table of Contents

- [Configuration file](#)
- [Variables](#)
- [Shortcuts](#)
- [Keywords](#)

Configuration file

Global configuration

There are 2 important configuration files. The global configuration files (aka master files) include device information, authentication etc that are used for all the test cases in the suite. The local configuration file `local.yaml` includes information about nodes, tester ports etc. that are used in a specific test case.

At the beginning, the module makes a local copy the master files and initialize necessary variables.

The RENAT framework utilized the YAML format for its configurations file.

The master files folder is defined by `renat-master-folder` in `$RENAT_PATH/config/config.yaml`. Usually, users do not need to modify the master files. The most common case is when new device is deployed, the `device.yaml` need to be update so that device could be used in the test cases.

1. device.yaml: contains global device information

Each device information is store under `device` block and has the following format:

```
<node_name>
  type:      <device type>
  description: <any useful description>
  ip:        <the IPv4 address of the device>
```

Where `<node_name>` is the name of the device. It could be the name of a switch, router or a web appliance box and should be uniq between the devices. `<description>` is any useful information and `<ip>` is the IP that RENAT uses to access the device.

`<type>` is important because it will be used as the ky of the `access_template` in template file. Usually users do not need to invent a new type but should use the existed type. When a new platform need to be supported, a new type will be introduced with the correspon template and authentication information.

Samples:

```
device:
  apollo:
    type: ssh-host
    description: main server
    ip: 10.128.3.101
  artermis:
    type: ssh-host
    description: second server
    ip: 10.128.3.91
  vmx11:
    type: juniper
    description: r1
    ip: 10.128.64.11
  vmx12:
    type: juniper
    description: r2
    ip: 10.128.64.12
```

2. template.yaml: contains device template information

The template file contains information about how to access to the device and how it should polling information (SNMP only for now). Each template has the following format:

`<type>`: access: <ssh or telnet> auth: <plaint-text or public-key> profile: <authentication profile name> prompt: <a regular expression for the PROMPT of the CLI device> (optional) login_prompt: <a login PROMPT for CLI device> (optional) password_prompt:<a PROMPT for asking password of CLI device> (optional) append: <a pharase to append automatically for every CLI command that executes> on this device (optional) init: <an array of command that will be executed automatically after a sucessful login of CLI device> (optional)

Note: Becareful about the prompt field. Usually RENAT will wait until it could see the prompt in its output. A wrong prompt will halt the system until it is timed out.

Samples:

```
access-template:
  ssh-host:
    access: ssh
    auth: public-key
    profile: default
    prompt: \$
    append:
    init: unalias -a
  juniper:
    access: telnet
    auth: plain-text
```

```

profile: default
prompt: "(#|>) "
append: ' | no-more'
init:
cisco:
  access: ssh
  auth: plain-text
  profile: default
  prompt: "\@.*(#|>) "
  append:
  init:
snmp-template:
  juniper:
    mib: ./mib-Juniper.json
    community: public
    poller: renat
  cisco:
    mib: ./mib-Cisco.json
    community: public

```

3. auth.yaml: contains authentication information

The file contains authentication information that system uses when access to a device. Each authentication type has following format:

```

plain-text
<profile>
  user: <user name>
  password: <password>

```

or

```

public-key:
<profile>:
  user: <user name>
  key: <public key path>

```

Where <profile> is the name of the authentication profile specified in the `access template` of the device

Sample:

```

auth:
  plain-text:
    default:
      user: user
      pass: nttXXX
    flets:
      user: user
      pass: lpcoXXXX
    arbor:
      user: admin
      pass: nttXXX

  public-key: # for Public Key authentication
    default:
      user: robot
      key: /home/user/.ssh/robot_id_rsa
    test:
      user: jenkins
      key: /var/lib/jenkins/.ssh/id_rsa

```

Local Configuration

Local configuration (aka `local.yaml`) was used by a test case or its sub test cases. Test cases could include several test cases (the sub level is not limited). The local configuration is defined by `local.yaml` in the `config` folder of each test case. If a test case does not have the `local.yaml` in its `config` folder, it will use the `local.yaml` file in its parent test case and so on. This will help users to share the test information for related test cases without having the same `local.yaml` for each test case (**Note:** this feature is enabled from RENAT 0.1.4). The `local.yaml` that is really used for the test is called `active local.yaml`.

When user used the wizard `item.sh` to create a new test case, they have the ability to create new `local.yaml` or not. `local.yaml` could be edited and inserted new information later to hold more information for the test case.

When a test is run, it will display its current active `local.yaml`

The local configuration file of each test item is stored in the `config` folder of the item as `local.yaml`

Usually the `local.yaml` has following parts:

- CLI node information: started by `node` keyword
- WEB node information: started by `webapp` keyword
- Tester device information: started by `tester` keyword
- Default information: automatically created and started by `default` keyword
- And other necessary information for the test by `yaml` format

Sample:

```

# CLI node
node:
  vmx11:
    device: vmx11
    snmp_polling: yes
  vmx12:
    device: vmx11
    snmp_polling: yes
  apollo:
    device: vmx11
    snmp_polling: yes

```

```
# web application information
webapp:
  arbor-sp-a:
    device: arbor-sp-a
    proxy:
      http: 10.128.8.210:8080
      ssl: 10.128.8.210:8080
      socks: 10.128.8.210:8080

# Tester information
tester:
  tester01:
    type: ixnet
    ip: 10.128.32.70
    config: vmx_20161129.ixncfg

# Other user information|
port-mapping:
  uplink01:
    device: vmx11
    port: ge-0/0/0
  downlink01:
    device: vmx12
    port: ge-0/0/2

# Default information
default:
  ignore_dead_node: yes
  terminal:
    width: 80
    height: 32
  result_folder: result
```

Variables

The module automatically create `GLOBAL` & `LOCAL` variable for other libraries. It also creates global list variables `GLOBAL`, `LOCAL` and `NODE` that could be accessed from Robot Framework test cases.

The `GLOBAL` variable holds all information defined by the master files and `LOCAL` variable holds all variables defined by active `local.yaml`. And `NODE` is a list that hold all active nodes defined in the `local.yaml`.

Users could access to the information of a key in `local.yaml` by `$(LOCAL['key'])`, information of a node by `$(LOCAL['node']['vmx11'])` or simply `$NODE['vmx']`. When a keyword need a list of current node, `@{NODE}` could be used.

Notes: By default, RENAT will stop and raise an exception if connection to a node is failed. But if `ignore_dead_node` is defined as `yes` (default) is the current active `local.yaml`, RENAT will omit an warning but keep running the test and remove the node from its active node list.

Shortcuts

Change Mod · Cleanup Result · Close Display · Convert Html To Pdf · Convert Xml · Count Keyword · Count Keyword Line · Count Match Regexp · Create Sequence · Csv Add · Csv Concat · Csv Create · Csv Merge · Csv Select · Diff File · Err · Error Line Should Not Be Bigger Than · Error Should Not Be Bigger Than · Explicit Run · File Md5 · Fold Str · Follow Syslog And Trap · Get Config Path · Get Config Value · Get File Without Error · Get Item Config Path · Get Item Name · Get Myid · Get Renat Path · Get Result Folder · Get Result Path · Get Test Device · Is Stable · Keyword Line Should Not Be Bigger Than · Keyword Should Not Be Bigger Than · Load Plugin · Log · Log Csv · Log To Console · Loop For Node Tag · Md 5 · Merge Files · Mib For Node · Node With Attr · Node With Tag · Node Without Tag · Pause · Ping Until Ok · Random Name · Random Number · Renat Version · Screenshot · Send · Set Multi Item Variable · Set Result Folder · Slack · Start Display · Str 2 Seq · Version · Wait

Keywords

Keyword	Arguments	Documentation
Change Mod	<i>name, mod, relative=False</i>	Changes file mod, likes Unix chmod <code>mod</code> is a string specifying the privilege mode <code>relative</code> is <code>False</code> or <code>True</code> Examples: <div>Common.Change Mod tmp 0775</div>
Cleanup Result	<i>ignore=^(log.html output.xml report.html)\$</i>	Cleans up the result folder Deletes all files in current active folder that does not match the <code>ignore</code> expression and are older than the time the test has started. Note: The keyword only removes files but not folders
Close Display		Closes the opened display
Convert Html To Pdf	<i>html_file, pdf_file</i>	Converts html file to pdf file
Convert Xml	<i>style, src, dst</i>	Converts XML by using XLS stylesheet Predefined stylesheets are store in <i>tools/xls</i> under current active RENAT folder Parameters: <ul style="list-style-type: none"> style: path to stylesheet src: path to the XML source dst: path to the output file
Count Keyword	<i>keyword, *pattern_list</i>	Count the keyword in files. Keyword is not case-sensitive
Count Keyword Line	<i>keyword, *pattern_list</i>	Count the number of lines contains the <code>keyword</code>

Count Match Regexp	regexp, *pattern_list	<p>Notes: Keyword is matched partially. For example, <code>error</code> or <code>errorXXX</code> will be matched by <code>error</code> keyword. Count the number of <code>regex</code> found in <code>pattern_list</code></p> <p>Examples:</p> <table><tr><td><code>\$(err_num)=</code></td><td><code>Count Match RegExp</code></td><td><code>.error.*</code></td><td><code>result/*.csv</code></td><td><code>result/*.txt</code></td></tr></table>	<code>\$(err_num)=</code>	<code>Count Match RegExp</code>	<code>.error.*</code>	<code>result/*.csv</code>	<code>result/*.txt</code>																											
<code>\$(err_num)=</code>	<code>Count Match RegExp</code>	<code>.error.*</code>	<code>result/*.csv</code>	<code>result/*.txt</code>																														
Create Sequence	start, end, interval, option=float	<p>Creates a list with number from <code>start</code> to <code>end</code> with <code>interval</code></p> <p>Example:</p> <table><tr><td><code>@{list}=</code></td><td><code>Create Sequence</code></td><td><code>10</code></td><td><code>15</code></td><td><code>0.5</code></td></tr></table> <p>will create a list of <code>[11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5]</code></p>	<code>@{list}=</code>	<code>Create Sequence</code>	<code>10</code>	<code>15</code>	<code>0.5</code>																											
<code>@{list}=</code>	<code>Create Sequence</code>	<code>10</code>	<code>15</code>	<code>0.5</code>																														
Csv Add	pathname, *items	<p>Add more data define by a list <code>items</code> to a existed CSV file</p> <p>Note:: do not check the consistency between item's number and header's number</p>																																
Csv Concat	src_pattern, dst_name, input_header=None, result_header=True	<p>Concatinates CSV files vertically If the CSV files has header, set <code>has_header</code> to <code>\$(TRUE)</code></p> <p>Examples:</p> <table><tr><td>Common.<code>CSV Concat</code></td><td>config/data0[3,4].csv</td><td>result/result2.csv</td><td></td></tr><tr><td>Common.<code>CSV Concat</code></td><td>config/data0[3,4].csv</td><td>result/result2.csv</td><td><code>has_header=\$(TRUE)</code></td></tr></table>	Common. <code>CSV Concat</code>	config/data0[3,4].csv	result/result2.csv		Common. <code>CSV Concat</code>	config/data0[3,4].csv	result/result2.csv	<code>has_header=\$(TRUE)</code>																								
Common. <code>CSV Concat</code>	config/data0[3,4].csv	result/result2.csv																																
Common. <code>CSV Concat</code>	config/data0[3,4].csv	result/result2.csv	<code>has_header=\$(TRUE)</code>																															
Csv Create	pathname, *header	<p>Create a CSV file with headers defined by a list <code>header</code></p> <p>The CSV file is opend with <code>UTF-8</code> encoding mode</p>																																
Csv Merge	src_pattern, dst_name, input_header=None, key=0, select_column=, result_header=True	<p>Merges all CSV files horizontally by <code>key</code> key from <code>src_pattern</code></p> <p><code>input_header</code> defines whether the input files have header row or not. If <code>input_header</code> is <code>\$(NULL)</code>, the keyword assume that input files have no header and automatically define columns name. When <code>input_header</code> is not null (default is zero), the row define by <code>input_header</code> will be used as header and data is counted from the next row.</p> <p><code>select_column</code> is a string that define the output columns and <code>key</code> is the column name that used to merge. When <code>input_header</code> is <code>\$(NULL)</code>, <code>select_column</code> and <code>key</code> is the index of columns. Otherwise, they are <code>column name</code>.</p> <p>The result header (column names) is decided by <code>result_header</code> (<code>True</code> or <code>False</code>)</p> <p>The keyword returns <code>False</code> if no file is found by the pattern</p> <p>Examples:</p> <table><tr><td>Common.<code>CSV Merge</code></td><td>config/data0[3,4].csv</td><td>result/result2.csv</td><td></td></tr><tr><td>Common.<code>CSV Merge</code></td><td>config/data0[3,4].csv</td><td>result/result2.csv</td><td><code>input_header=0</code></td></tr><tr><td>Common.<code>CSV Merge</code></td><td><code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code></td><td><code>input_header=0</code></td><td></td></tr><tr><td>...</td><td><code>dst_name=\$(RESULT_FOLDER)/result.csv</code></td><td><code>result_header=\$(FALSE)</code></td><td></td></tr><tr><td>...</td><td><code>key=Stat Name</code></td><td><code>select_column=Valid Frames Rx.</code></td><td></td></tr><tr><td>Common.<code>CSV Merge</code></td><td><code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code></td><td><code>input_header=\$(NULL)</code></td><td></td></tr><tr><td>...</td><td><code>dst_name=\$(RESULT_FOLDER)/result.csv</code></td><td><code>result_header=\$(FALSE)</code></td><td></td></tr><tr><td>...</td><td><code>key=0</code></td><td><code>select_column=5</code></td><td></td></tr></table>	Common. <code>CSV Merge</code>	config/data0[3,4].csv	result/result2.csv		Common. <code>CSV Merge</code>	config/data0[3,4].csv	result/result2.csv	<code>input_header=0</code>	Common. <code>CSV Merge</code>	<code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code>	<code>input_header=0</code>		...	<code>dst_name=\$(RESULT_FOLDER)/result.csv</code>	<code>result_header=\$(FALSE)</code>		...	<code>key=Stat Name</code>	<code>select_column=Valid Frames Rx.</code>		Common. <code>CSV Merge</code>	<code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code>	<code>input_header=\$(NULL)</code>		...	<code>dst_name=\$(RESULT_FOLDER)/result.csv</code>	<code>result_header=\$(FALSE)</code>		...	<code>key=0</code>	<code>select_column=5</code>	
Common. <code>CSV Merge</code>	config/data0[3,4].csv	result/result2.csv																																
Common. <code>CSV Merge</code>	config/data0[3,4].csv	result/result2.csv	<code>input_header=0</code>																															
Common. <code>CSV Merge</code>	<code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code>	<code>input_header=0</code>																																
...	<code>dst_name=\$(RESULT_FOLDER)/result.csv</code>	<code>result_header=\$(FALSE)</code>																																
...	<code>key=Stat Name</code>	<code>select_column=Valid Frames Rx.</code>																																
Common. <code>CSV Merge</code>	<code>src_pattern=\$(RESULT_FOLDER)/balance*.csv</code>	<code>input_header=\$(NULL)</code>																																
...	<code>dst_name=\$(RESULT_FOLDER)/result.csv</code>	<code>result_header=\$(FALSE)</code>																																
...	<code>key=0</code>	<code>select_column=5</code>																																
Csv Select	src_file, dst_file, str_row=, str_col=, has_header=None	<p>Select part of the CSV file and write it to other file <code>str_row</code> and <code>str_col</code> are used to specify necessary rows and columns. They are using the same format with slice for Python list.</p> <ul style="list-style-type: none">▪ <code>:</code> and <code>:</code> means all rows and columns▪ <code>:2</code> and <code>:</code> means first 2 rows and all columns▪ <code>:</code> and <code>1,2</code> means all rows and 2nd and 3rd columns▪ <code>0:3</code> and <code>1</code> means 3 rows from the 1st one(0,1,2) and second column▪ <code>0:5:2</code> and <code>1</code> means 3 rows(0,3,5) and second column <p>Notes:</p> <ul style="list-style-type: none">▪ Rows and columns are indexed from zero▪ When <code>:</code> is used, the string has format: <code><start>:<stop></code> or <code><start>:<stop>:<step></code> For convenience, <code>:</code> means all the data, <code>'x'</code> means first 'x' data <p>Examples:</p> <table><tr><td><code>CSV Select</code></td><td>result/data05.csv</td><td>result/result3.csv</td><td>0,1,2</td><td>0,1</td></tr><tr><td><code>CSV Select</code></td><td>result/data05.csv</td><td>result/result4.csv</td><td>:</td><td>0,1</td></tr><tr><td><code>CSV Select</code></td><td>result/data05.csv</td><td>result/result5.csv</td><td>:2</td><td>:</td></tr><tr><td><code>CSV Select</code></td><td>result/data05.csv</td><td>result/result6.csv</td><td>0:3</td><td>:</td></tr><tr><td><code>CSV Select</code></td><td>result/data05.csv</td><td>result/result7.csv</td><td>0:5:2</td><td>:</td></tr></table>	<code>CSV Select</code>	result/data05.csv	result/result3.csv	0,1,2	0,1	<code>CSV Select</code>	result/data05.csv	result/result4.csv	:	0,1	<code>CSV Select</code>	result/data05.csv	result/result5.csv	:2	:	<code>CSV Select</code>	result/data05.csv	result/result6.csv	0:3	:	<code>CSV Select</code>	result/data05.csv	result/result7.csv	0:5:2	:							
<code>CSV Select</code>	result/data05.csv	result/result3.csv	0,1,2	0,1																														
<code>CSV Select</code>	result/data05.csv	result/result4.csv	:	0,1																														
<code>CSV Select</code>	result/data05.csv	result/result5.csv	:2	:																														
<code>CSV Select</code>	result/data05.csv	result/result6.csv	0:3	:																														
<code>CSV Select</code>	result/data05.csv	result/result7.csv	0:5:2	:																														
Diff File	path1, path2, newline=True	<p>Shows difference between files</p> <p>Returns the diff result (multi lines) <code>path1</code>, <code>path2</code> are absolute paths.</p>																																
Err	msg	Prints error msg to console																																
Error Line Should Not Be Bigger Than	num, *pattern_list	Checks whether the number of lines that contains <code>error</code> be less than a number																																
Error Should Not Be Bigger Than	num, *pattern_list	Checks whether the number of <code>error</code> be less than a number																																
Explicit Run		skip the test case if global_variable <code>RUN_ME</code> is not defined																																

