Wry Documentation

Release 2.0.2

Ocado Technology

Jul 11, 2018

Contents

A Pythonic AMT provisioning, configuration and management library.

It is [very] loosely based on the OpenStack Ironic project, hence the name.

CHAPTER 1

OK, but what is AMT?

Intel AMT is a remote management technology, widely implemented in modern Intel chipsets, and often advertised under the 'vPro' marketing tag.

It provides functionality such as:

- Remote power control
- Remote control via Serial-over-LAN and VNC
- Packet filtering
- Arbitary key/value data storage in NVRAM

AMT is implemented in BMC firmware and is, thus, operating-system independent.

See [Wikipedia](https://en.wikipedia.org/wiki/Intel_Active_Management_Technology) for more information.

1.1 Introduction

Wry is a library that facilitates interaction with, and configuration and control of, hardware devices that implement Intel AMT (vPro) technology.

It uses the openwsman python bindings.

1.1.1 Quickstart

Wry's functionality is exposed through the AMTDevice class. Initialize it as such:

```
>>> from wry import AMTDevice
>>> dev = AMTDevice.AMTDevice(address, False, username, password)
```

You can then access different aspects of device functionality, through aspect-specific namespaces. For example:

```
>>> dev.power.turn_on()
>>> dev.power.state
StateMap(state='on', sub_state=None)
```

Currently, the following namespaces are implemented:

- dev.power, via *wry.AMTPower.AMTPower*, provides access to:
  - Power state and control
- dev.boot, via *wry.AMTBoot.AMTBoot*, provides access to:
  - Boot configuration
  - Boot medium selection
- dev.vnc, via *wry.AMTKVM.AMTKVM*, provides access to:
  - Remote KVM (VNC) state and configuration
  - Setting of [additonal] user opt-in policy for KVM
- dev.opt_in, via *wry.AMTOptIn.AMTOptIn*, provides access to:
  - Setting of opt-in policies for KVM, Serial-over-LAN and media redirection
- dev.redirection, via *wry.AMTRedirection.AMTRedirection*, provides access to:
  - State and control of media redirection (IDER)
  - State and control of Serial-over-LAN (SOL)

You can click on a class name above, to see documentation for the available methods.

1.1.2 Status

Wry is in the early stages of development, and the interfaces it exposes may change as a result. Issues and pull requests are more than welcome.

Wry currently supports Python 2.7 and 3.5, it may well work with other versions.

1.1.3 Compatibility

Wry relies on the wsman AMT protocol, and therefore supports AMT versions 7(?) onwards.

Tested on the following hardware/firmware:

- Intel NUC DC53427HYE (BIOS 0037, ME 8.1.40.1416)
- Intel NUC5i5MYBE

1.1.4 License

1.2 Usage

1.2.1 Quickstart

Wry's functionality is exposed through the AMTDevice class. Initialize it as such:

```
>>> from wry import AMTDevice
>>> dev = AMTDevice.AMTDevice(address, False, username, password)
```

You can then access different aspects of device functionality, through aspect-specific namespaces. For example:

```
>>> dev.power.turn_on()
>>> dev.power.state
StateMap(state='on', sub_state=None)
```

Currently, the following namespaces are implemented:

- dev.power, via *wry.AMTPower.AMTPower*, provides access to:
  - Power state and control
- dev.boot, via *wry.AMTBoot.AMTBoot*, provides access to:
  - Boot configuration
  - Boot medium selection
- dev.vnc, via *wry.AMTKVM.AMTKVM*, provides access to:
  - Remote KVM (VNC) state and configuration
  - Setting of [additonal] user opt-in policy for KVM
- dev.opt_in, via *wry.AMTOptIn.AMTOptIn*, provides access to:
  - Setting of opt-in policies for KVM, Serial-over-LAN and media redirection
- dev.redirection, via *wry.AMTRedirection.AMTRedirection*, provides access to:
  - State and control of media redirection (IDER)
  - State and control of Serial-over-LAN (SOL)

You can click on a class name above, to see documentation for the available methods.

1.2.2 In-Depth

As well as the above, the AMTDevice class provides more genearlized/low-level functionality.

class wry.AMTDevice.**AMTDevice**(*target=None*, *is_ssl=True*, *username=None*, *password=None*, *debug=False*, *showxml=False*)

A wrapper class which packages AMT functionality into an accessible, device-centric format.

bios

A property which returns the BIOS identifiers (for the code, not settings)

dump(*as_json=True*)

Print all of the known information about the device.

Returns WryDict or json.

class wry.AMTPower.**AMTPower**(*device*)

Control over a device's power state.

available_states()
Get a list of available power states given our current power state

request_power_state_change(*power_state*)
Change the NUC to the specified power state

reset()
Reboot the device.

state
A property which describes the machine's power state.

A `wry.device.StateMap` as described in `wry.device.AMT_POWER_STATE_MAP`.

toggle()
If the device is off, turn it on. If it is on, turn it off.

turn_off()
Turn off the device.

turn_on()
Turn on the device.

class wry.AMTKVM.**AMTKVM**(*device*)
Control over a device's KVM (VNC) functionality.

default_screen
Default Screen. An integer.

enabled
Whether KVM functionality is enabled or disabled.

True/False

Note: This will return True even if KVM is enabled, but no ports for it are.

---

enabled_ports
Tells you (and/or allows you to set) the enabled ports for VNC.

opt_in_timeout
User opt-in timeout for KVM access, in seconds.

If set to 0, opt-in will be disabled.

password
This doesn't fail but always appears to return None

port_5900_enabled
Whether the standard VNC port (5900) is enabled. True/False.

session_timeout
Session timeout. In minutes.

setup(*password=''*, *port5900Enabled=False*, *defaultScreen=0*, *optIn=True*, *optInTimeout=60*, *sessionTimeout=10*)
Set all basic KVM settings in one call

class wry.AMTBoot.**AMTBoot**(*device*)
Control how the machine will boot next time.

config
Get configuration for the machine's next boot.

supported_media
Media the device can be configured to boot from.

class wry.AMTOptIn.**AMTOptIn**(*device*)
Manage user consent and opt-in codes.

code_ttl
How long an opt-in code lasts, in seconds.

class wry.AMTRedirection.**AMTRedirection**(*device*)
Control over Serial-over-LAN and storage redirection.

class wry.WryDict.**WryDict**(**args*, ***kwargs*)
An OrderedDict with the ability to be generated from wman-returned XML

classmethod from_xml(*xmlstr*)
Construct a WryDict from an XML string

to_xml
Return the XML representation of this WryDict

class wry.wsmanResource.**wsmanResource**(*target=None*, *is_ssl=False*, *username=None*, *password=None*, *resource=None*, *debug=False*, *showxml=False*)
Class to represent a resource on a wsman compatible server

enumerate(***kwargs*)
Return all instances of this Resource

get(*setting=''*, ***kwargs*)
Send a get request and return the result

@param setting: the setting to get the value of (None for all in this resources)

invoke(*method*, ***kwargs*)
Call a method and return the result

@param method: the method name to call

put(***kwargs*)
Get the current values, fill in new values and put back

@param **kwargs: zero or more settings to put back to the wsman server

request(*doc=None*, *params={}*)
Send a request to the target and return the response

class wry.wsmanModule.**wsmanModule**(*device*)
Base class for all wry modules

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

W

Index