
arguspy Documentation

Release 2.0.0

Canux CHENG

Aug 05, 2018

Contents

| | | |
|----------|------------------------------|----------|
| 1 | Install | 3 |
| 2 | Quick Start | 5 |
| 3 | How to use | 7 |
| 4 | The User Guide | 9 |
| 4.1 | HTTP | 9 |
| 4.2 | FTP | 9 |
| 4.3 | SSH | 9 |
| 4.4 | Mysql | 9 |
| 4.5 | MSSQL | 10 |
| 4.6 | WinRM | 10 |
| 4.7 | WMI | 10 |
| 4.8 | SNMP | 10 |
| 4.9 | Indices and tables | 10 |

arguspy is a pure python package for [\[monitoring-plugins\]](#).

In this package you can use lots of protocols to get the monitoring data.

All you need to do is just focus on your monitoring business.

You don't need to pay attention on basic protocols connection and the basic functions about the monitoring tools.

CHAPTER 1

Install

Use pip:

```
$pip install super_devops  
$pip install arguspy
```

Use source code:

```
$git clone https://github.com/crazy-canux/arguspy.git  
$cd arguspy  
$python setup.py install
```


CHAPTER 2

Quick Start

Import the protocol you want:

```
from arguspy.http_requests import Http
from arguspy.ftp_ftplib import Ftp
from arguspy.ssh_paramiko import Ssh
from arguspy.mssql_pymssql import Mssql
from arguspy.mysql_pymysql import Mysql
from arguspy.winrm_pywinrm import WinRM
from arguspy.wmi_sh import Wmi
from arguspy.wmi_subprocess import Wmi
...
```

Write your own business monitoring class:

```
class HttpSearch(Http):
    def __init__(self):
        super(HttpSearch, self).__init__()
        self.logger.debug("Init HttpSearch")

    def define_sub_options(self):
        super(HttpSearch, self).define_sub_options()
        self.hs_parser = self.subparsers.add_parser("...",
                                                    help='...',
                                                    description='...')

        self.hs_parser.add_argument('-s', '--search',
                                    default=None,
                                    required=False,
                                    help='Search pattern',
                                    dest='search')

    def httpsearch_handle(self):
        # Set the default status to ok.
        status = self.ok
        # Define your variables
```

(continues on next page)

(continued from previous page)

```
...

# Call the API to get the monitoring data.
# Check the API from python/ipython console.
# help(Http)
# dir(Http)
...

# Get the last status after check(Warning/Critical/Ok).
...

# Set the output.
self.shortoutput = "...
self.longoutput.append(...)
self.perfdata.append(...)

# Exit and show output.
self.logger.debug("Return status and output.")
status(self.output())
```

If put more than one business monitoring class in one plugin:

```
class Register(HttpSearch, HttpCertificate):
    def __init__(self):
        super(Register, self).__init__()
```

Define main function:

```
def main():
    plugin = Register()
    if plugin.args.option == 'httpsearch':
        plugin.httpsearch_handle()
    else:
        plugin.unknown("Unknown Action.")

if __name__ == "__main__":
    main()
```

CHAPTER 3

How to use

If you want to use this library and the plugins.

You must know something about the monitoring tools and protocols.

Output:

```
- $SERVICEOUTPUT$ -> ShortOutput
- $SERVICEPERFDATA$ -> PerfData
- $LONGSERVICEOUTPUT$ -> LongOutput
```

Return code:

```
- 0 OK
- 1 Warning
- 2 Critical
- 3 Unknown
```


4.1 HTTP

HTTP is Hyper Text Transfer Protocol.

HTTPS is Hyper Text Transfer Protocol over Secure Socket Layer.

[Requests] is the best third party python library for http/https.

4.2 FTP

FTP is file transfer protocol.

[ftplib] is standard python library for FTP.

4.3 SSH

[SSH] is remote connect protocol.

[Paramiko] is the best third party python library for SSH.

4.4 Mysql

[Mysql] is the most popular open source database.

[pymysql] is the best third party python library for mysql.

4.5 MSSQL

[MSSQL] is Microsoft Sql Server database.

[pymssql] is the best third party python library for MSSQL.

4.6 WinRM

[WinRM] is Windows Remote Management.

[pywinrm] is third party python library for WinRM.

4.7 WMI

[WMI] is Windows Management Instrumentation.

[subprocess] is standard python library for WMI.

4.8 SNMP

[SNMP] is simple network management protocol.

[pysnmp] is the best third party python library for SNMP.

4.9 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)