Pythonect for Security Professionals

Itzik Kotler
Pythonect

- *Pythonect* is a portmanteau of the words Python and Connect
- New, experimental, general-purpose dataflow programming language based on Python
- Current “stable“ version (True to Feb 12 2013): 0.4.1
- Made available under 'Modified BSD License'
- Influenced by: Unix Shell Scripting, Python, Perl
- Cross-platform (should run on any Python supported platform)
- Website: [http://www.pythonect.org/](http://www.pythonect.org/)
A few words on the Development

- Written purely in Python (2.7)
  - Works on CPython 2.x, and Jython 2.7 implementations
- Tests written in PyUnit
- Hosted on GitHub
- Commits tested by Travis CI
Installing and Using The Pythonect Interpreter

• Install directly from PyPI using easy_install or pip:
  - easy_install Pythonect
  OR
  - pip install Pythonect

• Clone the git repository:
  - git clone git://github.com/ikotler/pythonect.git
  - cd pythonect
  - python setup.py install
The Pythonect Interpreter

• Written and integrated with the Python environment:

  % pythonect
  Pythonect 0.4.1
  >>>
Dataflow Programming

- Programming paradigm that treats data as something that originates from a source, flows through a number of components and arrives at some final destination.
- Most suitable when developing applications that are themselves focused on the "flow" of data.
Dataflow Example

- A video signal processor which perhaps starts with a video input, modifies it through a number of processing components (video filters), and finally outputs it to a video display.
Dataflow Example

• Let's say we want to change our feed from a local file to a remote file on a Website? No problem!
Dataflow Example

- Let's say we want to write the Video B&W Frame Processor output to both a screen and a local file? No problem!
Dataflow Programming Advantages

- Promotes some good programming practices
- Makes development and maintenance very intuitive
- Programs can be divided between threads, processors, or computers more easily
<Pythonect Examples>
'Hello, world' -> print
"Hello, world" -> [print, print]
["Hello, world", "Hello, world"] -> print
range(99, 0, -1) \n   | [ _ % 2 == 0 ] \n   -> str \n   -> __ + " bottle(s) of beer on the wall," \n   -> print \n   -> __.split(' on')[0] + "." \n   -> print \n   -> print("Take one down, pass it around,"\n
<Pythonect Security Scripts/Examples>
ROT13 Encrypt & Decrypt

raw_input() -> _ .encode('rot13') -> print
Check if FTP server supports Anonymous Login

'ftp.gnu.org'
    -> ftplib.FTP
    -> _.login()
    -> print("Allow anonymous")
sys.argv[1] \\  
-> [str(_, '/'+x) for x in open(sys.argv[2],'r').read().split('
')] \
-> [(_, urllib.urlopen(_))] \
-> _[1].getcode() != 404 \
-> print "%s returns %s" % (_[0], _[1], _[1].getcode())
Command line Fuzzer

```
["%s", "%n", 'A', 'a', '0', '!', '$', '%', '*', '+', ',', '-', '.', '/', ':']
| [___ * n for n in [256, 512, 1024, 2048, 4096]]
| os.system('/bin/ping ' + __)
```
open('dana.jpg', 'r').read() \
  -> itertools.permutations \
  -> open('output_' + hex(_.__hash__()) + '.jpg', 'w').write(''.join(_))
Compute MALWARE.EXE's MD5 & SHA1

"MALWARE.EXE"
-> [os.system("/usr/bin/md5sum " + _), os.system("/usr/bin/shasum " + _)]
Compute MALWARE.EXE's Entropy

**Entropy.py:**

```python
import math

def entropy(data):
    entropy = 0
    if data:
        for x in range(2**8):
            p_x = float(data.count(chr(x))) / len(data)
            if p_x > 0:
                entropy += - p_x * math.log(p_x, 2)
    return entropy
```

**Pythonect:**

```
"MALWARE.EXE" \
-> open(_, 'r').read() \
-> entropy.entropy \
-> print
```
References / More Examples

• My Blog
  – Scraping LinkedIn Public Profiles for Fun and Profit
  – Fuzzing Like A Boss with Pythonect
  – Automated Static Malware Analysis with Pythonect

• LightBulbOne (Blog)
  – Fuzzy iOS Messages!
Pythonect Roadmap

- Support Python 3k
- Support Stackless Python
- Support IronPython
- Support GPU Programming
- Fix bugs and etc.
Questions?
Thanks!

Website: http://www.pythonect.org
Mailing list: pythonect@googlegroups.com