### Posibles fases de la práctica

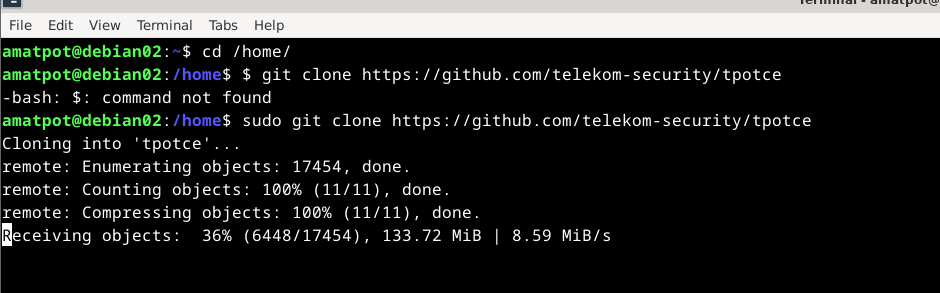
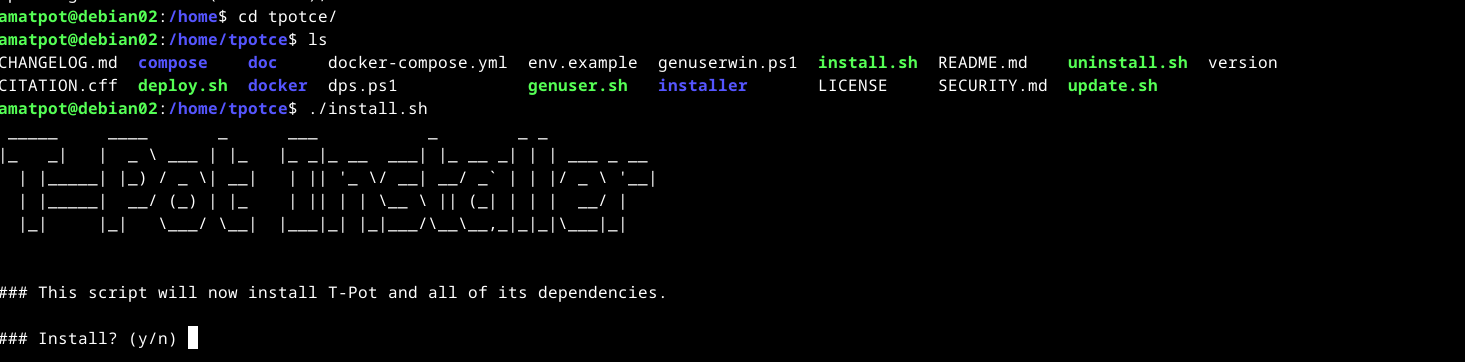
1. Montar Tpot de manera local para identificar configuraciones
2. Definir Honeypot a desplegar
3. Configurar de manera integral la implementación
4. Alta de servidores – configuraciones
5. Replica de configuraciones
6. Despliegue (Definir duración del Tpot online)
7. Primer avance: definir primer corte para revisar datos, primer progreso. Revisamos el tiempo restante.
8. Confeccionamos informe con el material previo
9. Cierre de T-pot en servidor
10. Recolectamos información, tableros, dashboard
11. Preparación informe ejecutivo, técnico
12. Ensayamos exposición

### Instalación Tpot – Local

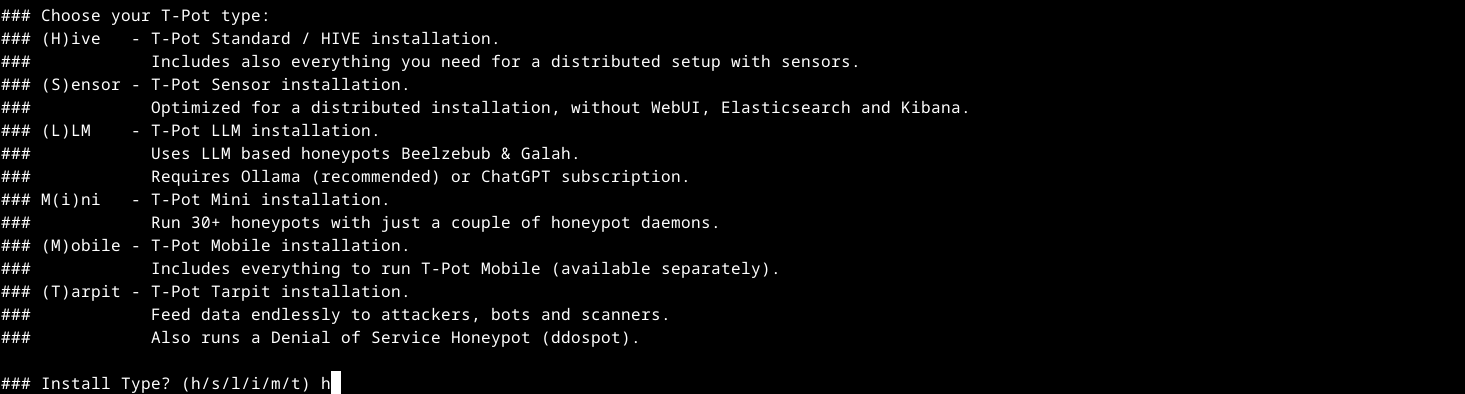
Instalamos de manera local la herramienta sobre la distribución Debian12. Nos vamos a respaldar del repositorio oficial <https://github.com/telekom-security/tpotce> (actualizado el 11.12.2024).

