BLACKPHENIX

v1.0.0

OPEN-SOURCE MALWARE ANALYSIS + AUTOMATION FRAMEWORK

Developed by Chris Navarrete @ Fortiguard Labs

DOCUMENT VERSION CONTROL

Version Number	Date Issued	Update Information
V1.0	9/17/2019	First published version

Contents

Getting Started

BLACKPHENIX is an open-source malware analysis automation framework composed of services, scripts, plug-ins, and tools and is based on a Command-and-Control (C&C) architecture.

This framework was released and presented at [BlackHat Arsenal 2019](#).

Features

- Easy Installation & Deployment
- Tool automation modules
- Virtual Machine management
- Scripting support (Python)
- Internet emulation
- Traffic redirection
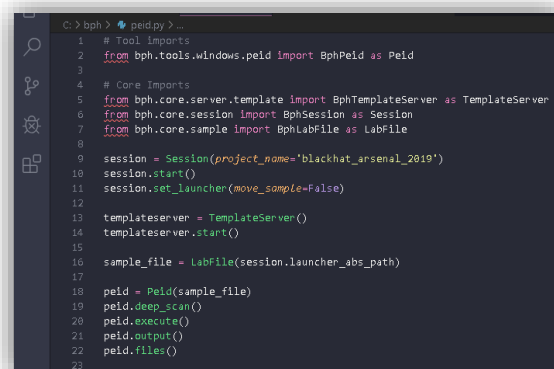- TOR support

How this tool can be used?

An analyst can use the framework to perform automated virtual machine-based malware analysis activities by automating the execution of well-known analysis tools, custom tools, and scripts that run in a remote virtual machine through the execution of python scripts called "**BPH Scripts**" and "**BPH Analysis Modules**" to perform parsing and further data analysis.

What is a BPH Script?

BPH Scripts are python scripts that import Windows tools python modules (**BPH Plug-ins**) and contain execution instructions for one or more of imported tools (bundled execution). For instance, a BPH script can call the UPX tool to unpack a compressed UPX executable and the next instruction can call another tool, such as ExeInfoPe or any other tool selected by the user.

The following picture shows the structure of the **PEiD BPH Script**. This script performs the following actions:

- Imports required libraries
- Creates a session
- Sets the malware sample to work on
- Initialize Template server
- Initialize PEiD's object
- Sets scan type (deep_scan) and execute the tool inside the target virtual machine
- Shows the tool's output in the console
- Lists the tool's log file in the console


```
C: > bph > 🐍 peid.py > ...
 1   # Tool imports
 2   from bph.tools.windows.peid import BphPeid as Peid
 3
 4   # Core Imports
 5   from bph.core.server.template import BphTemplateServer as TemplateServer
 6   from bph.core.session import BphSession as Session
 7   from bph.core.sample import BphLabFile as LabFile
 8
 9   session = Session(project_name='blackhat_arsenal_2019')
10   session.start()
11   session.set_launcher(move_sample=False)
12
13   templateserver = TemplateServer()
14   templateserver.start()
15
16   sample_file = LabFile(session.launcher_abs_path)
17
18   peid = Peid(sample_file)
19   peid.deep_scan()
20   peid.execute()
21   peid.output()
22   peid.files()
23
```

What is a BPH Analysis Module?

BPH Analysis Module are python scripts that can be imported and used to perform further analysis on the data collected by **BPH Scripts**. Users can also create code snippets "on-the-fly" within the BPH Script.

Malware Analysis Scenarios

Network Analysis Scenario

An analyst is interested in identifying which domains, IP addresses, ports and programs generate network traffic while executing a malware sample. A BPH Script can be developed to run the "***NetworkTrafficView***" tool and a BPH Analysis module to parse the tool's output (CSV) and extract the required information.

Behavioral Analysis Scenario

An analyst is interested in collecting behavioral data while executing a malware sample. A BPH Script can be developed to run "***Process Monitor***" tool and a BPH Analysis module to parse the tool's output (PML, XML).

Memory Analysis Scenario

An analyst is interested in collecting memory dump data while executing a malware sample. A BPH Script can be developed to run "***Pd32***" tool to generate memory dump data and a BPH Analysis module to work with the tool's output (EXE, DLL).

Packer Detection and Unpacking Scenario (Bundled Execution)

An analyst is interested in detecting if a malware sample is UPX packed and if the case, then unpacks and continues working on the unpacked version of the given sample. A BPH Script can be developed to run "**PEiD**" and "**UPX**" tools, and then perform further analysis activities on the received unpacked malware sample.

How does the tool execution work?

A BPH Script loads a tool module (BPH Module), which loads a tool profile (BPH Plug-ins) and generates a JSON serialized object and transmits to the guest virtual machine for execution. All data collected by the tools in the virtual machine is later reported back for further process by a BPH Analysis module.

Each analysis tool (e.g. PEiD, DUMPPE, ExeInfoPe, etc.) has its own BPH Module and Plug-in.

Virtual Machine Architecture

BLACKPHENIX relies on virtual machines to perform its operational activities and is composed by one host machine (**BPH VM Manager – Host**), one guest virtual machines (**BPH Controller – Guest VM**), and one - or more – guest virtual machine(s) (**BPH Analysis – Guest VM**), depending on whether the user will support analysis in several Windows OS'es AND/OR windows programs.
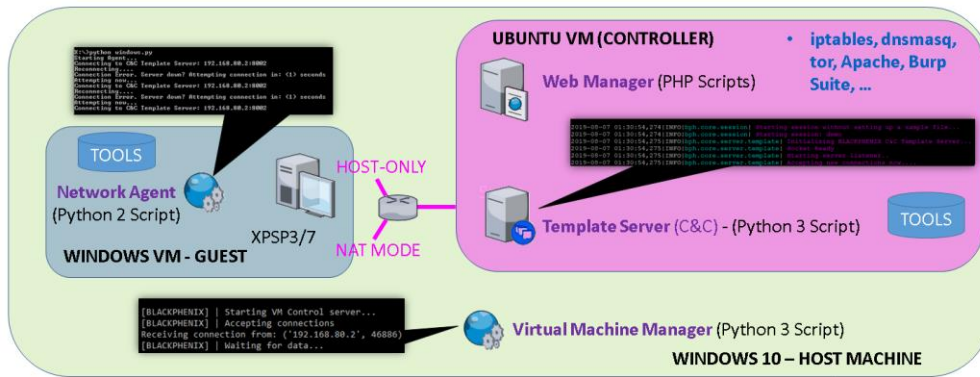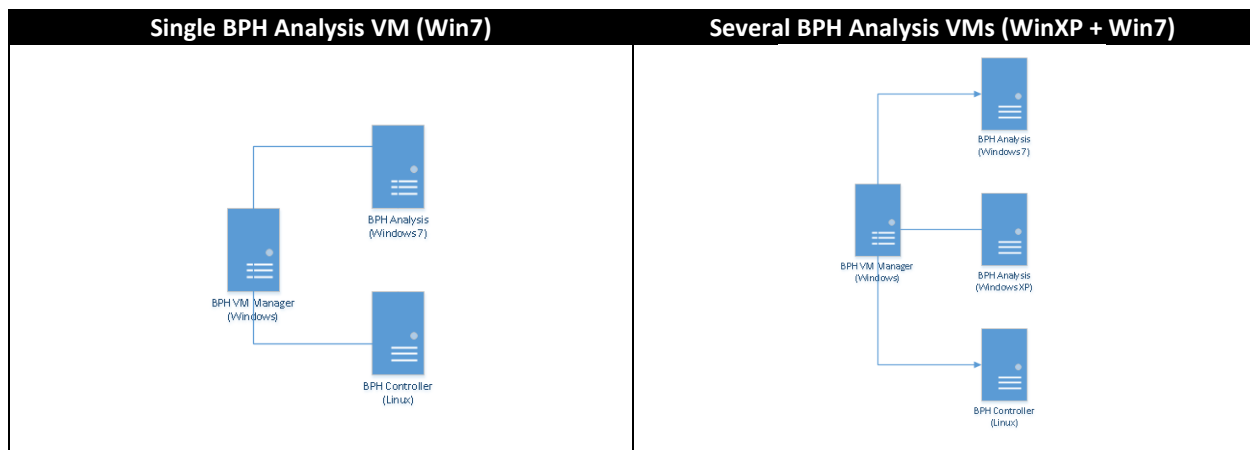
Figure 1 – Virtual Machine Infrastructure