		<div>Sample scenario:</div> <table><tr><td>00. Cabling</td><td></td></tr><tr><td>Common.<i>Explicit Run</i></td><td></td></tr><tr><td>Log To Console</td><td>cabling...</td></tr></table> <div>run.sh will bypass 00. Cabling by default. In other to run this test case <i>\$(FORCE)</i> needs declared globally run.sh -X -v FORCE</div>	00. Cabling		Common. <i>Explicit Run</i>		Log To Console	cabling...										
00. Cabling																		
Common. <i>Explicit Run</i>																		
Log To Console	cabling...																	
File Md5	<i>path</i>	Returns MD5 hash of a file <i>path</i> is an absolute path																
Fold Str	<i>str</i>	Folds a string by adding Non-Width-Space char (0x200b) at 6th char																
Follow Syslog And Trap	<i>pattern, log_file_name=syslog-trap.log, delay_str=1s</i>	Pauses the execution and wait for the pattern is matched if the file <i>log_file_name</i> located in the current result folder. By default the <i>log_file_name</i> is <i>./result/syslog-trap.log</i> which is created by <i>Follow Syslog and Trap</i> keyword. The keyword should be in tests between <i>Follow Syslog adn Trap Start</i> and <i>Follow Syslog and Trap Stop</i> keywords.																
Get Config Path		Returns absolute path of RENAT config folder path																
Get Config Value	<i>key, base=default, default=None</i>	Returns value of a key for renat configuration with this other LOCAL[base][key] > GLOBAL[base][key] > None																
Get File Without Error	<i>file_path</i>	Get content of the file and return null string if the file does not exist																
Get Item Config Path		Returns absolute path of current item config folder																
Get Item Name		Returns the name of the running item																
Get Myid																		
Get Renat Path		Returns the absolute path of RENAT folder																
Get Result Folder		Returns current result folder name. Default is <i>result</i> in current test case. Note: the keyword only returns the <i>name</i> of the result folder not its absolue path.																
Get Result Path		Returns absolute path of the current result folder																
Get Test Device		Return a list of all test device that is used in this test Notes: Device number could less than node number																
Is Stable	<i>seq, threshold, percentile=90</i>	Checks if the value sequence is stable or not																
Keyword Line Should Not Be Bigger Than	<i>num, keyword, *pattern_list</i>	Checks whether the number of line containing the keyword be less than a number																
Keyword Should Not Be Bigger Than	<i>num, keyword, *pattern_list</i>	Checks whether the number of keyword be less than a number																
Load Plugin		Load plugin in renat/plugin folder																
Log	<i>msg, level=1</i>	Logs <i>msg</i> to the current log file (not console) The <i>msg</i> will logged only if the level is bigger than the global level <i>\$(DEBUG)</i> which could be defined at runtime. If <i>\$(DEBUG)</i> is not defined, it will be considered as the default <i>level</i> as 1. Examples: <table><tr><td>Common.<i>Log</i> XXX</td><td># this always be logged</td><td></td></tr><tr><td>Common.<i>Log</i> AAA</td><td>level=2</td><td># this will not be logged with common run.sh</td></tr><tr><td>Common.<i>Log</i> BBB</td><td>level=2</td><td># ./run.sh -v DEBUG:2 will log the message</td></tr></table> Notes: For common use <ul style="list-style-type: none">level 1: is defaultlevel 2: is debug modelevel 3: is very informative mode	Common. <i>Log</i> XXX	# this always be logged		Common. <i>Log</i> AAA	level=2	# this will not be logged with common run.sh	Common. <i>Log</i> BBB	level=2	# ./run.sh -v DEBUG:2 will log the message							
Common. <i>Log</i> XXX	# this always be logged																	
Common. <i>Log</i> AAA	level=2	# this will not be logged with common run.sh																
Common. <i>Log</i> BBB	level=2	# ./run.sh -v DEBUG:2 will log the message																
Log Csv	<i>csv_file, index=False, border=0</i>	Logs a content of <i>csv_file</i> into default log.html <i>index, border</i> are table attributes																
Log To Console	<i>msg, level=1</i>	Logs a message to console See Common. <i>Print</i> for more details about debug level																
Loop For Node Tag	<i>var, tags, *keywords</i>	Repeatedly executes RF <i>keyword</i> for nodes that has tag <i>tags</i> multi tags are separated by <i>:</i> keywords has same meaning with <i>keywords</i> used by <i>Run Keywords</i> of RobotFramework (<i>keyword</i> and its arguments are separated by <i>AND</i> with the others. Example: <table><tr><td><i>Loop For Node Tag</i></td><td><i>\$(node)</i></td><td>tag1</td><td></td></tr><tr><td>...</td><td><i>Switch</i></td><td><i>\$(node)</i></td><td>AND</td></tr><tr><td>...</td><td><i>Cmd</i></td><td>show system user</td><td>AND</td></tr><tr><td>...</td><td><i>Cmd</i></td><td>show system uptime</td><td></td></tr></table> Note: <i>\$</i> in variable name must be escaped	<i>Loop For Node Tag</i>	<i>\$(node)</i>	tag1		...	<i>Switch</i>	<i>\$(node)</i>	AND	...	<i>Cmd</i>	show system user	AND	...	<i>Cmd</i>	show system uptime	
<i>Loop For Node Tag</i>	<i>\$(node)</i>	tag1																
...	<i>Switch</i>	<i>\$(node)</i>	AND															
...	<i>Cmd</i>	show system user	AND															
...	<i>Cmd</i>	show system uptime																
Md 5	<i>str</i>	Returns MD5 hash of a string																
Merge Files	<i>path_name, file_name</i>	Merges all the text files defined by <i>path_name</i> to <i>file_name</i>																

		<p>Example:</p> <div><div>Merge Files</div><div>./result/*.csv</div><div>./result/test.csv</div></div>
Mib For Node	node	<p>Returns the mib file name for this node mib file is define by mib keyword under the node in local.yaml</p> <div><pre>... node: vmx11: device: vmx11 snmp_polling: yes mib: mib11.txt ...</pre></div> <p>Default value is defined by mib keyword from global config/snmp-template.yaml for the type of the node</p> <p>Example:</p> <div><div>\$(mib)=</div><div>Common.MIB For Node</div><div>vmx11</div></div>
Node With Attr	attr_name, value	Returns a list of nodes which have attribute attr_name with value value
Node With Tag	*tag_list	<p>Returns list of node or webapp from local.yaml that has ALL tags defined by tag_list</p> <p>Tag was defined like this in local.yaml</p> <div><pre>vmx11: device: vmx11 snmp_polling: yes tag: - tag1 - tag2</pre></div> <p>Examples:</p> <div><div>\$(test3)=</div><div>Common.Node With Tag</div><div>tag1</div><div>tag3</div></div>
Node Without Tag	*tag_list	<p>Returns list of node from local.yaml that does not has ANY tags defined by tag_list</p> <p>Tag was defined like this in local.yaml</p> <div><pre>vmx11: device: vmx11 snmp_polling: yes tag: - tag1 - tag2</pre></div> <p>Examples:</p> <div><div>\$(test3)=</div><div>Common.Node Without Tag</div><div>tag1</div><div>tag3</div></div>
Pause	msg=, time_out=3h, error_on_timeout=True, default_input=	<p>Displays the message msg and pauses the test execution and wait for user input</p> <p>In case of error_on_timeout is True(default), the keyword will raise an error when timeout occurs. Otherwise, it will continue the test.</p> <p>Notes: If the variable \${RENAT_BATCH} was defined, the keyword will print out the message and keeps running without pausing.</p> <p>Examples:</p> <div><div>Common.Pause</div><div>Waiting...</div><div>10s</div><div>error_on_timeout=\${TRUE}</div><div>default input</div></div> <div><div>Common.Pause</div><div>Waiting...</div><div>10s</div><div></div><div></div></div>
Ping Until Ok	node, wait_str=5s, extra=-c 3	Ping a node until it gets response. Then wait for more wait_str Default extra option is -c 3
Random Name	base, a=0, b=99	<p>Returns a random name by a base and a random number between [a,b]</p> <p>Example:</p> <div><div>\$(FOLDER)=</div><div>Random Name</div><div>capture_%05d</div><div>0</div><div>99</div></div>
Random Number	a=0, b=99	Returns a random number between [a,b]
Renat Version		Returns RENAT version string
Screenshot	file_path	<p>Capture whole display to a file specified by file_path</p> <p>Notes: This keyword saves the whole virtual screen(monitor), while the familiar WebApp.Screenshot Capture only saves the portion of the web browser. But in contrast, the WebApp.Screenshot Capture could do fullpage capture depending on the content of the browser.</p>
Send	sock, data, recv_buffer_size=1024, encode=utf-8	<p>Sends bytes of data by socket sock and reicve the response</p> <p>When recv_buffer_size is zero, the function does not execept a response from the remote.</p>
Set Multi Item Variable	*vars	<p>Set multiple variables to be suite variable at the same time</p> <p>Suite variables (or item variable) could be access anywhere in all the item scenario.</p>
Set Result Folder	folder	<p>Sets the result folder to folder and return the old result folder. The result folder contains all output files from the test likes tester ouput, config file ...</p> <p>folder is a folder name that under current test case folder</p> <p>The system will create a new folder if it does not exist and set its mode to 0775</p> <p>Note: Result folder should be set at the begining of the test. Changing result folder only has effect on up comming connection</p>
Slack	msg, channel=#automation_dev, user=renat, host=10.128.3.103:4713	Post a message to Slack

Start Display		Starts a virtual display																
Str 2 Seq	<i>str_index, size</i>	Returns a sequence from string format Samples: <table><tr><td><i>Str2Seq</i></td><td>::</td><td>5</td><td># (0,1,2,3,4)</td></tr><tr><td><i>Str2Seq</i></td><td>:2</td><td>5</td><td># (0,1)</td></tr><tr><td><i>Str2Seq</i></td><td>1:3</td><td>5</td><td># (1,2)</td></tr><tr><td><i>Str2Seq</i></td><td>0:5:2</td><td>5</td><td># (0,2,4)</td></tr></table>	<i>Str2Seq</i>	::	5	# (0,1,2,3,4)	<i>Str2Seq</i>	:2	5	# (0,1)	<i>Str2Seq</i>	1:3	5	# (1,2)	<i>Str2Seq</i>	0:5:2	5	# (0,2,4)
<i>Str2Seq</i>	::	5	# (0,1,2,3,4)															
<i>Str2Seq</i>	:2	5	# (0,1)															
<i>Str2Seq</i>	1:3	5	# (1,2)															
<i>Str2Seq</i>	0:5:2	5	# (0,2,4)															
Version		Returns the current version of RENAT																
Wait	<i>wait_time, size=10</i>	Waits for <i>wait-time</i> and display the proress bar <i>wait_time</i> used RF <i>DateTime</i> format. Examples: <table><tr><td>Common.</td><td><i>Wait</i></td><td>wait_time=30s</td><td>size=10</td></tr></table>	Common.	<i>Wait</i>	wait_time=30s	size=10												
Common.	<i>Wait</i>	wait_time=30s	size=10															

Altogether 60 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:50:57.



VChannel

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A basic library that provides Terminal connection to routers/hosts

VChannel is a core RENAT library that maintains input/output to nodes with an attached virtual terminal. It encapsulates the SSH/Telnet connections behind and provides common usage of access and execute commands to the nodes. Each channel instance has its own log file and a virtual terminal.

Table of Contents

- [Device, Node and Channel](#)
- [Connections](#)
- [Shortcuts](#)
- [Keywords](#)

Device, Node and Channel

RENAT has 3 types of connection target. Device, Node and Channel.

Device

Each device stands for a real physical box that has its own IP address and is defined in the master file `device.yaml`. Users do not directly use device in keywords.

Node

Node is a logical instance of a device. It could stand for a logical instance of a router or just a virtual terminal to the router. Nodes were defined in `local.yaml` of the test case. Several nodes could point to a same device.

Channel

Each channel holds a session to a node. Each channel has its own log file and a virtual terminal. Any command used by `Cmd`, `Write` or `Read` will be logged to the log file. Each channel is identified by a name when it is created with `Connect` keyword and is released with `Close` keyword.

Notes: multi sessions to a same device could be done with predefined multi nodes to same device in the `local.yaml` file or by using multi `Connect` with different `name`.

Connections

The library provides a channel to a target node. Each channel is attached with a virtual terminal. Input and output to the node are made through this virtual terminal. This will help to provide the output looks like the output when operator is using the real terminal.

When keywords `Read`, `Write`, `Cmd` are used, if the connection is not available anymore, the system will try to reconnect to the host with the information provided in the 1st connect. It will try `max_retry_for_connect` times and wait for `interval_between_retry` seconds between retries. The values of `max_retry_for_connect` and `interval_between_retry` are defined in `./config/config.yaml`

Usually when RENAT could not make the connections to the target, the system will raise an exception. But if the `ignore_dead_node` is defined as `yes` in the current active `local.yaml`, the system will ignore the dead node, remove it from the global variable `LOCAL[node]` and `NODE` and keep running the test.