Creamos el usuario ordinario amatpot con permisos de sudo en Debian. Desde la ubicación $HOME clonaremos el repositorio, nos ubicamos en el directorio y lanzamos el instalador

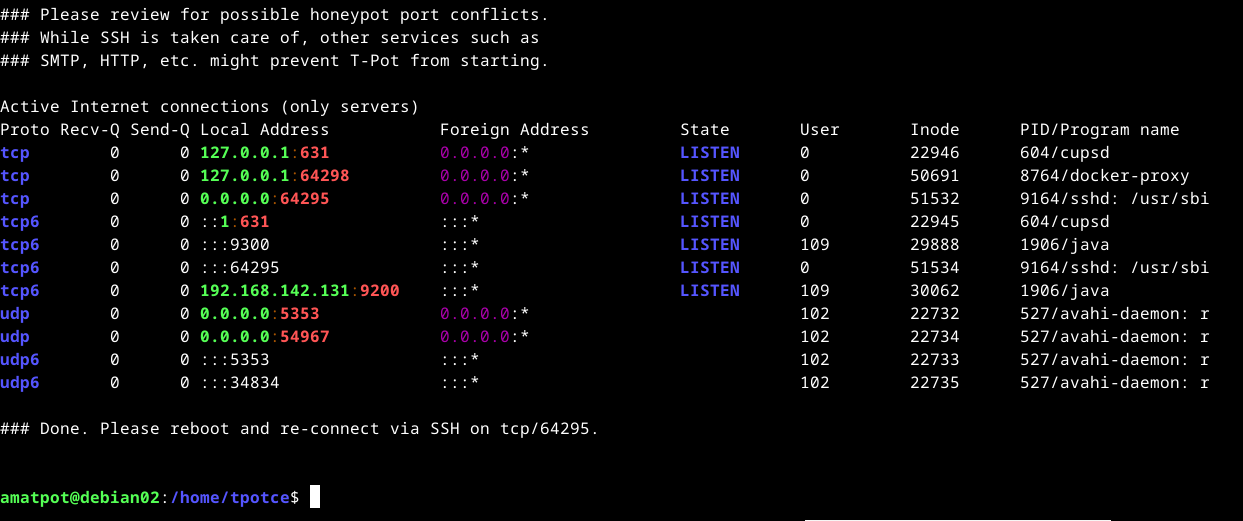
$ git clone <https://github.com/telekom-security/tpotce>  
$ cd tpotce  
$ ./install.sh

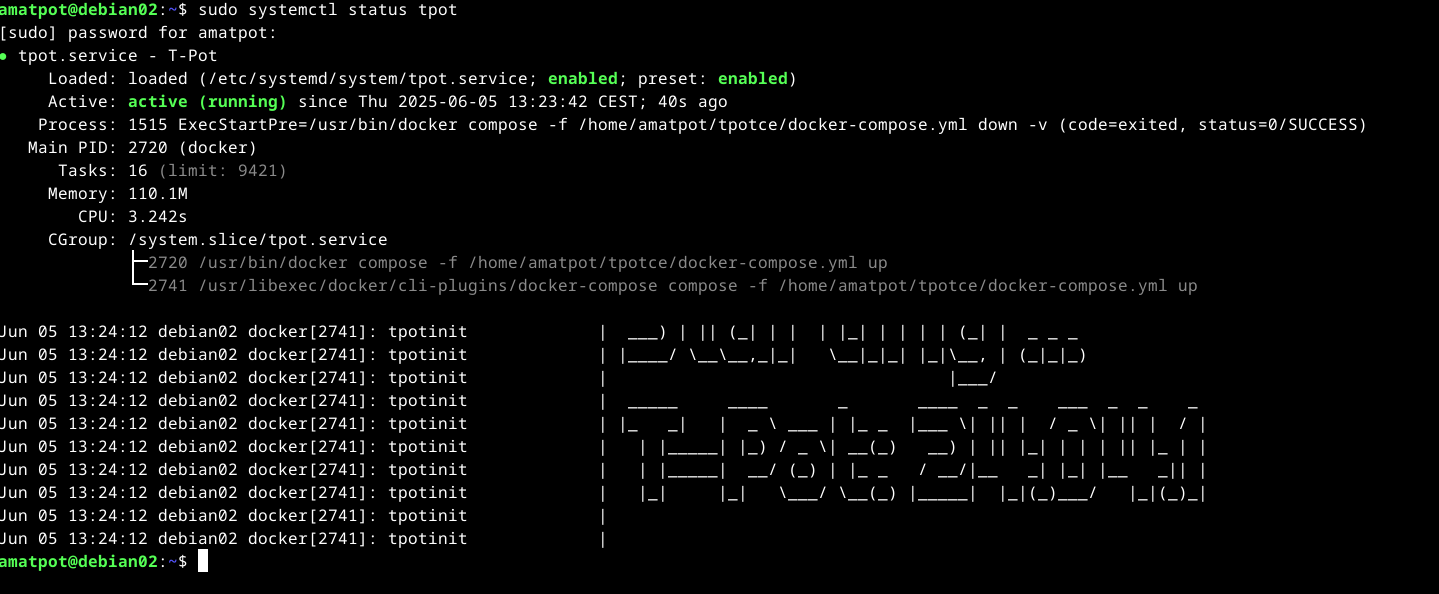
 

Instalaremos la T-Pot Standard / HIVE installation.



Asignaremos el user amatpot y el pass amatpot. Este tipo de credenciales tienen un nivel bajo de protección ya que solo haremos un acercamiento local a la herramienta y configuraciones antes del despliegue en el servidor.