Virtual Machine Deployment Scenarios

Single BPH Analysis VM (Win7)	Several BPH Analysis VMs (WinXP + Win7)

BPH Controller (Linux Guest)

Description: A virtual machine generates execution (JSON) templates from the instructions contained in a BPH Script, which are consumed by the BPH Analysis virtual machine network agent and receives the generated tool's data

Component	Language/Format	Description
Web Manager Server	PHP	Collects data and files generated by the Network agent
Template Server	Python	Generates and sends JSON execution data to the Network Agent
BPH Script	Python	Python scripts containing the instructions executed by the network agent remotely
BPH Analysis Module	Python	Python scripts that parse tool's content
BPH Plug-in	JSON	JSON files containing execution data for commands and tools used by the BPH scripts

BPH Analysis (Windows Guest)

Description: A virtual machine runs a Python script which is in charge of receiving tool execution instructions from the BPH Controller machine through the network

Component	Language/Format	Description
Network Agent	Python	Receives executes JSON execution data and reports results back to the C&C template server

BPH VM Manager (Windows Host)

Description: A host machine that runs a Python script which controls virtual machine operations, such as start, stop, and restore snapshot actions received through the network.

Component	Language/Format	Description
Virtual Machine Server	Python	A network server that manages the Virtual Machines used by the framework

Deployment Scenario

- **BPH Controller VM**
  - Operating System:
    - Ubuntu 18.04.2 Desktop (amd64)
    - Username (BPH USER): bph
  - Software:
    - Iptables, dnsmasq, tor, Apache2 + PHP (with ZipArchive support)
    - BurpSuite Free version
  - Network Configuration:
    - VirtualBox Interface 1: (enp0s3)
      - IP address: 192.168.56.90/24 - Static
      - Mode: Host-Only
    - VirtualBox Interface 2: (enp0s8)
      - IP Address: DHCP - Dynamic
      - Mode: NAT
  - BLACKPHENIX:
    - BPH Controller IP: 192.168.56.90
    - BPH Controller Web Port: 8194
    - BPH Web folder: bph
    - BPH Virtual Manager IP: 192.168.56.1
    - BPH Virtual Manager Port: 8003
- **BPH Analysis VM**
  - Operating System:
    - Windows 7 x64
  - Software:
    - Python 2.7.13

- o Network Configuration:
  - ▪ VirtualBox Interface 1:
    - • IP address: 192.168.56.91/24 - Static
    - • Mode: Host-Only
  - ▪ VirtualBox Interface 2:
    - • IP Address: DHCP - Dynamic
    - • Mode: NAT
  - ▪ Drive (BPH Tools):
    - • H:\
- o BLACKPHENIX:
  - ▪ BPH Controller IP: 192.168.56.90
  - ▪ BPH Controller Web Port: 8194
  - ▪ BPH Template Server Port: 8002
- **BPH Virtual Machine Manager (Host)**
  - o Operating System:
    - ▪ Windows 7 x64
  - o Software:
    - ▪ Virtual Box 6.0.8 r130520 (Qt5.6.2)
    - ▪ Python 3.7.3
  - o Network Configuration:
    - ▪ VirtualBox Host-Only Ethernet Adapter:
      - • IP address: 192.168.56.1/24 - Static
      - • DHCP Server: Disabled
  - o BLACKPHENIX:
    - ▪ BPH Virtual Machine Manager Port: 8002

Installing & Configuring (Steps)

BPH Controller – Ubuntu Guest VM

Software Requirements

- Python 3.6.8

OS Installation & User Creation

1. Create a new Virtual Machine (Ubuntu x64)
   - 15 GB HDD
   - 8GB RAM
2. Install Ubuntu (18.04.2) Guest Operating System
3. Create a system user named "**bph**"
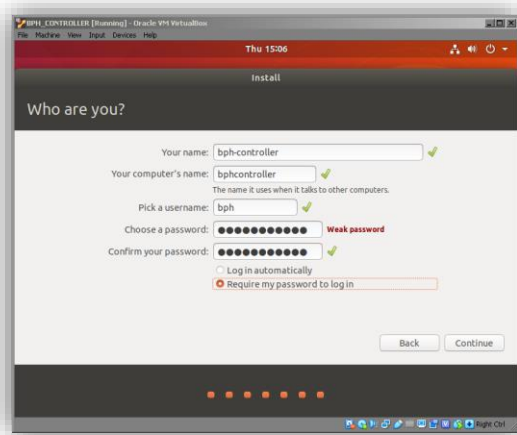
4. Conclude with the installation and turn off the machine

VirtualBox Network Interface Configuration

BLACKPHENIX uses two network interfaces to allow guest VM to communicate to the "real" Internet (NAT) and "emulated" Internet (Host-Only). See "**Emulated Internet**" section for more information.

