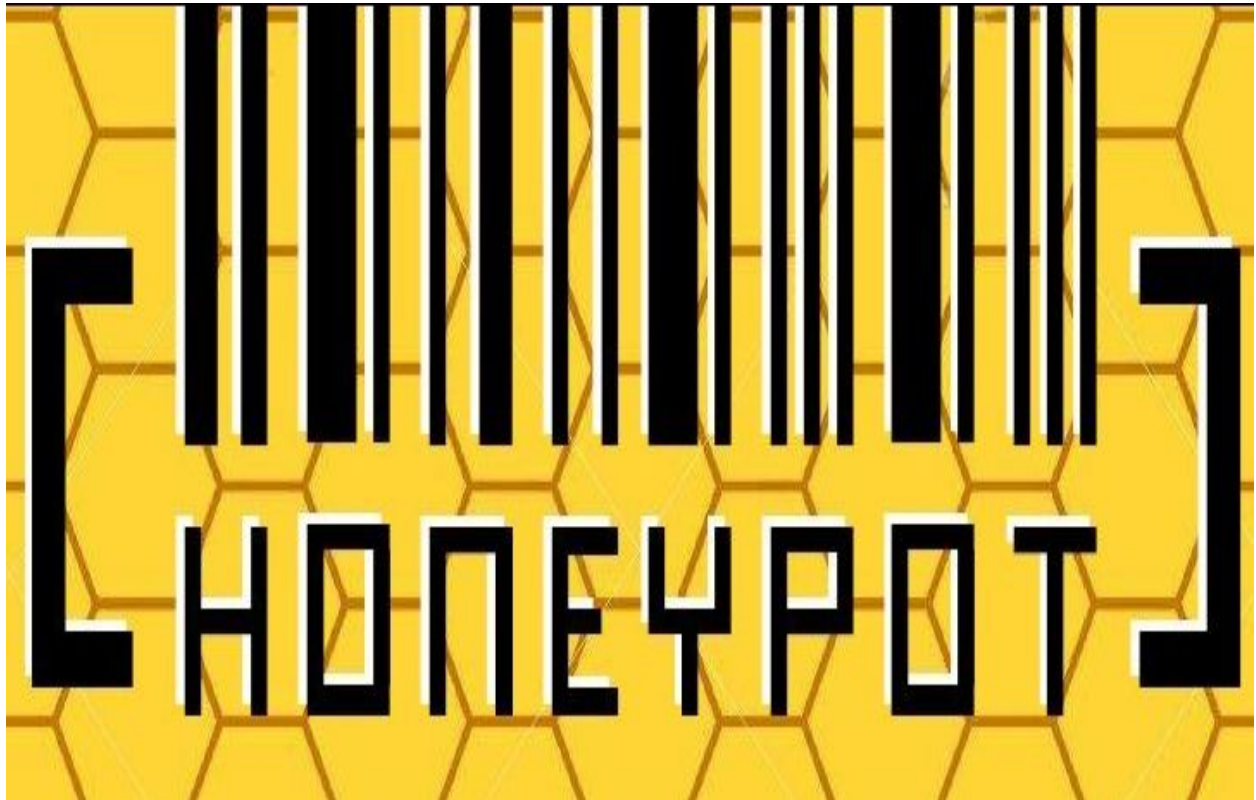


PROYECTO TRANSVERSAL

Conecta Empleo - Fundación Telefónica



IMPLEMENTACIÓN DE UNA RED DE 5 HONEYPOTS

Mentores:

Carolina Ramos
Nicolás Castellano

Realizado por:

Marc Vila
Nilo Challco
Toni Ariso

DICIEMBRE 2019



1. INTRODUCCIÓN	2
2. INSTALACIÓN	5
2.1 MHN SERVER	6
2.2 SENSORES / HONEYPOTS	15
2.2.1 Snort	15
2.2.2 Glastopf	18
2.2.3 Cowrie	20
2.2.4 p0f	23
2.2.5 Dionaea	25
2.2.6 HoneyDB Agent	27
2.2.7 Otros honeypots	33
3. ESTADÍSTICAS	35
3.1 MHN	35
3.2 HONEYDB	41
4. BIBLIOGRAFÍA	44
5. SCRIPTS	46
5.1 Snort Installation Script	46
5.2 Glastopf Installation Script	49
5.3 Cowrie Installation Script	53
5.4 p0f Installation Script	55
5.5 Dionaea Installation Script	57



1. INTRODUCCIÓN

¿Qué es un honeypot?

Un honeypot es una trampa que un profesional de TI pone para un hacker malicioso, con la esperanza de que interactúen con éste de una manera que proporcione inteligencia útil. Es una de las medidas de seguridad más antiguas en TI, pero tenga cuidado: atraer a los hackers a su red, incluso en un sistema aislado, puede ser un juego peligroso.

Se configurará deliberadamente con vulnerabilidades conocidas para ser un objetivo más tentador u obvio para los atacantes. Un honeypot no contendrá datos de producción ni participará en tráfico legítimo en su red; así es como puede darse cuenta de que cualquier cosa que ocurra dentro de él es el resultado de un ataque.

Historia

Una de las primeras historias de alto perfil de infosec se refería a lo que casi con toda seguridad es el primer uso de un honeypot. Como se detalla en su libro, *The Cuckoo's Egg*, en 1986 el administrador de sistemas de la Universidad de California en Berkeley, Clifford Stoll, trató de localizar un cargo aparentemente erróneo de \$0.75 por el uso de un sistema Unix en Lawrence Berkeley Lab; en el proceso, descubrió que alguien estaba marcando en el sistema y había logrado obtener acceso de superusuario. Stoll implementó dos defensas tipo honeypot para localizar al hacker: conectó terminales prestadas a las cincuenta líneas telefónicas entrantes durante un largo fin de semana y esperó a que el hacker llamara; una vez que se dio cuenta de que el hacker estaba buscando información sobre secretos de defensa nuclear, creó un departamento totalmente ficticio en LBL que supuestamente estaba trabajando en el sistema de defensa de misiles "Star Wars" para atraer al hacker a pasar tiempo allí. Finalmente, el atacante fue arrestado y se descubrió que era un alemán occidental que trabajaba para la KGB.



Otro incidente importante en los comienzos de los honeypot se produjo en 1990, cuando un hacker intentó entrar en los laboratorios Bell Labs de AT&T y robar su archivo de contraseñas. El pionero de Internet Bill Cheswick, que trabajaba para Bell Labs en ese momento, guió al atacante en lo que él llamó "una persecución alegre" a través de algunos sistemas de honeypot ad-hoc para rastrear su ubicación y aprender sus técnicas; su artículo sobre el incidente, "An Evening with Berferd" (Una noche con Berferd), fue extremadamente influyente.

Clasificación según el fin:

- Research: es usado por desarrolladores, administradores de sistema, y blue team managers que trabajan en universidades, escuelas, asociaciones o centros de investigación.
- Production: es usado tanto por instituciones públicas o privadas para analizar el comportamiento y técnicas de los hackers en la red.

Clasificación según la interactividad:

- Pure: un servidor físico configurado especialmente para atraer ataques y mostrarse como un objetivo real para hackers.
- High-interaction: se usan máquinas virtuales con el mayor número de servicios expuestos.
- Low-interaction: se usan máquinas virtuales con unas determinadas vulnerabilidades.




Clasificación de honeypots populares:

- SSH:
 - Kippo
 - Cowrie
- HTTP:
 - Glastopf
 - Nodepot
 - Google Hack HoneyPot
- Wordpress:
 - Formidable HoneyPot
 - Blackhole for Bad Bots
 - Wordpot
- Database:
 - ElasticHoney
 - HoneyMysql
 - MongoDB-HoneyProxy
- Email:
 - Honeymail
 - Mailoney
 - SpamHAT
- IoT:
 - HoneyThing
 - Kako



2 INSTALACIÓN

PANEL DE CONTROL

















HONEYPOT

DEFAULT

Class project / Educational purposes

[Resources](#) [Activity](#) [Settings](#)

DROPLETS (7)

  dionaea-honeypot	159.65.150.186	bangalore
  honeydb-server	159.203.8.93	toronto
  pof-honeypot	167.172.191.81	frankfurt
  cowrie-honeypot	206.189.43.28	singapore
  glastopf-honeypot	167.172.221.117	sanfrancisco
  snort-honeypot	167.99.3.150	newyork
  mhn-server	165.22.120.182	london



2.1 MHN SERVER

Decidimos montar la red honeypot en un servicio Cloud para que pudiera captar de forma global las amenazas que surfean Internet y que concienciarnos sobre la tendencias a nivel mundial de una serie de ataques.

De entre todos los proveedores de Cloud Computing Services:

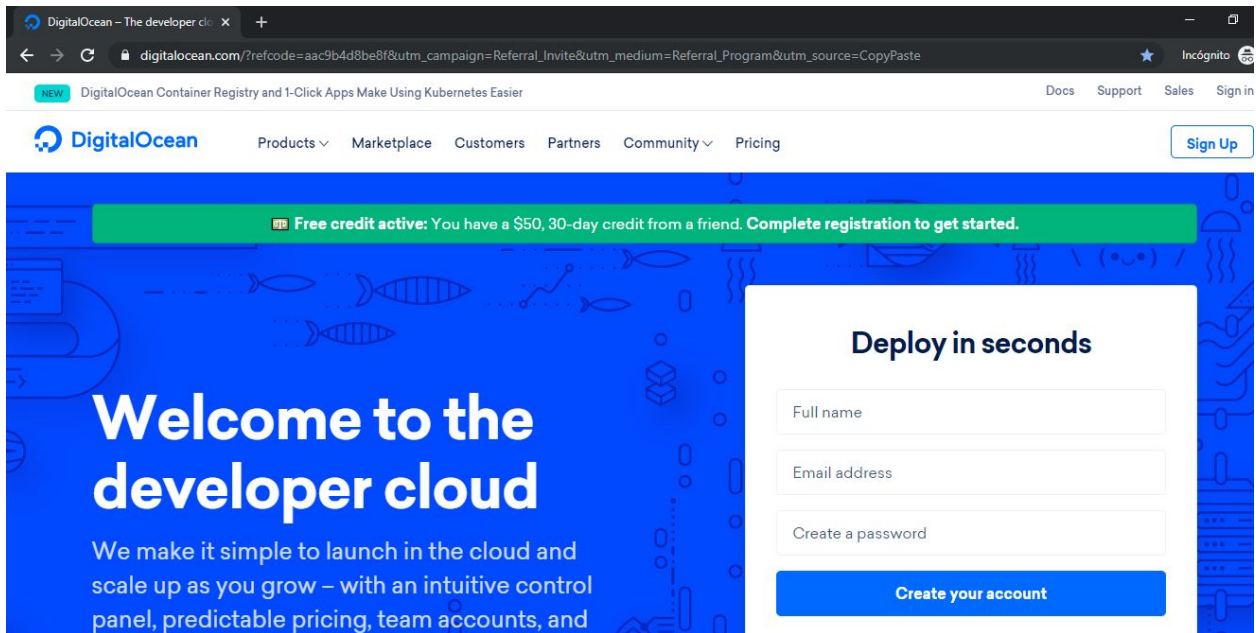
- Amazon Web Services
- Microsoft Azure
- Google Cloud
- IBM Cloud
- Oracle Cloud
- Digital Ocean

Nos decidimos por DigitalOcean por referencias de portales especializados y que recomendaban la utilización de este proveedor para la implementación de honeypots. Por ejemplo, tiene la opción predefinida de la instalación de HoneyDB que es líder en el campo de la recolección de datos agregados en comunidad.

Además ofrecía un cupón de 50 Us Dollars de crédito para un mes, lo que nos permitía sacarle el máximo partido a sus servidores.



Primero, nos registramos como clientes en la plataforma.

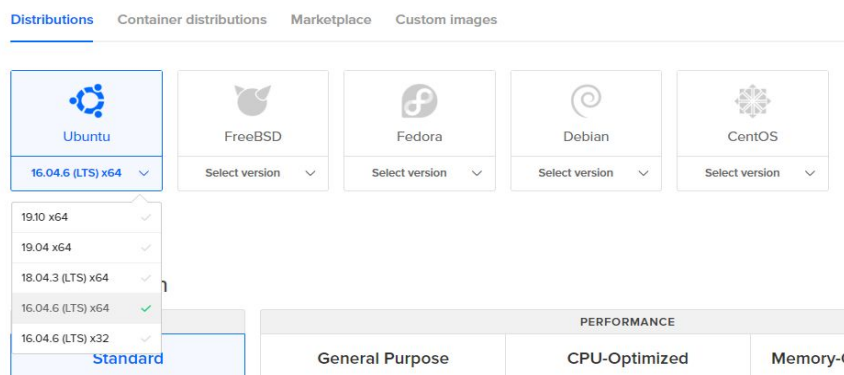


Una vez iniciada la sesión, procedemos a crear el servidor Modern HoneyPot Server (MHN) que nos ayudará a desplegar los sensores y recopilar información.