Nota: instalar Kibata, instalar Elastic

Configurar los .yml de cada herramienta, puerto 9200 – iplocal

### Configuraciones Honeypot

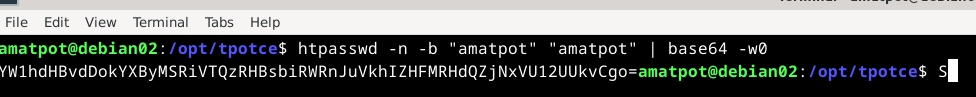
Ensayamos las configuraciones de los Honeypot a utilizar, nos respaldaremos en el repositorio <https://github.com/telekom-security/tpotce>. Haremos uso de los siguientes elementos (explicar motivos y enfoque de la elección):

* Cowrie
* Maps
* Dionaea
* Conpot
* Elasticpot
* Elasticsearch
* Kibana
* Mailoney
* Honeytrap
* Suricata
* DDoSPot
* WordPot
* Spiderfoot

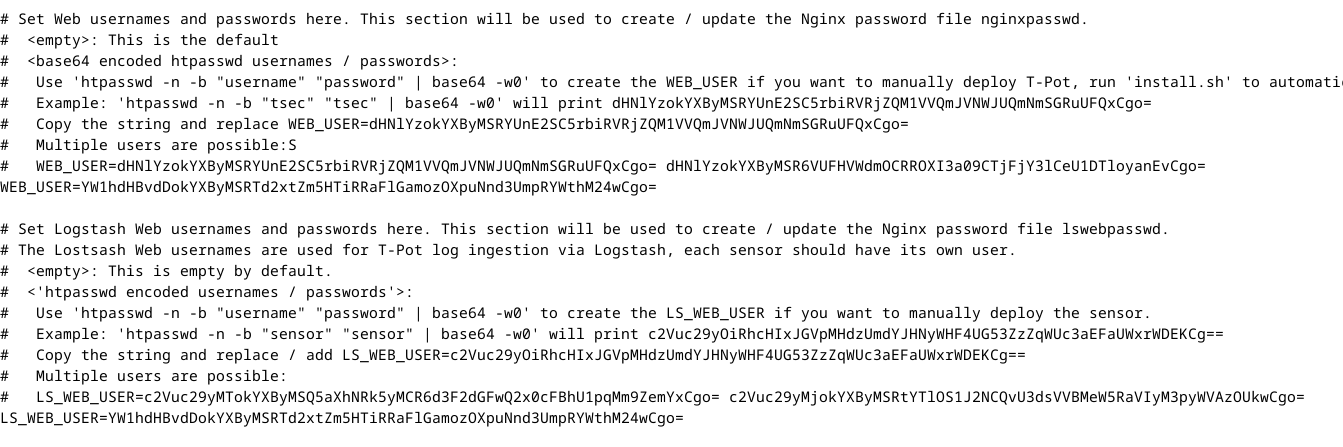
Previamente a las configuraciones, debemos definir el usuario WEB\_USER y LS\_WEB\_USER en el fichero oculto ~/tpotce/.env. Desde consola creamos el user y pass (utilizamos el mismo user para ambos usuarios):

(Todas las configuraciones se realizan con usuario ordinario)

htpasswd -n -b "amatpot" "amatpot" | base64 -w0



Reemplazamos la información en el fichero mencionado.



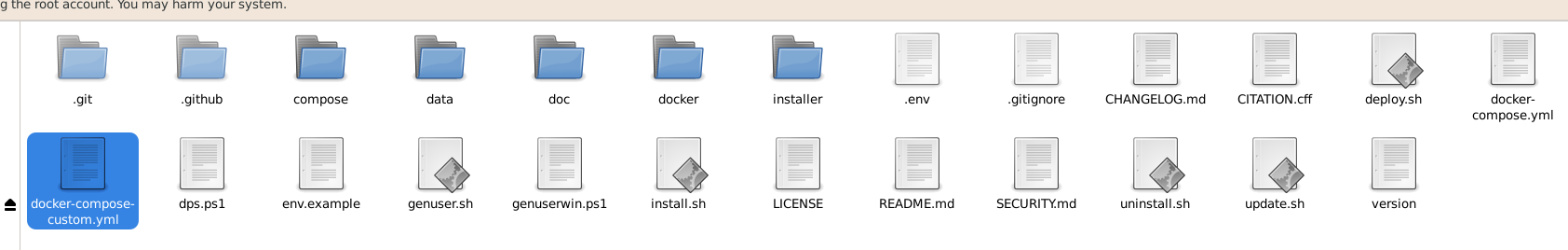
Ya realizada la configuración, lanzaremos desde el directorio ~/tpotce/compose el script customizer.py