Shortcuts

Change Log · Change Prompt · Close · Close All · Cmd · Cmd And Wait For · Cmd And Wait For Regex · Cmd More · Cmd Yesno · Connect · Connect All · Current Prompt · Exec File · Flush All · Get Channel · Get Channels · Get Current Channel · Get Current Name · Get Ip · Log · Read · Reconnect · Set Log Separator · Snap · Snap Diff · Start Screen Mode · Stop Screen Mode · Switch · Write

Keywords

Keyword	Arguments	Documentation												
Change Log	<code>log_file</code> , <code>mode=w</code>	Stops current log file and create a new log file. Every log from that point will be saved to the new log file Return old log filename												
Change Prompt	<code>str_prompt</code>	Changes the current prompt of the channel Returns previous prompt. User should change the prompt before execute the new command that expects to see new prompt. Example: <table><tr><td>Router.<code>Switch</code></td><td>vmx11</td><td></td></tr><tr><td>\${prompt}=</td><td>VChannel.<code>Change Prompt</code></td><td>%</td></tr><tr><td>VChannel.<code>Cmd</code></td><td>start shell</td><td></td></tr><tr><td>VChannel.<code>Cmd</code></td><td>ls</td><td></td></tr></table>	Router. <code>Switch</code>	vmx11		\${prompt}=	VChannel. <code>Change Prompt</code>	%	VChannel. <code>Cmd</code>	start shell		VChannel. <code>Cmd</code>	ls	
Router. <code>Switch</code>	vmx11													
\${prompt}=	VChannel. <code>Change Prompt</code>	%												
VChannel. <code>Cmd</code>	start shell													
VChannel. <code>Cmd</code>	ls													

		<div>VChannel.Change Prompt \${prompt}</div> <div>Vchannel.Cmdexit</div>												
Close		Closes current connection and returns the active channel name												
Close All		Closes all current sessions and flush out all log files. Current node name was reset to <code>None</code>												
Cmd	<i>command=, prompt=None, timeout=None, match_err= (unknown command./syntax error, expecting <command>.)</i>	Executes a <code>command</code> and wait until for the prompt. This is a blocking keyword. Execution of the test case will be postponed until the prompt appears. If <code>prompt</code> is a null string (default), its value is defined in the <code>./config/template.yaml</code> <i>timeout</i> is the timeout for this Cmd . If <i>timeout</i> is not define, the local <i>vchannel/cmd-timeout</i> or global <i>vchannel/cmd-timeout</i> will be used. The keyword returns error when the output matches the <code>match_err</code> and the default config value <i>cmd-auto-check</i> is <code>True</code> Output will be automatically logged to the channel current log file. See Common for details about the config files.												
Cmd And Wait For	<i>command, keyword, interval=30s, max_num=10, error_with_max_num=True</i>	Execute a command and expect <code>keyword</code> occurs in the output. If not wait for <code>interval</code> and repeat the process again After <code>max_num</code> , if <code>error_with_max_num</code> is <code>True</code> then the keyword will fail. Ortherwise the test continues.												
Cmd And Wait For Regex	<i>command, pattern, interval=30s, max_num=10, error_with_max_num=True</i>	Execute a command and expect <code>pattern</code> occurs in the output. If not wait for <code>interval</code> and repeat the process again When the keyword contains <code>not</code> : at the beginning, the matching logic is reversersed. After <code>max_num</code> , if <code>error_with_max_num</code> is <code>True</code> then the keyword will fail. Ortherwise the test continues.												
Cmd More	<i>cmd=, wait_prompt=.*---(more.*)---, press_key= , prompt=None</i>	Execute a command and press <i>press_key</i> when <i>wait_prompt</i> is displayed until the prompt												
Cmd Yesno	<i>cmd, ans=yes, question=? [yes,no] , timeout=5s</i>	Executes a <i>cmd</i> , waits for <i>question</i> and answers that by <i>ans</i>												
Connect	<i>node, name, log_file, timeout=20m, w=80, h=32, mode=w</i>	Connects to the node and create a VChannel instance Login information is automatically extracted from yaml configuration. By defaultt a virtual terminal (vty100) with size 80x64 is attached to this channel. If a login was successful, VChannel will create a log file name <i>log_file</i> for the connection in the current result folder of the test case. This log file will contain any command input/output executed on this channel. Multi sessions to the same node could be open with different names. Use Switch to change the current active session by its name Examples: <table><tr><td>Connect</td><td>vmx11</td><td>vmx11</td><td>vmx11.log</td><td></td><td></td></tr><tr><td>Connect</td><td>vmx11</td><td>vmx11</td><td>vmx11.log</td><td>80</td><td>64</td></tr></table> See Common for more detail about the yaml config files.	Connect	vmx11	vmx11	vmx11.log			Connect	vmx11	vmx11	vmx11.log	80	64
Connect	vmx11	vmx11	vmx11.log											
Connect	vmx11	vmx11	vmx11.log	80	64									
Connect All	<i>prefix=</i>	Connects to all nodes that are defined in active <i>local.yaml</i> . A prefix <i>prefix</i> was appended to the alias name of the connection. A new log file by <i><alias>.log</i> was automaticocally created. See Common for more detail about active <i>local.yaml</i>												
Current Prompt		Return current prompt												
Exec File	<i>file_name, vars=, comment=# , step=False, str_error=syntax,rror</i>	Executes commands listed in <i>file_name</i> Lines started with <i>comment</i> character is considered as comments <i>file_name</i> is a file located inside the <i>config</i> folder of the test case. This command file could be written in Jinja2 format. Default usable variables are <i>LOCAL</i> and <i>GLOBAL</i> which are identical to <i>Common.LOCAL</i> and <i>Common.GLOBAL</i> . More variables could be supplied to the template by <i>vars</i> . <i>vars</i> has the format: <i>var1=value1,var2=value2</i> If <i>step</i> is <code>True</code> , after very command the output is check agains an error list. And if a match is found, execution will be stopped. Error list is define by <i>str_err</i> , that contains multi regular expression separated by a comma. Default value of <i>str_err</i> is <i>error</i> A sample for command list with Jinja2 template: <pre>show interface {{ LOCAL['extra']['line1'] }} show interface {{ LOCAL['extra']['line2'] }}</pre>												

		<pre>{% for i in range(2) %} show interface et-0/0/{{ i }} {% endfor %}</pre> <p>Examples:</p> <table border="1"> <tr> <td>Router.Exec File</td><td>cmd.lst</td><td></td></tr> <tr> <td>Router.Exec File</td><td>step=\${TRUE}</td><td>str_error=syntax,error</td></tr> </table> <p>Note: Comment in the middle of the line is not supported For example if <code>comment</code> is "# "</p> <pre># this is comment line <-- this line will be ignored ## this is not an comment line, and will be entered to the router cli,</pre> <p>but the router might ignore this</p>	Router. Exec File	cmd.lst		Router. Exec File	step=\${TRUE}	str_error=syntax,error
Router. Exec File	cmd.lst							
Router. Exec File	step=\${TRUE}	str_error=syntax,error						
Flush All		Flushes all remain data into the logger						
Get Channel	<i>name</i>	Returns a channel by its <code>name</code>						
Get Channels		Returns all current vchannel instances						
Get Current Channel		Returns the current active channel						
Get Current Name		Returns the current active channel's name						
Get Ip		Returns the IP address of current node Examples:						
		<code>\$(router_ip)= Router.Get IP</code>						
Log	<i>msg</i>	Writes the log message <code>msg</code> to current log file of the channel						
Read	<i>silence=False</i>	Returns the current output of the virtual terminal and automatically logs to file. In <code>normal mode</code> this will return the unread output only, not all the content of the screen.						
Reconnect	<i>name</i>	Reconnects to the <code>name</code> node using existed information The only difference is that the mode of the log file is set to `a+` by default						
Set Log Separator	<i>sep=</i>	Set a separator between the log of <code>read</code> , <code>write</code> or <code>cmd</code> keywords						
Snap	<i>name, *cmd_list</i>	Remembers the result of a list of command defined by <code>cmd_list</code> Use this keyword with Snap Diff to get the difference between the command's result. The a new snapshot will override the previous result. Each snap is identified by its <code>name</code>						
Snap Diff	<i>name</i>	Executes the command that have been executed before by <code>name</code> snapshot and return the difference. Difference is in <code>context diff</code> format						
Start Screen Mode		Starts the <code>screen mode</code> . In the <code>screen mode</code> , the output is just the same with the real terminal. It means that any real-time application likes <code>top</code> will be captured as-is. Consecutive read from this VChannel instance may produce redundancy output.						
Stop Screen Mode		Stops the <code>screen mode</code> and returns to <code>normal mode</code> In <code>screen mode</code> , Write does not return any thing and no output is logged. In <code>normal mode</code> , escape sequences are not processed by the virtual terminal.						
Switch	<i>name</i>	Switches the current active channel to <code>name</code> . There only one active channel at any time Returns the current <code>channel_id</code> , <code>local_channel_id</code> and the output of current terminal. Notes: There is no assurance that the output of previous Write command will be in the return output because keywords like <code>Logger.Log All</code> will update every channels. Examples: <code>VChannel.Switch vmx12</code>						
Write	<i>str_cmd=, str_wait=0s, start_screen_mode=False</i>	Sends <code>str_cmd</code> to the target node and return after <code>str_wait</code> time. If <code>start_screen_mode</code> is <code>True</code> , the channel will be shifted to <code>Screen Mode</code> . Default value of <code>screen_mode</code> is <code>False</code> . In <code>normal mode</code> , a <code>new line</code> char will be added automatically to the <code>str_cmd</code> and the command return the output it could get at that time from the terminal and also logs that to the log file. In <code>screen Mode</code> , if it is necessary you need to add the <code>new line</code> char by your own and the output is not be logged or returned from the keyword. Parameters: <ul style="list-style-type: none"> <code>str_cmd</code>: the command <code>str_wait</code>: time to wait after apply the command <code>start_screen_mode</code>: whether start the <code>screen mode</code> right after writes the command 						

Special input likes Ctrl-C etc. could be used with global variable `${CTRL-<char>}`

Returns the output after writing the command the the channel.

When `str_wait` is not `0s`, the keyword `read` and `return` the output after waiting `str_wait`.
Otherwise, the keyword `return` without any output.

Notes: This is a non-blocking command.

Examples:

VChannel. <i>Write</i>	monitor interface traffic	start_screen_mode=\${TRUE}
VChannel. <i>Write</i>	<code>\${CTRL_C}</code>	# simulates Ctrl-C

Altogether 29 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:02.



Logger

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

Provides advanced logging functions. Every [Logger](#) instance has one [VChannel](#) object and the is synchronized with the current active [VChannel](#).

Notes: log file is not updated pararelly. Anytime a terminal is switched to, it will update its log file.

Shortcuts

Log · Log All · Switch

Keywords

Keyword	Arguments	Documentation								
Log	<i>msg,</i> <i>with_time=False,</i> <i>mark=***</i>	Inserts a message <code>msg</code> to the current <i>VChannel</i> log file. A default mark of <code>***</code> will be added at the beginning ant the end of this message. Example: <table><tr><td>Logger.Log</td><td>START TRAFFIC FROM HERE</td><td>\${TRUE}</td><td></td></tr><tr><td>Logger.Log</td><td>START TRAFFIC FROM HERE</td><td>\${False}</td><td>===</td></tr></table>	Logger. Log	START TRAFFIC FROM HERE	\${TRUE}		Logger. Log	START TRAFFIC FROM HERE	\${False}	===
Logger. Log	START TRAFFIC FROM HERE	\${TRUE}								
Logger. Log	START TRAFFIC FROM HERE	\${False}	===							
Log All	<i>msg,</i> <i>with_time=False,</i> <i>mark=***</i>	Inserts a message <code>msg</code> to current all VChannel log files. A default <code>mark</code> of <code>***</code> and newline will be added at the beggining and the end of this message. Example: <table><tr><td>Logger.Log All</td><td>START TRAFFIC FROM HERE</td><td>\${TRUE}</td><td></td></tr><tr><td>Logger.Log All</td><td>START TRAFFIC FROM HERE</td><td>\${TRUE}</td><td>===</td></tr></table> The log file will look likes this: <pre>user@vmx12> *** 06:01PM on August 13, 2017: START TRAFFIC FROM HERE *** === 06:01PM on August 13, 2017: START TRAFFIC FROM HERE === configure</pre>	Logger. Log All	START TRAFFIC FROM HERE	\${TRUE}		Logger. Log All	START TRAFFIC FROM HERE	\${TRUE}	===
Logger. Log All	START TRAFFIC FROM HERE	\${TRUE}								
Logger. Log All	START TRAFFIC FROM HERE	\${TRUE}	===							
Switch	<i>name</i>	Switches the current VChannel instance to <code>name</code> . <code>name</code> is the name of the VChannel (usually is the node name defined in the current active <code>local.yaml</code>) . Example: <table><tr><td>Logger.Switch</td><td>vmx11</td></tr></table>	Logger. Switch	vmx11						
Logger. Switch	vmx11									

Altogether 3 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:50:58.



OpticalSwitch

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A library provides control for L1 Optical Switch

Unlike other device, there is no *Switch* keyword with optical switch. Usually user only need to care about the interfaces not the ports of the switches.

Shortcuts

Add · **Clear By File** · **Close All** · **Connect All** · **Delete** · **Get Connection Info** · **Load From File** · **Save To File**

Keywords

Keyword	Arguments	Documentation						
Add	<i>dev1, intf1, dev2, intf2, direction=bi, force=False</i>	Adds a connection. See details in each module help						
Clear By File	<i>file_name=, comment=#</i>	Clears all x-connections defined in the <i>connection file</i> Default <i>connection file</i> is defined in <code>optic/connection</code> of <code>config/local.yaml</code>						
Close All		Close all connections						
Connect All		Connect to all L1 switch and read all necessary information						
Delete	<i>dev1, intf1, dev2, intf2, direction=bi, force=False</i>	Deletes a connection. See details in each module help						
Get Connection Info	<i>dev, intf</i>	Returns connection information. See details in each module help.						
Load From File	<i>file_name=, force=True, comment=#</i>	Loads the connection file and set the connections <code>filename</code> is the name of the connection file under the current config folder. If <code>filename</code> is empty, the value of <code>optic/connection</code> from <code>config/local.yaml</code> will be used. The <code>connection file</code> supports <code>jinja2</code> template language. Besides, <code>#</code> is the default comment char which could be changed The format of <code>connection file</code> follows: <ul style="list-style-type: none">■ each connection is described by 1 line■ <code>source</code> and <code>destination</code> are separated by <code>` - or > </code>, which mean <code>`bidirection</code> or <code>unidirection</code> (unidirection connects <code>source tx</code> to <code>dest rx</code>) Connection file sample: <pre>device1:port1 - device2:port2 device1:port3 > device2:port</pre> Examples: <table><tr><td><code>OpticalSwitch.</code></td><td><code>Load From File</code></td><td></td></tr><tr><td><code>OpticalSwitch.</code></td><td><code>Load From File</code></td><td><code>save1.conn</code></td></tr></table>	<code>OpticalSwitch.</code>	<code>Load From File</code>		<code>OpticalSwitch.</code>	<code>Load From File</code>	<code>save1.conn</code>
<code>OpticalSwitch.</code>	<code>Load From File</code>							
<code>OpticalSwitch.</code>	<code>Load From File</code>	<code>save1.conn</code>						
Save To File	<i>file_name</i>	Saves the current connection of all devices in this test. By default, all interfaces of the devices are save. If a connection file is given, only interfaces specified in the connection file are saved Examples: <table><tr><td><code>OpticalSwitch.</code></td><td><code>Save To File</code></td><td><code>save1.conn</code></td></tr></table>	<code>OpticalSwitch.</code>	<code>Save To File</code>	<code>save1.conn</code>			
<code>OpticalSwitch.</code>	<code>Save To File</code>	<code>save1.conn</code>						

Altogether 8 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:01.



calient

Library scope: global
Named arguments: supported

Introduction

A library provides control for Calient Optical Switch

Table of Contents

- Master file
- Connection file Format
- Shortcuts
- Keywords

Master file

The L1 switch provides a mechanism to remotely connect device interface. Each device interface has been wired to L1 switch already. The connection was described in the master file located specific by *calient-master-path* in the configuration file *renat/config/config.yaml*.
The master file includes several Calients in each tab. The column meaning and order is trivial.

Connection file Format

Keywords *Load From File*, *Clear By File* and *Save To File* use the x-connection file. X-connection files are text files and have the following format:

```
# this is the comment
device1,interface1,-,device2,interface2
device1,interface1,>,device2,interface2
```

The separator `-` means a bidirection connection and `>` means a unidirection connection. For a unidirection connection, `device1/interface1` TX will be connected to `device2/interface2` RX.
Note: The separator character must be surrounded by spaces or commas.
The connection file also support jinja2 template format. After the template is evaluated, comment could be used by `comment char`
There is no need to specify which L1 switch for the x-connection. The system will automatically find the appropriate switch.