The following pictures show the adapter configuration for each network interface:

Adapter 1 (Host-Only Adapter)	Adapter 2 (NAT)
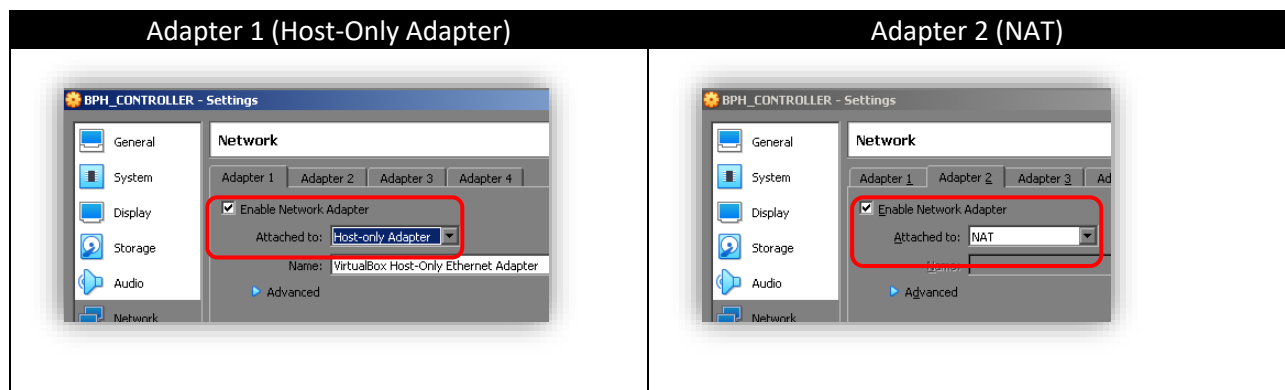
Once the Network adapters are properly configured as indicated, and then assign a static IP address to the Host-only interface. For the test deployment scenario used throughout the document, the IP address 192.168.56.90 will be used.

The following picture shows how the two network interfaces (enp0s3, enp0s8) must be configured as well as the default gateway that was automatically configured by the VirtualBox DHCP server.

Interface enp0s3 (Host-Only / Static IP address)	Interface enp0s8 (NAT / Dynamic IP address)

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.90  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe21:66cd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:21:66:cd  txqueuelen 1000  (Ethernet)
        RX packets 180  bytes 28138 (28.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 464  bytes 65738 (65.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.15  netmask 255.255.255.0  broadcast 10.0.3.255
        inet6 fe80::596:9665:afdd:334c  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:b1:d1:3e  txqueuelen 1000  (Ethernet)
        RX packets 763  bytes 514322 (514.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 641  bytes 126253 (126.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
bph@bphcontroller:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.3.2        0.0.0.0         UG    101    0        0 enp0s8
10.0.3.0        0.0.0.0         255.255.255.0   U     101    0        0 enp0s8
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s8
192.168.56.0    0.0.0.0         255.255.255.0   U     102    0        0 enp0s3
bph@bphcontroller:~$
```

NOTE: After setting up network configuration, make sure BPH Controller machine is able to reach the Internet since will be required to download additional packages.

Installation Steps

The following steps were tested assuming the configuration (recommended) used in the deployment scenario section. Choose the installation type of your election.

Automatic Installation

```
$ cd ~ && git clone <bph_github_url> && cd ~/bph-framework/

# bph_user = bph (Current Linux user)
# bph_controller_ip = 192.168.56.90
# bph_controller_web_port = 8194
# bph_controller_web_folder = bph

$ sudo python3 ./bph_install.py bph 192.168.56.90 8194 bph
```

Manual Installation

```
$ sudo apt-get install net-tools vim git python3 python3-pip default-jre -y
$ sudo systemctl disable systemd-resolved && sudo systemctl stop systemd-resolved
$ sudo rm -rf /etc/resolv.conf
$ sudo service network-manager restart
$ sudo sed -i -e 's/nameserver .*/nameserver 4.2.2.2/' /etc/resolv.conf
$ sudo sed -i "3i dns=none" /etc/NetworkManager/NetworkManager.conf
$ sudo service network-manager restart
$ sudo apt-get install dnsmasq apache2 libapache2-mod-php php-zip tor -y
$ sudo sed -i 's/APACHE_RUN_USER=www-data/APACHE_RUN_USER=bph/' /etc/apache2/envvars
$ sudo sed -i 's/APACHE_RUN_GROUP=www-data/APACHE_RUN_GROUP=bph/' /etc/apache2/envvars
$ sudo sed -i 's/Listen 80/Listen 8194/' /etc/apache2/ports.conf

$ cd /var/www/html && sudo mkdir bph && sudo chown -R bph:bph bph/ && cd bph && echo "<?php
phpinfo(); ?>" > test.php

$ cd ~ && git clone <bph_github_url> && cd ~/bph-framework/
```

```
bph@bphcontroller:~/bph-framework$ pwd
/home/bph/bph-framework
bph@bphcontroller:~/bph-framework$ ls
agent  auxiliary  bph  conf  LICENSE.txt  misc  plugins  README.md  requirements.txt  scripts  session
bph@bphcontroller:~/bph-framework$
```

```
$ cd /var/www/html/bph && ln -s ~/bph-framework/session/ session && ln -s ~/bph-
framework/plugins/ plugins && ln -s ~/bph-framework/agent/ agent
```

```
$ sed -i "4i export PYTHONPATH=\"/home/bph/bph-framework\"" ~/.bashrc
$ cd ~/bph-framework/ && sudo pip3 install -r ~/bph-framework/requirements.txt
$ cp ~/bph-framework/auxiliary/web/report.php /var/www/html/bph/
$ sudo sed -i 's/#SOCKSPort 192.168.0.1:9100/SOCKSPort 192.168.56.90:9050/' /etc/tor/torrc
$ cd ~/bph-framework/auxiliary/proxy/burp && sh download_burp.sh
```


Traffic Redirection & Emulation Activation

```
$ cd ~/bph-framework/auxiliary/network && sh activate.sh enp0s8 enp0s3
```


HTTP(S) + TOR Proxy Service Execution

```
$ cd ~/bph-framework/auxiliary/proxy/ && sed -i 's/123.123.123.123/192.168.56.90/'
tor_template.json && mv tor_template.json tor.json

$ cd ~/bph-framework/auxiliary/proxy/ && sudo ./proxy.sh
```

NOTE: The first time that Burp is executed, will require the user to accept the license agreement. The second time won't be required.

All connections received by the HTTP(S) proxy server, will be automatically redirected to the TOR service running on port 9050 (default TOR service port).

Testing Network Services Operation

After installation, all network services must be operating without any issue. To test its correct operation, all the network services are restarted.