sudo python3 customizer.py

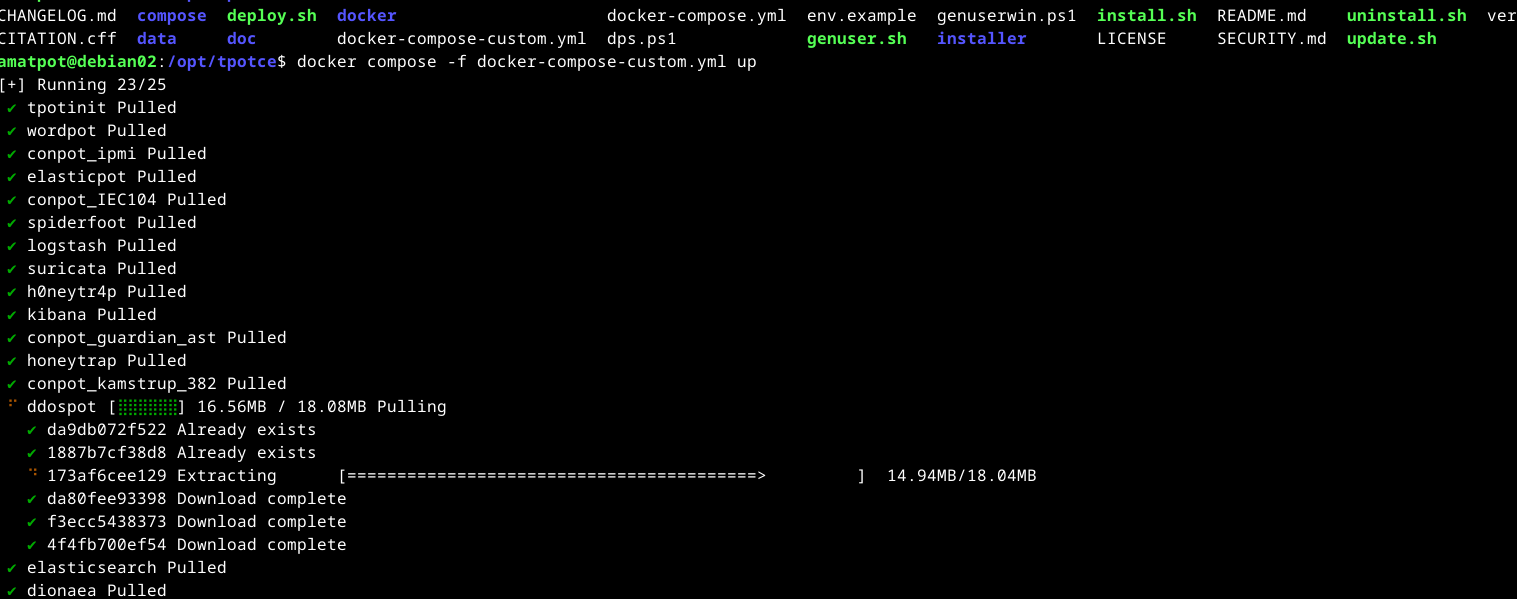


Ya creado el fichero customizado, lo copiamos en el directorio ~/tpotce

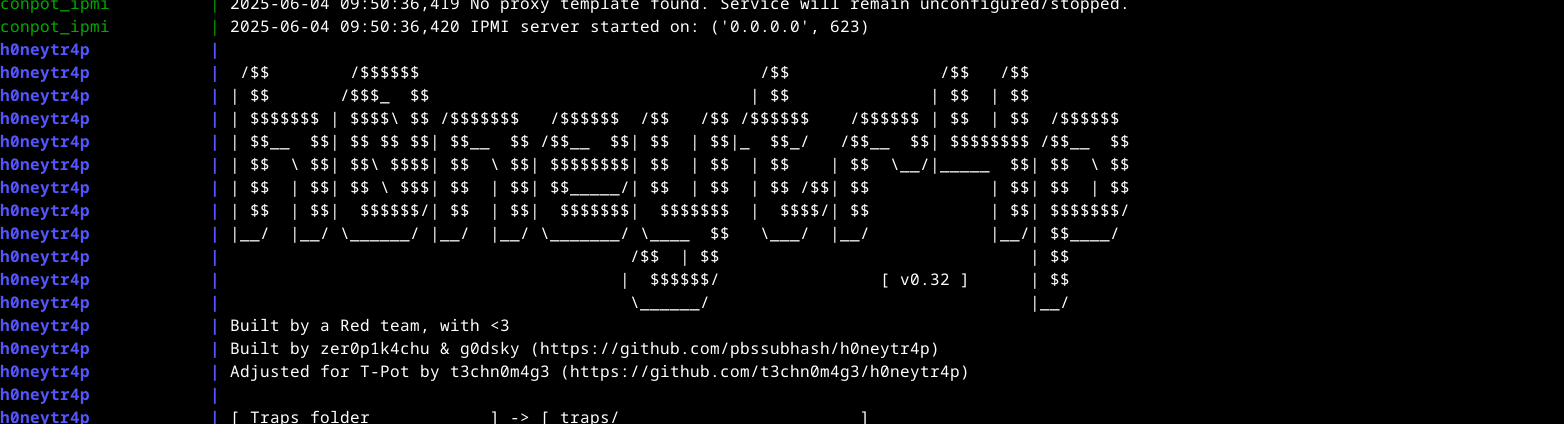
sudo cp docker-compose-custom.yml ~/tpotce/

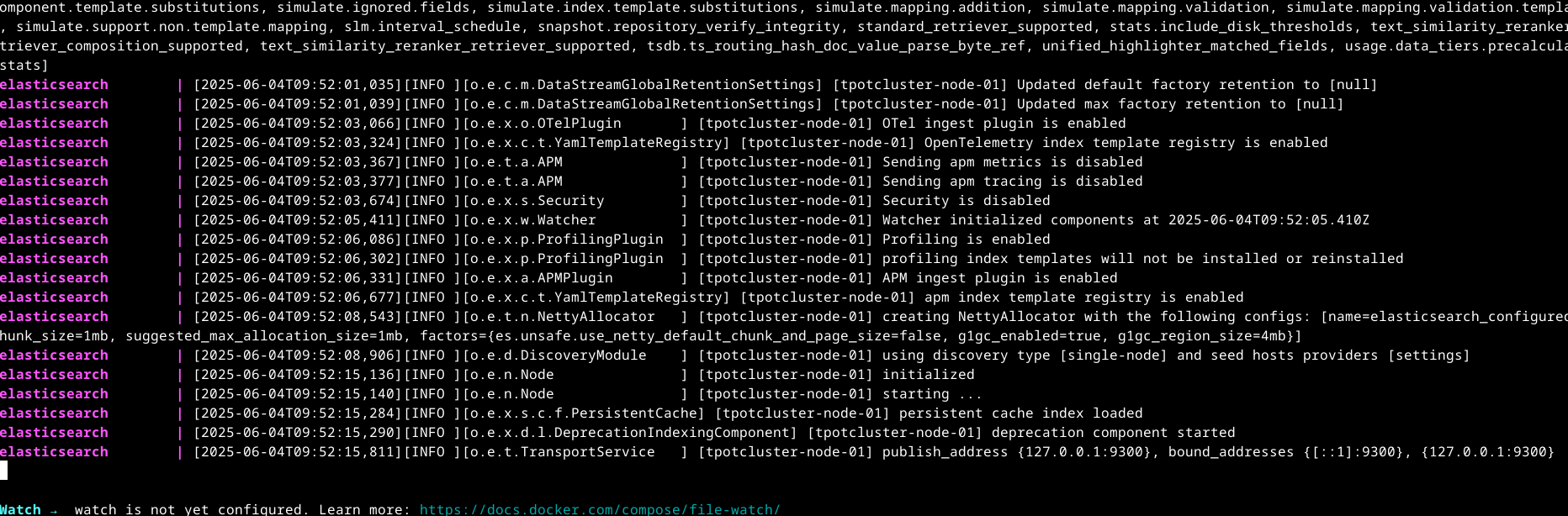


Nos posicionamos en el directorio donde colocamos el fichero y lanzamos el comando docker compose -f docker-compose-custom.yml up para revisar que funciona de manera correcta



