Python source code analysis using bandit

Install bandit

pip install bandit

Check python version

python --version
python3 --version

Sample python code to check vulnerability

app.py

```
def display():
    name = input ("Your name please : ")
    return name


display()
```

Running bandit scan

```
# bandit app.py
```

```
Test results:
>> Issue: [B322:blacklist] The input method in Python 2 will read from standard input, evaluate and run the
resulting string as python source code. This is similar, though in many ways worse, then using eval. On Python
2, use raw_input instead, input is safe in Python 3.
   Severity: High   Confidence: High
   Location: app.py:2
   More Info: https://bandit.readthedocs.io/en/latest/blacklists/blacklist_calls.html#b322-input
1       def display():
2           name = input ("Your name please : ")
3           return name

--------------------------------------------------


Code scanned:
        Total lines of code: 4
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 1
        Total issues (by confidence):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 1
Files skipped (0):
```

Exploiting the vulnerability in python2

```
# python app.py
Your name please : __import__('subprocess').call('last')
kali     tty7         :0                Fri Dec  4 01:32   still logged in
reboot   system boot  5.5.0-kali2-amd6 Fri Dec  4 01:31   still running
kali     pts/3        192.168.0.103    Tue Dec  1 05:01 - 05:33  (00:31)
kali     pts/2        192.168.0.103    Tue Dec  1 05:00 - 05:33  (00:32)

You can also try:
__import__('subprocess').call('uname')
__import__('subprocess').call('ls')
__import__('subprocess').call('hostname')
__import__('subprocess').call('history')
__import__('subprocess').call('uptime')
```

Testing the vulnerability in python3

```
root@kali:~/pysec# python3 app.py
Your name please : __import__('subprocess').call('last')
<< No output>>
```

Fixing the vulnerability

Fix the code by replacing **input** with **raw_input**

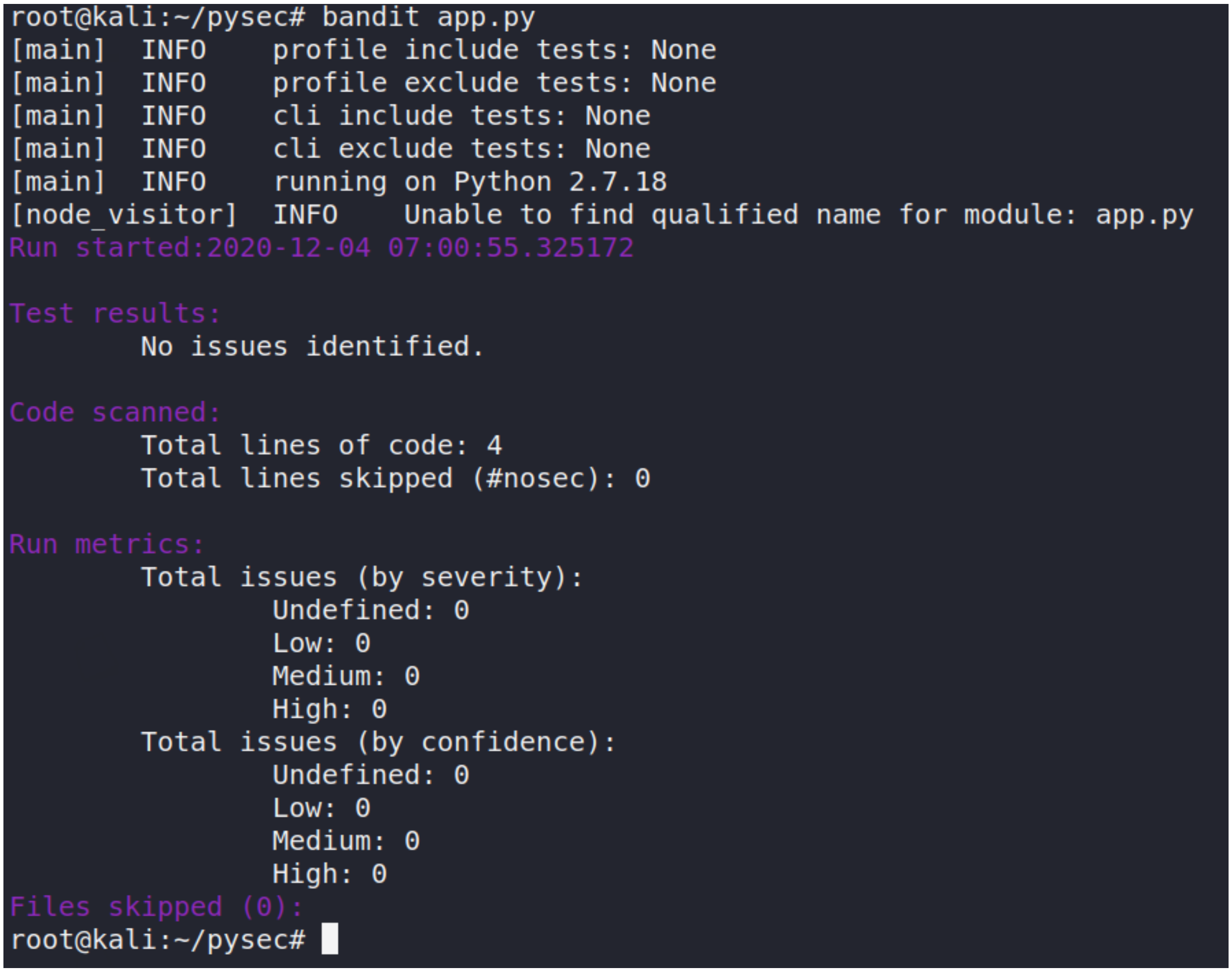
app.py

```
def display():
    name = raw_input ("Your name please : ")
    return name


display()
```

Scanning the vulnerability again after fixing

```
# bandit app.py
The output is clean (i.e. no errors)
```



```
root@kali:~/pysec# bandit app.py
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 2.7.18
[node_visitor]  INFO    Unable to find qualified name for module: app.py
Run started:2020-12-04 07:00:55.325172

Test results:
        No issues identified.

Code scanned:
        Total lines of code: 4
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 0
        Total issues (by confidence):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 0
Files skipped (0):
root@kali:~/pysec# 
```

Trying to exploit the vulnerability after fixing it

```
root@kali:~/pysec# python app.py
Your name please : __import__('subprocess').call('last')
<< No output>>
```