Python

- Flask
- Python Snippets
- Ubuntu 18.04.1, ISPConfig3, Python 3, Flask, Apache 2, and mod_wsgi
- Ansible and pipx
- Govee Bluetooth Hygrometers

Flask

Reference code bases:

- https://github.com/miguelgrinberg/flasky

Python Snippets

Useful Python3 snippets as I come across them.

Using xmltodict on the command line with ipset

Use ipset to dump a set list via xml, then convert that to json.

```
ipset list observed-unathorized -output xml | python3 -c 'import xmltodict, sys, json;
json.dump(xmltodict.parse(sys.stdin.read(), process_namespaces=True), sys.stdout, indent=4);'
```

#end

Ubuntu 18.04.1, ISPConfig3, Python 3, Flask, Apache 2, and mod_wsgi

Today I spent a lot of time trying to figure out how to get a Flask application started using Python 3 on Ubuntu 18.04.1. I had previously built an application using Python 2.7, Flask, and mod_wsgi, but it had been a while and the documentation I came across just wasn't connecting the dots properly. Here's my notes after the endeavor.

For the example, the root directory of the ISPConfig3 user is going to be /var/www/clients/client1/web1/.

```
# install
sudo apt install libapache2-mod-wsgi-py3 python3-pip
# disable mod_python if it's installed, or just uninstall it as mod_wsgi can't be loaded at
the same time

# install python3 virtualenv globally
sudo pip3 install virtualenv

# change to the private folder where we'll create the project
cd /var/www/clients/client1/web1/private/
# create the project folder and change into it
mkdir project1 && cd project1
# create the virtual environment in the env directory, which will be created for us
virtualenv -p python3 env
# activate the virtual environment
source env/bin/activate
# install flask
pip install flask
# setup some other directories and empty files
mkdir app log
touch wsgi.py run.py config.py app/__init__.py
```

wsgi.py

```
import sys
sys.path.insert(0, "/var/www/clients/client1/web1/private/project1")

from app import app as application
```

run.py

```
# WSGI Server for Development
# Use this during development vs. apache. Can view via [url]:8001
# Run using virtualenv. 'env/bin/python run.py'
from app import app

app.run(host='0.0.0.0', port=8001, debug=True)
```

app/__init__.py

```
from flask import Flask
app = Flask(__name__)

# Configurations
#app.config.from_object('config')

@app.route('/')
def hello_world():
    return 'Hello, World!'
```

Apache Directives for the WSGI configuration. This is to be added into the Options - Apache Directives section of the ISPConfig3 web interface. Note a change must be made to the ISPConfig3 vhost.conf.master file in order for the IfDefine statements to work. Diff included below. This is why I had to invert my logic and use IfDefine **!**ProtocolHTTP**S** instead of IfDefine ProtocolHTTP (see note below about the Define directive).

```
# For this to work /usr/local/ispconfig/server/conf/vhost.conf.master must be updated to
include the (Un)Define ProtocolHTTPS commands
<IfDefine !ProtocolHTTPS>
WSGIDaemonProcess client1web1 python-home=/var/www/clients/client1/web1/private/project1/env
python-path=/var/www/clients/client1/web1/private/project1
</IfDefine>
WSGIProcessGroup client1web1
```

```
WSGIScriptAlias / /var/www/clients/client1/web1/private/project1/wsgi.py
WSGIPassAuthorization On

<Directory /var/www/clients/client1/web1/private/project1>
    WSGIProcessGroup client1web1
    WSGIApplicationGroup %{GLOBAL}
WSGIScriptReloading On
    Require all granted
</Directory>
```

This is an example the generated Apache virtual host configuration. This is not the complete ISPConfig3 generated config file as it has all sorts of other configuration options.

```
<VirtualHost *:80>
ServerName test.domain.com
ServerAdmin webmaster@test.domain.com

ErrorLog /var/log/ispconfig/httpd/test.domain.com/error.log

<IfDefine !ProtocolHTTPS>
WSGIDaemonProcess client1web1 python-home=/var/www/clients/client1/web1/private/project1/env
python-path=/var/www/clients/client1/web1/private/project1
</IfDefine>
WSGIProcessGroup client1web1
WSGIScriptAlias / /var/www/clients/client1/web1/private/project1/wsgi.py
WSGIPassAuthorization On

<Directory /var/www/clients/client1/web1/private/project1>
    WSGIProcessGroup client1web1
    WSGIApplicationGroup %{GLOBAL}
WSGIScriptReloading On
    Require all granted
</Directory>

</VirtualHost>

<VirtualHost *:443>
Define ProtocolHTTPS

ServerName test.domain.com
```

```
ServerAdmin webmaster@test.domain.com

ErrorLog /var/log/ispconfig/httpd/test.domain.com/error.log

<IfModule mod_ssl.c>
SSLEngine on
SSLCertificateFile /var/www/clients/client1/web1/ssl/test.domain.com-le.crt
SSLCertificateKeyFile /var/www/clients/client1/web1/ssl/test.domain.com-le.key
SSLCertificateChainFile /var/www/clients/client1/web1/ssl/test.domain.com-le.bundle
</IfModule>

<IfDefine !ProtocolHTTPS>
WSGIDaemonProcess client1web1 python-home=/var/www/clients/client1/web1/private/project1/env
python-path=/var/www/clients/client1/web1/private/project1
</IfDefine>
WSGIProcessGroup client1web1
WSGIScriptAlias / /var/www/clients/client1/web1/private/project1/wsgi.py
WSGIPassAuthorization On

<Directory /var/www/clients/client1/web1/private/project1>
    WSGIProcessGroup client1web1
    WSGIApplicationGroup %{GLOBAL}
WSGIScriptReloading On
    Require all granted
</Directory>

UnDefine ProtocolHTTPS
</VirtualHost>
```