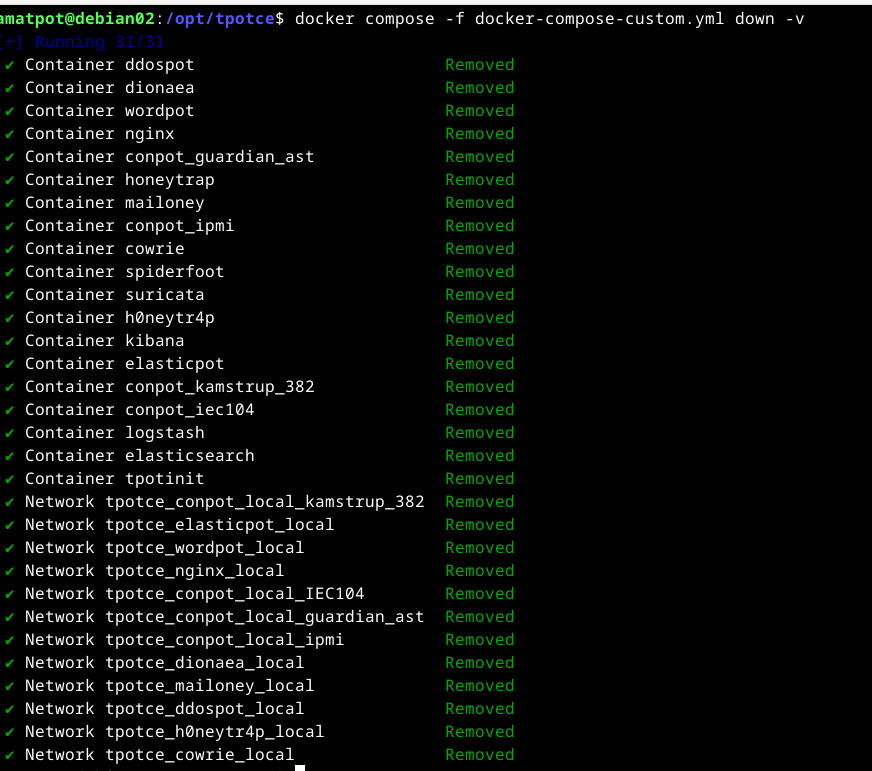


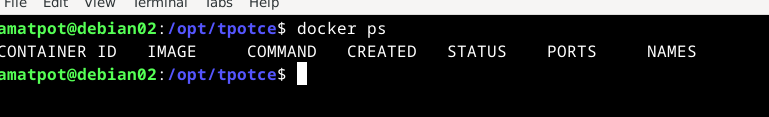




Docker irá lanzando el contenedor de cada Honeypot seleccionado. Considerando que las modificaciones se realizaron de manera correcta, interrumpimos el proceso y detenemos los contenedores con el comando

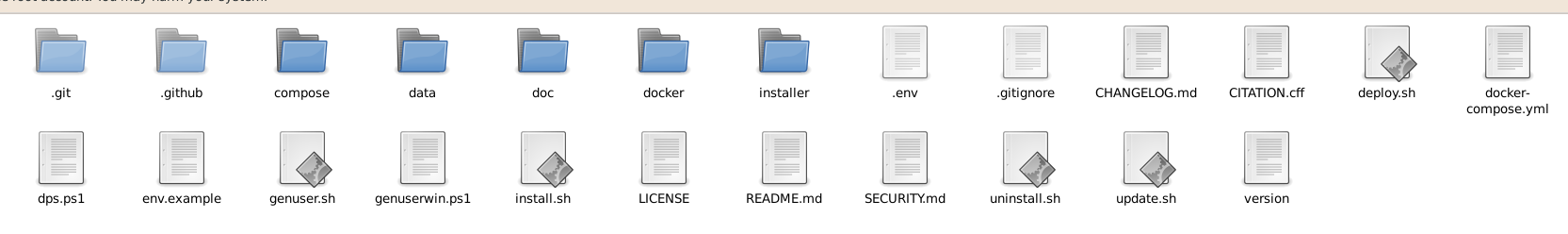
docker compose -f docker-compose-custom.yml down -v



