Workbench Documentation

Release 0.1

Brian Wylie

February 16, 2016

Contents

Please note the coverage/health (typically 95%) are super bad right now. It's a temporary issue that we're working on. :)

Contents:

CHAPTER 1

About Workbench

1.1 A medium-data framework for security research and development teams.

Workbench focuses on simplicity, transparency, and easy on-site customization. As an open source python project it provides light-weight task management, execution and pipelining for a loosely-coupled set of python classes.

1.2 Detailed Project Description

The workbench project takes the workbench metaphore seriously. It's a platform that allows you to do work; it provides a flat work surface that supports your ability to combine tools (python modules) together. In general a workbench never constrains you (oh no! you can't use those 3 tools together!) on the flip side it doesn't hold your hand either. Using the workbench software is a bit like using a Lego set, you can put the pieces together however you want AND adding your own pieces is super easy!.

1.2.1 Loosely coupled

- No inheritance relationships
- No knowledge of data structures
- Just take some input and barf some output (no format requirements)

1.2.2 Flat

- Workers (that's it... everything is a worker)
- Server dynamically loads workers from a directory called 'workers'

1.2.3 Robust

- Worker fails to load (that's fine)
- Worker crashes (no sweat, that request fails but system chugs on)

1.2.4 Transparency

- All worker output is reflected in the data store (currently Mongo)
- Use RoboMongo (see below) to inspect exactly what workers are outputting.

1.2.5 Small Granularity

- The system works by passing references from one worker to another so there is NO benefit to large granularity workers.
- It's super easy to have a worker that aggregates information from a set of workers, the opposite (breaking apart a large code chunk into smaller units) is almost never easy.
- Pull just what you want, workers and views (which are just workers) can be selectve about exactly which fields get pulled from which workers.

CHAPTER 2

Installing Workbench:

2.1 Workbench Client:

```
$ pip install zerorpc; echo 'Done!'
```

2.2 Workbench Server:

The indexers 'Neo4j' and 'ElasticSearch' are optional. We strongly suggest you install both of them but we also appreciate that there are cases where that's not possible or feasible.

2.2.1 Mac/OSX

- brew install mongodb
- brew install yara
- brew install libmagic
- brew install bro
- Put the bro executable in your PATH (/usr/local/bin or wherever bro is)

2.2.2 Ubuntu (14.04 and 12.04)

- sudo apt-get install mongodb
- sudo apt-get install python-dev
- sudo apt-get install g++
- sudo apt-get install libssl0.9.8
- Bro IDS:
- Put the bro executable in your PATH (/opt/bro/bin or wherever bro is)

  In general the Bro debian package files are WAY too locked down with dependencies on exact versions of libc6 and python2.6. We have a more 'flexible' version Bro-2.2-Linux-x86_64_flex.deb.

  - sudo dpkg -i Bro-2.2-Linux-x86_64_flex.deb

If using the Debian package above doesn't work out: - Check out the Installation tutorial here - or this one here - or go to offical Bro Downloads www.bro.org/download/

2.3 Install Indexers

2.3.1 Mac/OSX

- brew install elasticsearch
- pip install -U elasticsearch
- brew install neo4j
  - Note: You may need to install Java JDK 1.7 Oracle JDK 1.7 DMG for macs.

2.3.2 Ubuntu (14.04 and 12.04)

- Neo4j: See official instructions for Neo4j here
  - Note: You may need to install Java JDK 1.7. If you have Java 1.7 installed , and error says otherwise, run update-alternatives –config java and select Java 1.7
- ElasticSearch:
  - wget https://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-1.2.1.deb
  - sudo dpkg -i elasticsearch-1.2.1.deb
  - sudo update-rc.d elasticsearch defaults 95 10
  - sudo /etc/init.d/elasticsearch start
  - Any issues see elasticsearch_webpage

2.4 Pull the repository

Warning!: The repository contains malcious data samples, be careful, exclude the workbench directory from AV, etc...

2.5 Install Python Modules

Note: Workbench is continuously tested with python 2.7. We're currently working on Python 3 support (Issue 92).