Seleccionamos la distribución Ubuntu 16.04.6 (LTS) x64 recomendada como estable por los creadores de MHN.

Create Droplets

Choose an image ?





En este apartado aprovechamos el crédito que tenemos y seleccionamos una máquina con suficiente capacidad para que haga de host de nuestro servidor principal de todo el proyecto.

\$5/mo \$0.007/hour	\$10/mo \$0.015/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$20/mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Estará situado en Londres.

 New York	 San Francisco	 Amsterdam	 Singapore	 London	 Frankfurt
1 2 3	1 2	2 3	1	1	1

Lo definiremos con “mhn-server”.

— 1 Droplet +

mhn-server

Finalmente se crea con la siguiente dirección IP: 165.22.120.182

mhn-server LONT / 4GB / 80GB Disk	165.22.120.182 Copy	london		
---	---------------------	--------	--	--

Accedemos a la consola y nos pedirá reiniciar el password.

```
password:
You are required to change your password immediately (root enforced)
Changing password for root.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-169-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@mhn-server:~#
```



Crearemos un usuario “No Root” por recomendaciones de MHN, ya que al desplegar los sensores honeypot en root ha dado varios problemas.

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@mhn-server:~# adduser toni
Adding user `toni' ...
Adding new group `toni' (1000) ...
Adding new user `toni' (1000) with group `toni' ...
Creating home directory `/home/toni' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for toni
Enter the new value, or press ENTER for the default
  Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@mhn-server:~# visudo_
```

Asignamos al nuevo usuario privilegios root.

```
GNU nano 2.5.3      File: /etc/sudoers.tmp

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
toni    ALL=(ALL:ALL) ALL_

# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

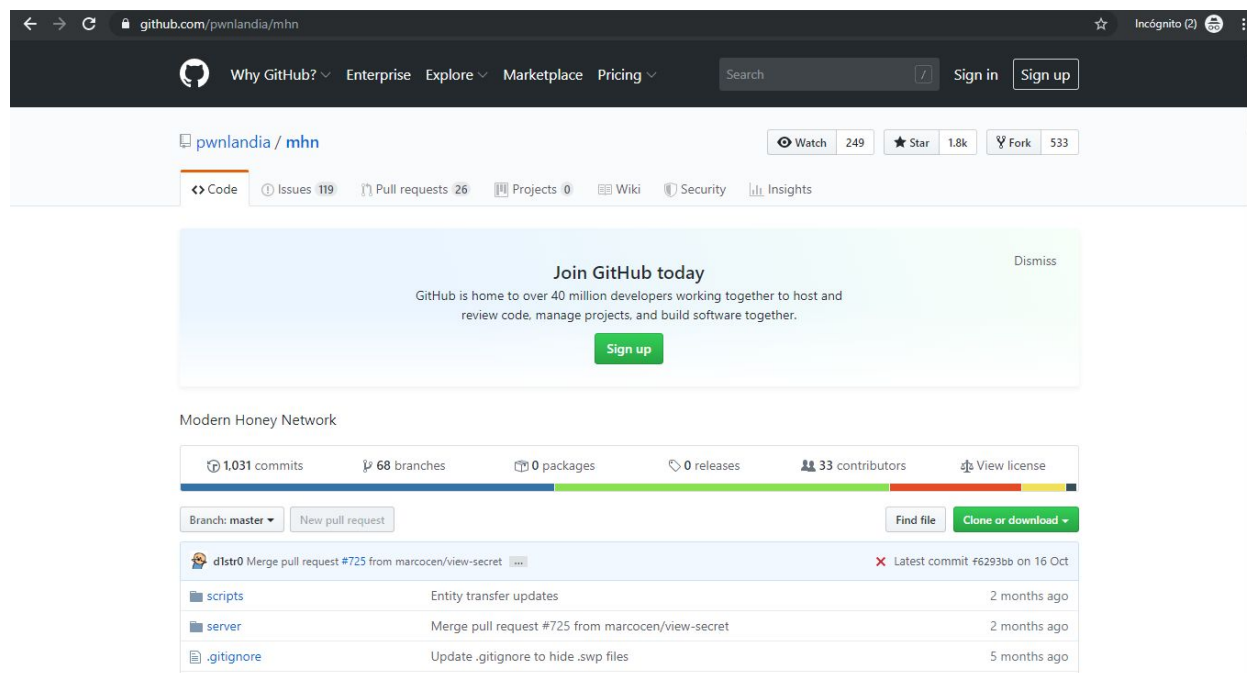
# See sudoers(5) for more information on "#include" directives:

#include_dir /etc/sudoers.d

[ Wrote 31 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File  ^_ Replace    ^U Uncut Text ^T To Spell
```



Como usuario clonamos el repositorio oficial de Github de MHN, y procedemos a la instalación.



```
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
#include_dir /etc/sudoers.d

root@mhn-server:~# su toni
toni@mhn-server:/root$ cd
toni@mhn-server:~$ ls
toni@mhn-server:~$ git clone https://github.com/pwnlandia/mhn
Cloning into 'mhn'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 7410 (delta 0), reused 0 (delta 0), pack-reused 7406
Receiving objects: 100% (7410/7410), 3.66 MiB | 2.44 MiB/s, done.
Resolving deltas: 100% (4028/4028), done.
Checking connectivity... done.
toni@mhn-server:~$ cd mhn/
toni@mhn-server:~/mhn$ sudo ./install.sh
```



MongoDB activo

```
Not creating home directory '/home/mongodb'.
Adding group 'mongodb' (GID 116) ...
Done.
Adding user 'mongodb' to group 'mongodb' ...
Adding user mongodb to group mongodb
Done.
Setting up mongodb-org-mongos (3.6.15) ...
Setting up mongodb-org-tools (3.6.15) ...
Setting up mongodb-org (3.6.15) ...
+ sed -i s/127.0.0.1/0.0.0.0/g /etc/mongod.conf
+ cat
+ systemctl start mongodb
+ systemctl status mongodb
* mongodb.service - High-performance, schema-free document-oriented database
   Loaded: loaded (/etc/systemd/system/mongodb.service; disabled; vendor preset:
   Active: active (running) since Mon 2019-12-02 18:37:48 UTC; 34ms ago
   Main PID: 12261 (mongod)
     Tasks: 1
        Memory: 5.2M
         CPU: 26ms
    CGroup: /system.slice/mongodb.service
            └─12261 /usr/bin/mongod --quiet --config /etc/mongod.conf

Dec 02 18:37:48 mhn-server systemd[1]: Started High-performance, schema-free doc
lines 1-11/11 (END)
```

Para Debug mode seleccionamos: “n”

```
Successfully installed Flask-0.12.4 Flask-Login-0.3.2 Flask-Mail-0.9.1 Flask-Mig
rate-1.8.1 Flask-Principal-0.4.0 Flask-SQLAlchemy-2.3.2 Flask-Script-2.0.6 Flask
-Security-1.7.5 Flask-Testing-0.7.1 Flask-WTF-0.14.2 Jinja2-2.10.3 Mako-1.1.0 Ma
rkupSafe-1.1.1 SQLAlchemy-1.3.11 WTForms-2.2.1 Werkzeug-0.16.0 alembic-1.3.1 amq
p-2.5.2 bcrypt-3.1.4 billiard-3.6.1.0 blinker-1.4 celery-4.3.0 certifi-2019.11.2
8 cffi-1.13.2 chardet-3.0.4 click-7.0 configparser-4.0.2 contextlib2-0.6.0.post1
geopip2-2.9.0 hpfeds-logger-0.0.7.7 hpfeds-threatstream-1.1 idna-2.8 importlib
-metadata-1.1.0 ipaddress-1.0.23 itsdangerous-1.1.0 kombu-4.6.6 maxminddb-1.5.1
more-iterertools-5.0.0 passlib-1.7.2 pathlib2-2.3.5 pycparser-2.19 pygal-2.4.0 pym
ongo-3.7.2 pyopenssl-2.0.3 python-dateutil-2.2 python-editor-1.0.4 pytz-2019.3 re
dis-3.2.1 requests-2.21.0 scandir-1.10.0 six-1.13.0 uWSGI-2.0.18 urllib3-1.24.3
vine-1.3.0 xmldict-0.12.0 zipp-0.6.0
+ '[' -f /etc/redhat-release ']'
+ echo 'DONE installing python virtualenv'
DONE installing python virtualenv
+ mkdir -p /var/log/mhn
+ cd /home/toni/mhn/server/
+ echo =====
=====
+ echo ' MHN Configuration'
MHN Configuration
+ echo =====
=====
+ python generateconfig.py
Do you wish to run in Debug mode?: y/n n_
```

Elegimos un email para el superuser y el resto de campos se puede dejar en blanco.

```
+ echo ' MHN Configuration'
MHN Configuration
+ echo =====
=====
+ python generateconfig.py
Do you wish to run in Debug mode?: y/n n
Superuser email: toni_ladrillo@yahoo.com
Superuser password:
Superuser password: (again):
Server base url ["http://165.22.120.182"]:
Honeytrap url ["http://165.22.120.182:3000"]:
Mail server address ["localhost"]:
Mail server port [25]:
Use TLS for email?: y/n n
Use SSL for email?: y/n n
Mail server username [""]:
Mail server password [""]:
Mail default sender [""]:
Path for log file ["/var/log/mhn/mhn.log"]:
+ echo -e '\nInitializing database, please be patient. This can take several min
utes'

Initializing database, please be patient. This can take several minutes
+ python initdatabase.py
```



Integrar con Splunk: “n”

```
_reset.py /home/toni/mhn/server/mhn /home/toni/mhn/server/mhn.db /home/toni/mhn/
Server/mhn.log /home/toni/mhn/server/mhn.py /home/toni/mhn/server/migration_remo
ve-hostname-and-name-uniq-constraints.sql /home/toni/mhn/server/requirements.txt
+ supervisorctl update
mhn-celery-beat: added process group
mhn-celery-worker: added process group
mhn-collector: added process group
mhn-uwsgi: added process group
+ /etc/init.d/nginx restart
[ ok ] Restarting nginx (via systemctl): nginx.service.
++ date
+ echo '[Mon Dec 2 18:56:17 UTC 2019] ===== MHN Server Install Finished ===
=====
[Mon Dec 2 18:56:17 UTC 2019] ===== MHN Server Install Finished =====
+ echo ''
+ true
+ echo -n 'Would you like to integrate with Splunk? (y/n) '
Would you like to integrate with Splunk? (y/n) + read SPLUNK
+ '[' 'y' == y -o '' == Y ']'
+ '[' 'n' == n -o '' == N ']'
+ true
+ echo -n 'Would you like to integrate with Splunk? (y/n) '
Would you like to integrate with Splunk? (y/n) + read SPLUNK
n_
```

Integrar con ELK: “n”

```
+ true
+ echo -n 'Would you like to integrate with Splunk? (y/n) '
Would you like to integrate with Splunk? (y/n) + read SPLUNK
n
+ '[' 'n' == y -o 'n' == Y ']'
+ '[' 'n' == n -o 'n' == N ']'
+ echo 'Skipping Splunk integration'
Skipping Splunk integration
+ echo 'The splunk integration can be completed at a later time by running this:
The splunk integration can be completed at a later time by running this:
+ echo ' cd /opt/mhn/scripts/'
cd /opt/mhn/scripts/
+ echo ' sudo ./install_splunk_universalforwarder.sh <SPLUNK_HOST> <SPLUNK_PO
RT>'
sudo ./install_splunk_universalforwarder.sh <SPLUNK_HOST> <SPLUNK_PORT>
+ echo ' sudo ./install_hpfeds-logger-splunk.sh'
sudo ./install_hpfeds-logger-splunk.sh
+ break
+ true
+ echo -n 'ELK Script will only work on Debian Based systems like Ubuntu'
ELK Script will only work on Debian Based systems like Ubuntu+ echo -n 'Would yo
u like to install ELK? (y/n) '
Would you like to install ELK? (y/n) + read ELK
n
```

Añadir MHN rules a UFW: “n”