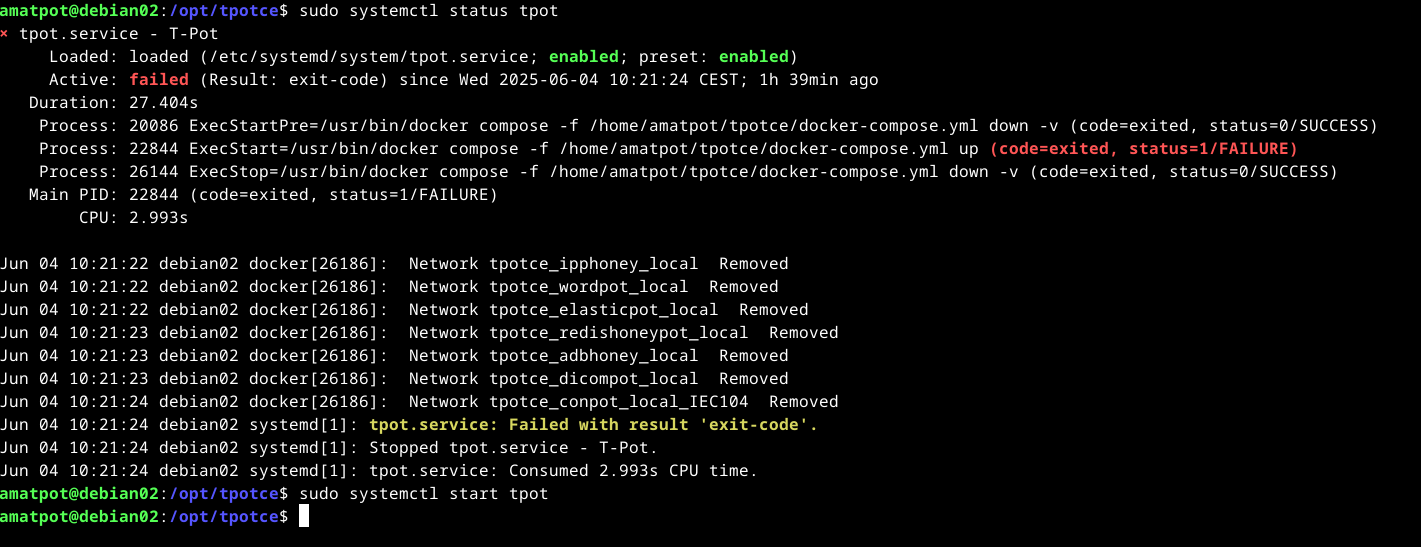


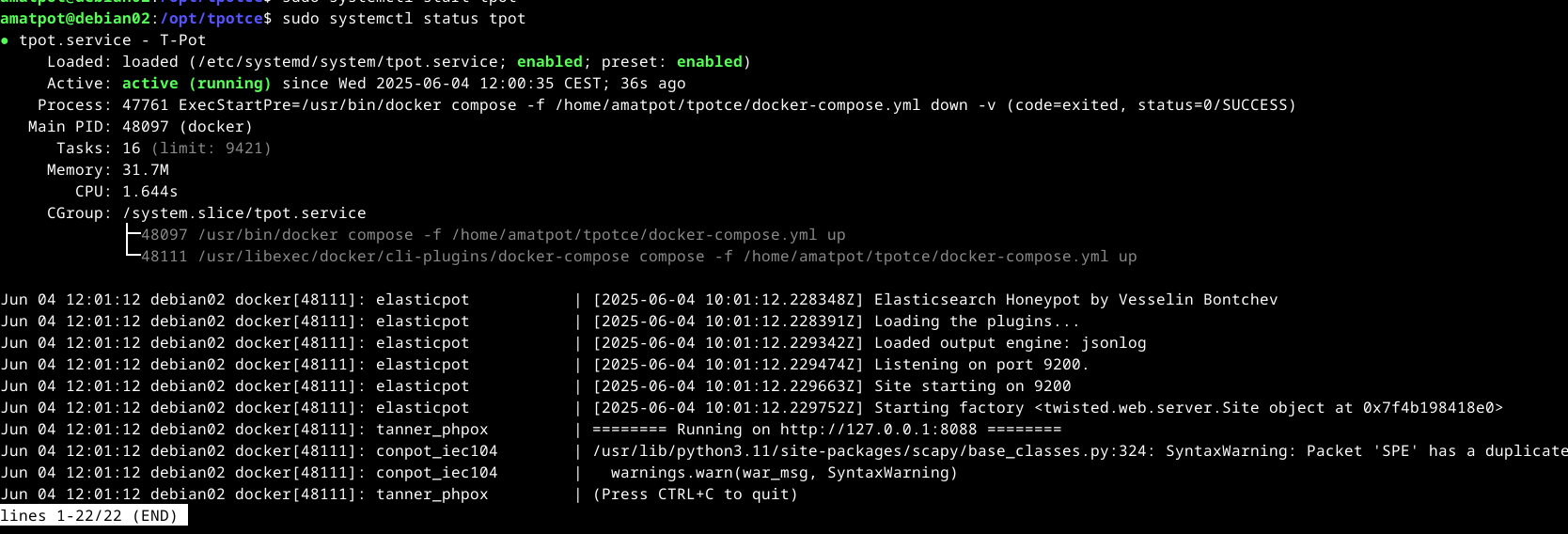
En este punto reemplazamos el fichero docker-compose.yml por las configuraciones personalizadas.