Diff of the vhost.conf.master file.

```
# diff -u vhost.conf.master.1 vhost.conf.master
--- vhost.conf.master.1 2018-08-23 03:50:52.539521052 -0500
+++ vhost.conf.master   2018-08-23 03:43:36.470905313 -0500
@@ -12,6 +12,9 @@

 <tmpl_loop name='vhosts'>
 <VirtualHost {tmpl_var name='ip_address'}:{tmpl_var name='port'}>
+<tmpl_if name='ssl_enabled'>
+Define ProtoHTTPS
```

```
+</tmpl_if>
 <tmpl_hook name='apache2_vhost:vhost_header'>
 <tmpl_if name='php' op='==' value='suphp'>
                DocumentRoot <tmpl_var name='web_document_root'>
@@ -524,6 +527,9 @@

 <tmpl_var name='apache_directives'>
 <tmpl_hook name='apache2_vhost:vhost_footer'>
+<tmpl_if name='ssl_enabled'>
+UnDefine ProtoHTTPS
+</tmpl_if>
 </VirtualHost>

 <tmpl_if name='apache_version' op='>=' value='2.4' format='version'>
```

Still to do

- ~~Figure out how to configure this properly using just the ISPConfig3 control panel. I can make it work editing the config files by hand, but they eventually get overwritten by ISPConfig3. The big problem I have right now is the WSGIDaemonProcess directive can only exist once... not in both the http and https VirtualHost definitions.~~ This was resolved as shown above.

Source material

- coxley/flask-file-structure (Source)
- proper structure of the wsgi.py file (Source)
- Flask 1.0.2 Quickstart (Source)
- Flask Deployment Options - mod_wsgi / Apache (Source)

Note regarding the Apache Define directive

"While this directive is supported in virtual host context, the changes it makes are visible to any later configuration directives, beyond any enclosing virtual host."

In order to workaround this behavior, I used a Define statement just after the opening <VirtualHost> tag, and an UnDefine statement just before the closing </VirtualHost> tag. This effectively maintains the scope of the Define to the VirtualHost.

(Source)

Ansible and pipx

The easiest way to make an up-to-date installation is by using pipx:

```
# initial installation
pipx install --include-deps ansible
pipx ensurepath
pipx completions
pipx inject ansible ansible-lint
pipx inject ansible jmespath
pipx inject ansible librouteros
pipx inject ansible paramiko
```

```
# future upgrades
pipx upgrade --include-injected ansible
```

Govee Bluetooth Hygrometers

Playing with some Govee H5074 and H5075 Bluetooth hygrometers. All code below was grabbed from Google search AI responses and then altered to suite.

```
import asyncio
import struct

from bleak import BleakScanner

timeout_seconds = 20
address_to_look_for = 'A4:C1:38'
service_id_to_look_for = '0000ec88-0000-1000-8000-00805f9b34fb'



class MyScanner:
    def __init__(self):
        self._scanner = BleakScanner()
        self._scanner.register_detection_callback(self.detection_callback)
        self.scanning = asyncio.Event()

    def detection_callback(self, device, advertisement_data):
        if device.address.startswith(address_to_look_for):
            if "H5074" in device.name:
                print(f"[FOUND] [{device.address}] Govee H5074: {device.name}")
            elif "H5075" in device.name:
                print(f"[FOUND] [{device.address}] Govee H5075: {device.name}")
            else:
                print(f"[FOUND] [{device.address}] Other Govee device: {device.name}")

            byte_data = advertisement_data.service_data.get(service_id_to_look_for)
            print(f"     -> {advertisement_data}")
            print(f"     -> byte_data -> {byte_data}")
```

```
#        if device.address.startswith(address_to_look_for)
#            byte_data = advertisement_data.service_data.get(service_id_to_look_for)
#            num_to_test, = struct.unpack_from('<I', byte_data, 0)
#            if num_to_test == 62976:
#                print('\t\tDevice found so we terminate')
#                self.scanning.clear()

    async def run(self):
        await self._scanner.start()
        self.scanning.set()
        end_time = loop.time() + timeout_seconds
        while self.scanning.is_set():
            if loop.time() > end_time:
                self.scanning.clear()
                print('\t\tScan has timed out so we terminate')
            await asyncio.sleep(0.1)
        await self._scanner.stop()


if __name__ == '__main__':
    my_scanner = MyScanner()
    loop = asyncio.get_event_loop()
    loop.run_until_complete(my_scanner.run())
```

Script output with MAC addresses redacted:

```
[FOUND] [A4:C1:38:xx:xx:xx] Govee H5074: Govee_H5074_24XX
     -> AdvertisementData(local_name='Govee_H5074_24XX', manufacturer_data={60552:
b'\x00e\x08\x86\x13d\x02', 76: b'\x02\x15INTELLI_ROCKS_HWPu\xf2\xff\xc2'},
service_uuids=['0000ec88-0000-1000-8000-00805f9b34fb'], rssi=-55)
[FOUND] [A4:C1:38:xx:xx:xx] Govee H5075: GVH5075_7EXX
     -> AdvertisementData(local_name='GVH5075_7EXX', manufacturer_data={60552: b'\x00\x03:-
d\x00', 76: b'\x02\x15INTELLI_ROCKS_HWPu\xf2\xff\x0c'}, service_uuids=['0000ec88-0000-1000-
8000-00805f9b34fb'], rssi=-62)
```