Shortcuts

Add · Delete · Get Connection Info

Keywords

Keyword	Arguments	Documentation
Add	self, dev1, intf1, dev2, intf2, direction=bi, force=False	<p>Adds x-connection between <code>dev1:intf1</code> and <code>dev2:intf2</code></p> <p><code>direction</code> is <code>bi</code> for bi-direction or <code>uni</code> for uni-direction. If <code>direction</code> is <code>uni</code>, the tx of <code>dev 1:port 1</code> will be connected to <code>dev 2:port 2</code>.</p> <p>With <code>force</code> mode, existed connection that use those ports will be deleted. Without <code>force</code> mode, an existed connection will make the keyword fails</p> <p>Examples:</p> <div>OpticalSwitch.<i>Add</i> mx2008-31-33 xe-3/0/0 mx2008-31-33 xe-3/0/1 bi \${TRUE}</div> <p>Note: when <code>force</code> is <code>False</code> but the current ports is owned by the same connection endpoints, keyword will succeed.</p> <p>For a bidirection connection, 2 single uni-direction connection will be made instead of 1 bi-direction connection. This will make the link could be simulated tx/rx failure later.</p>
Delete	self, dev1, intf1, dev2, intf2, direction=bi, force=False	<p>Deletes the connection between <code>dev1:intf1</code> - <code>dev2:intf2</code></p> <p>Examples:</p> <div>OpticalSwitch.<i>Delete</i> mx2008-31-33 xe-3/0/1 mx2008-31-33 xe-3/0/1 uni</div>
Get Connection Info	self, dev, intf	<p>Returns information of the optic switch port that connected to <code>dev:intf</code>. The information is in jason format.</p> <p>Examples:</p> <div>OpticalSwitch.<i>Get Connection Info</i> mx2008-31-33 xe-3/0/1</div> <p>return information looks like below:</p> <div>result = {u'outoc': u'NOHW', u'outopwdh': u'-20.0', u'inos': u'OOS', u'outalias': u'', u'inowner': u'TRANSIT', u'outopwct': u'-23.0', u'inpower': u'-3.4', u'inas': u'IS', u'outpower': u'-4.8', u'outas': u'OOS-NP', u'inopt': u'-17.0', u'inoph': u'13.0', u'incircuit': u'3.3.1>3.3.2', u'inalias': u''</div>

	<code>u'inoc': u'NOHW', u'inopte': u'-20.0', u'outos': u'OOS', u'port': u'3.3.1', u'outowner': u'NONE', u'outcircuit': u''}</code>
--	--

Altogether 3 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:07.



g4ntm

Library scope: global
Named arguments: supported

Introduction

A library provides control for Telescent Network Topology Management (NTM) robot patch.

Shortcuts

Add · **Delete** · **Get Connection Info**

Keywords

Keyword	Arguments	Documentation
Add	<i>self, dev1, intf1, dev2, intf2, direction=bi, force=False</i>	
Delete	<i>self, dev1, intf1, dev2, intf2, direction=bi, force=False</i>	Deletes the connection between dev1:intf1 - dev2:intf2
Get Connection Info	<i>self, dev, intf</i>	Returns information about the connection by router/interface

Altogether 3 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:08.



Router

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A class provides keywords for router control. An instance of Router class automatically assigned methods of a VChannel class (**Note:** this is not an inheritance but rather 1-to-1 relation)

See [VChannel](#) for more details about *VChannel*.

Device's `type` is defined in master `device.yaml`. The system will load appropriate modules for each device.

Details about keywords provided by modules could be found in document of each module likes:

- [Juniper module](#)
- [Cisco module](#)
- [GR module](#)

Keywords provides by above module could be executed through *Xrun* keyword or directly called from `Router`. Examples:

Router. <i>Switch</i>	vmx12
Router. <i>Xrun</i>	Load Config
Router. <i>Load Config</i>	

Shortcuts

Follow Mib · Xrun

Keywords

Keyword	Arguments	Documentation
Follow Mib	<i>node_list</i> , <i>wait_time=10s</i> , <i>interval_time=5s</i> , <i>len=12</i> , <i>percentile=80</i> , <i>threshold=75</i> , <i>max_len=300</i> , <i>factor=1</i>	<p>Waits until all the nodes defined in <code>node_list</code> become <code>stable</code>.</p> <p>Stableness is checked by SNMP polling result. The MIB list is define by <code>mib</code> in <code>node</code> section Parameter:</p> <ul style="list-style-type: none">▪ <code>wait_time(1)</code>: the time before the evaluation starting▪ <code>interval_time(2)</code>: interval between SNMP polling time▪ <code>threshold</code>: below this value is evaluated as <code>stable</code>▪ <code>len(3)</code>: the size of the evaluation window (number of values that are used in each valuation)▪ <code>percentile</code>: real useful percentage of data (ignore top 100-percentile percent)▪ <code>max_len(4)</code>: maximum waiting <code>lend</code> for this checking <p>time sequence: --(1)--(2)- ----- ----- ----- </p> <p><-----> poll poll</p> <p><-----></p> <p><-----></p>
Xrun	<i>cmd</i> , <i>*args</i> , <i>**kwargs</i>	<p>Runs the vendor independent keywords.</p> <p>Parametes:</p> <ul style="list-style-type: none">▪ <code>cmd</code>: a keyword▪ <code>args</code>: other argumemts <p>Examples:</p> <p>Router.<i>Xrun</i> Flap Interface ge-0/0/0</p> <p>This keyword will then actually calling the correspond keyword for the device type.</p>



cisco

Library scope: global
Named arguments: supported

Introduction

Documentation for test library `cisco`.

Shortcuts

Get User · Get Version

Keywords

Keyword	Arguments	Documentation
Get User	<i>self</i>	Return the current login user
Get Version	<i>self</i>	return router version information

Altogether 2 keywords.
Generated by [Libdoc](#) on 2019-01-15 10:51:03.



gr

Library scope: global
Named arguments: supported

Introduction

Provides keywords for Hitachi GR platform

Shortcuts

Get Chassis Serial · **Get Version**

Keywords

Keyword	Arguments	Documentation
Get Chassis Serial	<i>self</i>	Returns the serial number of the chassis
Get Version	<i>self</i>	return router version information

Altogether 2 keywords.
Generated by [Libdoc](#) on 2019-01-15 10:51:03.



juniper

Library scope: global
Named arguments: supported

Introduction

Provides keywords for Juniper platform

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

Shortcuts

Create Best Path Select Data · Disable Interface · Enable Interface · Flap Interface · Get Chassis Serial · Get Cli Mode · Get Config · Get Current Datetime · Get File · Get Intf Addr · Get Route Number · Get Version · Link Status · Load Config · Number Of Bgp Neighbor · Number Of Ospf Neighbor · Number Of Ospf3 Neighbor

Keywords

Keyword	Arguments	Documentation
Create Best Path Select Data	<i>self</i> , <i>route_content</i> , <i>output_excel=best.xlsx</i>	Creates the matrix of best path selection Provides the test described in <i>smb://10.128.3.91/SharePoint01/31_VerificationRoom/31_13_検証環境セット/BGP-Best-Path-SelectionのAll-in-One設定_20161118改良/</i> The test uses predefined Ixia config and follows predefined steps
Disable Interface	<i>self</i> , <i>intf</i>	Disables an interface <i>intf</i>
Enable Interface	<i>self</i> , <i>intf</i>	Enables an interface <i>intf</i>
Flap Interface	<i>self</i> , <i>intf</i> , <i>time_str=10s</i>	Simulates an interface flap for interface <i>intf</i> Disables the interface and wait for a while before turning it up again
Get Chassis Serial	<i>self</i>	Returns the serial number of the chassis
Get Cli Mode	<i>self</i>	Returns current mode of the CLI. Return value is <i>config</i> for configuration mode or <i>command</i> for command mode
Get Config	<i>self</i> , <i>dst_name=</i>	Gets the current configuration file of the router to current <i>result</i> folder. Default <i>dst_name</i> is <i>juniper.conf.gz</i>
Get Current Datetime	<i>self</i> , <i>time_format=%H:%M:%S</i> , <i>delta_time=0s</i> , <i>dir=+</i> , <i>**kwargs</i>	Returns the current date time with vendor format <i>delta_time</i> will be added or subtracted to current time, default is <i>0s</i> <i>time_format</i> decides the time part of the output. Example result are : <div>May 24 04:14:25 May 4 04:14:25</div> Note: The date part is padded by space, and the result is allways 15 characters
Get File	<i>self</i> , <i>src_file</i> , <i>dst_file=</i>	Gets a file from router <ul style="list-style-type: none"><i>src_file</i> is a absolute path insides the router<i>dst_file</i> is a file name under <i>result</i> folder
Get Intf Addr	<i>self</i> , <i>intf_name</i> , <i>family=inet</i>	Returns the tuple of address and netmask of an interface <i>family</i> should be <i>inet</i> or <i>inet6</i> If the address is not set, (" <i>,</i> ") will be returned.
Get Route Number	<i>self</i> , <i>table=inet.0</i>	Returns number of active route in the <i>table</i> <i>table</i> could be <i>inet.0</i> or <i>inet.6</i>
Get Version	<i>self</i>	Returns router version information
Link Status	<i>self</i> , <i>if_name</i>	Returns link physical status as string (aka: "up down", "up up")
Load Config	<i>self</i> , <i>mode=set</i> , <i>config_file=</i> , <i>confirm=0s</i> , <i>vars=</i> , <i>err_match=(error:)</i>	Loads configuration to a router. Usable <i>mode</i> is <i>set</i> , <i>override</i> , <i>merge</i> and <i>replace</i> <i>set</i> mode uses configuration that contains <i>set</i> command. Mode <i>override</i> , <i>merge</i> and <i>replace</i> use ordinary JunOS configuration file with appropriate mode. <i>config_file</i> is a configuration file inside the <i>config</i> folder of the current test case. Config file could includes jinja2 template. The template will be evaluated with <i>LOCAL</i> , <i>GLOBAL</i> and variables defined by <i>vars</i> . The <i>vars</i> has the format: <i>var1=value1,var2=value2 ...</i> If the loading has no error that match the <i>error_match</i> , the configuration will be committed. The keywordl waits for <i>confirm</i> seconds before rollback the committed configuration. A zero value indicates an immediatly commit
Number Of Bgp Neighbor	<i>self</i> , <i>state=Established</i>	Returns number of BGP neighbor in <i>state</i> state

Number Of Ospf Neighbor	<i>self, state=Full</i>	Returns number of OPSF neighbors with status <i>state</i>
Number Of Ospf3 Neighbor	<i>self, state=Full</i>	Returns number of OPSFv3 neighbors with status <i>state</i>

Altogether 17 keywords.
Generated by [Libdoc](#) on 2019-01-15 10:51:03.



WebApp

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A library provides common keywords for web applications (aka Samurai, Arbor TMS)

The library utilize *Selenium2Library* and adds more functions to control Samurai application easily.

The *WebApp* uses the configuration in `local.yaml` in `webapp` section. The webapp device has following format:

```
<test node name>:
  device: <device name>
  proxy:
    http: <proxy for http>
    https: <proxy for http>
    ssl: <proxy for http>
```

Where `<device name>` is defined in master `device.yaml`, `proxy` section could be optional.

In order to download files from link, a *download-path* and MIME of that download need to be listed in *profile* section of the application setting. Default *download-path* is the current active result folder.

Samples:

```
...
webapp:
  samurai-1:
    device: samurai-b
```

profile: auto-save-mime: application/octet-stream

```
  proxy:
    http: 10.128.8.210:8080
    ssl: 10.128.8.210:8080
  arbor-1:
    device: arbor-sp-a
```

profile: auto-save-mime: application/xml,application/octet-stream

```
  proxy:
    http: 10.128.8.210:8080
    ssl: 10.128.8.210:8080
```

...

Selenium2Library keywords still could be used along with this library like this:

Click Link	//a[contains(.,'ユーザ設定')]
Sleep	2s
Click Link	Home設定
Sleep	2s
Samurai.Capture Screenshot	

See [Selenium2Library](#) for more details.

The module *Samurai* and *Arbor* based on this module. See [Arbor](#), [Samurai](#) for details about keywords of each application.

Shortcuts

Capture Screenshot · Close · Close All · Get Verbose · Mark Element · Open Ff With Profile · Reset Capture Counter · Set Ajax Wait · Set Capture Counter · Set Capture Format · Set Verbose · Switch · Verbose Capture · Wait Until Element Changes

Keywords

Keyword	Arguments	Documentation															
Capture Screenshot	<i>filename=None</i> , <i>extra=</i>	<p>Captures the current screen to file</p> <p>Using the internal counter for filename if <code>filename</code> is not specified. In this case, the filename is defined by a pre-set format. Set Capture Format could be used to change the current format.</p> <p>An extra information will be add to the filename if <code>extra</code> is defined</p> <p>Examples:</p> <table><tr><td>Samurai.Capture Screenshot</td><td></td><td># samurai_0000000001.png</td></tr><tr><td>Samurai.Capture Screenshot</td><td>extra=_list</td><td># samurai_0000000002_list.png</td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td># arbor_0000000001.png</td></tr><tr><td>Arbor.Capture Screenshot</td><td>extra=_xxx</td><td># arbor_0000000001_xxx.png</td></tr><tr><td>Samurai.Capture Screenshot</td><td>filename=1111.png</td><td># 1111.png</td></tr></table>	Samurai. Capture Screenshot		# samurai_0000000001.png	Samurai. Capture Screenshot	extra=_list	# samurai_0000000002_list.png	Arbor. Capture Screenshot		# arbor_0000000001.png	Arbor. Capture Screenshot	extra=_xxx	# arbor_0000000001_xxx.png	Samurai. Capture Screenshot	filename=1111.png	# 1111.png
Samurai. Capture Screenshot		# samurai_0000000001.png															
Samurai. Capture Screenshot	extra=_list	# samurai_0000000002_list.png															
Arbor. Capture Screenshot		# arbor_0000000001.png															
Arbor. Capture Screenshot	extra=_xxx	# arbor_0000000001_xxx.png															
Samurai. Capture Screenshot	filename=1111.png	# 1111.png															

Close		Close the web application
Close All		Closes all current opened applications
Get Verbose		Get current verbose mode
Mark Element	<i>xpath</i>	Marking an element to check its status later
Open Ff With Profile	<i>app, name</i>	
Reset Capture Counter		Resets the counter of the screen capture
Set Ajax Wait	<i>wait_time=2s</i>	Set the ajax wait time
Set Capture Counter	<i>value=0</i>	Sets the counter of the screen capture to <code>value</code>
Set Capture Format	<i>format</i>	<p>Sets the format for the screen capture file</p> <p>The format does not include the default prefix <code>.png</code>. The default format is <code><mod>_%010d.</code>. <code>mod</code> could be <code>samurai</code> or <code>arbor</code></p> <p>See https://docs.python.org/2/library/string.html#format-specification-mini-language for more details about the format string.</p> <p>Examples:</p> <pre>Samurai.Set Capture Format \${case}_%010d # \${case} is a predefined variable</pre>
Set Verbose	<i>verbose=False</i>	Set current verbose mode to <code>verbose</code>
Switch	<i>name</i>	Switches the current browser to <code>name</code>
Verbose Capture	<i>*args, **kwargs</i>	Capture screenshot if verbose mode is <code>True</code> otherwise do nothing
Wait Until Element Changes	<i>interval=5s, timeout=180s, error_on_timeout=False</i>	Wait until the marked element has been changed

Altogether 14 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:09.



Samurai

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A library provides functions to control Samurai application

The library utilize *SeleniumLibrary* and adds more functions to control Samurai application easily. Without other furthur mentions, all of the concepts of `user`, `user group` are Samurai concepts. By default, RENAT will try to connec to all Samurai nodes defined in active `local.yaml` at the beginning of the test and disconnect from them at the end of the test automatically. Usually user does not need to use `Connect All` and `Close` explicitly.

Currently, this module supposed that Samurai is used in Japanese locale. When Samurai module has error, it tried to make the last snapshot in `result/selenium-screenshot-x.png`. Checking this capture will help to understand the reason of the error.

Currently the module support Samurai 09/14/16

Some keywords of `Samurai` is using `xpath` to identify elements. See *Selenium2Library* for more details about xpath.

See [WebApp](#) for common keywords of web applications and how to configure the `local.yaml` file.

Selenium2Library keywords still could be used together within this library. See [Selenium2Library](#) for more details.