```
$ sudo service dnsmasq restart
$ sudo service tor restart
$ sudo service apache2 restart

$ sudo netstat -apnt | grep -E '(apache2|dnsmasq|tor|java)'
```


Finally, a quick DNS test is executed to ensure correct resolving operation

```
$ dig +short google.com @192.168.56.90
```


Testing BPH Web Access

http://192.168.56.90:8194/bph/test.php

http://192.168.56.90:8194/bph/
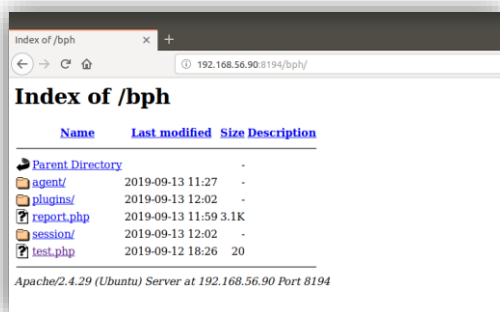
NOTE: The tests must be completed as shown above; otherwise the BPH Controller will present anomalous behavior or won't run at all

Framework Configuration – Part 1

The first step before running a **BPH Script** is to configure the framework is renaming the configuration file located in the *conf* folder from "***blackphenix_template.conf***" to "***blackphenix.conf***". The configuration used is based on the test deployment scenario.

Web Controller Section

```
# BLACKPHENIX TEMPLATE CONFIGURATION FILE

[WEB_CONTROLLER]
#
# WebControllerIp = 192.168.80.2
# WebControllerPort = 8194
# WebControllerPath = bph
#
WebControllerIp = 192.168.56.90
WebControllerPort = 8194
WebControllerPath = bph
```

Windows Agent Section

The "RemoteToolsDrive" value is the drive where all the Windows tools and scripts will be located in the BPH Analysis machine. See "BPH Analysis Tools Update" section for more information.

```
[WINDOWS_AGENT]
#
# RemoteToolsDrive = E:\
#
RemoteToolsDrive = H:\
```

Template Server Section

Template server is the network component that allows interaction between the BPH Analysis guest network agent and the BPH controller. The test deployment scenario will run the Template Server on the same machine, then "TemplateServerIP" value will be set to the BPH Controller IP address. The "TemplateServerPort" value is set to 8002. It is up to the user to select any port. This port must match with the port when running the network agent (agent.py) on the BPH Analysis machine.

The "TemplateServerOutput" value is set to "False". When set to "True", the JSON execution data will be shown in the console when a BPH Script is executed. This option can be used mostly for troubleshooting purposes.

```
[TEMPLATE_SERVER]
#
# TemplateServerIp = 192.168.80.2
# TemplateServerPort = 8002
# TemplateServerBufferSize = 8192
# TemplateServerOutput = True
#
TemplateServerIp = 192.168.56.90
TemplateServerPort = 8002
TemplateServerBufferSize = 8192
TemplateServerOutput = False
```

BLACKPHENIX can control the VirtualBox actions executed by the Host, such as starting and restoring virtual machines. This can be done by the Virtual Machine Manager server (vm_manager.py) which will be running on the host machine. In this configuration section, the IP address of the host machine and the port is where the manager will be listening.
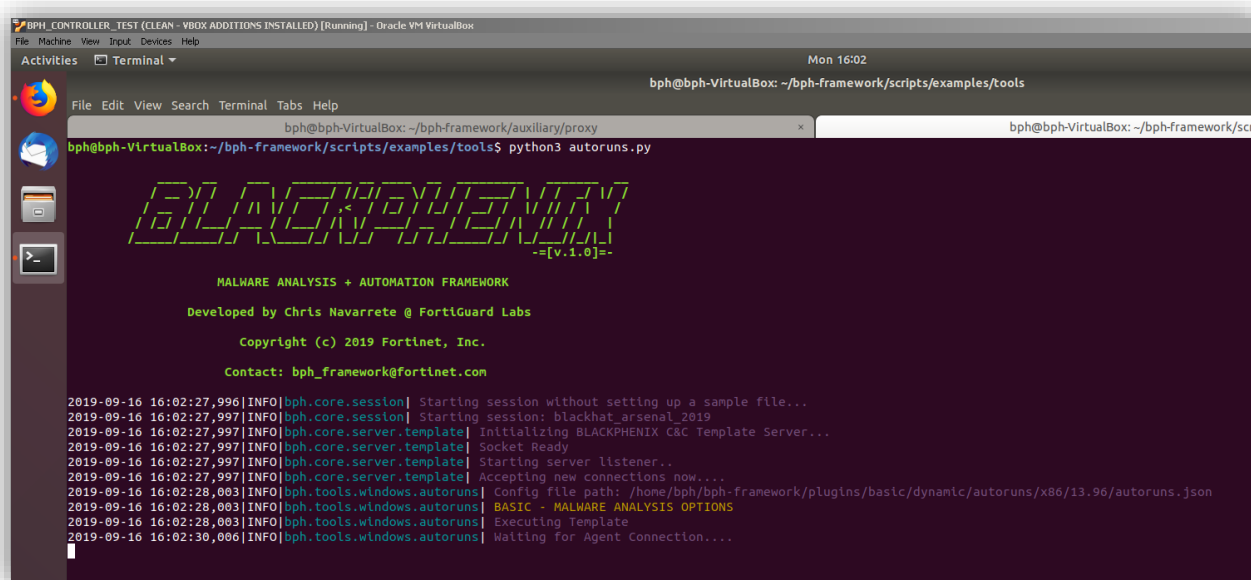
Note: The IP address should be the one used by the "Ethernet adapter VirtualBox Host-Only Network" adapter.

```
[VIRTUALBOX_SERVER]
#
# VirtualBoxServerIp = 192.168.80.1
# VirtualBoxServerPort = 8990
#
VirtualBoxServerIp = 192.168.56.1
VirtualBoxServerPort = 8003
```

In "Framework Configuration – Part 2" will configure the VM Snapshot that will be used for executing malware analysis.

BPH Controller – Script Execution Test