```
+ echo -n 'ELK Script will only work on Debian Based systems like Ubuntu'
ELK Script will only work on Debian Based systems like Ubuntu+ echo -n 'Would yo
u like to install ELK? (y/n) '
Would you like to install ELK? (y/n) + read ELK
n
+ '[' 'n' == y -o 'n' == Y ']'
+ '[' 'n' == n -o 'n' == N ']'
+ echo 'Skipping ELK installation'
Skipping ELK installation
+ echo 'The ELK installation can be completed at a later time by running this:'
The ELK installation can be completed at a later time by running this:
+ echo ' cd /opt/mhn/scripts/'
cd /opt/mhn/scripts/
+ echo ' sudo ./install_elk.sh'
sudo ./install_elk.sh
+ break
+ true
+ echo -n 'A properly configured firewall is highly encouraged while running MHN
A properly configured firewall is highly encouraged while running MHN.+ echo -n
'This script can enable and configure UFW for use with MHN.'
This script can enable and configure UFW for use with MHN.+ echo -n 'Would you l
ike to add MHN rules to UFW? (y/n) '
Would you like to add MHN rules to UFW? (y/n) + read UFW
n
```



Instalación finalizada

```
mhn-celery-worker: started
++ date
+ echo '[Mon Dec  2 18:57:31 UTC 2019] Completed Installation of all MHN package
s
[Mon Dec  2 18:57:31 UTC 2019] Completed Installation of all MHN packages
toni@mhn-server:~/mhn$ ifconfig
eth0      Link encap:Ethernet  HWaddr ee:7b:b1:18:03:b7
          inet addr:165.22.120.182  Bcast:165.22.127.255  Mask:255.255.240.0
          inet6 addr: fe80::ec7b:b1ff:fe18:3b7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:43143 errors:0 dropped:0 overruns:0 frame:0
          TX packets:22903 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:448417031 (448.4 MB)  TX bytes:2049292 (2.0 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1711 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1711 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:201357 (201.3 KB)  TX bytes:201357 (201.3 KB)

toni@mhn-server:~/mhn$ _
```

Podemos acceder a la de nuestro MHN: 165.22.120.182

Datos de acceso: Email que asignamos al usuario creado y su password.



Welcome to the Modern
HoneyPot Network Server

Log In

Email

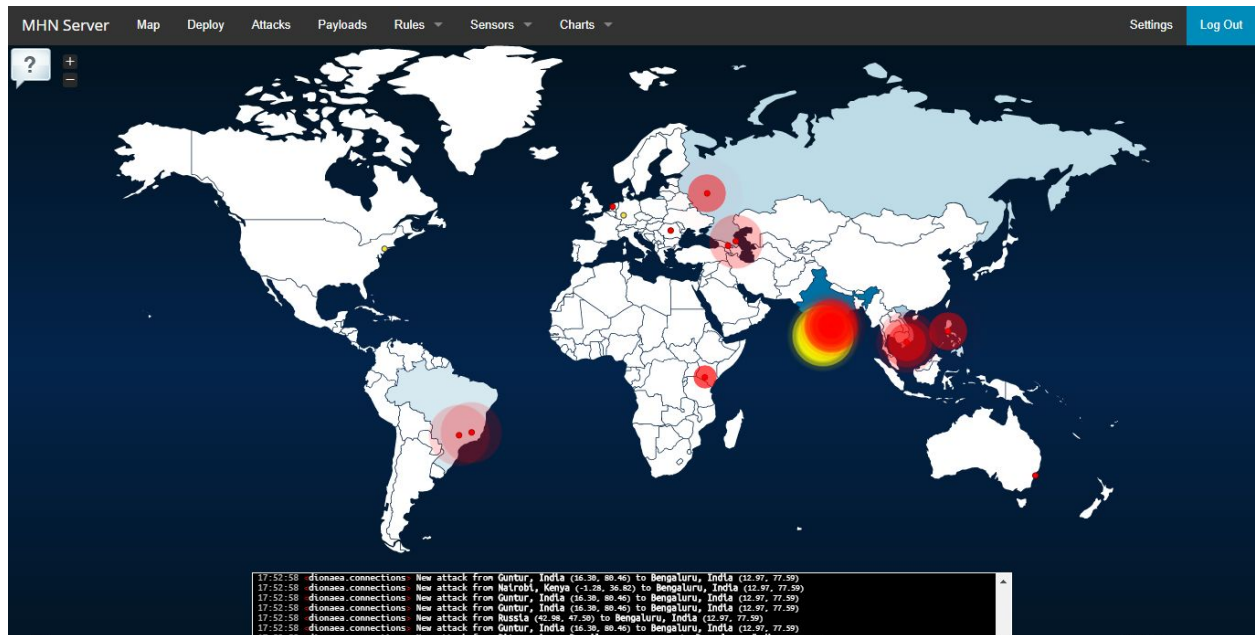
Password

[Forgot password?](#)

Modern HoneyNet Framework is an open source project by: [Pwnlandia](#)



Cuando instalemos los sensores, en la primera pestaña veremos un mapa como el siguiente, con las amenazas que se vayan recogiendo por los honeypots situados en diferentes localizaciones del mundo.





2.2 SESORES / HONEYPOTS

2.2.1 Snort



Es un sistema de detección de intrusos en red que implementa un motor de detección de ataques y escaneo de puertos.

En nuestro caso, este IDS lo hemos hecho funcionar como un honeypot-sniffer que registra los paquetes que coinciden con alguno de los patrones establecidos en la instalación (reglas para backdoor, DDoS, finder, FTP, ataques web, CGI, Nmap...). Así se sabe, cuándo, de dónde y cómo se produjo el ataque.

Las reglas están definidas en el archivo `/opt/snort/rules/local.rules`

El rango de reglas es el máximo definido por la configuración de MHN para obtener la mayor diversidad de casos a examinar.

<https://rules.emergingthreats.net/open/snort-2.9.0/emerging.rules.tar.gz>



INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Ubuntu 16.04.6 (LTS) x64

Create Droplets

Choose an image ?

Distributions Container distributions Marketplace Custom images

 Ubuntu 16.04.6 (LTS) x64	 FreeBSD Select version	 Fedora Select version	 Debian Select version	 CentOS Select version
---------------------------------	-------------------------------	------------------------------	------------------------------	------------------------------

19.10 x64
19.04 x64
18.04.3 (LTS) x64
16.04.6 (LTS) x64
16.04.6 (LTS) x32

PERFORMANCE
General Purpose
CPU Optimized
Memory Optimized

Seleccionamos la máquina menos potente.

\$5/mo \$0.007/hour	\$10/mo \$0.015/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$20/mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Localización: Nueva York

 New York	 San Francisco	 Amsterdam	 Singapore	 London	 Frankfurt
1 2 3	1 2	2 3	1	1	1

Nombre: "snort-honeypot"

1 Droplet

snort-honeypot

IP: 167.99.3.150

snort-honeypot
NYC1 / 1GB / 25GB Disk

167.99.3.150 Copy

newyork



Desde la pestaña Deploy seleccionamos el Script Ubuntu-Snort para linkar el sensor con MHN.



Accedemos a la consola e introducimos la comanda.

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

root@snort-honeypot:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 0a:0b:02:49:77:7d
          inet addr:167.99.3.150  Bcast:167.99.15.255  Mask:255.255.240.0
          inet6 addr: fe80::80b:2ff:fe49:777d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:256 errors:0 dropped:0 overruns:0 frame:0
          TX packets:182 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:568048 (568.0 KB)  TX bytes:16777 (16.7 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:160 errors:0 dropped:0 overruns:0 frame:0
          TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:11040 (11.0 KB)  TX bytes:11040 (11.0 KB)

root@snort-honeypot:~# wget "http://165.22.120.182/api/script/?text=true&script_id=4" -O deploy.sh && sudo bash deploy.sh http://165.22.120.182 vSBvMrGF
```

Una vez acaba la instalación automática, tendremos Snort activo.



2.2.2 GLASTOPF



Glastopf es un servidor web minimalista escrito en Python. Esta herramienta honeypot recopila información sobre ataques basados en aplicaciones web.

La principal diferencia entre Glastopf y otros honeypots web consiste en que estos últimos se enfocan en el uso de plantillas para su funcionamiento. Las cuales acarrearán desventajas inherentes relacionadas con el mantenimiento y el desarrollo continuo que requieren, además de su limitada capacidad para hacer frente a ataques multietapas.

Glastopf utiliza un “emulador de vulnerabilidades”, el cual posibilita la generación de respuestas válidas sin necesidad de utilizar plantillas de aplicaciones web modificadas. Glastopf analiza solicitudes entrantes en busca de cadenas como “=http://” o “CAST(0x”. Si encuentra una coincidencia, intentará descargar y analizar el archivo y responder al atacante de la forma más cercana posible a sus expectativas.

INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Ubuntu 16.04.6 (LTS) x64

Create Droplets

Choose an image ?

Distributions Container distributions Marketplace Custom images

Image	Select version
Ubuntu	Select version
FreeBSD	Select version
Fedora	Select version
Debian	Select version
CentOS	Select version

16.04.6 (LTS) x64

- 19.10 x64
- 19.04 x64
- 18.04.3 (LTS) x64
- 16.04.6 (LTS) x64 ✓
- 16.04.6 (LTS) x32

PERFORMANCE

General Purpose	CPU Optimized	Memory Optimized



Seleccionamos la máquina menos potente.

\$5/mo \$0.007/hour	\$10/mo \$0.015/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$20/mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Localización: San Francisco

 New York	 San Francisco	 Amsterdam	 Singapore	 London	 Frankfurt
1 2 3	1 2	2 3	1	1	1

Nombre: glastopf-honeypot

—

1 Droplet

+

glastopf-honeypot

IP: 167.172.221.117

glastopf-honeypot SFO2 / 1GB / 25GB Disk	167.172.221.117 Copy	sanfrancisco		
--	--------------------------------------	--------------	--	--

Desde la pestaña Deploy seleccionamos el Script Ubuntu-Glastopf para linkar el sensor con MHN.

MHN Server	Map	Deploy	Attacks	Payloads	Rules	Sensors	Charts	Settings	Log Out
------------	-----	--------	---------	----------	-------	---------	--------	----------	---------

Select Script

Ubuntu - Glastopf

Deploy Command

```
wget "http://165.22.120.182/api/script/?text=true&script_id=7" -O deploy.sh && sudo bash deploy.sh  
http://165.22.120.182 vSBvMrGf
```

Accedemos a la consola e introducimos la comanda.

```
You are required to change your password immediately (root enforced)  
Changing password for root.  
(current) UNIX password:  
Enter new UNIX password:  
Retype new UNIX password:  
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-169-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
0 packages can be updated.  
0 updates are security updates.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
root@glastopf-honeypot:~# wget "http://165.22.120.182/api/script/?text=true&script_id=7" -O deploy.sh && sudo bash deploy.sh http://165.22.120.182 vSBvMrGf
```

Una vez acaba la instalación automática, tendremos Glastopf activo.



2.2.3 COWRIE



El honeypot Cowrie tiene características muy interesantes una de ellas es simular un sistema de archivos completo con la posibilidad de crear y borrar archivos, de esta manera, un posible atacante podría creerse que está en el sistema operativo real, cuando en realidad está dentro del Honeypot. Otra particularidad interesante es que podremos añadir ficheros falsos para que si por ejemplo el atacante hace un «cat» a un archivo como /etc/passwd se crea que está leyendo todos los usuarios del propio sistema.

Cowrie también puede guardar archivos descargados de Internet a través de wget o cURL, o también de archivos subidos a través del protocolo SFTP o SCP para posteriormente estudiar a fondo qué son esos archivos que el ciberdelincuente ha intentado colarnos en el sistema.

Cowrie está basado en el honeypot Kippo, pero tiene características adicionales que lo hacen mucho más interesante. Por ejemplo, soporta comandos ejecutados a través de SSH (SSH exec), de esta manera, el atacante podrá enviar comandos.



INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Ubuntu 16.04.6 (LTS) x64

Create Droplets

Choose an image ?

Distributions Container distributions Marketplace Custom images

 Ubuntu 16.04.6 (LTS) x64	 FreeBSD Select version	 Fedora Select version	 Debian Select version	 CentOS Select version
---------------------------------	-------------------------------	------------------------------	------------------------------	------------------------------

19.10 x64
19.04 x64
18.04.3 (LTS) x64
16.04.6 (LTS) x64
16.04.6 (LTS) x32

PERFORMANCE

General Purpose CPU Optimized Memory

Seleccionamos la máquina menos potente.