- cd workbench
- pip install -r requirements.txt
- Go have a large cup of coffee...

2.5.1 Optional Tools

Robomongo

Robomongo is a shell-centric cross-platform MongoDB management tool. Simply, it is a handy GUI to inspect your mongodb.

- http://robomongo.org/
- download and follow install instructions
- create a new connection to localhost (default settings fine). Name it as you wish.

2.6 Dependency Installation Errors

Python Modules

Note: If you get a bunch of clang errors about unknown arguments or 'cannot link a simple C program' add the following FLAGs:

```
```
$ export CFLAGS=-Qunused-arguments
$ export CPPFLAGS=-Qunused-arguments
```
```

Errors when running Tests

If when running the worker tests you get some errors like 'MagicError: regexec error 17, (illegal byte sequence)' it's an issue with libmagic 5.17, revert to libmagic 5.16. Using brew on Mac:

```
$ cd /usr/local
$ brew versions libmagic # Copy the line for version 5.16, then paste (for me it looked like the foll
$ git checkout bfb6589 Library/Formula/libmagic.rb
$ brew uninstall libmagic
$ brew install libmagic
```

2.7 Deprecated Stuff

Scapy Install

- brew tap Homebrew/python
- brew install scapy
- brew install pypcap
- If you get error about pyrex.distutils:
  - pip install pyrex (or if this doesn't work do easy_install pyrex)
  - and then retry the 'brew install pypcap'
- Still not working try pyrex from scatch pyrex

(2-5-14): For scapy python binding you have to manually install the latest release from secdev.org and follow the instructions (like first 5 lines)

2.8 Deprecated Instructions for Ubuntu 12.04

2.8.1 Ubuntu (tested on 12.04)

- Mongo: Go through the steps given at MongoDB Installation Tutorial
- Bro IDS: Check out the Installation tutorial here
- Yara: Read the installation instructions here
- sudo apt-get install libmagic-dev
- sudo apt-get install libxml2-dev
- sudo apt-get install libxslt-dev
- sudo apt-get install libevent-dev

CHAPTER 3

Running WorkBench

3.1 Server (localhost or server machine)

3.2 Example Clients (use -s for remote server)

There are about a dozen example clients showing how to use workbench on pcaps, PEfiles, pdfs, and log files. We even have a simple nodes.js client (looking for node devs to pop some pull requests :).

3.3 Configuration File Information

When you first run workbench it copies default.ini to config.ini within the workbench/server directory, you can make local changes to this file without worrying about it getting overwritten on the next 'git pull'. Also you can store API keys in it because it never gets pushed back to the repository.

```
# Example/default configuration for the workbench server
[workbench]

# Server URI (server machine ip or name)
# Example: mybigserver or 12.34.56.789
server_uri = localhost

# DataStore URI (datastore machine ip or name)
# Example: mybigserver or 12.34.56.789
datastore_uri = localhost

# Neo4j URI (Neo4j Graph DB machine ip or name)
# Example: mybigserver or 12.34.56.789
neo4j_uri = localhost

# ElasticSearch URI (ELS machine ip or name)
# Example: mybigserver or 12.34.56.789
els_uri = localhost

# DataStore Database
# Example: customer123, ml_talk, pdf_deep
database = workbench

# Storage Limits (in MegaBytes, 0 for no limit)
worker_cap = 10
```

```
samples_cap = 200

# VT API Key
# Example: 93748163412341234v123947
vt_apikey = 123
```

3.4 Workbench Examples

Please note that all of these notebooks are 'clients' hitting the workbench server. Making your own client is super easy! See Making a Client

- PCAP to Graph (A short teaser)
- Workbench Demo
- Adding a new Worker (super hawt)
- PCAP to Dataframe
- PCAP DriveBy Analysis
- Using Neo4j for PE File Sim Graph
- Generator Pipelines Notebook
- WIP Notebooks
  - Network Stream Analysis Notebook
  - PE File Static Analysis Notebook

