Netmiko Documentation

Release 1.0

Kirk Byers

Oct 10, 2018

Contents

Contents:

CHAPTER 1

Netmiko Classes

1.1 BaseConnection

class netmiko.base_connection.**BaseConnection**(*ip=u", host=u", username=u", password=u", secret=u", port=None, device_type=u", verbose=False, global_delay_factor=1, use_keys=False, key_file=None, pkey=None, passphrase=None, allow_agent=False, ssh_strict=False, system_host_keys=False, alt_host_keys=False, alt_key_file=u", ssh_config_file=None, timeout=100, session_timeout=60, auth_timeout=None, blocking_timeout=8, keepalive=0, default_enter=None, response_return=None, serial_settings=None, fast_cli=False, session_log=None, session_log_record_writes=False, session_log_file_mode=u'write', allow_auto_change=False, encoding=u'ascii'*)

Defines vendor independent methods.

Otherwise method left as a stub method.

__enter__()
Establish a session using a Context Manager.

__exit__(*exc_type, exc_value, traceback*)
Gracefully close connection on Context Manager exit.

__init__(*ip=u'', host=u'', username=u'', password=u'', secret=u'', port=None, device_type=u'', verbose=False, global_delay_factor=1, use_keys=False, key_file=None, pkey=None, passphrase=None, allow_agent=False, ssh_strict=False, system_host_keys=False, alt_host_keys=False, alt_key_file=u'', ssh_config_file=None, timeout=100, session_timeout=60, auth_timeout=None, blocking_timeout=8, keepalive=0, default_enter=None, response_return=None, serial_settings=None, fast_cli=False, session_log=None, session_log_record_writes=False, session_log_file_mode=u'write', allow_auto_change=False, encoding=u'ascii'*)

Initialize attributes for establishing connection to target device.

param ip IP address of target device. Not required if *host* is provided.

type ip str

param host Hostname of target device. Not required if *ip* is provided.

type host str

param username Username to authenticate against target device if required.

type username str

param password Password to authenticate against target device if required.

type password str

param secret The enable password if target device requires one.

type secret str

param port The destination port used to connect to the target device.

type port int or None

param device_type Class selection based on device type.

type device_type str

param verbose Enable additional messages to standard output.

type verbose bool

param global_delay_factor Multiplication factor affecting Netmiko delays (default: 1).

type global_delay_factor int

param use_keys Connect to target device using SSH keys.

type use_keys bool

param key_file Filename path of the SSH key file to use.

type key_file str

param pkey SSH key object to use.

type pkey paramiko.PKey

param passphrase Passphrase to use for encrypted key; password will be used for key decryption if not specified.

type passphrase str

param allow_agent Enable use of SSH key-agent.

type allow_agent bool

param ssh_strict Automatically reject unknown SSH host keys (default: False, which means unknown SSH host keys will be accepted).

type ssh_strict bool

param system_host_keys Load host keys from the user's 'known_hosts' file.

type system_host_keys bool

param alt_host_keys If *True* host keys will be loaded from the file specified in 'alt_key_file'.

type alt_host_keys bool

param alt_key_file SSH host key file to use (if alt_host_keys=True).

type alt_key_file str

param ssh_config_file File name of OpenSSH configuration file.

type ssh_config_file str

param timeout Connection timeout.

type timeout float

param session_timeout Set a timeout for parallel requests.

type session_timeout float

param auth_timeout Set a timeout (in seconds) to wait for an authentication response.

type auth_timeout float

param keepalive Send SSH keepalive packets at a specific interval, in seconds. Currently defaults to 0, for backwards compatibility (it will not attempt to keep the connection alive).

type keepalive int

param default_enter Character(s) to send to correspond to enter key (default: '

').

type default_enter str

param response_return Character(s) to use in normalized return data to represent enter key (default: '

')

type response_return str

param fast_cli Provide a way to optimize for performance. Converts select_delay_factor to select smallest of global and specific. Sets default global_delay_factor to .1 (default: False)

type fast_cli boolean

param session_log File path or BufferedIOBase subclass object to write the session log to.

type session_log str

param session_log_record_writes The session log generally only records channel reads due to eliminate command duplication due to command echo. You can enable this if you want to record both channel reads and channel writes in the log (default: False).

type session_log_record_writes boolean

param session_log_file_mode "write" or "append" for session_log file mode (default: "write")

type session_log_file_mode str

param allow_auto_change Allow automatic configuration changes for terminal settings. (default: False)

type allow_auto_change bool

param encoding Encoding to be used when writing bytes to the output channel. (default: 'ascii')

type encoding str

`__weakref__`
list of weak references to the object (if defined)

`check_config_mode`(*check_string=u", pattern=u"*)
Checks if the device is in configuration mode or not.