\$5/mo \$0.007/hour	\$10/mo \$0.015/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$20/mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Localización: Singapore

 New York	 San Francisco	 Amsterdam	 Singapore	 London	 Frankfurt
1 2 3	1 2	2 3	1	1	1

Nombre: "cowrie-honeypot"

1 Droplet

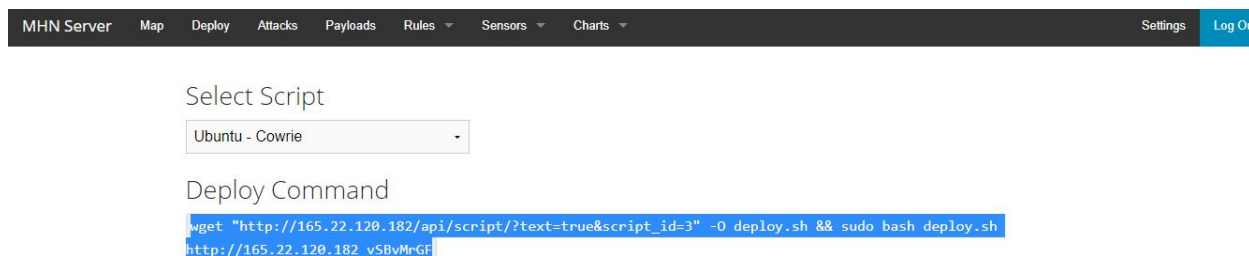
cowrie-honeypot

IP: 206.189.43.28

 cowrie-honeypot SGP1 / 1GB / 25GB Disk	206.189.43.28 Copy	singapore	...
---	--------------------	-----------	-----



Desde la pestaña Deploy seleccionamos el Script Ubuntu-Glastopf para linkar el sensor con MHN.



Accedemos a la consola e introducimos la comanda.

```
root@cowrie-honeypot:~# ifconfig
eth0      Link encap:Ethernet  HWaddr b6:1a:2b:80:2b:64
          inet addr:206.189.43.28  Bcast:206.189.47.255  Mask:255.255.240.0
          inet6 addr: fe80::b41a:2bff:fe80:2b64/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:339168 errors:0 dropped:0 overruns:0 frame:0
          TX packets:328206 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:493593077 (493.5 MB)  TX bytes:48684929 (48.6 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:160 errors:0 dropped:0 overruns:0 frame:0
          TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:11840 (11.8 KB)  TX bytes:11840 (11.8 KB)

root@cowrie-honeypot:~# WGET "HTTP://165.22.120.182/API/SCRIPT/?TEXT=TRUE&SCRIPT_ID=3" -O DEPLOY.SH && SUDO BASH DEPLOY.SH HTTP://165.22.120.182 VsbVmRgF
```

Una vez acaba la instalación automática, tendremos Cowrie activo.



2.2.4 p0f

Es una herramienta de “fingerprinting” pasivo. Puede analizar el tráfico que pasa por las redes a la que la máquina esté conectada, ya sea que este esté dirigido hacia la misma o no.

Toma huellas digitales del sistema operativo del host remoto mediante el análisis de determinados campos en los paquetes capturados.






INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Ubuntu 16.04.6 (LTS) x64

Create Droplets

Choose an image ?

Distributions Container distributions Marketplace Custom images

 Ubuntu 16.04.6 (LTS) x64	 FreeBSD Select version	 Fedora Select version	 Debian Select version	 CentOS Select version
19.10 x64 ✓ 19.04 x64 ✓ 18.04.3 (LTS) x64 ✓ 16.04.6 (LTS) x64 ✓ 16.04.6 (LTS) x32 ✓				
PERFORMANCE				

Seleccionamos la máquina menos potente.

\$5 /mo \$0.007/hour	\$10 /mo \$0.015/hour	\$15 /mo \$0.022/hour	\$15 /mo \$0.022/hour	\$15 /mo \$0.022/hour	\$20 /mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer



Localización: Frankfurt

 New York 1 2 3	 San Francisco 1 2	 Amsterdam 2 3	 Singapore 1	 London 1	 Frankfurt 1
-----------------------	--------------------------	----------------------	--------------------	-----------------	--------------------

Nombre: “pof-honeypot”

— 1 Droplet +

pof-honeypot

IP: 167.172.191.81

 pof-honeypot FRA1 / 1GB / 25GB Disk	167.172.191.81 Copy	frankfurt	3 ...
--	---------------------	-----------	-------

Desde la pestaña Deploy seleccionamos el Script Ubuntu-Glastopf para linkar el sensor con MHN.

MHN Server Map Deploy Attacks Payloads Rules Sensors Charts Settings Log Out

Select Script

Ubuntu - pOf

Deploy Command

wget "http://165.22.120.182/api/script/?text=true&script_id=9" -O deploy.sh && sudo bash deploy.sh
http://165.22.120.182 vSBvMrGF

Accedemos a la consola e introducimos la comanda.

```
You are required to change your password immediately (root enforced)
Changing password for root.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-169-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@pof-honeypot:~# wget "http://165.22.120.182/api/script/?text=true&script_id=9" -O deploy.sh && sudo bash deploy.sh http://165.22.120.182 vSBvMrGF
```

Una vez acaba la instalación automática, tendremos pOf activo.



2.2.5 DIONAEA



Una de las principales características de este honeypot, es que soporta rutinas no bloqueantes, lo que mejora considerablemente el desempeño general del sistema y permite ejecutar múltiples operaciones de escritura y lectura sobre uno o varios sockets sin necesidad de esperar una respuesta por parte del receptor.

En este caso utiliza el módulo `asyncio` con las librerías `libudns` y `libev` para peticiones DNS no bloqueantes y recibir notificaciones sobre diferentes tipos de eventos en el entorno de red respectivamente.

Los protocolos soportados por Dionaea son implementaciones propias del honeypot.

INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Ubuntu 16.04.6 (LTS) x64

Create Droplets

Choose an image ?

[Distributions](#) [Container distributions](#) [Marketplace](#) [Custom images](#)

Distribution	Version
Ubuntu	16.04.6 (LTS) x64
FreeBSD	Select version
Fedora	Select version
Debian	Select version
CentOS	Select version

PERFORMANCE

General Purpose	CPU Optimized	Memory Optimized



Seleccionamos la máquina menos potente.

\$5/mo \$0.007/hour	\$10/mo \$0.015/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$15/mo \$0.022/hour	\$20/mo \$0.030/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Localización: Bangalore

 Toronto	 Bangalore
1	1

Nombre: "dionaea-honeypot"

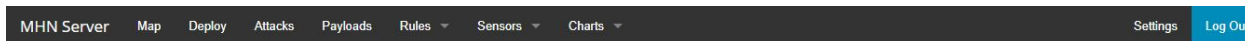
— 1 Droplet +

dionaea-honeypot

IP: 159.65.150.186

dionaea-honeypot BLR1 / 1GB / 25GB Disk	159.65.150.186 Copy	bangalore		
---	-------------------------------------	---------------------------	--	--

Desde la pestaña Deploy seleccionamos el Script Ubuntu-Glastopf para linkar el sensor con MHN.



Select Script

Ubuntu/Raspberry Pi - Dionaea

Deploy Command

```
wget "http://165.22.120.182/api/script/?text=true&script_id=2" -O deploy.sh && sudo bash deploy.sh  
http://165.22.120.182 vSBvMrGF
```

Accedemos a la consola e introducimos la comanda.

```
You are required to change your password immediately (root enforced)  
Changing password for root.  
(current) UNIX password:  
Enter new UNIX password:  
Retype new UNIX password:  
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-169-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
0 packages can be updated.  
0 updates are security updates.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
root@dionaea-honeypot:~# wget "http://165.22.120.182/api/script/?text=true&script_id=2" -O deploy.sh && sudo bash deploy.sh http://165.22.120.182 vSBvMrGF
```

Una vez acaba la instalación automática, tendremos Dionaea activo.



2.2.6 HONEYDB Agent

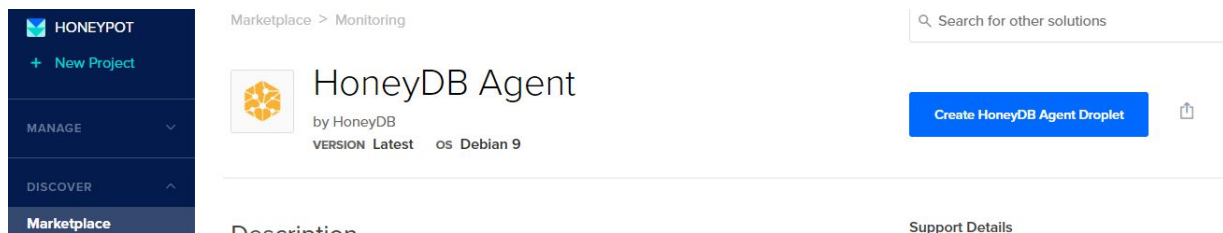


Es un honeypot con una seguridad medio-baja de interacción, que soporta la emulación de servicios de red TCP, UDP, FTP, Telnet, SSH, DNS y MySQL principalmente.

A parte de los honeypots/sensores del servidor MHN, instalamos un honeypot extra como ejemplo. En este caso la instalación del honeypot va como el resto por Digital Ocean pero el servidor que recibirá la información generada será la plataforma HoneyDB.

El servidor también muestra la información almacenada por todos los usuarios que han instalado un sensor. Aunque nosotros podremos filtrar y mostrar únicamente la nuestra.

Procedemos a la instalación. Iremos a la pestaña de Marketplace dentro de nuestro proyecto, y seleccionaremos el HoneyDB Agent





INSTALACIÓN EN DIGITALOCEAN:

Distribución recomendada Debian 9.7 x64

Create Droplets

Choose an image ?

Distributions Container distributions Marketplace Custom images

 Ubuntu Select version ▾	 FreeBSD Select version ▾	 Fedora Select version ▾	 Debian 9.7 x64 ▾ 9.7 x64 ✓ 10.0 x64 ✓	 CentOS Select version ▾
--------------------------------	---------------------------------	--------------------------------	--	--------------------------------

Seleccionamos la máquina menos potente.

\$5/mo \$0.007/hour 1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer	\$10/mo \$0.015/hour 2 GB / 1 CPU 50 GB SSD disk 2 TB transfer	\$15/mo \$0.022/hour 3 GB / 1 CPU 60 GB SSD disk 3 TB transfer	\$15/mo \$0.022/hour 2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer	\$15/mo \$0.022/hour 1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer	\$20/mo \$0.030/hour 4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer
---	---	---	--	--	--

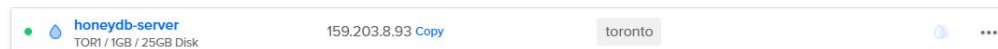
Localización: Toronto

 Toronto 1	 Bangalore 1
------------------	--------------------

Nombre: "honeydb-server"

—	1 Droplet	+	honeydb-server
---	-----------	---	----------------

IP: 159.209.8.93



Accedemos a la consola. Are your ready to enter your agent keys?: “y”

```

Welcome to the One-Click HoneyDB Agent Droplet.

You can continue to use port 22 (SSH) to access this host. However, if you
want to make port 22 (SSH) a honeypot service you'll need to configure your SSH
service to listen on a different port and then configure honeydb-agent to listen
on port 22 (by default honeydb-agent is using port 2222 to emulate SSH).

To view your honeypot data, login at: http://honeymb.io
* The honeydb-agent configuration files are located at /etc/honeydb
* For information on agent configuration, visit:
  https://honeymb-agent-docs.readthedocs.io/en/latest/configuration/#agent-configuration
* For information on service configuration, visit:
  https://honeymb-agent-docs.readthedocs.io/en/latest/configuration/#service-configuration
* If you need agent keys to enable sending data to HoneyDB,
  visit: http://honeymb.io

=====
To delete this message of the day: rm -rf /etc/update-notif.d/99-honeydb-agent

The programs included with the Debian GNU/Linux system are free software:
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

=====
Configure HoneyDB Agent!

Generate your agent keys at honeymb.io

This script will only run automatically once.
You can run it again later but running the command:
/opt/honeydb/configure_agent.sh

WARNING: If you are manually executing this script it will
overwrite your existing agent.conf file. Any previous edits
to /etc/honeydb/agent.conf will be lost.


Are you ready to enter your agent keys? [Y/n] Y
```

Vamos a la pestaña “User” de la dirección web HoneyDB:

<https://riskdiscovery.com/honeydb/>

Las dos únicas formas de Login son mediante GitHub o Twitter. En nuestra caso, seleccionamos Twitter.