```
$ cd ~/bph-framework/scripts/examples/tools/ && python3 autoruns.py
```


If "Waiting for Agent Connection" is displayed, that means that BPH Controller is waiting for a connection coming from BPH Analysis machine. The script can be safely interrupted.

Windows Tool Repository

BLACKPHENIX has bundled the majority of the tools and the ones required by the analyst should be downloaded and copied into its BPH folder location for proper functioning.

Each tool has its own location. For instance, "PEiD v0.95" tool should be downloaded and moved to its corresponding folder, which in this case the location would be *"/home/<user>/bph-framework/tools/basic/static/peid/x86/0.95/"* respectively.

A BPH Analysis VM can use a script called "update.bat" to fetch tools from BPH Controller through the Web (Apache's "bph" web folder) automatically.

BPH Analysis – Guest VM

Operating System Requirements
- UAC disabled
- VC redist/.NET Framework

Software Requirements
- Python 2.7.13

VirtualBox Interface Configuration
BPH Analysis machine (Windows guest) requires two network interfaces to work. Adapter 1 (Host-Only) will act as a gateway between the BPH Analysis VM and an "emulated" internet, and Adapter 2 (NAT) in case any direct-Internet access is required by the user.

<p style="text-align:center"><strong style="color:red">IMPORTANT NOTE:</p>

Due to the nature of the risks involved while analyzing malware, an analyst must ensure that production machines are NOT located on the same network segment as the BPH Virtual Machine Manager.

Network Configuration
The network interfaces configuration should look as follows:

1. **Adapter 1 => Host-Only**
    a. IP Address: 192.168.56.91 - Static
    b. Netmask: 255.255.255.0
    c. Gateway: 192.168.56.90
    d. DNS: 192.168.56.90
2. **Adapter 2 => NAT**
    a. DHCP Enabled

Emulated Internet
Emulated internet means that the "Host-Only" special configuration, allows guest VM (BPH Analysis) resolve internet hosts (IP addresses and domain names) but direct connections to the host's IP addresses will be redirected transparently to the BPH Controller machine.

The following screen shows how from BPH Analysis machine attempts a telnet connection to a public IP address, and BPH Controller (Fake Telnet server) receives such connection.

BPH Analysis connection attempt	BPH Controller connection reception
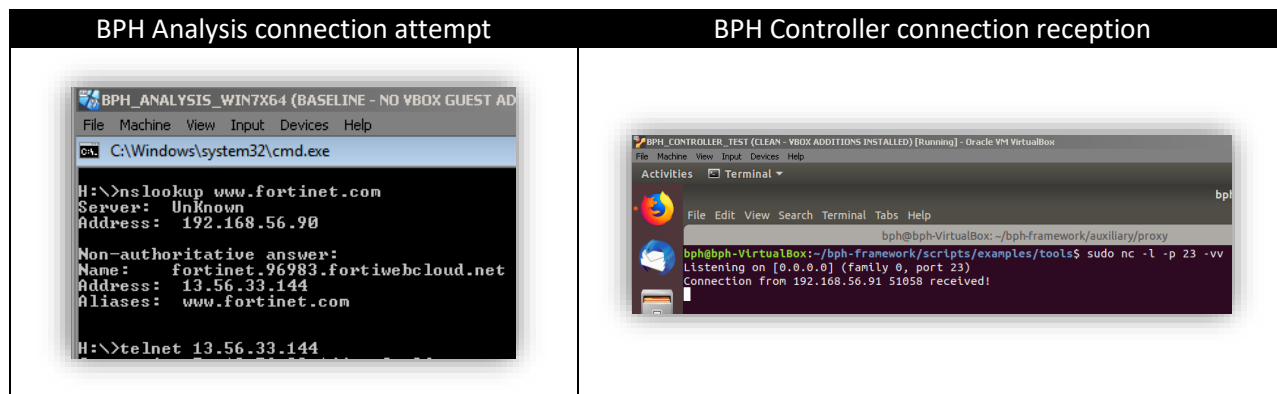
HTTP(S)/TOR Traffic Redirection

The current support for traffic redirection is for HTTP/HTTPS that redirects traffic through a TOR channel. This happens when a malware attempt to connect to an HTTP(S) server, the request is handled by the BPH Controller (iptables) and forwarded to a user's selected port (Burp Suite) server, which forward it to the TOR.
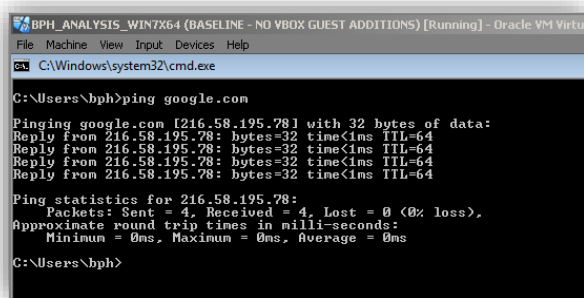
Any manual redirection should be modifying the "PORT REDIRECTION" section located on the **"~/bph-framework/auxiliary/network/activate.sh**" script in the BPH Controller VM.