Shortcuts

Add Policy · **Add Policy Group** · **Add User** · **Capture Screenshot** · **Change Policy View Group** · **Click All Elements** · **Close** · **Close All** · **Close Window** · **Connect** · **Connect All** · **Delete Policy** · **Delete Policy Group** · **Delete User** · **Edit Mitigation Controller** · **Edit Policy** · **Get Mitigation List** · **Get Verbose** · **Left Menu** · **Login** · **Logout** · **Make Item Map** · **Mark Element** · **Open Ff With Profile** · **Reconnect** · **Reset Capture Counter** · **Select Items In Table** · **Select Window** · **Set Ajax Wait** · **Set Capture Counter** · **Set Capture Format** · **Set Verbose** · **Show Detail Mitigation** · **Show Policy Basic** · **Show Policy Detection** · **Show Policy Display** · **Show Policy Mitigation** · **Show Policy Mo** · **Show Policy Monitor** · **Show Policy Notify** · **Show Policy Nw Monitor** · **Show Policy Traffic** · **Start Mitigation** · **Stop Mitigation** · **Switch** · **Update Mitigation Controller Info** · **Verbose Capture** · **Wait Until Element Changes**

Keywords

Keyword	Arguments	Documentation				
Add Policy	**policy	Adds a new Samurai policy				
		policy is a map containing the below information to create the new policy.				
		key	meaning	mandatory	sample	
		name	name of the policy	yes	test001	
		basic_alias	alias name of the policy		test001	
		basic_port_id	another alias			
		basic_facing	customer or backbone		customer	
		basic_intf_list	list of router and interface pair, separated by comma	yes	10.128.18.31:xe-0/0/0.1	
		basic_cidr_list	list of CIDR separate by comma			
		basic_option_filter	optinal filter			
		basic_direction	direction of the traffic (incoming or outgoing)		incoming	
		traffic_enabled	Enable traffic monitoring or not	yes	\${TRUE} or \${FALSE}	
		detection_enabled	Enable detection or not	yes	\${TRUE} or \${FALSE}	
		detection_direction	change detect direction fo all attack type	incomming,outgoing,'both both:check		
		mitigation_enabled	Enable Mitigation or not	yes	\${TRUE} or \${FALSE}	
		mitigation_zone_name	Name of the zone for mitigation		zone001	
		mitigation_zone_prefix	Prefixes that could mitigate		1.1.1.1/32	
		mitigation_thr_bps	Upper limit (bps)		800,000,000	
		mitigation_thr_pps	Upper limit (pps)		54,000,000	
		mitigation_auto_enabled	Enable automitigation or not		\${TRUE} or \${FALSE}	
		mitigation_auto_level	Automitigation level		0:overLow 1:overMedium 2:High	
		mitigation_auto_time	Automitigation detect attack time (min)		default is 15	
		mitigation_mo_enabled	Using Arbor TMS MO or not	yes	\${TRUE} or \${FALSE}	
		mitigation_auto_stop_enabled	Enable		\${TRUE} or \${FALSE}	

		mitigation_auto_stop_enabled	enable automitigation stop or not		$\$(TRUE) or \$(FALSE)$
		mitigation_auto_stop_level	Automitigation level		0:overLow 2:High
		mitigation_auto_stop_time	Automitigation stop detect attack time (min)		default is 15
		mitigation_device_list	Devices used for TMS, separated by comma		ArborSP-A
		mitigation_mo_name	MO name, separated by comma		OCN12(ALU)_LOOSE
		mitigation_comm_list	commna separated peer/community list		1.10(180.0.1.10)/2914:666,1.11(180.0.1.11)/2914:777
		nw_monitor_gre1	1st GRE address for NW monitor		210.0.1.1
		nw_monitor_gre2	2nd GRE address for NW monitor		210.0.1.1
		nw_monitor_ce1	1st CE address for NW monitor		210.0.1.2
		nw_monitor_ce2	2nd CE address for NW monitor		210.0.1.2
		nw_monitor_pe1	1st PE for NW monitor (list)		edge01hige-MX2020-15(118.23.176.244)
		nw_monitor_pe2	2nd PE for NW monitor (list)		edge01hige-MX2020-15(118.23.176.244)
		event_name	name of the message event to make		info1
		event_addr	address to send the events		user@mail.com
		view_group	user group that could view this policy, separated by comma	yes	SuperGroup,test_group_007

Example:

Samurai. Switch	samurai-1	
Samurai. Add Policy	name=\${POLICY_NAME}	basic_alias=\${POLICY_NAME}
...	basic_facing=customer	basic_intf_list=10.128.18.31:xe-0/0/0.1
...	basic_cidr_list=1.1.1.0/24	basic_direction=incoming
...	traffic_enabled=\${TRUE}	
...	detection_enabled=\${TRUE}	
...	mitigation_zone_name=test_zone001	mitigation_zone_prefix=1.1.1.1/32
...	mitigation_device_list=ArborSP-A,ArborSP-B	
...	mitigation_mo_enabled=\${TRUE}	
...	mitigation_mo_name=N000000012_LOOSE	
...	mitigation_comm_list=1.10(180.0.1.10)/2914:666,1.11(180.0.1.11)/2914:777	
...	event_name=test	event_addr=user@mail.com
...	view_group=SuperGroup	

Add Policy Group	group_name, policy_list=*, limit_bps=4000000000, limit_pps=2700000	Add a new policy group group_name is the name of the new group. policy_list is a comma separated of existed policy that should be bound to this policy. An asterisk for this parameter (*) means all of the existed policy. limit_bps and limit_pps are the mitigation capacity threshold of this group.																
Add User	group, **user_info	Adds user to the current group user_info is a dictionary contains user information that has following keys: name, password, privilege and policy privilege is existed privilege that has been created (e.g: system_admin. policy could be * for all current policies or a list of policy names that are binded to this user. group is the user group. Dot(.) means current group Examples: <table><tr><td>Samurai.Add User</td><td>OCNDDoS</td><td>name=user000</td><td>password=Test12345678</td></tr><tr><td>...</td><td>privilege=system_admin</td><td>policy=*</td><td></td></tr><tr><td>Samurai.Add User</td><td>OCNDDoS</td><td>username=user001</td><td>password=Test12345678</td></tr><tr><td>...</td><td>privilege=system_admin</td><td>policy=OCN11,OCN12</td><td></td></tr></table>	Samurai.Add User	OCNDDoS	name=user000	password=Test12345678	...	privilege=system_admin	policy=*		Samurai.Add User	OCNDDoS	username=user001	password=Test12345678	...	privilege=system_admin	policy=OCN11,OCN12	
Samurai.Add User	OCNDDoS	name=user000	password=Test12345678															
...	privilege=system_admin	policy=*																
Samurai.Add User	OCNDDoS	username=user001	password=Test12345678															
...	privilege=system_admin	policy=OCN11,OCN12																
Capture Screenshot	filename=None, extra=	Captures the current screen to file Using the internal counter for filename if filename is not specified. In this case, the filename is defined by a pre-set format. Set Capture Format could be used to change the current format. An extra information will be add to the filename if extra is defined Examples: <table><tr><td>Samurai.Capture Screenshot</td><td></td><td># samurai_0000000001.png</td></tr><tr><td>Samurai.Capture Screenshot</td><td>extra=_list</td><td># samurai_0000000002_list.png</td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td># arbor_0000000001.png</td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td># arbor_0000000001.png</td></tr></table>	Samurai.Capture Screenshot		# samurai_0000000001.png	Samurai.Capture Screenshot	extra=_list	# samurai_0000000002_list.png	Arbor.Capture Screenshot		# arbor_0000000001.png	Arbor.Capture Screenshot		# arbor_0000000001.png				
Samurai.Capture Screenshot		# samurai_0000000001.png																
Samurai.Capture Screenshot	extra=_list	# samurai_0000000002_list.png																
Arbor.Capture Screenshot		# arbor_0000000001.png																
Arbor.Capture Screenshot		# arbor_0000000001.png																

		<table><tr><td>Samurai.Capture Screenshot</td><td>filename=1111.png</td><td># 1111.png</td></tr></table>	Samurai. Capture Screenshot	filename=1111.png	# 1111.png															
Samurai. Capture Screenshot	filename=1111.png	# 1111.png																		
Change Policy View Group	name, *group_name	<p>Changes the groups that could see this policy</p> <p>name is the policy name. group_name is a list of policies</p> <p>Example:</p> <table><tr><td>Samurai.Change Policy View Group</td><td>super_admin</td><td>test_group001</td></tr></table>	Samurai. Change Policy View Group	super_admin	test_group001															
Samurai. Change Policy View Group	super_admin	test_group001																		
Click All Elements	xpath	<p>Click all element in current page defined by xpath</p> <p>Returns the number of elements that have been clicked</p>																		
Close		Close the web application																		
Close All		Closes all current opened applications																		
Close Window		Closes the current window																		
Connect	app, name	<p>Opens a web browser and connects to application and assigns a name .</p> <ul style="list-style-type: none">app is the name of the application (e.ge. samurai-1)name is the name of the browser <p>If not defined in local.yaml those following key will have default values:</p> <table><tr><td>browser</td><td>firefox</td><td>optional</td></tr><tr><td>login_url</td><td>/</td><td>optiona</td></tr><tr><td>proxy:</td><td></td><td>optional</td></tr><tr><td>http: 10.128.8.210:8080</td><td>optional</td><td></td></tr><tr><td>ssl: 10.128.8.210:8080</td><td>optional</td><td></td></tr><tr><td>socks: 10.128.8.210:8080</td><td>optional</td><td></td></tr></table>	browser	firefox	optional	login_url	/	optiona	proxy:		optional	http: 10.128.8.210:8080	optional		ssl: 10.128.8.210:8080	optional		socks: 10.128.8.210:8080	optional	
browser	firefox	optional																		
login_url	/	optiona																		
proxy:		optional																		
http: 10.128.8.210:8080	optional																			
ssl: 10.128.8.210:8080	optional																			
socks: 10.128.8.210:8080	optional																			
Connect All		<p>Connects to all applications defined in local.yaml</p> <p>The name of the connection will be the same of the webapp name</p>																		
Delete Policy	*policy_names	<p>Deletes poilcies by their names</p> <p>Returned the number of deleted users</p> <p>Notes: If the policy does not exists, the system will not report any error.</p> <p>Examples:</p> <table><tr><td>Samurai.Delete Policy</td><td>test001</td><td>test002</td></tr></table>	Samurai. Delete Policy	test001	test002															
Samurai. Delete Policy	test001	test002																		
Delete Policy Group	*group_list	<p>Deletes policy groups</p> <p>See Select Items In Table for more detail about how to choose group_list</p> <p>Returns the number of deleted policy groups Example:</p> <table><tr><td>Samurai.Delete Policy Group</td><td>test_group001</td><td>test_group002</td></tr></table>	Samurai. Delete Policy Group	test_group001	test_group002															
Samurai. Delete Policy Group	test_group001	test_group002																		
Delete User	group, *user_list	<p>Deletes user from the user group</p> <p>group is the user group. And . means current group Returns the number of deleted users</p> <p>Examples:</p> <table><tr><td>Samurai.Delete User</td><td>SuperGroup</td><td>user001</td><td>user002</td></tr><tr><td>Samurai.Delete User</td><td>.</td><td>user002</td><td></td></tr></table>	Samurai. Delete User	SuperGroup	user001	user002	Samurai. Delete User	.	user002											
Samurai. Delete User	SuperGroup	user001	user002																	
Samurai. Delete User	.	user002																		
Edit Mitigation Controller	controller, **config	<p>Change the setting of the mitigation control</p> <ul style="list-style-type: none">control: name of the mitigation controllerconfig: configuration need to be changed. Currently only tms_group is configurable with the following format: groupname1:action1,groupname2:action2. groupname is currently set TMS group name and action could be click,check or uncheck. <p>Example:</p> <table><tr><td>Samurai.Edit Mitigation Controller</td><td>controller=vSP-A</td><td>tms_group=Logical0_SOCN_IPv4:uncheck</td></tr></table>	Samurai. Edit Mitigation Controller	controller=vSP-A	tms_group=Logical0_SOCN_IPv4:uncheck															
Samurai. Edit Mitigation Controller	controller=vSP-A	tms_group=Logical0_SOCN_IPv4:uncheck																		
Edit Policy	**policy	<p>Edits a Samurai policy</p> <p>policy contains information about the policy. See Add Policy for more details about policy format</p>																		
Get Mitigation List	status=実行中	<p>Gets current mitigation list</p> <p>Return current active mitigation name, ID and the number of them</p> <p>Example:</p> <table><tr><td>\$(MITI) \$(IDS) \$(NUM)=</td><td>Samurai.Get Mitigation List</td></tr></table>	\$(MITI) \$(IDS) \$(NUM)=	Samurai. Get Mitigation List																
\$(MITI) \$(IDS) \$(NUM)=	Samurai. Get Mitigation List																			
Get Verbose		Get current verbose mode																		
Left Menu	menu, locator=None, ignore_first_element=True	<p>Chooses the left panel menu by its displayed name</p> <p>When locator is not null, the keyword will return a list of text attribute of all elements specified by the locator. locator could be a xpath or a predefined string.</p> <p>locator predefined strings are: MITIGATE_REALTIME, MITIGATE_LIST, DETECT_LIST</p> <p>For example, a xpath //div[@id='infoareain2']*/td[1]/a means the list of link of all elements in a 1st column of a table insides a div with id infoareain2.</p> <p>Examples:</p> <table><tr><td>Samurai.Left Menu</td><td>Traffic</td><td></td><td></td></tr><tr><td>Samurai.Left Menu</td><td>Detection</td><td></td><td></td></tr><tr><td>Samurai.Left Menu</td><td>ポリシー管理</td><td></td><td></td></tr><tr><td>@{LIST}=</td><td>Samurai.Left Menu</td><td>Active Mitigation</td><td>//div[@id='infoareain2']*/td[1]/a</td></tr></table>	Samurai. Left Menu	Traffic			Samurai. Left Menu	Detection			Samurai. Left Menu	ポリシー管理			@{LIST}=	Samurai. Left Menu	Active Mitigation	//div[@id='infoareain2']*/td[1]/a		
Samurai. Left Menu	Traffic																			
Samurai. Left Menu	Detection																			
Samurai. Left Menu	ポリシー管理																			
@{LIST}=	Samurai. Left Menu	Active Mitigation	//div[@id='infoareain2']*/td[1]/a																	
Login		Logs-in into the application																		