 **HoneyDB**

Community driven honeypot sensor data collection and aggregation.

IP Address

Search

Home

Hosts

Services

Threat Info API

Deploy

Project Sponsors

Twitter

Stats


Downloads


User

Status

About

Login

 Sign in with GitHub

 Sign in with Twitter

Login to generate a Threat Info API access key and/or a HoneyDB agent API key.

- Threat Info API access keys - Used to query the HoneyDB API.
- HoneyDB Agent API keys - Used when deploying your own HoneyDB Agent.
 - When you login to HoneyDB you can filter graphs to show only the data from your agents.
- Creating an account will add you to the HoneyDB newsletter mailing list. Emails will contain an unsubscribe link if you no longer want to receive newsletter emails.

Not another password! HoneyDB does not want to manage user passwords, and you probably don't want to create another password for another web site. Currently HoneyDB support sign in via GitHub and Twitter.

Project sponsors [Novcon](#) | [Cloudsmith](#) | [NoVirusThanks](#) | [ThreatSTOP](#) | [Hoplite](#)
Copyright © HoneyDB

Authorize HoneyDB to access your account?


Authorize app

Cancel

This application will be able to:

- See Tweets from your timeline (including protected Tweets) as well as your Lists and collections.
- See your Twitter profile information and account settings.
- See accounts you follow, mute, and block.
- See your email address.

[Learn more about third-party app permissions in the Help Center.](#)

**HoneyDB**
riskdiscovery.com/honeydb

HoneyDB provides real time data of honeypot activity. This data comes from honeypots deployed on the Internet using the HoneyPy honeypot.

[Privacy Policy](#)
[Terms and Conditions](#)

We recommend reviewing the app's terms and privacy policy to understand how it will use data from your Twitter account. You can revoke access to any app at any time from the **Apps and sessions** section of your Twitter account settings.

By authorizing an app you continue to operate under Twitter's **Terms of Service**. In particular, some usage information will be shared back with Twitter.

For more, see our [Privacy Policy](#).

Se generan tres Keys:

- HoneyDB API ID: nuestro identificador como usuario registrado.
- Threat information API: para realizar las queries a HoneyDB API.
- HoneyDB Agent: el identificador para hacer queries de nuestro sensor.



HoneyDB

Community driven honeypot sensor data collection and aggregation.

IP Address

Search

Home

Hosts

Services

Threat Info API

Deploy

Project Sponsors

Twitter

Stats

Downloads

User

Status

About

API Keys

HoneyDB API ID

The HoneyDB API ID is your identifier to use when querying HoneyDB APIs.

1634880241394ee9e039f95c806eed53f3c7f515685b94f3ef81873d2c96028

Secret Keys

Note, there are two types of secret keys. How to know which one to use:

If you are going to be calling the Threat Info API via a tool or script to download threat information, you need the **Threat Information API** key.

If you are deploying a honeypot with the HoneyDB Agent or HoneyPy, you need the **HoneyDB Agent/HoneyPy Sensor API** key.

Threat information API

The threat information API key can be used to query HoneyDB APIs.

1e8b0c39255093fa68b904f140d6e0478415a01f959c234c7d9be224718f1185

Generate Key

HoneyDB Agent/HoneyPy Sensor API

The HoneyDB Agent/HoneyPy sensor API key is used to log sensor data to HoneyDB.

a9a18eeb468232929c6f8f9f3e9181211af8be7ea896cabd65347394d7571f82

Introducimos las keys de nuestro ID y del honeypot.



```
service to listen on a different port and then configure honeydb-agent to listen
on port 22 (by default honeydb-agent is using port 2222 to emulate SSH).

To view your honeypot data, login at: http://honeydb.io
* The honeydb-agent configuration files are located at /etc/honeydb
* For information on agent configuration, visit:
  https://honeydb-agent-docs.readthedocs.io/en/latest/configuration/#agent-configuration
* For information on service configuration, visit:
  https://honeydb-agent-docs.readthedocs.io/en/latest/configuration/#service-configuration
* If you need agent keys to enable sending data to HoneyDB,
  visit: http://honeydb.io

=====
To delete this message of the day: rm -rf /etc/update-motd.d/99-honeydb-agent

The programs included with the Debian GNU/Linux system are free software:
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

=====
Configure HoneyDB Agent!

Generate your agent keys at honeydb.io

This script will only run automatically once.
You can run it again later but running the command:
/opt/honeydb/configure_agent.sh

WARNING: If you are manually executing this script it will
overwrite your existing agent.conf file. Any previous edits
to /etc/honeydb/agent.conf will be lost.

Are you ready to enter your agent keys? [Y/n] Y

Great! First enter your API ID: 1634880241394ee9e039f95c806eef453f3c7f515605b94f3ef01873d2c9602b
Next, enter your API Key: a9a18eeb468232929c6f8f9f3e9181211af8be7ea896cabd65347394d7571f82
Restarting honeydb-agent...

HoneyDB Agent configuration complete!
Remember, you can always modify agent config by editing
these files: /etc/honeydb/agent.conf and /etc/honeydb/services.conf

Happy Honeypotting!
```

Tenemos activo nuestro HoneyDB Agent.

En el archivo `/etc/honeydb/services.conf` podemos activar/desactivar los servicios que deseemos para recolectar datos.

Plugin	Port	Description	Enabled
Echo_tcp	7	Echo back data received via tcp.	Yes



Echo_udp	7	Echo back data received via udp.	Yes
MOTD_tcp	8	Send a message via tcp and close connection.	Yes
MOTD_udp	8	Send a message via udp.	Yes
FTP_tcp	21	FTP service.	Yes
SSH_tcp	2222	SSH service.	Yes
Telnet_tcp	23	Telnet service.	Yes
SMTP_tcp	25	SMTP service.	Yes
DNS_udp	53	DNS service.	Yes
TFTP_udp	69	TFTP service.	Yes
HTTP_tcp	80	HTTP service.	Yes
Modbus_tcp	502	Modbus service.	Yes
Random_tcp	2048	Send random data via tcp.	Yes
Telnet_iot_tcp	2223	Send random data via tcp.	Yes
MySQL_tcp	3306	MySQL server.	Yes
HashCountRandom_tcp	4096	Send random data prefixed with a hash of a counter via tcp.	Yes
RDP_tcp	3389	Windows RDP service.	Yes
SIP_udp	5060	The Session Initiation Protocol (SIP)	Yes
VNC_tcp	5900	Low interaction VNC service.	Yes
Redis_tcp	6379	Basic Redis.	Yes
Weblogic_tcp	7001	Send basic HTTP replies.	Yes
Elasticsearch_tcp	9200	Send basic elasticsearch replies.	Yes
Memecached_tcp	11211	Send basic memcached replies.	Yes



2.2.7 Otros honeypots de interés

KIPPO



Kippo es una solución honeypot de media interacción para sistemas Linux que registra ataques de fuerza bruta sobre SSH (22/tcp). Una de sus cualidades más es que una vez logrado el acceso permite cierta interacción con una Shell simulada que posteriormente ofrecerá información al administrador sobre los comandos ejecutados.

Kippo ofrece la visión de un sistema Linux fuertemente fortificado, ya que si solo se instala kippo sobre el sistema original, este solo tendrá abiertos los puertos 22/tcp para SSH y 80/tcp para HTTP.

Entre las características de Kippo destacan:

- Simula un sistema de ficheros falsos que permite la eliminación y creación de archivos. El sistema de ficheros emula una arquitectura Debian 5.0.

- Posibilidad de añadir contenido falso a los archivos, de forma que al realizar operaciones como `cat /etc/passwd` se mostraría información personalizada. Tras la instalación solo se incluye un número muy limitado de información en determinados archivos clave.

- Registro de sesiones almacenado en un formato compatible con ULM, que ofrece información sobre las marcas de tiempo de los accesos.

- Almacena los archivos descargados para una inspección posterior.

- Puesto que el servicio SSH es simulado permite modificar el comportamiento de ciertas sentencias para que no se comporten sobre el sistema como lo harían usualmente.



WORDPOT



Wordpot es un honeypot WordPress que detecta sondas para plugins, temas (themes), TimThumb y otros archivos comunes utilizados para obtención de huellas digitales de una instalación de WordPress.

Se puede utilizar un tema de WordPress como lo haría en una instalación de WordPress normal, situando la carpeta del tema en el directorio de temas.

También podría ser necesario editar el esqueleto HTML que se almacena en la carpeta de plantillas “templates/” y debe ser nombrado como su tema (por ejemplo: “themename.html”).

ELASTICHONEY



Este honeypot es bastante simple. Toma solicitudes en los puntos finales /, / _search y / _nodes y devuelve una respuesta JSON que es idéntica a una instancia ES vulnerable (debería ser idéntica).

Los ataques se registran tan pronto como se detectan. Por defecto, elastichoney registra los ataques en formato JSON en stdout, así como en un archivo llamado elastichoney.log.

Es importante tener en cuenta que esto no es infalible. Las personas con experiencia pueden echar un vistazo al código y encontrar rápidamente formas de detectar el honeypot. No es perfecto, pero funciona. Echemos un vistazo a algunos resultados.

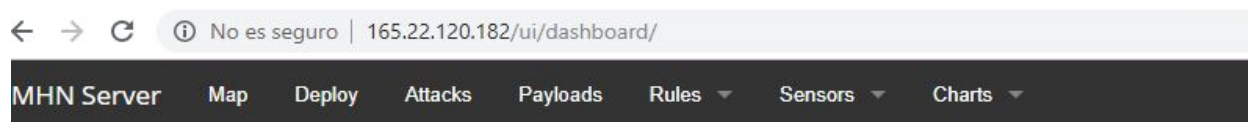


3. ESTADÍSTICAS

3.1 MHN

Esta es una selección de estadísticas que recoge en las últimas 24h nuestro servidor MHN.

Como se aprecia en un solo día, se llegan a contabilizar 581.273 intentos de intrusión o escaneo de puertos para encontrar alguna vulnerabilidad.



Attack Stats

Attacks in the last 24 hours: **581,273**

Estos ataques provienen de IPs de países en vías de desarrollo, ya sea como origen o a través de proxies registrados en ellos. Probablemente, debido a que la legislación y la persecución de un posible delito es más laxa, y el presunto atacante puede actuar con mayor impunidad.

TOP 5 Attackers IPs	IP	Country	Number of Attacks
1	118.173.93.86	Tailandia	71.207
2	27.71.206.25	Vietnam	70.654
3	91.235.227.119	Ucrania	66.241
4	49.205.203.9	India	65.690
5	45.172.144.201	Brasil	58.766



Por la información que tenemos sobre los puertos con más actividad, vemos que se busca acceder a obtener el control de la máquina o acceso a la base de datos para analizar la información almacenada.

TOP 5 Attacked ports	Port	Description	Number of times
1	1433	Microsoft-SQL-Server	558.101
2	445	Microsoft-DS (Active Directory)	12.189
3	22	SSH, scp, SFTP	3.170
4	23	Telnet remote control	1.021
5	2222	SSH	530

El sensor que registró una mayor actividad fue Dionaea, debido a su característica principal de no bloquear entradas.

TOP 5 Sensors	Honeypot	Number of attacks
1	Dionaea	571.359
2	p0f	7.860
3	Snort	1.647
4	Cowrie	338
5	Glastopf	29



Una de las principales razones para la que incluimos Snort como uno de los sensores es su capacidad de definir los posibles payloads que se han utilizado.

Payloads Report

Search Filters

Payload

snort.alerts

Regex Term

pcre regex

GO

source_ip	destination_port	priority	classification	signature
185.175.93.78	2589	2	30	ET DROP Dshield Block Listed Source group 1
80.82.78.100	1027	2	30	ET CINS Active Threat Intelligence Poor Reputation IP UDP group 70
159.226.124.234	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433
222.173.102.2	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433
222.173.102.2	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433
222.173.102.2	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433
175.7.37.247	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433
159.203.201.21	33315	2	30	ET DROP Dshield Block Listed Source group 1
185.156.73.66	7353	2	30	ET DROP Dshield Block Listed Source group 1
122.165.179.176	1433	2	3	ET SCAN Suspicious inbound to MSSQL port 1433

TOP 5 Attacks Signatures	Description	Number of times
1	ET DROP Dshield Block Listed Source group 1	503
2	ET SCAN Suspicious inbound to MSSQL port 1433	471
3	ET CINS Active Threat Intel Poor Reputation IP TCP group 87	48
4	ET CINS Active Threat Intel Poor Reputation IP TCP group 70	45
5	ET SCAN Sipvicious User-Agent Detected (friendly-scanner)	43



Glastopf también genera reporte de payloads, en este caso, vemos que se intentan ejecutar comandos vía url.