Testing Network Connectivity

Make sure the BPH Analysis machine can reach BPH Controller and emulated Internet by executing a "ping" to the controller's IP address and to a domain (google.com).

NOTE: The reply will be valid, but even that the domain resolves to its original IP address, all connection attempts will be automatically redirected to the BPH Controller machine.

Security Certificate Installation

BLACKPHENIX uses Burp Suite to establish HTTP(S) connections. In order to support transparent internet access to HTTP(S) servers, the BPH Analysis machine must install Burp's security certificate first. The installation steps are shown as follows:

1. Open the URL: http://192.168.56.90:8080
2. Click on the "CA Certificate" orange link and save the "cacert.der" file
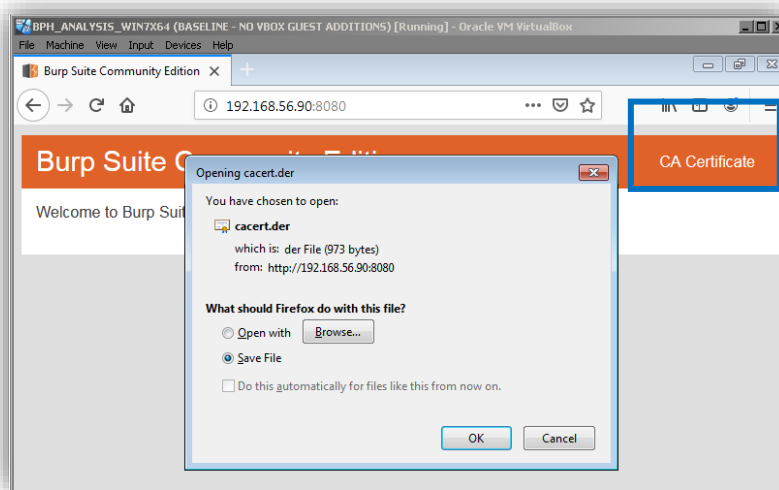
3. Right-click on the "cacert.der" file, and select "Install Certificate:
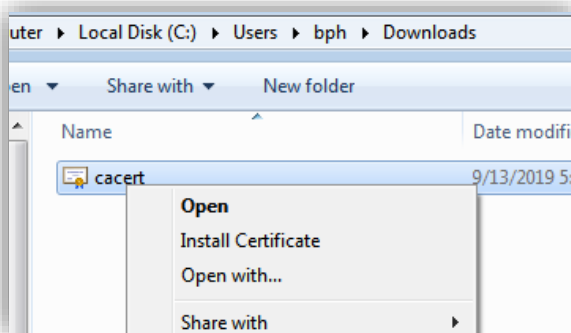
4. Select "Place all certificates in the following store" => "Trusted Root Certification Authorities"
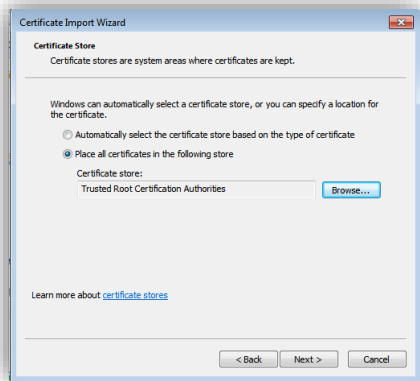
5. Click on "Yes" in the Security Warning window
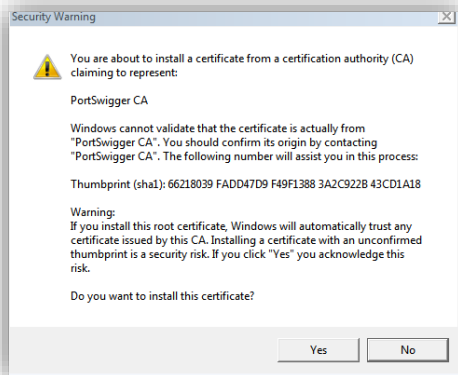
6. Restart the Virtual Machine
7. After reboot, browse any website with SSL enabled and a security certificate warning should not be displayed on the screen
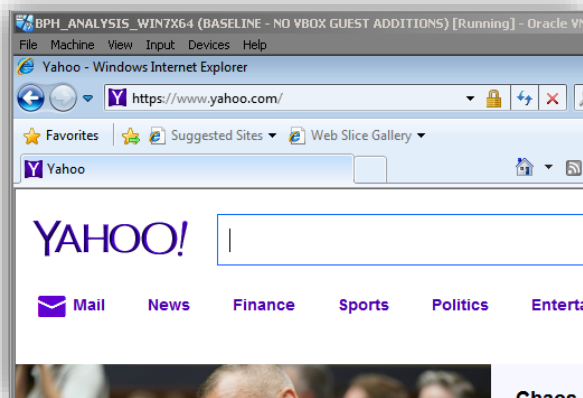
BPH Agent Download

In order to receive commands and submit data from/to the BPH Controller, a network agent (agent.py) must be running on the BPH Analysis machine.

The download and execution instructions are as follows:

1. Open the following URL: http://192.168.56.90:8194/bph/agent/windows/
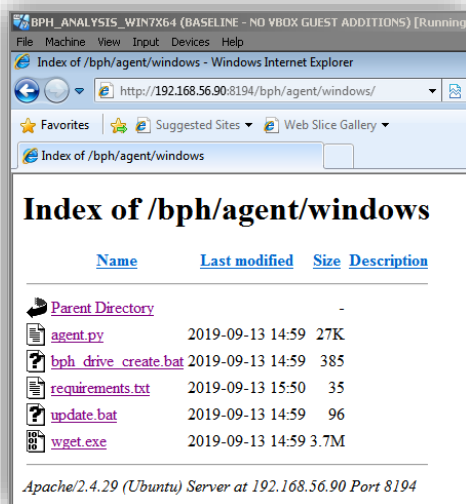
2. Download all files and put them in a temporary folder

BPH Analysis Tools Drive Creation

3. Following the previous steps, double-click on "bph_drive_create.bat" and will create a new drive with the letter "H:\" assigned to it
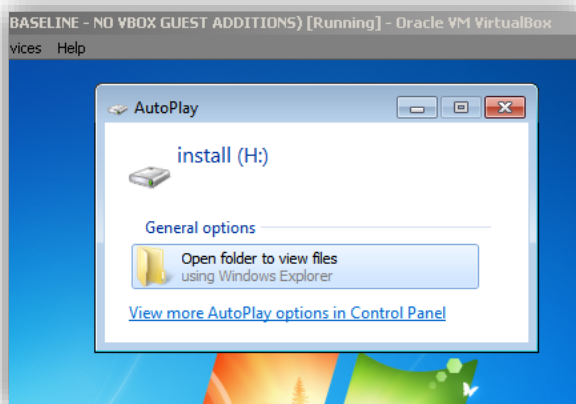
4. Move all the files saved to the temporary location to the new H:\ drive

5. Install Python 2.7.13 and install the script dependencies by running pip (pip install –r H:\requirements.txt)

BPH Analysis Tools Update

6. Edit "update.bat" URL to point to the IP address (192.168.56.90), Web Port (8194) and Web path (bph) of BPH Controller.
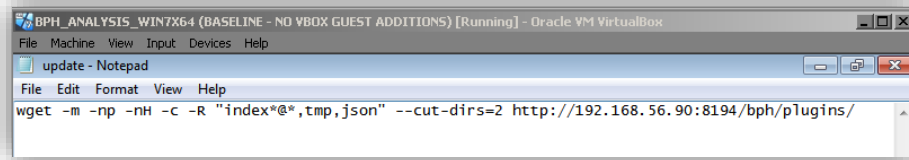
7. Execute "*update.bat*" batch script to download all the tools from the BPH Controller machine
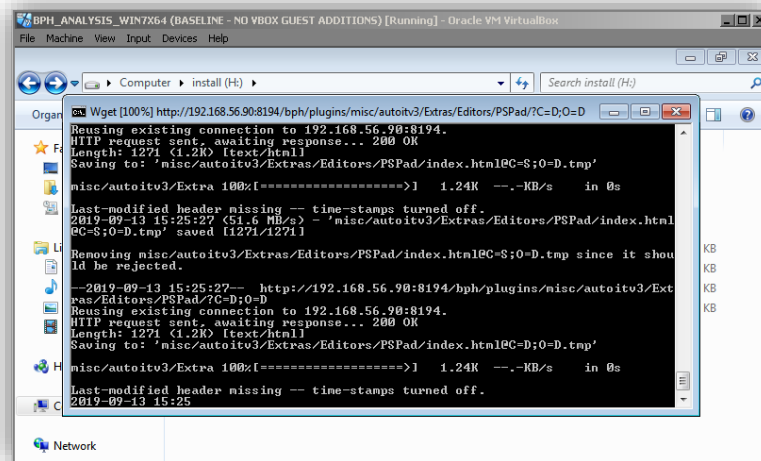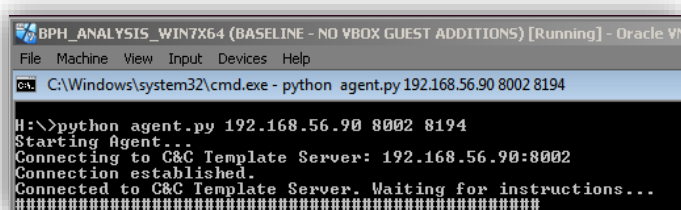
8. Finally, all the tools will be downloaded on H:\
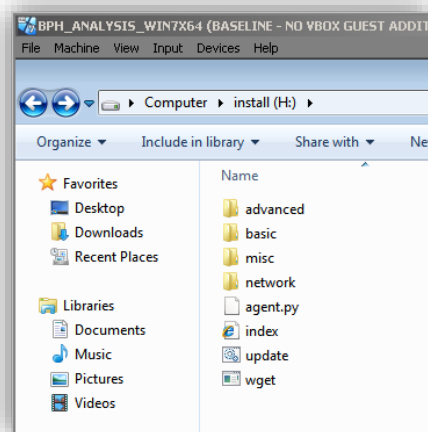
BPH Agent – Agent Execution Test
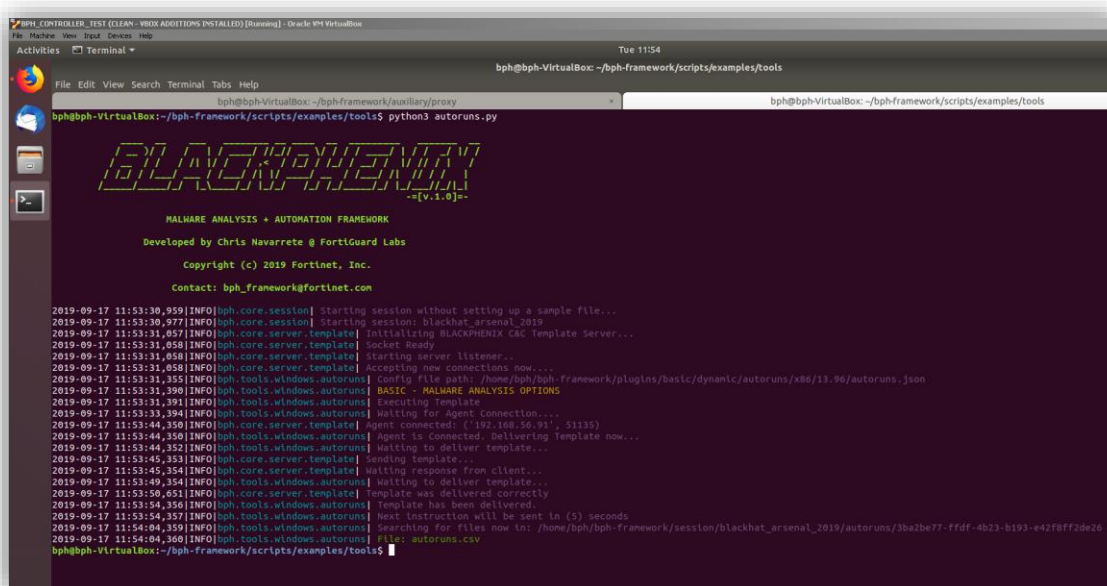
Before starting the network agent, make sure the "autoruns" BPH Script (/home/bph/bph-framework/scripts/examples/tools/autoruns.py) is executed and is waiting for agent connection.