		User and password is set by the template and authentication methods in the master files
Logout		Logs-out the current application, the browser remains
Make Item Map	<i>xpath</i>	Makes a item/webelement defined <i>xpath</i> The map is a dictionary from <i>item</i> to the <i>WebElement</i> Items name found by <i>xpath</i> are used as keys
Mark Element	<i>xpath</i>	Marking an element to check its status later
Open Ff With Profile	<i>app, name</i>	
Reconnect		Reconnects to the server
Reset Capture Counter		Resets the counter of the screen capture
Select Items In Table	<i>xpath, xpath2, *item_list</i>	Checks items in Samurai table by <i>xpath</i> <i>xpath</i> points to the column that used as key and <i>xpath2</i> is the relative <i>xpath</i> contains the target column. <i>item_list</i> is a list of item and its action that need to check. Item in the list could be a regular expression with the format <i>re: <regular expression> action</i> . The default action for the item could be <i>click`'(default)`,check</i> or <i>uncheck</i> The keyword is called with assuming that the table is already visible. Returns the tuple of all items and selected items Note: Non-width-space (\u200b) will be take care by the keyword. Note: if the first <i>item_list</i> is <i>*</i> then the keyword will try to click a link named <i>すべてを選択</i> .
Select Window	<i>title</i>	Selects a window by its title
Set Ajax Wait	<i>wait_time=2s</i>	Set the ajax wait time
Set Capture Counter	<i>value=0</i>	Sets the counter of the screen capture to <i>value</i>
Set Capture Format	<i>format</i>	Sets the format for the screen capture file The format does not include the default prefix <i>.png</i> The default format is <i><mod>_%010d</i> . <i>mod</i> could be <i>samurai</i> or <i>arbor</i> See https://docs.python.org/2/library/string.html#format-specification-mini-language for more details about the format string. Examples: <div>Samurai.<i>Set Capture Format</i> \${case}_%010d # \${case} is a predefined variable</div>
Set Verbose	<i>verbose=False</i>	Set current verbose mode to <i>verbose</i>
Show Detail Mitigation	<i>id</i>	Shows detail information of a mitigation
Show Policy Basic	<i>policy_name</i>	Makes the virtual browser show basic setting of the policy <i>name</i> . A following Samurai. <i>Capture Screenshot</i> is necessary to capture the result.
Show Policy Detection	<i>policy_name</i>	Shows the detection pannel of <i>policy_name</i> policy
Show Policy Display	<i>policy_name</i>	Make a virtual browser show the display setting of a policy Notes: Depending on the setting of the policy, MO panel may not be existed. In this case, if <i>strict</i> is <i>True</i> , then the keyword will fail.
Show Policy Mitigation	<i>policy_name</i>	Make the virtual browser show the mitigation setting of a policy
Show Policy Mo	<i>policy_name, strict=False</i>	Make the virtual browser show the MO setting of a policy Automatically expand the MO section of other devices. Notes: Depending on the setting of the policy, MO panel may not be existed. In this case, if <i>strict</i> is <i>True</i> , then the keyword will fail.
Show Policy Monitor	<i>policy_name</i>	
Show Policy Notify	<i>policy_name</i>	Make a virtual browser show the mitigation setting of a policy
Show Policy Nw Monitor	<i>policy_name, strict=False</i>	Make a virtual browser show the NW monitor setting of a policy Notes: Depending on the setting of the policy, MO panel may not be existed. In this case, if <i>strict</i> is <i>True</i> , then the keyword will fail.
Show Policy Traffic	<i>policy_name</i>	Makes the virtual browser show the traffic setting of the policy <i>name</i> . A following Samurai. <i>Capture Screenshot</i> is necessary to capture the result.
Start Mitigation	<i>policy, prefix, comment=mitigation started by RENAT, device=None, force=False</i>	Starts a mitigation with specific <i>prefix</i> <i>device</i> is used for matching real device name configured by Samurai If <i>force</i> is <i>TRUE</i> then the keyword will fail if selected device does not contain <i>device</i> Returns mitigation <i>id</i> and selected <i>arbor</i> device Example: <div>\${id} \${device}= Samurai.<i>Start Mitigation</i> 211.1.12.1/32 mitigation by RENAT SP-A \${TRUE}</div>
Stop Mitigation	<i>id, raise_error=True</i>	Stops a mitigation by its ID The keyword will raise an error if <i>raise_error</i> is <i>True</i> . Otherwise it will ignore any errors. Example: <div>Samurai.<i>Stop Mitigation</i> 700</div>
Switch	<i>name</i>	Switches the current browser to <i>name</i>
Update	<i>controller wait=10s</i>	Updates information of <i>controller</i>

Update Mitigation Controller Info	<i>controller, wait=True</i>	Updates information of controller
Verbose Capture	<i>*args, **kwargs</i>	Capture screenshot if verbose mode is <code>True</code> otherwise do nothing
Wait Until Element Changes	<i>interval=5s, timeout=180s, error_on_timeout=False</i>	Wait until the marked element has been changed

Altogether 48 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:10.



Arbor

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A library provides functions to control Arbor application

The library utilize *SeleniumLibrary* and adds more functions to control Arbor application easily.

See [WebApp](#) for common keywords of web applications.

SeleniumLibrary keywords still could be used along with this library. See [SeleniumLibrary](#) for more details.

Notes: From 0.1.10, move from *Selenium2Library* to *SeleniumLibrary*

Shortcuts

Capture Screenshot · Clean Status Msg · Close · Close All · Commit · Connect · Connect All · Detail First Mitigation · Get Status Msg · Get Verbose · Login · Logout · Mark Element · Menu · Open Ff With Profile · Reconnect · Reset Capture Counter · Set Ajax Wait · Set Capture Counter · Set Capture Format · Set Verbose · Show All Mitigations · Show Detail Countermeasure · Show Detail First Mitigation · Show Detail Mitigation · Show Detail Mitigation With Order · Switch · Verbose Capture · Wait Until Element Changes

Keywords

Keyword	Arguments	Documentation															
Capture Screenshot	filename=None, extra=	<p>Captures the current screen to file</p> <p>Using the internal counter for filename if filename is not specified. In this case, the filename is defined by a pre-set format. <i>Set Capture Format</i> could be used to change the current format.</p> <p>An extra information will be add to the filename if extra is defined</p> <p>Examples:</p> <table><tr><td>Samurai.<i>Capture Screenshot</i></td><td></td><td># samurai_0000000001.png</td></tr><tr><td>Samurai.<i>Capture Screenshot</i></td><td>extra=_list</td><td># samurai_0000000002_<i>list</i>.png</td></tr><tr><td>Arbor.<i>Capture Screenshot</i></td><td></td><td># arbor_0000000001.png</td></tr><tr><td>Arbor.<i>Capture Screenshot</i></td><td>extra=_xxx</td><td># arbor_0000000001_xxx.png</td></tr><tr><td>Samurai.<i>Capture Screenshot</i></td><td>filename=1111.png</td><td># 1111.png</td></tr></table>	Samurai. <i>Capture Screenshot</i>		# samurai_0000000001.png	Samurai. <i>Capture Screenshot</i>	extra=_list	# samurai_0000000002_ <i>list</i> .png	Arbor. <i>Capture Screenshot</i>		# arbor_0000000001.png	Arbor. <i>Capture Screenshot</i>	extra=_xxx	# arbor_0000000001_xxx.png	Samurai. <i>Capture Screenshot</i>	filename=1111.png	# 1111.png
Samurai. <i>Capture Screenshot</i>		# samurai_0000000001.png															
Samurai. <i>Capture Screenshot</i>	extra=_list	# samurai_0000000002_ <i>list</i> .png															
Arbor. <i>Capture Screenshot</i>		# arbor_0000000001.png															
Arbor. <i>Capture Screenshot</i>	extra=_xxx	# arbor_0000000001_xxx.png															
Samurai. <i>Capture Screenshot</i>	filename=1111.png	# 1111.png															
Clean Status Msg		Disposes any alert or status messgae at the top of the current page															
Close		Close the web application															
Close All		Closes all current opened applications															
Commit	strict=True	<p>Commit the current changes</p> <p>If <i>strict</i> is \${TRUE} then the keyword fails if there is not change to commit. Otherwise, it does nothing</p>															
Connect	app, name	<p>Opens a web browser and connects to application and assigns a name .</p> <p>Extra information could be added to the webapp sections likes login_url , browser or profile_dir . Default values are:</p> <table><tr><td>browser</td><td>firefox</td></tr><tr><td>login_url</td><td>/</td></tr><tr><td>profile_dir</td><td>./config/samurai.profile</td></tr></table>	browser	firefox	login_url	/	profile_dir	./config/samurai.profile									
browser	firefox																
login_url	/																
profile_dir	./config/samurai.profile																
Connect All		<p>Connects to all applications defined in local.yaml</p> <p>The name of the connection will be the same of the webapp name</p>															
Detail First Mitigation																	
Get Status Msg		<p>Get current status message</p> <p>Return null string if no message exists</p>															
Get Verbose		Get current verbose mode															
Login		Logs into the Arbor application															
Logout		Logs-out the current application, the browser remains															
Mark Element	xpath	Marking an element to check its stattus later															
Menu	order, wait=2s, capture_all=False, prefix=menu_, suffix=.png,	Access to Arbor menu															

	<code>partial_match=False</code>	<p>Parameters</p> <ul style="list-style-type: none">▪ <code>order</code> is the list of top menu items separated by '/'▪ <code>wait</code> is the wait time after the last item is clicked▪ if <code>capture_all</code> is <code>True</code> then a screenshot is captured for each menu item automatically. In this case, the image file is appended by <code>prefix</code> and <code>suffix</code>. <p>by default, the system try to match the menu item in full, when <code>partial_match</code> is <code>True</code>, partial match is applied.</p> <p>Samples:</p> <table><tr><td>Arbor.Menu</td><td>order=Alerts/Ongoing</td><td></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td></tr><tr><td>Arbor.Menu</td><td>order=Alerts/All Alerts</td><td></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td></tr><tr><td>Arbor.Menu</td><td>order=System/Status/Deployment Status</td><td></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td></tr><tr><td>Arbor.Menu</td><td>order=System/Status/Signaling Status/Appliance Status</td><td><code>partial_match=\${TRUE}</code></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td></tr></table>	Arbor. Menu	order=Alerts/Ongoing		Arbor. Capture Screenshot			Arbor. Menu	order=Alerts/All Alerts		Arbor. Capture Screenshot			Arbor. Menu	order=System/Status/Deployment Status		Arbor. Capture Screenshot			Arbor. Menu	order=System/Status/Signaling Status/Appliance Status	<code>partial_match=\${TRUE}</code>	Arbor. Capture Screenshot		
Arbor. Menu	order=Alerts/Ongoing																									
Arbor. Capture Screenshot																										
Arbor. Menu	order=Alerts/All Alerts																									
Arbor. Capture Screenshot																										
Arbor. Menu	order=System/Status/Deployment Status																									
Arbor. Capture Screenshot																										
Arbor. Menu	order=System/Status/Signaling Status/Appliance Status	<code>partial_match=\${TRUE}</code>																								
Arbor. Capture Screenshot																										
Open Ff With Profile	<code>app, name</code>																									
Reconnect		Reconnect to server if necessary																								
Reset Capture Counter		Resets the counter of the screen capture																								
Set Ajax Wait	<code>wait_time=2s</code>	Set the ajax wait time																								
Set Capture Counter	<code>value=0</code>	Sets the counter of the screen capture to <code>value</code>																								
Set Capture Format	<code>format</code>	<p>Sets the format for the screen capture file</p> <p>The format does not include the default prefix <code>.png</code> The default format is <code><mod>_%010d</code>. <code>mod</code> could be <code>samurai</code> or <code>arbor</code></p> <p>See https://docs.python.org/2/library/string.html#format-specification-mini-language for more details about the format string.</p> <p>Examples:</p> <table><tr><td>Samurai.Set Capture Format</td><td><code>\${case}_%010d</code></td><td># <code>\${case}</code> is a predefined variable</td></tr></table>	Samurai. Set Capture Format	<code>\${case}_%010d</code>	# <code>\${case}</code> is a predefined variable																					
Samurai. Set Capture Format	<code>\${case}_%010d</code>	# <code>\${case}</code> is a predefined variable																								
Set Verbose	<code>verbose=False</code>	Set current verbose mode to <code>verbose</code>																								
Show All Mitigations		Shows all mitigations																								
Show Detail Countermeasure	<code>name, *method_list</code>	<p>Shows detail informatin about a countermeasure</p> <p><code>name</code> is used to search the the mitigation and <code>method_list</code> is a list of countermeasures that are listed in Arbor Countermeasures panel</p> <p>Example:</p> <table><tr><td><code>\${NAME}</code></td><td><code>\${ID}=</code></td><td>Show Detail First Mitigation</td><td></td></tr><tr><td>Arbor.Show Detail Countermeasure</td><td><code>\${NAME}</code></td><td>DNS Malformed</td><td></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td><td></td></tr><tr><td>Sleep</td><td>10s</td><td></td><td></td></tr><tr><td>Arbor.Show Detail Countermeasure</td><td><code>\${NAME}</code></td><td>Zombie Detection</td><td>HTTP Malformed</td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td><td></td></tr></table>	<code>\${NAME}</code>	<code>\${ID}=</code>	Show Detail First Mitigation		Arbor. Show Detail Countermeasure	<code>\${NAME}</code>	DNS Malformed		Arbor. Capture Screenshot				Sleep	10s			Arbor. Show Detail Countermeasure	<code>\${NAME}</code>	Zombie Detection	HTTP Malformed	Arbor. Capture Screenshot			
<code>\${NAME}</code>	<code>\${ID}=</code>	Show Detail First Mitigation																								
Arbor. Show Detail Countermeasure	<code>\${NAME}</code>	DNS Malformed																								
Arbor. Capture Screenshot																										
Sleep	10s																									
Arbor. Show Detail Countermeasure	<code>\${NAME}</code>	Zombie Detection	HTTP Malformed																							
Arbor. Capture Screenshot																										
Show Detail First Mitigation		<p>Shows details about the 1st mitigation on the list</p> <p>The keyword returns the <code>mitigation ID</code> and its name</p>																								
Show Detail Mitigation	<code>search_str</code>	<p>Shows detail information of a mitigation by its <code>search_str</code></p> <p>Note: the result could include multi mitigations</p>																								
Show Detail Mitigation With Order	<code>order</code>	<p>Shows details about the <code>order</code>(th) mitigation in the current list</p> <p><code>order</code> is counted from <code>1</code>. The keyword returns the <code>mitigation_id</code> and its name</p> <p>Example:</p> <table><tr><td><code>\${NAME}</code></td><td><code>\${ID}=</code></td><td>Arbor.Show Detail Mitigation With Order</td><td>3</td></tr><tr><td>Log To Console</td><td><code>\${NAME}:\${ID}</code></td><td></td><td></td></tr><tr><td>Arbor.Capture Screenshot</td><td></td><td></td><td></td></tr></table>	<code>\${NAME}</code>	<code>\${ID}=</code>	Arbor. Show Detail Mitigation With Order	3	Log To Console	<code>\${NAME}:\${ID}</code>			Arbor. Capture Screenshot															
<code>\${NAME}</code>	<code>\${ID}=</code>	Arbor. Show Detail Mitigation With Order	3																							
Log To Console	<code>\${NAME}:\${ID}</code>																									
Arbor. Capture Screenshot																										

Switch	<i>name</i>	Switches the current browser to <code>name</code>
Verbose Capture	<i>*args, **kwargs</i>	Capture screenshot if verbose mode is <code>True</code> otherwise do nothing
Wait Until Element Changes	<i>interval=5s, timeout=180s, error_on_timeout=False</i>	Wait until the marked element has been changed

Altogether 29 keywords.
Generated by [Libdoc](#) on 2019-01-15 10:51:10.



Tester

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A class provides keywords for controlling testers and traffic generators.

It could load predefined traffic file, manipulate traffic items, start and stop traffic flows. It also could generate traffic reports and support QuickTest for IxNetwork.

Tester information is stored in the active `local.yaml`. The syntax is slightly different depending on the tester platform. Below is sample for IxNetwork.

```
tester:
  tester01:
    device: ixnet03_8009
    config: vmx_20161129.ixncfg
    real_port:
      - description: to egde router
```

chassis: 10.128.32.71

```
card: 6
port: 11
- description: to backbone router
```

chassis: 10.128.32.71

```
card: 6
port: 9
```

where `device` is the tester defined in the master `device.yaml` file. If `real_port` does not exist, port remapping will not take place. Otherwise, port remapping will use the `real_port` information to reassign all existed ports and map to Ixia ports.

In this case, the order will be the order when user created the port in Ixia GUI.

Note: User can always confirm the created order by `clear sorting` in Ixia GUI.

Examples:

Tester. Connect All	
Tester. Switch	tester01
Tester. Load And Start Traffic	
Sleep	30s
Tester. Stop Traffic	

Time format used in this module is same with `time string` format of Robot Framework. For more details about this, see [DateTime](#) library of Robot Framework.

Note: See [IxNet module](#), [IxLoad module](#) for details about keyword of each module.

Shortcuts

Close All · Connect · Connect All · Switch

Keywords

Keyword	Arguments	Documentation
Close All		Closes all connections
Connect	<i>name</i>	Connect to the tester <code>name</code>
Connect All		Connects to all testers
Switch	<i>name</i>	Switchs the current tester to <code>name</code>

Altogether 4 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:00.



ixload

Library scope: global
Named arguments: supported

Introduction

provides functions for IxLoad

To use IxLoad module, a IxLoad TCL server should be started properly.

RENAT runs a virtual IxLoad client locally in the background that connects to a Windows App server. Keywords from test case will send control messages to the client, which in turn will control the test ports.

Different to IxNetwork, an IxLoad test case usually stops within predefined time before `Stop Traffic` was called.