sudo mv docker-compose-custom.yml docker-compose.yml

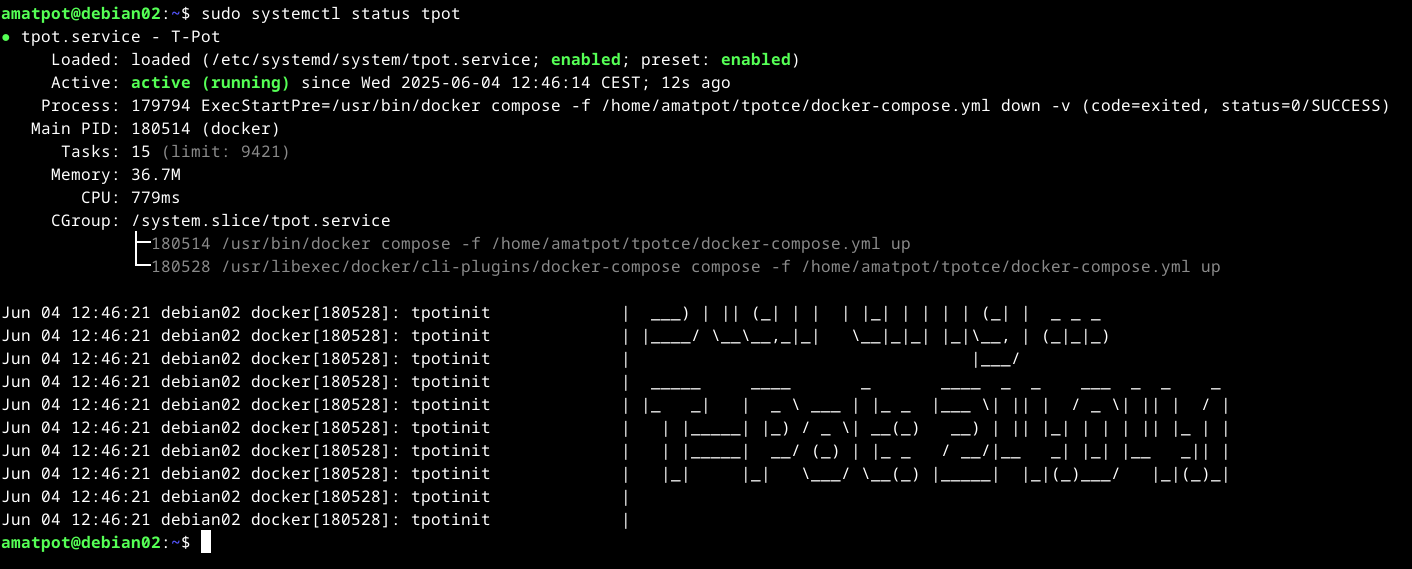


Validamos el estado actual de Tpot y lo iniciamos



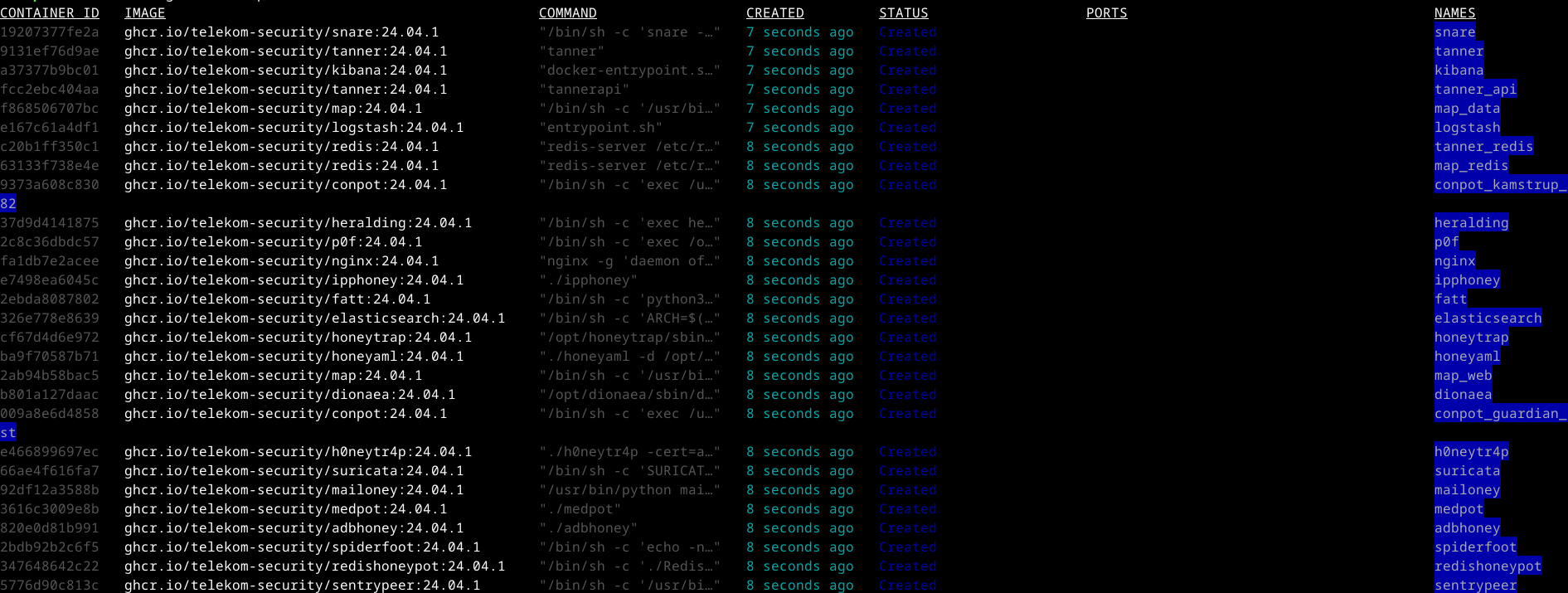


Realizamos un reinicio del sistema para el impacto correcto de las configuraciones sudo reboot