Now, run the network agent script as follows: **python agent.py 192.168.56.90 8002 8194**

After execution, in BPH Controller will show the interaction between the BPH Analysis machine and the controller and as result of the execution, the file "autoruns.csv" was received successfully.

NOTE: The first time autoruns tool executes, an accept warning will be displayed.

At this point in time, communication between BPH Analysis and BPH Controller machines is working as expected. Now, the next step is to generate a virtual machine snapshot of the BPH Analysis machine that will be used for automated malware analysis.

BPH Analysis VM - Snapshot

If the "BPH Agent – Agent Execution Test" passed successfully, then a snapshot must be taken. This snapshot will be restored every time a new malware sample will be analyzed.

The steps to take the snapshot are described as follows:

1. Execute the agent.py script along with its required arguments (BPH Controller IP address, Template Server port, BPH Web port) once again. The agent will attempt to connect back to the BPH Controller (port 8002) several times until a BPH Script gets executed on the controller side.
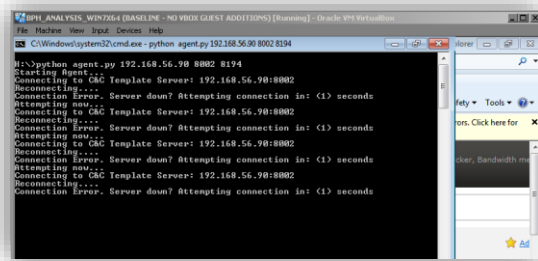
2. While the agent is running, in VirtualBox, go to "Machine"=>"Take Snapshot", and provide a name for it. In this case, the name was "BASELINE".
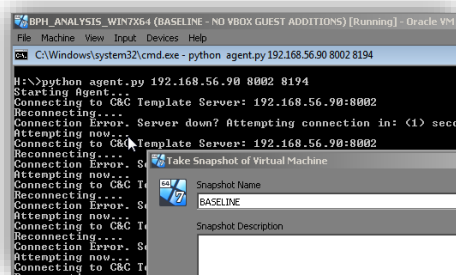
3. In Windows console, type: "**cd C:\Program Files\Oracle\VirtualBox && VBoxManage.exe showvminfo BPH_ANALYSIS_WIN7X64 | findstr "BASELINE"**". The line ending in "*" is the latest snapshot taken.
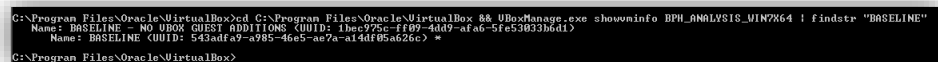
4. Take note of the Snapshot's UUID that will be used in the second part of the Framework's configuration shortly.

Framework Configuration – Part 2

The last step for BPH configuration is to set the snapshot(s) that will be used to execute malware samples on it.

Virtual Machines Section

In "VIRTUALMACHINES" section four virtual machines can be configured and used. Each virtual machine profile is supposed to be used for the specific analysis profile, which can be used for Basic Static, Basic Dynamic, Advanced Dynamic or Advanced Static malware analysis and each virtual machine configuration has three values to be updated.

The "alias", which is how BLACKPHENIX will identify a virtual machine, the "MachineId" which is the name assigned to the virtual machine in VirtualBox, "SnapshotId" which is the UUID captured in the previous step, and finally "NetworkConnection", which is the network interface that will be used by the virtual machine for outgoing connections. For this particular case, the "Basic Static Analysis" profile was selected with interface number one (1), which is the "TOR network". If the value is set to two (2), then the outgoing connections will go through the NAT interface (open Internet access).