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

Shortcuts

Close · Collect Data · Get Test Report · Load Config · Load Traffic · Start Traffic · Stop Traffic

Keywords

Keyword	Arguments	Documentation
Close	<i>self</i>	Disconnects the current tester client
Collect Data	<i>self</i> , <i>prefix</i> =, <i>more_file</i> =, <i>ignore_not_found</i> =True	<p>Collects all result data and save them to the current active <code>result</code> folder</p> <p>A <code>prefix</code> will be automatically added to the file names.</p> <p>Currently the follow data will be downloaded to the local machine</p> <ul style="list-style-type: none">■ HTTP_Server.csv■ HTTP_Client.csv■ HTTP_Client - Per URL.csv■ HTTP_Server - Per URL.csv■ L2-3 Stats for Client Ports.csv■ L2-3 Stats for Server Ports.csv■ L2-3 Throughput Stats.csv■ Port CPU Statistics.csv <p>Extra files could be add by <code>more_file</code> which is a comma separated filename string</p> <p>When <code>ignore_not_found</code> is True, the keyword will not terminate even when the expected file is not found.</p>
Get Test Report	<i>self</i> , <i>prefix</i> =	Get the test report(PDF) and put it into the active result folder
Load Config	<i>self</i> , <i>config_name</i> =	<p>Loads the test traffic defined by <code>config_name</code></p> <p><code>file_path</code> is the path of the test file on the remote App server A path to a remote network drive could be use to load a config file on Renat server.</p>
Load Traffic	<i>self</i> , <i>file_path</i>	
Start Traffic	<i>self</i>	Starts the test traffic
Stop Traffic	<i>self</i>	<p>Stops the current running test</p> <p>Returns the elapsed time in seconds</p>

Altogether 7 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:05.



ixnet

Library scope: global
Named arguments: supported

Introduction

provides functions for IxNetwork

To use IxNetwork module, a IxNetwork TCL server should be started properly.

RENAT will connect to the App server and control the test ports. Test files and result will be inside 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 · **Add Quicktest** · **Apply Traffic** · **Change Frame Rate** · **Change Frame Rate Dynamic** · **Change Frame Size** · **Close** · **Collect All Data** · **Collect Data** · **Csv Snapshot** · **Get All Test Result** · **Get Quicktest List** · **Get Quicktest Result** · **Get Quicktest Result Path** · **Get Test Composer Result** · **Get Test Report** · **Get Test Result** · **Load And Start Traffic** · **Load Config** · **Load Traffic** · **Loss From File** · **Ping** · **Regenerate** · **Reset Config** · **Run Quicktest** · **Set All Traffic Item** · **Set Bgp Items** · **Set Bgp Neighbor** · **Set Capture Port** · **Set Traffic Item** · **Should Be Pingable** · **Start Capture** · **Start Protocol** · **Start Test Composer** · **Start Traffic** · **Stop All Protocols** · **Stop And Save Capture** · **Stop Quicktest** · **Stop Test Composer** · **Stop Traffic** · **Wait Until Connected**

Keywords

Keyword	Arguments	Documentation
Add Port	<i>self</i> , <i>force=True</i> , <i>time_out=2m</i> , <i>learn_time=2m</i>	<p>Add ports using the <code>real-port</code> information from active local config</p> <ul style="list-style-type: none"><code>time_out</code> is the wait time until port is connected (default is 2m)<code>learn_time</code> is the time waiting for arp to be learned (default is 2m) <p>Sample of local config tester:</p> <pre>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</pre> <p>port: 4</p> <pre>ip: 10.100.14.2 mask: 24 gw: 10.100.14.1</pre>
Add Quicktest	<i>self</i> , <i>name</i> , <i>test_type=rfc2544throughput</i> , <i>tx_mode=interleaved</i> , <i>clear_all=True</i>	<p>Create a new Quicktest with default value</p> <p>Type could be one of following: <code>rfc2544throughput</code>, <code>rfc2544frameLoss</code>, <code>rfc2544back2back</code>. Use Tester.Load Config to load a customized quicktest</p> <p>When <code>clear_all</code> is True, any existed quicktests will be cleared.</p> <p>Transmit mode <code>tx_mode</code> takes following values: <code>interleaved</code> (default) or <code>sequential</code>. The mode should be identical with the transmit mod of the ports.</p> <p>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.</p>
Apply Traffic	<i>self</i> , <i>refresh=True</i>	<p>Applies the current traffic configuration</p> <p><code>refresh</code>: Refreshed the learned information before apply the traffic or not Note: This is a blocking command</p>
Change Frame Rate	<i>self</i> , <i>value</i> , <i>pattern=.*</i> , <i>flow_pattern=.*</i>	<p>Changes the frame rate</p> <p>Parameter:</p> <ul style="list-style-type: none"><code>value</code>: value to set. Depends on the current configuration, this could be <code>percent line rate</code> or <code>bit per second</code> etc.<code>pattern</code>: a regular expression to identify traffic item name, default is everything <code>*</code><code>flow_pattern</code>: a regular expression to identify flow group inside the item
Change Frame Rate Dynamic	<i>self</i> , <i>value</i> , <i>pattern=.*</i>	<p>Changes the traffic flow rate on-fly</p> <p>No need to stop the running traffic to change the rate</p> <p>Parameter:</p> <ul style="list-style-type: none"><code>value</code>: value to set. Depend on the current configuration, this could be <code>percent line rate</code> or <code>bit per second</code> etc.<code>pattern</code>: a regular expression to identify traffic item name, default is everything <code>*</code>

Change Frame Size	<code>self, type, value, pattern=.*, flow_pattern=.*</code>	<p>Changes the frame size</p> <p>Parameter:</p> <ul style="list-style-type: none"> ■ <code>type</code>: could be <code>fixed size</code>, <code>increment_from`</code>, <code>increment_step</code> or <code>increment_to</code> ■ <code>value</code>: value to set ■ <code>pattern</code>: a regular expression to identify traffic item name, default is everything <code>.*</code> ■ <code>flow_pattern</code>: a regular expression to identify flow group inside the item
Close	<code>self</code>	Disconnects the current tester client
Collect All Data	<code>self, prefix=stat_</code>	Deprecated. Use
Collect Data	<code>self, view, prefix=stat_</code>	Deprecated. Use Get Test Result
Csv Snapshot	<code>self, prefix=snapshot_, *views</code>	<p>Get current CSV snapshot</p> <p>Parameters:</p> <ul style="list-style-type: none"> ■ <code>prefix</code>: prefix that be added to the filename. Default is <code>snapshot_</code> ■ <code>views</code>: list of target views (eg: <code>Port Statistics</code>, <code>Flow Statistics</code> ...). If <code>view</code> is <code>None</code>, all current available views will be target
Get All Test Result	<code>self, prefix=stat_</code>	<p>Collects all Ixia traffic data after traffic is stopped.</p> <p>Results are CSV files that are stored in <code>result</code> folder. The prefix <code>prefix</code> is appended to the original view name</p>
Get Quicktest List	<code>self</code>	Returns current loaded Quicktest list
Get Quicktest Result	<code>self, test_index=-1, prefix=, enable_all=True</code>	<p>Get the result.csv file from the latest Quicktests</p> <p><code>test_index</code> is a index of the current Quicktest. <code>-1</code> means that last one.</p>
Get Quicktest Result Path	<code>self, test_index=-1</code>	<p>Returns the path of the newest run of a Quicktest</p> <p><code>test_index</code> is a index of the current Quicktest. <code>-1</code> means that last one.</p>
Get Test Composer Result	<code>self, result_file=composer.log</code>	Get the result of test composer script
Get Test Report	<code>self, local_name=ixnet_report.pdf, enable_all=True</code>	<p>Generates and get report of the current active test in PDF format</p> <p><code>local_name</code>: name of the report on local machine. Default is <code>ixnet_report.pdf</code></p>
Get Test Result	<code>self, view, prefix=stat_</code>	<p>Collects traffic data of a <code>view</code> and export to a CSV file in <code>result</code> folder</p> <p>Currently, supported views are:</p> <p><code>Port Statistics</code>, <code>Global Protocol Statistics</code>, <code>BGP Aggregated Statistics</code>, <code>BGP Aggregated State Counts</code>, <code>OSPF Aggregated Statistics</code>, <code>OSPF Aggregated State Counts</code>, <code>OSPFv3 Aggregated Statistics</code>, <code>OSPFv3 Aggregated State Counts</code>, <code>L2-L3 Test Summary Statistics</code>, <code>Flow Statistics</code>, <code>Flow Detective</code>, <code>Data Plane Port Statistics</code>, <code>User Defined Statistics</code>, <code>Traffic Item Statistics</code></p> <p>Result were store as CSV files in <code>result</code> folder. If there is no valid data, view will be silently ignored</p> <p>The prefix <code>prefix</code> is appended to the view name for the CSV file.</p>
Load And Start Traffic	<code>self, wait_time1=10s, wait_time2=10s</code>	Combines Load Traffic and Start Traffic to one keyword.
Load Config	<code>self, config_name=, wait_time=2m, wait_time2=2m, apply=True, protocol=True, force=True, wait_time3=30s</code>	<p>loads traffic configuration, applies and start protocol if necessary.</p> <p>The config file name was defined in the <code>local.yaml</code> which is a Ixia Network configuration file and located in the <code>config</code> folder of the test.</p> <p>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 <code>tx_mode</code> to set the TxMode of the remapped ports.</p> <p>Parameters:</p> <ul style="list-style-type: none"> ■ <code>apply</code>: applies traffic when <code>True</code> otherwise ■ <code>protocol</code>: starts all protocols when <code>True</code> otherwise ■ <code>force</code>: force to reclaim the ports when <code>True</code> otherwise ■ <code>wait_time</code>: wait time after applying protocols ■ <code>wait_time2</code>: maximum wait time befor all ports become available. In common case, this is calculated automatically so user does not need to change this value. ■ <code>wait_time3</code>: default waiting time after config file is loaded (30s) <p>More information about ports could be define in <code>real_port</code> section like this:</p> <pre># tester information tester: tester: device: ixnet03_8009 config: bgp.ixncfg real-port: - chassis: 10.128.4.41 card: 4 port: 7 media: fiber tx_mode: interleaved</pre> <p>Configurable port parameters ares:</p> <ul style="list-style-type: none"> ■ <code>tx_mode</code>: <code>sequential</code> or <code>interleaved</code>(default)

		<ul style="list-style-type: none">media : copper or fiber (Note: no default 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, index=0	Returns packet loss by milliseconds and delta frame. Parameters: <ul style="list-style-type: none">file_name: flow information (csv format). Default is Flow_Statistics.csvindex: row index of the result(counted from zero) 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 <div><div>- Port <portName>: ping failed: port not assigned</div><div>- Response received from <sourceIp>/unknown . Sequence Number <sequenceNumber></div><div>- Ping request to <destinationIp>/unknown ip failed: <GenericPingError>/<error>: <genericError>unknown reason</div><div>- Error: Couldn't find any source interface for Send Ping to <destinationIp> on <portName> Id <id></div><div>- Error: Couldn't find any source IP for Send Ping to <destinationIp> on <portName> Id <id></div></div> Parameters: <ul style="list-style-type: none">src_port_index: index of Ixia port (starts from 0)src_intf_index: index of interface insides the port (starts from 0) Examples: <table><tr><td>Tester.Ping</td><td>1.1.1.1</td><td>0</td><td>0</td></tr><tr><td>Tester.Ping</td><td>1.1.1.1</td><td></td><td></td></tr></table>	Tester. Ping	1.1.1.1	0	0	Tester. Ping	1.1.1.1				
Tester. Ping	1.1.1.1	0	0									
Tester. Ping	1.1.1.1											
Regenerate	self	Regenerates all flow of current traffic items										
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: <ul style="list-style-type: none">port_index : index of the portneighbor_index : index of the neighbor or *route_range_index : index of the route range or *is_enable : \${TRUE} or \${FALSE} Note Examples: <table><tr><td>Tester.Set BGP Items</td><td>0</td><td>*</td><td>*</td><td>\${FALSE}</td></tr><tr><td>Tester.Set BGP Items</td><td>0</td><td>*</td><td>*</td><td>\${TRUE}</td></tr></table>	Tester. Set BGP Items	0	*	*	\${FALSE}	Tester. Set BGP Items	0	*	*	\${TRUE}
Tester. Set BGP Items	0	*	*	\${FALSE}								
Tester. Set BGP Items	0	*	*	\${TRUE}								
Set Bgp Neighbor	self, *indexes, **kwargs	Enables/Disables BGP entry by neighbor index kwargs contains following parameters: <ul style="list-style-type: none">indexes: is a list of index of BGP neighbor (index is started from zero)vport_index: is the target vport indexenabled: TRUE or FALSE Examples: <table><tr><td>Tester.Set BGP Item</td><td>0</td><td>1</td><td>vport_index=0</td><td>enabled=\${FALSE}</td></tr><tr><td>Tester.Set BGP Item</td><td>0</td><td>1</td><td>vport_index=1</td><td>enabled=\${TRUE}</td></tr></table>	Tester. Set BGP Item	0	1	vport_index=0	enabled=\${FALSE}	Tester. Set BGP Item	0	1	vport_index=1	enabled=\${TRUE}
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: <table><tr><td>Tester.Set Capture Port</td><td>0</td><td></td><td></td></tr><tr><td>Tester.Set Capture Port</td><td>control_mode=\${TRUE}</td><td>0</td><td>1</td></tr></table>	Tester. Set Capture Port	0			Tester. Set Capture Port	control_mode=\${TRUE}	0	1		
Tester. Set Capture Port	0											
Tester. Set Capture Port	control_mode=\${TRUE}	0	1									
Set Traffic Item	self, *items, **kwargs	Enables/Disables some traffic items items Parameters: <ul style="list-style-type: none">items : a list of Ixia traffic item nameenabled : 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										

		<p>Test frame item could be specified by item name. In this case the item is the start of frame item count from zero.</p> <p>Returns <code>True</code> if all items are set coordinately or otherwise</p> <p>Examples:</p> <table><tr><td>Set Traffic Item</td><td>Traffic Item 1</td><td>Traffic Item 2</td></tr><tr><td>Set Traffic Item</td><td>@{item_list}</td><td></td></tr><tr><td>Set Traffic Item</td><td>Traffic Item 1</td><td>enabled = \${FALSE}</td></tr></table>	Set Traffic Item	Traffic Item 1	Traffic Item 2	Set Traffic Item	@{item_list}		Set Traffic Item	Traffic Item 1	enabled = \${FALSE}
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	<code>self, dst_ip, src_port_index=0, src_intf_index=0</code>	<p>Ping from Ixia and raise an error if ping fails</p> <p>The keyword return <code>True</code> if succeeds</p>									
Start Capture	<code>self, wait_time=30s</code>	<p>Start packet capture</p> <p>Target ports are set by the configuration file or by [Set Capture] keyword</p>									
Start Protocol	<code>self, wait_time=1m</code>	<p>Starts all protocols and wait for <code>wait_time</code></p> <p>Default <code>wait_time</code> is 1 minute. Make sure <code>wait_time</code> is big engouh to start all protocols.</p>									
Start Test Composer	<code>self, script_name=Main_Procedure, run_num=1, wait_for_test=True, parameter=, wait=10s</code>	<p>Run a test composer script.</p> <p>The test composer script should be included in an Ixia Network configuration file and loaded properly with Load Config</p> <p>Parameters:</p> <ul style="list-style-type: none"><code>script_name</code> is the name of the script to run. Default value is <code>Main_Procedure</code>.<code>wait_for_test</code>: if <code>\$(TRUE)</code> then wait until the script finishes.<code>parameter</code>: parameter that is passed to the script. Parameter could be in 2 formats: <code>{{VAR1 VALUE1}} {VAR2 VALUE2}}</code> or simply as <code>VALUE1 VALUE2</code>. <p>The script must prepare <code>VAR1</code> and <code>VAR2</code> properly by <i>Test parameter</i>. See Ixia Network anout composer script for more details.</p> <ul style="list-style-type: none"><code>wait</code>: wait time before go to next keyword <p>Examples:</p> <table><tr><td>Tester.Start Test Composer</td><td>parameter=XXX YYY</td></tr><tr><td>Tester.Get Test Composer Result</td><td>result_file=script1.log</td></tr><tr><td>Tester.Start Test Composer</td><td>parameter={{VAR1 AAA}} {VAR2 BBB}}</td></tr><tr><td>Tester.Get Test Composer Result</td><td>result_file=script1.log</td></tr></table>	Tester. Start Test Composer	parameter=XXX YYY	Tester. Get Test Composer Result	result_file=script1.log	Tester. Start Test Composer	parameter={{VAR1 AAA}} {VAR2 BBB}}	Tester. Get Test Composer Result	result_file=script1.log	
Tester. Start Test Composer	parameter=XXX YYY										
Tester. Get Test Composer Result	result_file=script1.log										
Tester. Start Test Composer	parameter={{VAR1 AAA}} {VAR2 BBB}}										
Tester. Get Test Composer Result	result_file=script1.log										
Start Traffic	<code>self, wait_time=30s</code>	<p>Starts the current traffic settiing and wait for <code>wait_time</code>.</p> <p>Note: This is a asynchroneus action. After called, the keyword finishes immediatly but it will take a while before traffic starts</p> <p>By default the keyword will wait for 30 seconds.</p>									
Stop All Protocols	<code>self, wait_time=30s</code>	<p>Stop all running protocols</p>									
Stop And Save Capture	<code>self, prefix=, wait_until_finish=True, monitor_interval=5s</code>	<p>Stop current capture and save the results to folder specified by <code>path</code></p> <p>Captured files will be saved in current <code>result</code> folder with <code>prefix</code> appended in their names.</p> <p>Examples:</p> <table><tr><td>Tester.Start Capture</td><td></td></tr><tr><td>Sleep</td><td>10s</td></tr><tr><td>Tester.Stop And Save Capture</td><td>\$(RESULT_FOLDER)/capture.zip</td></tr></table>	Tester. Start Capture		Sleep	10s	Tester. Stop And Save Capture	\$(RESULT_FOLDER)/capture.zip			
Tester. Start Capture											
Sleep	10s										
Tester. Stop And Save Capture	\$(RESULT_FOLDER)/capture.zip										
Stop Quicktest	<code>self, test_index=0</code>	<p>Stops a running test</p>									
Stop Test Composer	<code>self, wait=10s</code>	<p>Stop a running composer</p> <p>Do nothing when a test composer has already stopped or no composer has been prepared.</p>									
Stop Traffic	<code>self, stop_protocol=False, wait_time=10s</code>	<p>Stops the current traffic and wait for <code>wait_time</code></p> <p>Parameters:</p> <ul style="list-style-type: none"><code>stop_protocol</code>: if <code>True</code> also stops all running protocols									
Wait Until Connected	<code>self, timeout_str=5m</code>	<ul style="list-style-type: none"><code>wait_time</code>: time to wait after apply the command <p>Waits until ports become enabled and connected</p>									

Altogether 41 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:04.



ixbps

Library scope: global
Named arguments: supported

Introduction

provides functions for Ixia Breaking Point

Breaking Point testers setting is set by `tester` section in `local.yaml`. Users need to specify the physical ports used by the test by its card and port number.