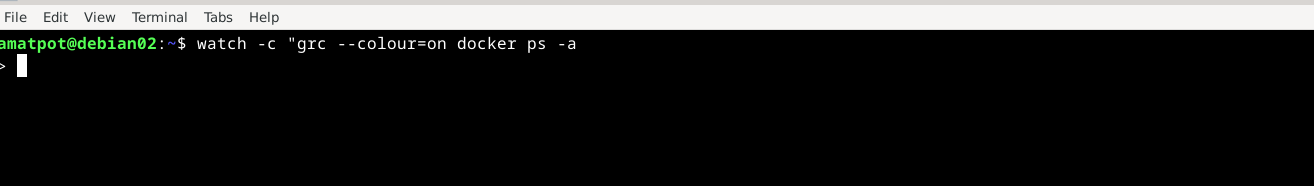


Una vez activo Tpot, podremos revisar los contenedores activos con el comando

grc docker ps -a

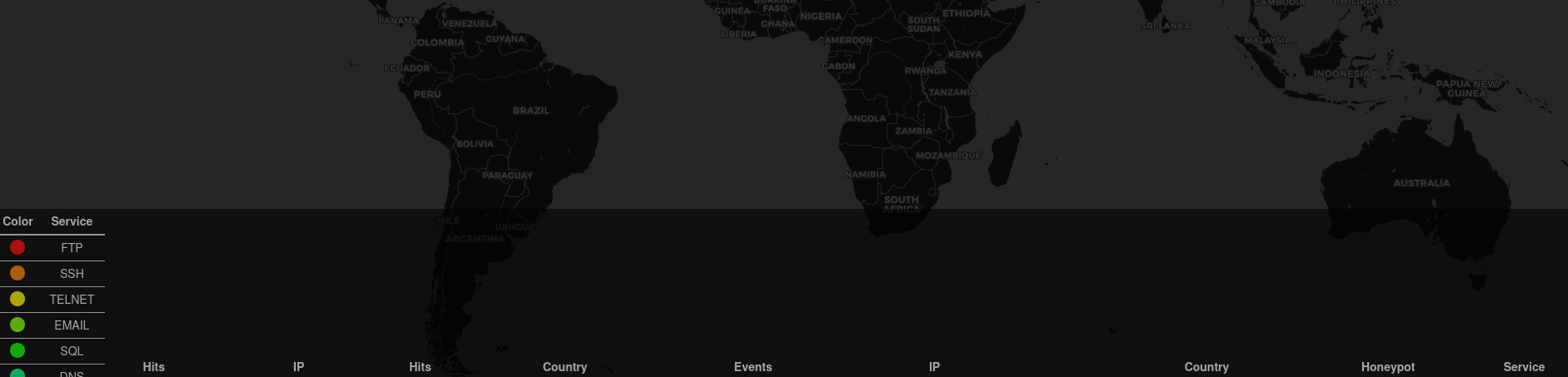


También podremos monitorizar en directo con watch -c "grc --colour=on docker ps -a



De manera local, desde el navegador, podremos acceder al apartado Web desde el siguiente enlace https://192.168.142.131:64297/ (ip local)





**Configuraciones en servidor AWS (primera instancia)**

Abordamos las configuraciones del servidor AWS con los recursos recomendados (16GB ram + 150GB almacenamiento)

Generamos un par de claves únicas para el acceso SSH de manera externa.

Nos conectamos al servidor con las credenciales que ofrece AWS, replicamos la instalación y configuraciones que realizamos de manera local de las herramientas y sistema. Reiniciamos Debían, y modificamos los puertos en la instancia.

64295 > para conectarnos por ssh y realizar las modificaciones y ajustes correspondientes, monitoreo de sistema

64297 > para acceder a la interfaz web de Tpot, revisar dashboard, análisis de métricas

0-64000 > exponemos Tpot (multiples Honeypot)

ICMP > para validar la conexión externa de la IP pública asignada



**Configuraciones en servidor AWS (segunda instancia)**

Luego de haber realizado las configuraciones correspondientes, haber definido los Hoyneypot que lanzaremos en el primer ensayo, nos percatamos que al lanzar Tpot, la herramienta presenta intermitencias en la conexión. Dejando sin utilidad la instancia. Revisando la demanda de la herramienta y la configuración actual, decidimos probar reduciendo el Q de Honeypot notando un mejor rendimiento de la instancia. En este punto decidimos modificar los recursos del servidos y llegarlo a 32GB de RAM.



Con el cambio ya realizado, definimos los Honeypot que vamos a utilizar en la práctica:

* **Cowrie**: Honeypot de SSH y Telnet diseñado para registrar ataques de fuerza bruta y la interacción de los atacantes con el sistema.
* **Logstash**: Aunque no es un honeypot en sí, es una herramienta de procesamiento de logs que puede integrarse con honeypots para analizar eventos de seguridad.
* **DDoSPot**: Honeypot especializado en la detección de ataques DDoS basados en UDP, simulando servicios como DNS, NTP y SSDP.
* **Dionaea**: Honeypot diseñado para capturar malware, emulando servicios vulnerables como SMB, HTTP, FTP y MSSQL.
* **ElasticPot**: Honeypot que simula un servidor Elasticsearch vulnerable, utilizado para detectar ataques dirigidos a bases de datos abiertas.
* **Endlessh**: Honeypot de SSH que ralentiza los ataques de fuerza bruta al enviar un banner interminable, haciendo perder tiempo a los atacantes.
* **Heralding**: Honeypot de captura de credenciales que simula múltiples servicios como FTP, SSH, HTTP, SMTP y VNC para registrar intentos de acceso.
* **Honeytrap**: Framework avanzado de honeypots que permite combinar múltiples servicios en un solo sistema para atraer atacantes.
* **Mailoney**: Honeypot de SMTP diseñado para capturar intentos de acceso no autorizado y ataques de spam.
* **Tanner**: Sistema de análisis que trabaja junto con Snare para evaluar solicitudes HTTP y emular vulnerabilidades en aplicaciones web.
* **Suricata**: No es un honeypot, sino un IDS/IPS que analiza tráfico de red en busca de amenazas.
* **Elasticsearch**: En el contexto de honeypots, se usa para almacenar y analizar datos de ataques registrados por otros honeypots.
* **Kibana**: Herramienta de visualización de datos que se integra con honeypots para analizar patrones de ataque.
* **Nginx Honeypot**: Configuración de Nginx que bloquea bots maliciosos al detectar intentos de acceso a rutas sospechosas.
* **SpiderFoot**: No es un honeypot, sino una herramienta de OSINT que recopila información sobre amenazas y posibles atacantes.
* **Snare**: Honeypot de aplicaciones web que simula vulnerabilidades y analiza ataques dirigidos a sitios web.

Análisis Spiderfoot