Payloads Report

Search Filters

Payload

glastopf.events

Regex Term

pcr regex

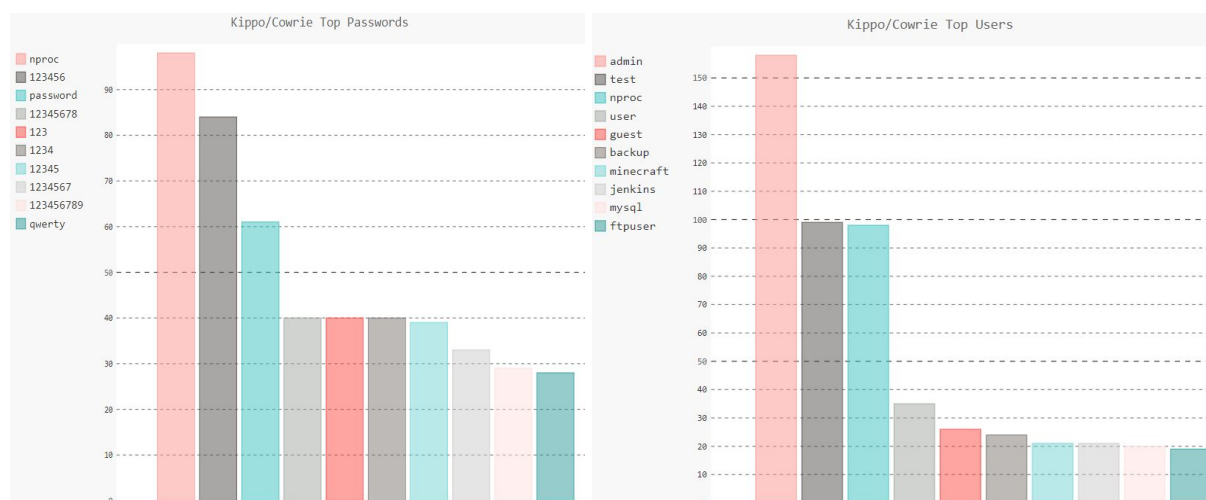
GO

source	request_url
[u'220.248.165.19', 37784]	/TP/public/index.php?s=index/\think\app\invokefunction&function=call_user_func_array&vars[0]=phpinfo&vars[1][]=1
[u'220.248.165.19', 57313]	/TP/public/index.php
[u'108.29.127.202', 39392]	/
[u'23.251.84.126', 47779]	/editBlackAndWhiteList
[u'173.249.51.143', 61000]	/
[u'185.153.196.97', 49598]	/?XDEBUG_SESSION_START=phpstorm
[u'223.18.145.232', 49853]	/setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=busybox&curpath=/¤tsetting.htm=1
[u'42.157.192.132', 47112]	/
[u'5.189.188.207', 61000]	/
[u'178.238.226.57', 61000]	/
[u'188.113.139.237', 41191]	/
[u'201.229.4.50', 49088]	/
[u'185.53.88.5', 35704]	//admin/config.php?password%5B0%5D=bebydviyx&username=admin
[u'207.180.223.221', 61000]	/



El honeypot Cowrie nos ha ayudado a recoger una lista de usuarios y passwords más utilizados en los intentos de intrusión. Según se aprecia en la lista, los usuarios seguimos registrándonos con información fácil de recordar o que viene por defecto en la configuración.

TOP Users	TOP Passwords
admin	nproc
test	123456
nproc	password
user	12345678
guest	123



Nos llamó la atención que se utilizará “nproc” en este apartado. A continuación, explicamos en qué consiste este comando de linux.

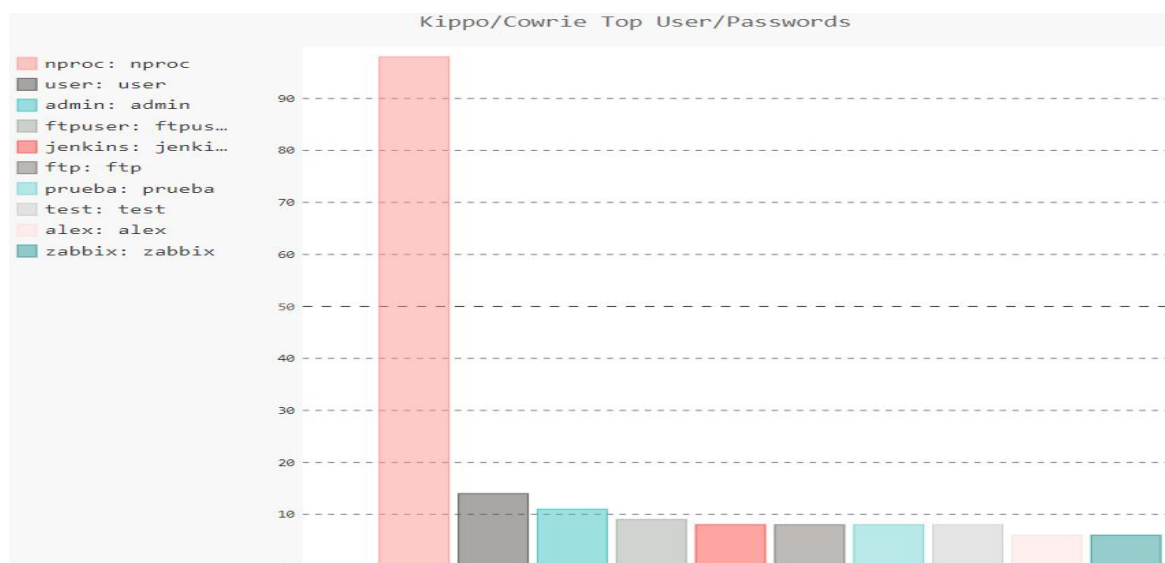


nproc

Cada proceso que se ejecuta en un sistema requiere de CPU para realizar lo que se espera de él. En muchas ocasiones la CPU del sistema está sobrecargada por los procesos que están en marcha, es aquí donde la herramienta nproc no servirá para confirmar el número disponible de unidades de procesamiento para nuevos procesos.

Entendemos que los atacantes desean saber a qué nivel está trabajando la CPU para provocar un “overload” y reiniciar la máquina con la posibilidad de entrar con privilegios “root”.

En la combinación de user-password apreciamos que también se busca el acceso a servicios como ftp, jenkins y zabbix.





3.2 HoneyDB

HoneyDB a través de la Threat Information API y su integración con la aplicación Postman, nos permite realizar varias queries predefinidas. Podemos obtener la información almacenada en formato HTML, XML, JSON... y una vez parseada a Excel o con librerías python para Data Analysis, podemos analizarla en más profundidad.

The screenshot shows the Postman interface with a GET request to `https://riskdiscovery.com/honeydb/api/bad-hosts`. The request headers include `X-HoneyDb-ApId` and `X-HoneyDb-APIKey`. The response body is a JSON array of threat intelligence data.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> X-HoneyDb-ApId	1634880241394ee9e039f95c806eef53f3c7f515685b94...	The HoneyDB API ID is your identifier to use when queryl...
<input checked="" type="checkbox"/> X-HoneyDb-APIKey	c1b40819e815cf4d2e11550196eb1534e6383a53174db...	The threat information API key is required to query infor...
Key	Value	Description

Temporary Headers (8)

Body: 200 OK, Time: 2.21s, Size: 471.67 KB, Save Response

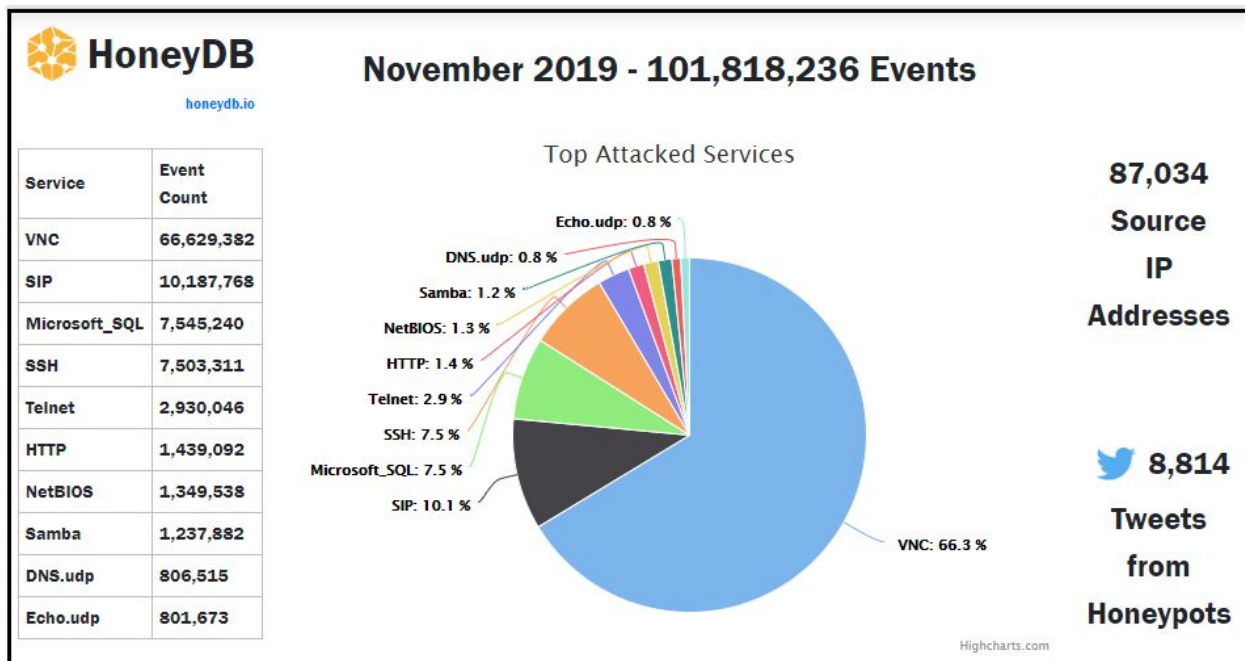
Response Body (JSON):

```
1 [{"remote_host": "193.188.22.114", "count": "613214", "last_seen": "2019-12-06"}, {"remote_host": "185.156.177.44", "count": "571656", "last_seen": "2019-12-06"}, {"remote_host": "185.156.177.11", "count": "570916", "last_seen": "2019-12-06"}, {"remote_host": "185.153.198.197", "count": "315152", "last_seen": "2019-12-06"}, {"remote_host": "185.153.197.251", "count": "296764", "last_seen": "2019-12-06"}, {"remote_host": "185.153.196.159", "count": "170097", "last_seen": "2019-12-06"}, {"remote_host": "89.144.47.5", "count": "71265", "last_seen": "2019-12-06"}, {"remote_host": "119.48.181.233", "count": "53412", "last_seen": "2019-12-05"}, {"remote_host": "159.138.5.185", "count": "44726", "last_seen": "2019-12-06"}, {"remote_host": "128.209.99.194", "count": "36262", "last_seen": "2019-12-06"}, {"remote_host": "213.186.168.178", "count": "34048", "last_seen": "2019-12-06"}, {"remote_host": "222.186.173.183", "count": "29960", "last_seen": "2019-12-06"}, {"remote_host": "104.247.243.15", "count": "25423", "last_seen": "2019-12-05"}, {"remote_host": "200.187.174.127", "count": "25244", "last_seen": "2019-12-06"}, {"remote_host": "187.145.117.234", "count": "23070", "last_seen": "2019-12-05"}, {"remote_host": "185.222.59.1", "count": "21961", "last_seen": "2019-12-06"}, {"remote_host": "186.159.6.116", "count": "21741", "last_seen": "2019-12-06"}]
```



HoneyDB recoge los datos de nuestro honeypot y los agrega a su base datos global.

Podemos comparar la información generada por nuestros honeypots de la red MHN con la de HoneyDB. Confirmamos que en ambas redes los atacantes buscan el control remoto del sistema o el acceso a bases de datos.