```
[VIRTUALMACHINES]
#
# BasicStaticAnalysisBphVmAlias = basic_static
# BasicStaticAnalysisMachineId = BPH_ANALYSIS_WINXPSP3
# BasicStaticAnalysisSnapshotId = c8d81fc1-9866-41eb-8f0d-14c8b3516ffd
# BasicStaticAnalysisNetworkConnection = 1
#

BasicStaticAnalysisBphVmAlias = basic_static
BasicStaticAnalysisMachineId = BPH_ANALYSIS_WIN7X64
BasicStaticAnalysisSnapshotId = 543adfa9-a985-46e5-ae7a-a14df05a626c
BasicStaticAnalysisNetworkConnection = 1
```

This snapshot will be restored by the VM Manager per user (BPH Script instruction) request.

BPH VM Manager – Host

Software Requirements

- Python 3.7.3

1. Copy the VM manager script (*/home/bph/bph-framework/auxiliary/server/vm_manager.py*) to the BPH VM Manager host, open a command-line window and run the script:

   ```
   C:\bph\python vm_manager.py 192.168.56.1 8003
   ```

2. As a result, the script will indicate that the server has started and is awaiting connections.

```
C:\bph>python vm_manager.py 192.168.56.1 8003
       -=[B L A C K P H E N I X]=-
    by Chris Navarrete @ FortiGuard Labs

      [VirtualMachine Server Manager]

[BLACKPHENIX] ¦ Starting VM Control server...
[BLACKPHENIX] ¦ Accepting connections
```

BLACKPHENIX BPH VM CONTROL TEST
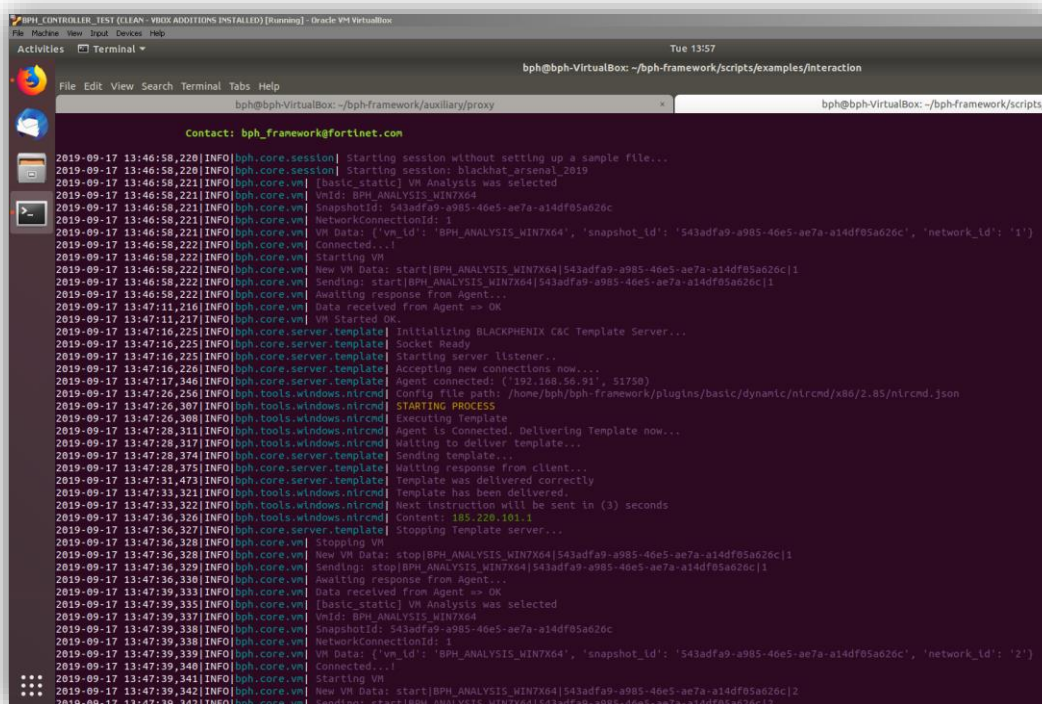
BPH Script – Python + TOR Execution Testing

To make sure that virtual machine management and networking is operating as expected, and then the Python/TOR BPH Script can help with that. The script is located on **"~/bph-framework/scripts/examples/interaction/python_and_tor.py"** and the purpose of it is to test how Python can be executed inside the virtual machine and also how within the same script, the virtual machine network configuration can be modified to provide different network access, through TOR network (Host-Only) and open-access Internet (NAT).

The script instructions indicate BPH Analysis machine to execute a Python on-liner to fetch an Internet resource that gets back a public IP address going through the network interface 1 (Host-Only/TOR), showing as result "185.220.101.1" which is a TOR IP address.
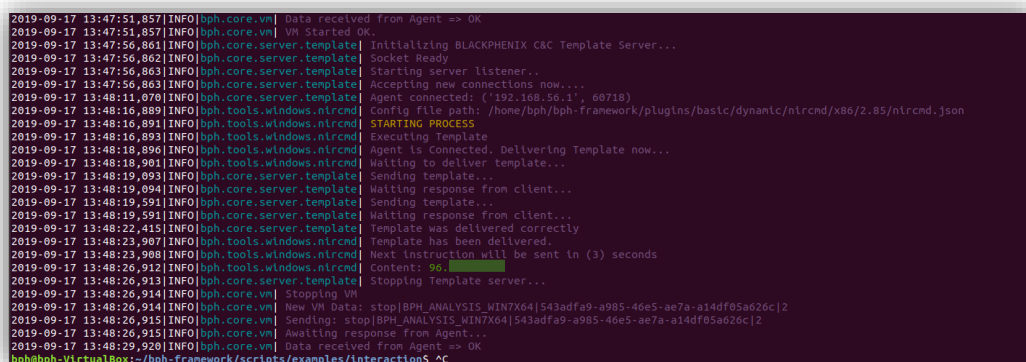
NOTE: If the TOR IP is not shown, increase the time delay (nircmd.execute(delay=3)) from 3 to 5 or a higher number. A delay set to 3 works in most cases.

The script also instructs BPH Analysis machine to repeat the same process but switching the network interface to 2 (NAT network), showing as result the router's public IP address.

BPH Script – UPX Sample Execution Test

The following test consists in executing the **UPX BPH Script** that:

1. Sets a UPX'd target sample (/home/bph/bph-framework/misc/bph_upx.exe)
2. Submits the sample to the BPH Analysis virtual machine:

```
$ cd ~/bph-framework/scripts/examples/tools/ && python3 upx.py ~/bph-
framework/misc/bph_upx.exe
```

3. Receive the tool's (output) log file as well as its unpacked executable file

The successful execution of the previous final tests fulfills the requirements to fully operate and is ready to go.

Enjoy!!

Running Example BPH Scripts

BPH Framework includes several working examples that show how each supported tool works. These samples are the ones showcased at BlackHat Arsenal 2019 as part of the tool's release.

All files can be found under "*/home/bph/bph-framework/scripts/examples/*" folder.

BPH Script Development & Tool Customization

Please, refer to the PDF files located in the "docs" folder.

Contact

For general information regarding the framework, please use the following email

- bph_framework@fortinet.com