Parameters

- **`check_string`** (*`str`*) – Identification of configuration mode from the device
- **`pattern`** (*`str`*) – Pattern to terminate reading of channel

`check_enable_mode`(*check_string=u"*)
Check if in enable mode. Return boolean.

Parameters **`check_string`** (*`str`*) – Identification of privilege mode from device

`cleanup`()
Any needed cleanup before closing connection.

`clear_buffer`()
Read any data available in the channel.

`close_session_log`()
Close the session_log file (if it is a file that we opened).

`commit`()
Commit method for platforms that support this.

`config_mode`(*config_command=u", pattern=u"*)
Enter into config_mode.

Parameters

- **`config_command`** (*`str`*) – Configuration command to send to the device
- **`pattern`** (*`str`*) – Pattern to terminate reading of channel

`disable_paging`(*command=u'terminal length 0', delay_factor=1*)
Disable paging default to a Cisco CLI method.

Parameters

- **`command`** (*`str`*) – Device command to disable pagination of output
- **`delay_factor`** (*`int`*) – See __init__: global_delay_factor

`disconnect`()
Try to gracefully close the SSH connection.

enable(*cmd=u”*, *pattern=u’ssword’*, *re_flags=2*)
Enter enable mode.

Parameters

- **cmd** (*str*) – Device command to enter enable mode
- **pattern** (*str*) – pattern to search for indicating device is waiting for password
- **re_flags** (*int*) – Regular expression flags used in conjunction with pattern

establish_connection(*width=None*, *height=None*)
Establish SSH connection to the network device

Timeout will generate a NetMikoTimeoutException Authentication failure will generate a NetMikoAuthenticationException

width and height are needed for Fortinet paging setting.

Parameters

- **width** (*int*) – Specified width of the VT100 terminal window
- **height** (*int*) – Specified height of the VT100 terminal window

exit_config_mode(*exit_config=u”*, *pattern=u”*)
Exit from configuration mode.

Parameters

- **exit_config** (*str*) – Command to exit configuration mode
- **pattern** (*str*) – Pattern to terminate reading of channel

exit_enable_mode(*exit_command=u”*)
Exit enable mode.

Parameters **exit_command** (*str*) – Command that exits the session from privileged mode

find_prompt(*delay_factor=1*)
Finds the current network device prompt, last line only.

Parameters **delay_factor** (*int*) – See __init__: global_delay_factor

is_alive()
Returns a boolean flag with the state of the connection.

normalize_cmd(*command*)
Normalize CLI commands to have a single trailing newline.

Parameters **command** (*str*) – Command that may require line feed to be normalized

normalize_linefeeds(*a_string*)
Convert ‘

, ‘, ‘

‘ to ‘ .‘

param a_string A string that may have non-normalized line feeds i.e. output returned from device, or a device prompt

type a_string str

open_session_log(*filename*, *mode=u’write’*)
Open the session_log file.

`paramiko_cleanup`()
Cleanup Paramiko to try to gracefully handle SSH session ending.

`read_channel`()
Generic handler that will read all the data from an SSH or telnet channel.

`read_until_pattern`(**args*, ***kwargs*)
Read channel until pattern detected. Return ALL data available.

`read_until_prompt`(**args*, ***kwargs*)
Read channel until self.base_prompt detected. Return ALL data available.

`read_until_prompt_or_pattern`(*pattern=u''*, *re_flags=0*)
Read until either self.base_prompt or pattern is detected.