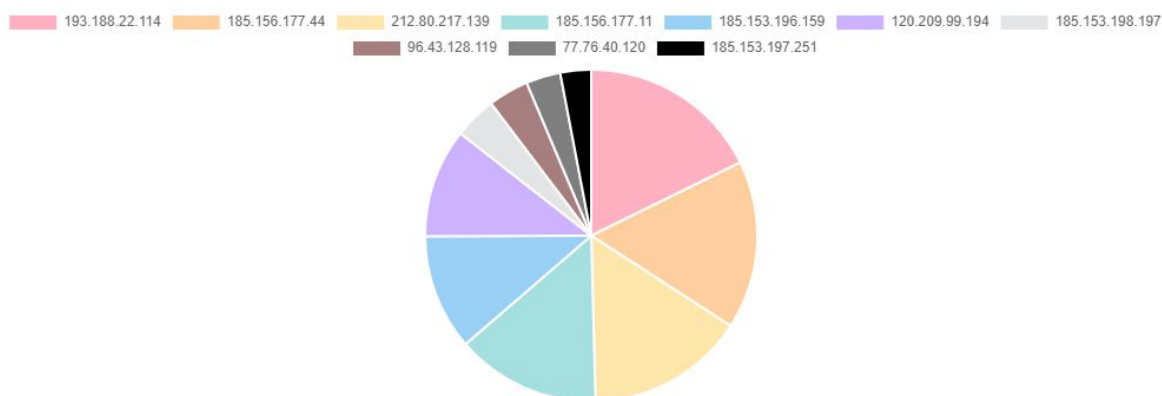




Respecto al origen de los ataques si que difieren completamente de los datos de nuestra Modern HoneyPot Network.

Top Attack Hosts

All Data ▼



Hosts

Date	Remote Host	Country	Count	Service
2019-12-07	193.188.22.114	Russia	192413	VNC
2019-12-07	185.156.177.44	Russia	178312	VNC
2019-12-07	212.80.217.139	Netherlands	167609	VNC
2019-12-07	185.156.177.11	Russia	152821	VNC
2019-12-07	185.153.196.159	Republic of Moldova	122519	VNC
2019-12-07	120.209.99.194	China	115273	Telnet
2019-12-07	185.153.198.197	Republic of Moldova	44836	VNC
2019-12-07	96.43.128.119	United States	42871	Microsoft_SQL
2019-12-07	77.76.40.120	Bulgaria	36415	Microsoft_SQL
2019-12-07	185.153.197.251	Republic of Moldova	32554	VNC



4. BIBLIOGRAFÍA:

Información general sobre honeypots:

Wikipedia:

<https://es.wikipedia.org/wiki/Honeypot>

<https://es.wikipedia.org/wiki/Honeynet>

Incibe:

<https://www.incibe-cert.es/blog/honeypot-herramienta-conocer-al-enemigo>

Digital Ocean código para la creación de cuenta con 50\$ de crédito:

https://www.digitalocean.com/?refcode=aac9b4d8be8f&utm_campaign=Referral_Invite&utm_medium=Referral_Program&utm_source=CopyPaste

Modern Honeypot Network:

Github:

<https://github.com/pwnlandia/mhn>

Documentación:

<https://github.com/pwnlandia/mhn/wiki>

API:

<https://github.com/pwnlandia/mhn/wiki/MHN-REST-APIs>

HoneyDB:

Web oficial:

<https://riskdiscovery.com/honeydb/>

Twitter:

<https://twitter.com/Honeydblo>

API:

<https://riskdiscovery.com/honeydb/threats>

Snort

Web oficial:

<https://www.snort.org/>

Documentación:

<https://www.snort.org/documents#OfficialDocumentation>

Wikipedia:

<https://es.wikipedia.org/wiki/Snort>



Glastopf

Github:

<https://github.com/mushorg/glastopf>

Web oficial:

<http://mushmush.org/>

Cowrie

Github:

<https://github.com/cowrie/cowrie>

Web oficial:

<https://www.cowrie.org/>

Documentación:

<https://cowrie.readthedocs.io/en/latest/>

Twitter:

<https://twitter.com/micheloosterhof>

p0f

Web oficial:

<http://lcamtuf.coredump.cx/p0f3/>

Wikipedia:

<https://en.wikipedia.org/wiki/P0f>

Dionaea

Github:

<https://github.com/DinoTools/dionaea>

Documentación:

<https://dionaea.readthedocs.io/en/latest/>

Kippo

Wikipedia:

<https://en.wikipedia.org/wiki/Kippo>

Github:

<https://github.com/desaster/kippo>

ElasticHoney

Github:

<https://github.com/jordan-wright/elasticchoney>

Wordpot

Github:

<https://github.com/gbrindisi/wordpot>



5. SCRIPTS

5.1 Snort Installation Script

```
#!/bin/bash
INTERFACE=$(basename -a /sys/class/net/e*)
set -e
set -x

if [ $# -ne 2 ]
then
    if [ $# -eq 3 ]
    then
        INTERFACE=$3
    else
        echo "Wrong number of arguments supplied."
        echo "Usage: $0 <server_url> <deploy_key>."
        exit 1
    fi
fi

compareint=$(echo "$INTERFACE" | wc -w)

if [ "$INTERFACE" = "e*" ] || [ "$compareint" -ne 1 ]
then
    echo "No Interface selectable, please provide manually."
    echo "Usage: $0 <server_url> <deploy_key> <INTERFACE>"
    exit 1
fi

server_url=$1
deploy_key=$2
apt-get update
DEBIAN_FRONTEND=noninteractive apt-get -y install build-essential libpcap-dev libjansson-dev libpcap3-dev libdnet-dev
libdumbnet-dev libdaq-dev flex bison python-pip git make automake libtool zlib1g-dev
pip install --upgrade distribute
pip install virtualenv

# Install hpfeeds and required libs...
cd /tmp
rm -rf libev*
wget https://github.com/pwnlandia/hpfeeds/releases/download/libev-4.15/libev-4.15.tar.gz
tar zxvf libev-4.15.tar.gz
cd libev-4.15
./configure && make && make install
ldconfig
```




```
cd /tmp
rm -rf hpfeeds
git clone https://github.com/pwnlandia/hpfeeds.git
cd hpfeeds/appsupport/libhpfeeds
autoreconf --install
./configure && make && make install

cd /tmp
rm -rf snort
git clone -b hpfeeds-support https://github.com/threatstream/snort.git
export CPPFLAGS=-I/include
cd snort
./configure --prefix=/opt/snort && make && make install

# Register the sensor with the MHN server.
wget $server_url/static/registration.txt -O registration.sh
chmod 755 registration.sh
# Note: this will export the HPF_* variables
./registration.sh $server_url $deploy_key "snort"

mkdir -p /opt/snort/etc /opt/snort/rules /opt/snort/lib/snort_dynamicrules /opt/snort/lib/snort_dynamicpreprocessor /var/log/snort/
cd etc
cp snort.conf classification.config reference.config threshold.config unicode.map /opt/snort/etc/
touch /opt/snort/rules/white_list.rules
touch /opt/snort/rules/black_list.rules

cd /opt/snort/etc/
# out prefix is /opt/snort not /usr/local...
sed -i 's#/usr/local/#/opt/snort/#' snort.conf

# disable all the built in rules
sed -i -r 's,include $RULE_PATH/(.*) ,# include $RULE_PATH/\1,' snort.conf

# enable our local rules
sed -i 's,# include $RULE_PATH/local.rules,include $RULE_PATH/local.rules,' snort.conf

# enable hpfeeds
sed -i "s/# hpfeeds/# hpfeeds\noutput log_hpfeeds: host $HPF_HOST, ident $HPF_IDENT, secret $HPF_SECRET, channel\nsnort.alerts, port $HPF_PORT/" snort.conf

#Set HOME_NET
IP=$(ip -f inet -o addr show $INTERFACE|head -n 1|cut -d\ -f 7 | cut -d/ -f 1)
sed -i "/ipvar HOME_NET/c\ipvar HOME_NET $IP" /opt/snort/etc/snort.conf

# Installing snort rules.
# mhn.rules will be used as local.rules.
rm -f /etc/snort/rules/local.rules
ln -s /opt/mhn/rules/mhn.rules /opt/snort/rules/local.rules

# Supervisor will manage snort-hpfeeds
apt-get install -y supervisor
```



```
# Config for supervisor.
cat > /etc/supervisor/conf.d/snort.conf <<EOF
[program:snort]
command=/opt/snort/bin/snort -c /opt/snort/etc/snort.conf -i $INTERFACE
directory=/opt/snort
stdout_logfile=/var/log/snort.log
stderr_logfile=/var/log/snort.err
autostart=true
autorestart=true
redirect_stderr=true
stopsignal=QUIT
EOF

cat > /etc/cron.daily/update_snort_rules.sh <<EOF
#!/bin/bash
mkdir -p /opt/mhn/rules
rm -f /opt/mhn/rules/mhn.rules.tmp

echo "[`date`] Updating snort signatures ..."
wget $server_url/static/mhn.rules -O /opt/mhn/rules/mhn.rules.tmp && \
    mv /opt/mhn/rules/mhn.rules.tmp /opt/mhn/rules/mhn.rules && \
    (supervisorctl update ; supervisorctl restart snort ) && \
    echo "[`date`] Successfully updated snort signatures" && \
    exit 0

echo "[`date`] Failed to update snort signatures"
exit 1
EOF
chmod 755 /etc/cron.daily/update_snort_rules.sh
/etc/cron.daily/update_snort_rules.sh

supervisorctl update
```



5.2 Glastopf Installation Script

```
#!/bin/bash
set -e
set -x

if [ $# -ne 2 ]
then
    echo "Wrong number of arguments supplied."
    echo "Usage: $0 <server_url> <deploy_key>."
    exit 1
fi

# Check if Ubuntu 18.04 or 16.04
if [ "$(lsb_release -r -s)" != "18.04" ] && [ "$(lsb_release -r -s)" != "16.04" ]; then
    echo "WARNING: This operating system may not be supported by this script."
    echo "Continue? (y/n)"
    read PROMPT
    if [ "$PROMPT" == "n" -o "$PROMPT" == "N" ]
    then
        exit
    fi
fi

server_url=$1
deploy_key=$2
GLASTOPF_HOME=/opt/glastopf

# Update repository
apt-get update

# Install Prerequisites
if [ "$(lsb_release -r -s)" == "14.04" ]; then
    apt-get install -y python2.7 python-openssl python-gevent libevent-dev python2.7-dev build-essential make python-chardet
python-requests python-sqlalchemy python-xml python-beautifulsoup mongodb python-pip python-dev python-setuptools g++ git
php5 php5-dev liblapack-dev gfortran libmysqlclient-dev libxml2-dev libxslt-dev supervisor
else
    apt-get install -y apache2 python2.7 python-openssl python-gevent libevent-dev python2.7-dev build-essential make
python-chardet python-requests python-sqlalchemy python-xml python-beautifulsoup mongodb python-pip python-dev
python-setuptools g++ git php php-dev liblapack-dev gfortran libmysqlclient-dev libxml2-dev libxslt-dev supervisor
fi

pip install -e git+https://github.com/pwnlandia/hpfeeds.git#egg=hpfeeds-dev

# Install and configure the PHP sandbox
cd /opt
git clone git://github.com/mushorg/BFR.git
cd BFR
phpize
./configure --enable-bfr
make && make install
```



```
# Updated php.ini to add bfr.so
if [ "$(lsb_release -r -s)" == "14.04" ]; then
    BFR_BUILD_OUTPUT=`find /usr/lib/php5/ -type f -name "bfr.so" | awk -F"/" '{print $5}'`
    echo "zend_extension = /usr/lib/php5/$BFR_BUILD_OUTPUT/bfr.so" >> /etc/php5/apache2/php.ini
else
    BFR_BUILD_OUTPUT=`find /usr/lib/php/ -type f -name "bfr.so" | awk -F"/" '{print $5}'`
    echo "zend_extension = /usr/lib/php/$BFR_BUILD_OUTPUT/bfr.so" >> /etc/php/7.0/fpm/php.ini
fi

# Stop apache2 and disable it from start up
service apache2 stop
update-rc.d -f apache2 remove

# Upgrade python-greenlet
pip install --upgrade greenlet

# Install glastopf
pip install --upgrade pgen
pip install --upgrade cython
pip uninstall --yes setuptools
git clone https://github.com/mushorg/glastopf.git $GLASTOPF_HOME
cd $GLASTOPF_HOME
python setup.py install

# Register the sensor with the MHN server.
wget $server_url/static/registration.txt -O registration.sh
chmod 0755 registration.sh
# Note: This will export the HPF_* variables
./registration.sh $server_url $deploy_key "glastopf"

# Add the modified glastopf.cfg
cat > $GLASTOPF_HOME/glastopf.cfg <<EOF
[webserver]
host = 0.0.0.0
port = 80
uid = nobody
gid = nogroup
proxy_enabled = False

[ssl]
enabled = False
certfile =
keyfile =

#Generic logging for general monitoring
[logging]
consolelog_enabled = False
filelog_enabled = True
logfile = log/glastopf.log

[dork-db]
enabled = True
pattern = rfi
```



```
#Extracts dorks from a online dorks service operated by The Honeynet Project
mnem_service = True
```

```
[hpfeed]
enabled = True
host = $HPF_HOST
port = $HPF_PORT
secret = $HPF_SECRET
# channels comma separated
chan_events = glastopf.events
chan_files = glastopf.files
ident = $HPF_IDENT
```

```
[main-database]
#If disabled a sqlite database will be created (db/glastopf.db)
#to be used as dork storage.
enabled = True
#mongodb or sqlalchemy connection string, ex:
#mongodb://localhost:27017/glastopf
#mongodb://james:bond@localhost:27017/glastopf
#mysql://james:bond@somehost.com/glastopf
connection_string = sqlite:///db/glastopf.db
```

```
[surfcertids]
enabled = False
host = localhost
port = 5432
user =
password =
database = idserver
```

```
[syslog]
enabled = False
socket = /dev/log
```

```
[mail]
enabled = False
# an email notification will be sent only if a specified matched pattern is identified.
# Use the wildcard char *, to be notified every time
patterns = rfi,lfi
user =
pwd =
mail_from =
mail_to =
smtp_host = smtp.gmail.com
smtp_port = 587
```

```
[taxii]
enabled = False
host = taxiitest.mitre.org
port = 80
inbox_path = /services/inbox/default/
use_https = False
use_auth_basic = False
```



```
auth_basic_username = your_username
auth_basic_password = your_password
use_auth_certificate = False
auth_certificate_keyfile = full_path_to_keyfile
auth_certificate_certfile = full_path_to_certfile
include_contact_info = False
contact_name = ...
contact_email = ...

[logstash]
enabled = False
host = localhost
port = 5659
handler = AMQP/TCP/UDP

[misc]
# set webserver banner
banner = Apache/2.0.48

[surface]
#https://www.google.com/webmasters/
google_meta =
#http://www.bing.com/toolbox/webmaster
bing_meta =

[sensor]
sensorid = None
[profiler]
enabled = False

[s3storage]
enabled = False
endpoint = http://localhost:8080/
aws_access_key_id = YOUR_aws_access_key_id
aws_secret_access_key = YOUR_aws_access_key_id
bucket = glastopf
region = eu-west-1
signature_version = s3
EOF

# Set up supervisor
cat > /etc/supervisor/conf.d/glastopf.conf <<EOF
[program:glastopf]
command=/usr/bin/python /usr/local/bin/glastopf-runner
directory=$GLASTOPF_HOME
stdout_logfile=/var/log/glastopf.out
stderr_logfile=/var/log/glastopf.err
autostart=true
autorestart=true
redirect_stderr=true
stopsignal=QUIT
EOF
supervisorctl update
supervisorctl restart all
```