3.4.1 Making your own Worker

Fill in info

3.4.2 Making your own Client

Although the Workbench repository has dozens of clients (see workbench/clients)there is NO official client to workbench. Clients are examples of how YOU can just use ZeroRPC from the Python, Node.js, or CLI interfaces. See ZeroRPC.

```
import zerorpc
c = zerorpc.Client()
c.connect("tcp://127.0.0.1:4242")
with open('evil.pcap','rb') as f:
    md5 = c.store_sample('evil.pcap', f.read())
print c.work_request('pcap_meta', md5)
```

Output from above 'client': `python {'pcap_meta': {'encoding': 'binary', 'file_size': 54339570, 'file_type': 'tcpdump (little-endian) - version 2.4 (Ethernet, 65535)', 'filename': 'evil.pcap', 'import_time': '2014-02-08T22:15:50.282000Z', 'md5': 'bba97e16d7f92240196dc0caef9c457a', 'mime_type': 'application/vnd.tcpdump.pcap'}}` ### Running the IPython Notebooks * brew install freetype * brew install gfortran * pip install -r requirements_notebooks.txt * Go to Starbucks..

3.4.3 Workbench Conventions

Workers should adhere to the following naming conventions (not enforced)

- If you work on a specific type of sample than start the name with that
- Examples: pcap_bro.py, pe_features.py, log_meta.py
- A worker that is new/experimental should start with 'x_' (x_pcap_razor.py)
- A 'view'(worker that handles 'presentation') should start with 'view_'
- Examples: view_log_meta.py, view_pdf.py, view_pe.py

3.5 Running Tests

Unit testing and sub-pipeline tests

Full pipeline tests (clients exercise a larger set of components)

3.6 Test Coverage

If you want to run the test code coverage properly you'll need to create a ~/.noserc file with these options:

```
[nosetests]
with-coverage=1
cover-erase=1
cover-inclusive=1
cover-min-percentage=90
cover-package=.
```

3.7 Benign Error

We have no idea why occasionaly you see this pop up in the server output. To our knowledge it literally has no impact on any functionality or robustness. If you know anything about this please help us out by opening an issue and pull request. :)

3.7.1 VirusTotal Warning

The vt_query.py worker uses a shared 'low-volume' API key provided by SuperCowPowers LLC. When running the vt_query worker the following warning happens quite often:

```
"VirusTotal Query Error, no valid response... past per min quota?"
```

If you'd like to use the vt_query worker on a regular basis, you'll have to put your own VirusTotal API key in the workbench/server/config.ini file.

CHAPTER 4

Contributions/Support/Getting Involved

Workbench is committed to providing open source security software. If you're a developer looking to chip-in or want to support the project please contact us at support@supercowpowers.com or visit one of the links below:

- Users email list: workbench-users
- Developer email list: workbench-devs
- Feature requests: issue tracker
- **Earn a T-Shirt!**: All issues have cow points, get 100+ for T
- **Buy a T-Shirt!**: SuperCowPowers
- Donate: SuperCowPowers

4.1 Git Development Model

We're going to use the 'GitHub Flow' model.

- To work on something new, create a descriptively named branch off of master (ie:new-oauth2-scopes)
- Commit to that branch locally and regularly push your work to the same named branch on the server
- When you need feedback or help, or you think the branch is ready for merging, open a pull request
- After someone else has reviewed and signed off on the feature, you can merge it into master

4.2 Git Example

```
$ git checkout -b my-awesome
$ git push -u origin my-awesome
$ <code for a bit>; git push
$ <code for a bit>; git push
$ Go to github and hit 'New pull request'
```

4.2.1 Bounties (Rewards for contributing to Workbench)

Top Bounties

- Bro Scripts for OWASP Top 10 (1000 Cow Points)

- Python based SWF Decompiler/Decompression (500 Cow Points)
- Deep PDF Static Analysis (500 Cow Points)
- Worker for Cab File extraction (100 Cow Points)

FAQ about Cow Points

- Are Cow Points worth anything? : No
- Will Cow Points ever be worth anything? : Maybe
- Are Cow Points officially tracked? : Yes
- Will I receive good Karma for Cow Points? : Yes