Parameters

- **`pattern`** (*regular expression string*) – the pattern used to identify that the output is complete (i.e. stop reading when pattern is detected). pattern will be combined with self.base_prompt to terminate output reading when the first of self.base_prompt or pattern is detected.
- **`re_flags`** (*int*) – regex flags used in conjunction with pattern to search for prompt (defaults to no flags)

`save_config`(*cmd=u''*, *confirm=True*, *confirm_response=u''*)
Not Implemented

`select_delay_factor`(*delay_factor*)
Choose the greater of delay_factor or self.global_delay_factor (default). In fast_cli choose the lesser of delay_factor of self.global_delay_factor.

Parameters **`delay_factor`** (*int*) – See __init__: global_delay_factor

`send_command`(*command_string*, *expect_string=None*, *delay_factor=1*, *max_loops=500*, *auto_find_prompt=True*, *strip_prompt=True*, *strip_command=True*, *normalize=True*, *use_textfsm=False*)
Execute command_string on the SSH channel using a pattern-based mechanism. Generally used for show commands. By default this method will keep waiting to receive data until the network device prompt is detected. The current network device prompt will be determined automatically.

Parameters

- **`command_string`** (*str*) – The command to be executed on the remote device.
- **`expect_string`** (*str*) – Regular expression pattern to use for determining end of output. If left blank will default to being based on router prompt.
- **`delay_factor`** (*int*) – Multiplying factor used to adjust delays (default: 1).
- **`max_loops`** (*int*) – Controls wait time in conjunction with delay_factor. Will default to be based upon self.timeout.
- **`strip_prompt`** (*bool*) – Remove the trailing router prompt from the output (default: True).
- **`strip_command`** (*bool*) – Remove the echo of the command from the output (default: True).
- **`normalize`** (*bool*) – Ensure the proper enter is sent at end of command (default: True).
- **`use_textfsm`** – Process command output through TextFSM template (default: False).

`send_command_expect`(**args*, ***kwargs*)
Support previous name of send_command method.

Parameters

- **`args`** (*`list`*) – Positional arguments to send to send_command()
- **`kwargs`** (*`dict`*) – Keyword arguments to send to send_command()

`send_command_timing`(*command_string*, *delay_factor=1*, *max_loops=150*, *strip_prompt=True*, *strip_command=True*, *normalize=True*, *use_textfsm=False*)

Execute command_string on the SSH channel using a delay-based mechanism. Generally used for show commands.

Parameters

- **`command_string`** (*`str`*) – The command to be executed on the remote device.
- **`delay_factor`** (*`int or float`*) – Multiplying factor used to adjust delays (default: 1).
- **`max_loops`** (*`int`*) – Controls wait time in conjunction with delay_factor. Will default to be based upon self.timeout.
- **`strip_prompt`** (*`bool`*) – Remove the trailing router prompt from the output (default: True).
- **`strip_command`** (*`bool`*) – Remove the echo of the command from the output (default: True).
- **`normalize`** (*`bool`*) – Ensure the proper enter is sent at end of command (default: True).
- **`use_textfsm`** – Process command output through TextFSM template (default: False).

`send_config_from_file`(*config_file=None*, ***kwargs*)

Send configuration commands down the SSH channel from a file.

The file is processed line-by-line and each command is sent down the SSH channel.

**kwargs are passed to send_config_set method.

Parameters

- **`config_file`** (*`str`*) – Path to configuration file to be sent to the device
- **`kwargs`** (*`dict`*) – params to be sent to send_config_set method

`send_config_set`(*config_commands=None*, *exit_config_mode=True*, *delay_factor=1*, *max_loops=150*, *strip_prompt=False*, *strip_command=False*, *config_mode_command=None*)

Send configuration commands down the SSH channel.

config_commands is an iterable containing all of the configuration commands. The commands will be executed one after the other.

Automatically exits/enters configuration mode.

Parameters

- **`config_commands`** (*`list or string`*) – Multiple configuration commands to be sent to the device
- **`exit_config_mode`** (*`bool`*) – Determines whether or not to exit config mode after complete
- **`delay_factor`** (*`int`*) – Factor to adjust delays
- **`max_loops`** (*`int`*) – Controls wait time in conjunction with delay_factor (default: 150)
- **`strip_prompt`** (*`bool`*) – Determines whether or not to strip the prompt

- **strip_command** (*bool*) – Determines whether or not to strip the command
- **config_mode_command** (*str*) – The command to enter into config mode

session_preparation()

Prepare the session after the connection has been established

This method handles some differences that occur between various devices early on in the session.