5.3 Cowrie Installation Script

```
#!/bin/bash
set -e
set -x

if [ $# -ne 2 ]
then
    echo "Wrong number of arguments supplied."
    echo "Usage: $0 <server_url> <deploy_key>."
    exit 1
fi

apt-get update
apt-get install -y python
server_url=$1
deploy_key=$2

apt-get update
apt-get -y install python-dev git supervisor authbind openssl python-virtualenv build-essential python-gmpy2 libgmp-dev libmpfr-dev
libmpc-dev libssl-dev python-pip libffi-dev

pip install -U supervisor
/etc/init.d/supervisor start || true

sed -i 's/#Port/Port/g' /etc/ssh/sshd_config
sed -i 's/Port 22$/Port 2222/g' /etc/ssh/sshd_config
service ssh restart
useradd -d /home/cowrie -s /bin/bash -m cowrie -g users
cd /opt
git clone https://github.com/micheloosterhof/cowrie.git cowrie
cd cowrie

# Most recent known working version
git checkout 34f8464

# Config for requirements.txt
cat > /opt/cowrie/requirements.txt <<EOF
twisted>=17.1.0
cryptography>=2.1
configparser
pyopenssl
pyparsing
packaging
appdirs>=1.4.0
pyasn1_modules
attrs
service_identity
python-dateutil
tftpy
bcrypt
EOF
```



```
virtualenv cowrie-env #env name has changed to cowrie-env on latest version of cowrie
source cowrie-env/bin/activate
# without the following, i get this error:
# Could not find a version that satisfies the requirement csirtgsdk (from -r requirements.txt (line 10)) (from versions: 0.0.0a5,
0.0.0a6, 0.0.0a5.linux-x86_64, 0.0.0a6.linux-x86_64, 0.0.0a3)
pip install csirtgsdk==0.0.0a6
pip install -r requirements.txt

# Register sensor with MHN server.
wget $server_url/static/registration.txt -O registration.sh
chmod 755 registration.sh
# Note: this will export the HPF_* variables
./registration.sh $server_url $deploy_key "cowrie"

cd etc
cp cowrie.cfg.dist cowrie.cfg
sed -i 's/hostname = svr04/hostname = server/g' cowrie.cfg
sed -i 's/listen_endpoints = tcp:2222:interface=0.0.0.0/listen_endpoints = tcp:22:interface=0.0.0.0/g' cowrie.cfg
sed -i 's/version = SSH-2.0-OpenSSH_6.0p1 Debian-4+deb7u2/version = SSH-2.0-OpenSSH_6.7p1 Ubuntu-5ubuntu1.3/g'
cowrie.cfg
sed -i 's/#[output_hpfeeds]/[output_hpfeeds]/g' cowrie.cfg
sed -i 's/#[output_hpfeeds]/!b;n;cenabled = true' cowrie.cfg
sed -i 's/#server = hpfeeds.mysite.org/server = $HPF_HOST/g' cowrie.cfg
sed -i 's/#port = 10000/port = $HPF_PORT/g' cowrie.cfg
sed -i 's/#identifier = abc123/identifier = $HPF_IDENT/g' cowrie.cfg
sed -i 's/#secret = secret/secret = $HPF_SECRET/g' cowrie.cfg
sed -i 's/#debug=false/debug=false/' cowrie.cfg
cd ..

chown -R cowrie:users /opt/cowrie/
touch /etc/authbind/byport/22
chown cowrie /etc/authbind/byport/22
chmod 770 /etc/authbind/byport/22

# start.sh is deprecated on new Cowrie version and substituted by "bin/cowrie [start/stop/status]"
sed -i 's/AUTHBIND_ENABLED=no/AUTHBIND_ENABLED=yes/' bin/cowrie
sed -i 's/DAEMONIZE=""/DAEMONIZE="-n/" bin/cowrie

# Config for supervisor
cat > /etc/supervisor/conf.d/cowrie.conf <<EOF
[program:cowrie]
command=/opt/cowrie/bin/cowrie start
directory=/opt/cowrie
stdout_logfile=/opt/cowrie/var/log/cowrie/cowrie.out
stderr_logfile=/opt/cowrie/var/log/cowrie/cowrie.err
autostart=true
autorestart=true
stopasgroup=true
killasgroup=true
user=cowrie
EOF

supervisorctl update
```




5.4 p0f Installation Script

```
#!/bin/bash
INTERFACE=$(basename -a /sys/class/net/e*)
set -e
set -x

if [ $# -ne 2 ]
then
    if [ $# -eq 3 ]
    then
        INTERFACE=$3
    else
        echo "Wrong number of arguments supplied."
        echo "Usage: $0 <server_url> <deploy_key>."
        exit 1
    fi
fi

compareint=$(echo "$INTERFACE" | wc -w)

if [ "$INTERFACE" = "e*" ] || [ "$compareint" -ne 1 ]
then
    echo "No Interface selectable, please provide manually."
    echo "Usage: $0 <server_url> <deploy_key> <INTERFACE>"
    exit 1
fi

server_url=$1
deploy_key=$2

apt update
apt install -y git supervisor libpcap-dev libjansson-dev gcc

# install p0f
cd /opt
git clone https://github.com/threatstream/p0f.git
cd p0f
git checkout origin/hpfeeds
./build.sh
useradd -d /var/empty/p0f -M -r -s /bin/nologin p0f-user || true
mkdir -p -m 755 /var/empty/p0f

# Register the sensor with the MHN server.
wget $server_url/static/registration.txt -O registration.sh
chmod 755 registration.sh
# Note: this will export the HPF_* variables
./registration.sh $server_url $deploy_key "p0f"
```



```
# Note: This will change the interface and the ip in the p0f config
sed -i "/INTERFACE=/c\INTERFACE=$INTERFACE" /opt/p0f/p0f_wrapper.sh
sed -i "/MY_ADDRESS=/c\MY_ADDRESS=\$(ip -f inet -o addr show \${INTERFACE}|head -n 1|cut -d\\  -f 7 | cut -d/ -f 1)"
/opt/p0f/p0f_wrapper.sh

cat > /etc/supervisor/conf.d/p0f.conf <<EOF
[program:p0f]
command=/opt/p0f/p0f_wrapper.sh
directory=/opt/p0f
stdout_logfile=/var/log/p0f.out
stderr_logfile=/var/log/p0f.err
autostart=true
autorestart=true
redirect_stderr=true
stopsignal=TERM
environment=HPFEEDS_HOST="$HPF_HOST",HPFEEDS_PORT="$HPF_PORT",HPFEEDS_CHANNEL="p0f.events",HPFEEDS
_IDENT="$HPF_IDENT",HPFEEDS_SECRET="$HPF_SECRET"
EOF

supervisorctl update
```



5.5 Dionaea Installation Script

```
#!/bin/bash
set -e
set -x

if [ $# -ne 2 ]
then
    echo "Wrong number of arguments supplied."
    echo "Usage: $0 <server_url> <deploy_key>."
    exit 1
fi

server_url=$1
deploy_key=$2

# Install dependencies
apt update
apt --yes install \
    git \
    supervisor \
    build-essential \
    cmake \
    check \
    cython3 \
    libcurl4-openssl-dev \
    libemu-dev \
    libev-dev \
    libglib2.0-dev \
    libloudmouth1-dev \
    libnetfilter-queue-dev \
    libnl-3-dev \
    libpcap-dev \
    libssl-dev \
    libtool \
    libudns-dev \
    python3 \
    python3-dev \
    python3-bson \
    python3-yaml \
    python3-boto3

git clone https://github.com/DinoTools/dionaea.git
cd dionaea

# Latest tested version with this install script
git checkout baf25d6
```



```
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX:PATH=/opt/dionaea ..
```

```
make
make install
```

```
wget $server_url/static/registration.txt -O registration.sh
chmod 755 registration.sh
# Note: this will export the HPF_* variables
. ./registration.sh $server_url $deploy_key "dionaea"
```

```
cat > /opt/dionaea/etc/dionaea/ihandlers-enabled/hpfeeds.yaml <<EOF
- name: hpfeeds
  config:
    # fqdn/ip and port of the hpfeeds broker
    server: "$HPF_HOST"
    # port: $HPF_PORT
    ident: "$HPF_IDENT"
    secret: "$HPF_SECRET"
    # dynip_resolve: enable to lookup the sensor ip through a webservice
    dynip_resolve: "http://canhazip.com/"
    # Try to reconnect after N seconds if disconnected from hpfeeds broker
    # reconnect_timeout: 10.0
EOF
```

```
# Editing configuration for Dionaea.
mkdir -p /opt/dionaea/var/log/dionaea/wwwroot /opt/dionaea/var/log/dionaea/binaries /opt/dionaea/var/log/dionaea/log
chown -R nobody:nogroup /opt/dionaea/var/log/dionaea
```

```
mkdir -p /opt/dionaea/var/log/dionaea/bistreams
chown nobody:nogroup /opt/dionaea/var/log/dionaea/bistreams
```

```
# Config for supervisor.
cat > /etc/supervisor/conf.d/dionaea.conf <<EOF
[program:dionaea]
command=/opt/dionaea/bin/dionaea -c /opt/dionaea/etc/dionaea/dionaea.cfg
directory=/opt/dionaea/
stdout_logfile=/opt/dionaea/var/log/dionaea.out
stderr_logfile=/opt/dionaea/var/log/dionaea.err
autostart=true
autorestart=true
redirect_stderr=true
stopsignal=QUIT
EOF
```

```
supervisorctl update
```