Setting example:

```
ixbps01:  
  device: ixbps01  
  config: test.bpt  
  real-port:  
    - card: 1  
      port: 0  
    - card: 1  
      port: 1
```

Notes: Ignore the *self* parameters when using those keywords. The module requires Breaking Point Python library installed properly to work.

Shortcuts

Cleanup Tests · Close · Get Card Config · Get Card Mode · Get Test Report · Load Config · Set Card Config · Start Test · Stop Test · Wait Until Finish

Keywords

Keyword	Arguments	Documentation
Cleanup Tests	<i>self</i>	Cleans up running test and release their ports
Close	<i>self</i>	Closes the connection to the BP box
Get Card Config	<i>self</i> , <i>slot_num</i>	Get card configuration for <i>slot_num</i> Parameter: <ul style="list-style-type: none">▪ <i>slot_num</i> is <i>all</i> or an integer start from 1 Result is a json formatted string contains the information for specific slot or <i>all</i> slots.
Get Card Mode	<i>self</i> , <i>slot_num</i>	Gets the <i>mode</i> of a specific slot
Get Test Report	<i>self</i> , <i>report_name=result</i> , <i>format=csv</i>	Gets and saves the test report to local disk The report will be in PDF format. If <code>export_csv</code> is <code>True</code> then test results are also exported by CSV format.
Load Config	<i>self</i> , <i>config_name=</i> , <i>force=True</i>	Loads test configuration <i>config_name</i> is defined in <code>local.yaml</code> or specific by user in the main scenario.
Set Card Config	<i>self</i> , <i>slot</i> , <i>action=mode</i> , <i>param=ixload</i>	Changes the configuration of BPS card Parameters: <ul style="list-style-type: none">▪ slot: slot number▪ action: <code>mode</code> or <code>perfacc</code>▪ param: depending on <i>action</i> Values of <i>param</i> : <ul style="list-style-type: none">▪ if <i>action</i> is <code>mode</code> then <i>param</i> should be <code>ixload</code>, <code>bp</code> or <code>bpl23</code> (BreakingPoint L2/3)▪ if <i>action</i> is <code>perfacc</code> then <i>param</i> should be <code>\$(TRUE)</code> or <code>\$(FALSE)</code>
Start Test	<i>self</i> , <i>test_name=None</i> , <i>force=True</i>	Starts a test by its name The system automatically reserve the ports defined in <code>local.yaml</code> . The reserved ports are released when the test is stopped <i>test_name</i> is the name of the testmodel saved in the configuration. If <i>test_name</i> is <code>None</code> , the 1st testmodel will be used. If <i>force</i> is <code>True</code> then all running tests and their reserved ports will be released.
Stop Test	<i>self</i> , <i>wait=5s</i>	Stops a running test
Wait Until Finish	<i>self</i> , <i>interval=30s</i> , <i>timeout=30m</i> , <i>verbose=False</i>	Waits until the test finished or timeout Notes: This is a blocking keyword

Altogether 10 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:06.



avaproxy

Library scope: global
Named arguments: supported

Introduction

Provides Spirent Avalance keywords

Table of Contents

- [Usage](#)
- [Test Result](#)
- [Keywords](#)

Usage

The module need to be used with a AVA proxy server with topology looks like this:

```
this module <=> AVA proxy server (32bit) <=> AVA 32bit library <=> AVA hardware
```

SPF test files should be exported from AVA GUI.

Sample `local.yaml` setting for AVA

```
# tester information
tester:
  ava01:
    device: ava01
    config: config.spf
    real-port:
      - chassis: 10.128.64.231
        card: 1
        port: 1
      - chassis: 10.128.64.232
        card: 1
        port: 1
```

The device `ava01` is define as below in global `device.yaml`:

```
ava01:
  type: avaproxy
  description: a AVA proxy (32bit)
  license-server: 10.128.64.222
  ip: 10.128.64.104
  port: 5001
```

Note: the `license-server` field is optional.

and the account is define in `auth.yaml` under `avaproxy` section

```
avaproxy:
  user: admin
```

Test Result

The AVA result is collected and save into the foler `ava` under current active `result` folder. The merged results for client/server is a XML file as `ava/result/merged/results.xml`.

Some predefined stylesheets are provided under `tools/xslt` folder and could be use to reformat this result like belows samples:

Common.Convert XML	\${RENAT_PATH}/tools/xslt/merged2html.xls	\${CURDIR}/\${RESULT_FOLDER}/ava/merged/results.xml	\${CURDIR}/\${RESULT_FOLDER}/ava_merged.html
Common.Convert XML	\${RENAT_PATH}/tools/xslt/merged2csv.xls	\${CURDIR}/\${RESULT_FOLDER}/ava/merged/results.xml	\${CURDIR}/\${RESULT_FOLDER}/ava_merged.csv

Note: AVA proxy server is not a part of Spirent solutions.

Shortcuts

Close · Get Test Result · Load Config · Start Test

Keywords

Keyword	Arguments	Documentation
Close	<code>self</code>	Disconnect to the remote server
Get Test Result	<code>self</code> , <code>name=ava</code>	Get test result folder The results will be saved under <i>current result</i> folder with the name <i>name</i>
Load Config	<code>self</code> , <code>config_name=</code>	Loads the SPF config file
Start Test	<code>self</code> , <code>timeout=5m</code>	Starts the test and wait until it finishes or timeout

Altogether 4 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:06.



Hypervisor

Library version: RENAT 0.1.11
Library scope: test suite
Named arguments: supported

Introduction

A module controls Hypervisors

A hypervisor is declared in `local.yaml` like this:

```
# esxi information
hypervisor:
  esxi-server:
    device: esxi-3-15
```

Notes: Currently support VMWare(Esxi) only

Shortcuts

Capture Mks Screenshot · Click Mks · Close · Close All · Connect · Connect All · Get Mks Ticket · Get Vm Id · Get Vm List · Get Vm Power State · Open Console · Power Off · Power On · Reset Capture Counter · Send Mks Cmd · Send Mks Key · Set Capture Format · Switch · Xrun

Keywords

Keyword	Arguments	Documentation
Capture Mks Screenshot	<i>*args, **kwargs</i>	
Click Mks	<i>*args, **kwargs</i>	
Close		Closes and disconnects from a hypervisor
Close All		Closes all current opend hypervisor connection
Connect	<i>hyper, name</i>	Connects to a Hypervisor
Connect All	<i>prefix=</i>	Connect to all hypervisor listed in local config yaml
Get Mks Ticket	<i>*args, **kwargs</i>	
Get Vm Id	<i>*args, **kwargs</i>	
Get Vm List	<i>*args, **kwargs</i>	
Get Vm Power State	<i>*args, **kwargs</i>	
Open Console	<i>*args, **kwargs</i>	
Power Off	<i>*args, **kwargs</i>	
Power On	<i>*args, **kwargs</i>	
Reset Capture Counter	<i>*args, **kwargs</i>	
Send Mks Cmd	<i>*args, **kwargs</i>	
Send Mks Key	<i>*args, **kwargs</i>	
Set Capture Format	<i>*args, **kwargs</i>	
Switch	<i>name</i>	Switch the current hypervisor to a new one
Xrun	<i>cmd, *args, **kwargs</i>	

Altogether 19 keywords.

Generated by [Libdoc](#) on 2019-01-15 10:51:12.



vmware

Library scope: global
Named arguments: supported

Introduction

provides function for VMware ESXI

Although this module provides some keywords for interact with the console (WebConsole), they only support very primitive actions. Do not use this to accomplish complex tasks. Instead, using VChannel through a SSH/Telnet channel.

Shortcuts

Capture Mks Screenshot · Click Mks · Get Mks Ticket · Get Vm Id · Get Vm List · Get Vm Power State · Open Console · Power Off · Power On · Reset Capture Counter · Send Mks Cmd · Send Mks Key · Set Capture Format

Keywords

Keyword	Arguments	Documentation																														
Capture Mks Screenshot	<i>self</i> , <i>filename=None</i> , <i>extra=</i>	<p>Captures the current web console to <i>filename</i></p> <p>If <i>filename</i> is <code>None</code>, the captured filename will be decided by the current <code>format</code> with an auto-increment counter and a <code>extra</code> at the end.</p> <p>Example:</p> <table><tr><td><code>Hypervisor.Capture Mks Screenshot</code></td><td><code># will create a file console_0000000001.png</code></td><td></td></tr><tr><td><code>Hypervisor.Capture Mks Screenshot</code></td><td><code>xxx.png</code></td><td><code># will create a file xxx.png</code></td></tr><tr><td><code>Hypervisor.Capture Mks Screenshot</code></td><td><code># will create a file console_0000000002.png</code></td><td></td></tr></table>	<code>Hypervisor.Capture Mks Screenshot</code>	<code># will create a file console_0000000001.png</code>		<code>Hypervisor.Capture Mks Screenshot</code>	<code>xxx.png</code>	<code># will create a file xxx.png</code>	<code>Hypervisor.Capture Mks Screenshot</code>	<code># will create a file console_0000000002.png</code>																						
<code>Hypervisor.Capture Mks Screenshot</code>	<code># will create a file console_0000000001.png</code>																															
<code>Hypervisor.Capture Mks Screenshot</code>	<code>xxx.png</code>	<code># will create a file xxx.png</code>																														
<code>Hypervisor.Capture Mks Screenshot</code>	<code># will create a file console_0000000002.png</code>																															
Click Mks	<i>self</i> , <i>xoffset</i> , <i>yoffset</i>	<p>Click on the MKS console at <i>xoffset,yoffset</i> coordinate</p> <p>Notes: The coordinate (0,0) is at the left corner of the console screen</p>																														
Get Mks Ticket	<i>self</i> , <i>vm_name</i>	Returns a MSK ticket for WebConsole																														
Get Vm Id	<i>self</i> , <i>vm_name</i>	Returns a VMID of a VM																														
Get Vm List	<i>self</i>	Returns current VM name list of the hypervisor																														
Get Vm Power State	<i>self</i> , <i>vm_name</i>	<p>Get vm power status</p> <p>Return <code>on</code> or <code>off</code></p>																														
Open Console	<i>self</i> , <i>vm_name</i> , <i>width=None</i> , <i>height=None</i>	<p>Opens a web console for a VM <i>vm_name</i></p> <p>Returns the <i>width</i> and <i>height</i> of the console Examples:</p> <table><tr><td><code>Hypervisor.Set Capture Format</code></td><td><code>console_%010d</code></td><td></td></tr><tr><td><code>Open Console</code></td><td><code>\${VM_NAME}</code></td><td></td></tr><tr><td><code>Hypervisor.Capture Mks Screenshot</code></td><td></td><td></td></tr><tr><td><code>Send Mks Cmd</code></td><td><code>root</code></td><td></td></tr><tr><td><code>Send Mks Cmd</code></td><td><code>password</code></td><td><code>wait=10s</code></td></tr><tr><td><code>Send Mks Cmd</code></td><td><code>ls</code></td><td></td></tr><tr><td><code>Hypervisor.Capture Mks Screenshot</code></td><td></td><td></td></tr><tr><td><code>Send Mks Key</code></td><td><code>\${CTRL_L}</code></td><td></td></tr><tr><td><code>Send Mks Cmd</code></td><td><code>whoami</code></td><td></td></tr><tr><td><code>Hypervisor.Capture Mks Screenshot`</code></td><td></td><td></td></tr></table>	<code>Hypervisor.Set Capture Format</code>	<code>console_%010d</code>		<code>Open Console</code>	<code>\${VM_NAME}</code>		<code>Hypervisor.Capture Mks Screenshot</code>			<code>Send Mks Cmd</code>	<code>root</code>		<code>Send Mks Cmd</code>	<code>password</code>	<code>wait=10s</code>	<code>Send Mks Cmd</code>	<code>ls</code>		<code>Hypervisor.Capture Mks Screenshot</code>			<code>Send Mks Key</code>	<code>\${CTRL_L}</code>		<code>Send Mks Cmd</code>	<code>whoami</code>		<code>Hypervisor.Capture Mks Screenshot`</code>		
<code>Hypervisor.Set Capture Format</code>	<code>console_%010d</code>																															
<code>Open Console</code>	<code>\${VM_NAME}</code>																															
<code>Hypervisor.Capture Mks Screenshot</code>																																
<code>Send Mks Cmd</code>	<code>root</code>																															
<code>Send Mks Cmd</code>	<code>password</code>	<code>wait=10s</code>																														
<code>Send Mks Cmd</code>	<code>ls</code>																															
<code>Hypervisor.Capture Mks Screenshot</code>																																
<code>Send Mks Key</code>	<code>\${CTRL_L}</code>																															
<code>Send Mks Cmd</code>	<code>whoami</code>																															
<code>Hypervisor.Capture Mks Screenshot`</code>																																
Power Off	<i>self</i> , <i>vm_name</i> , <i>graceful=True</i>	<p>Shutdowns a VM</p> <p>If <i>graceful</i> is <code>True</code>, a graceful shutdown is tried before a power off.</p> <p>Note: if VMware tools is not install on the VM, graceful shutdown is not available</p>																														
Power On	<i>self</i> , <i>vm_name</i>	Power on a VM																														
Reset Capture Counter	<i>self</i>																															
Send Mks Cmd	<i>self</i> , <i>cmd</i> , <i>wait=2s</i>	<p>Sends command to current web console and wait for a while</p> <p>By default, <i>wait</i> time is <code>2s</code> and the keyword will automaticall add a <code>Newline</code> char after sending the <i>cmd</i></p>																														
Send Mks Key	<i>self</i> , <i>key</i> , <i>wait=1s</i>	<p>Sends key strokes to current web console</p> <p>Special Ctrl char could be used as <code>\${CTRL_A}</code> to <code>\${CTRL_Z}</code></p> <p>Examples:</p> <table><tr><td></td><td></td><td></td></tr></table>																														

		Send MKS Key \${CTRL_L}
Set Capture Format	<i>self, format</i>	Set console capture format Initialized format is 'vmware_%010d'

Altogether 13 keywords.
Generated by [Libdoc](#) on 2019-01-15 10:51:13.