In general, it should include: self._test_channel_read() self.set_base_prompt() self.disable_paging() self.set_terminal_width() self.clear_buffer()

set_base_prompt(*pri_prompt_terminator=u'#'*, *alt_prompt_terminator=u'>'*, *delay_factor=1*)

Sets self.base_prompt

Used as delimiter for stripping of trailing prompt in output.

Should be set to something that is general and applies in multiple contexts. For Cisco devices this will be set to router hostname (i.e. prompt without '>' or '#').

This will be set on entering user exec or privileged exec on Cisco, but not when entering/exiting config mode.

Parameters

- **pri_prompt_terminator** (*str*) – Primary trailing delimiter for identifying a device prompt
- **alt_prompt_terminator** (*str*) – Alternate trailing delimiter for identifying a device prompt
- **delay_factor** (*int*) – See __init__: global_delay_factor

set_terminal_width(*command=u''*, *delay_factor=1*)

CLI terminals try to automatically adjust the line based on the width of the terminal. This causes the output to get distorted when accessed programmatically.

Set terminal width to 511 which works on a broad set of devices.

Parameters

- **command** (*str*) – Command string to send to the device
- **delay_factor** (*int*) – See __init__: global_delay_factor

special_login_handler(*delay_factor=1*)

Handler for devices like WLC, Extreme ERS that throw up characters prior to login.

strip_ansi_escape_codes(*string_buffer*)

Remove any ANSI (VT100) ESC codes from the output

http://en.wikipedia.org/wiki/ANSI_escape_code

Note: this does not capture ALL possible ANSI Escape Codes only the ones I have encountered

Current codes that are filtered: ESC = '' or chr(27) ESC = is the escape character [^ in hex ('') ESC[24;27H Position cursor ESC[?25h Show the cursor ESC[E Next line (HP does ESC-E) ESC[K Erase line from cursor to the end of line ESC[2K Erase entire line ESC[1;24r Enable scrolling from start to row end ESC[?6l Reset mode screen with options 640 x 200 monochrome (graphics) ESC[?7l Disable line wrapping ESC[2J Code erase display ESC[00;32m Color Green (30 to 37 are different colors) more general pattern is

ESC[dd;ddm and ESC[dd;dd;ddm

ESC[6n Get cursor position

HP ProCurve and Cisco SG300 require this (possible others).

Parameters **`string_buffer`** (*`str`*) – The string to be processed to remove ANSI escape codes

`static strip_backspaces`(*output*)

Strip any backspace characters out of the output.

Parameters **`output`** (*`str`*) – Output obtained from a remote network device.

`strip_command`(*command_string*, *output*)

Strip command_string from output string

Cisco IOS adds backspaces into output for long commands (i.e. for commands that line wrap)

Parameters

- **`command_string`** (*`str`*) – The command string sent to the device
- **`output`** (*`str`*) – The returned output as a result of the command string sent to the device

`strip_prompt`(*a_string*)

Strip the trailing router prompt from the output.

Parameters **`a_string`** (*`str`*) – Returned string from device

`telnet_login`(*pri_prompt_terminator=u'#\\s*$'*, *alt_prompt_terminator=u'>\\s*$'*, *username_pattern=u'(?:user:|username|login|user name)'*, *pwd_pattern=u'assword'*, *delay_factor=1*, *max_loops=20*)

Telnet login. Can be username/password or just password.

Parameters

- **`pri_prompt_terminator`** (*`str`*) – Primary trailing delimiter for identifying a device prompt
- **`alt_prompt_terminator`** (*`str`*) – Alternate trailing delimiter for identifying a device prompt
- **`username_pattern`** (*`str`*) – Pattern used to identify the username prompt
- **`delay_factor`** (*`int`*) – See __init__: global_delay_factor
- **`max_loops`** – Controls the wait time in conjunction with the delay_factor

(default: 20)

`write_channel`(*out_data*)

Generic handler that will write to both SSH and telnet channel.

Parameters **`out_data`** (*`str (can be either unicode/byte string)`*) – data to be written to the channel

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Index