Proyecto Final ASIR

Institut Puig Castellar

HONEYPOTSEC

MONITORIZA Y APRENDE

Autores: Sergio Fermiñan y Gerard Martinez

ÍNDICE

1. Introducción

1.1. Que es HoneypotSEC

HoneypotSEC es una plataforma integral de ciberseguridad diseñada para transformar la defensa de red mediante el uso estratégico de sistemas de engaño. En su esencia, el proyecto consiste en una infraestructura de señuelos (honeypots) multiprotocolo que actúan como trampas digitales para atraer y analizar el comportamiento de posibles atacantes en un entorno controlado.

Esta solución se diferencia de los sistemas tradicionales por su enfoque híbrido, que abarca los siguientes puntos:

- **Infraestructura Técnica:** Se basa en la migración y el despliegue de una red de señuelos capaz de emular diversos protocolos, centralizando toda la actividad capturada en un Dashboard accesible vía web.

- **Modelo de Negocio SaaS:** El proyecto plantea la evolución de esta tecnología hacia un modelo de Software as a Service (SaaS) orientado al mercado B2B, permitiendo a las organizaciones monitorizar amenazas en tiempo real sin necesidad de gestionar infraestructuras complejas.

- **Capacitación Interactiva:** Su principal valor diferencial radica en la formación corporativa. La plataforma utiliza la inteligencia de amenazas y los ataques reales capturados por los honeypots para generar simulaciones, cuestionarios y rutas de aprendizaje. De este modo, las empresas pueden utilizar la misma herramienta de monitorización para concienciar a sus empleados frente a las técnicas de cibercrimen más recientes.

En conclusión, **HoneypotSEC** va más allá de la monitorización básica; es una plataforma que combina la detección de amenazas con el aprendizaje organizacional para reforzar la postura de seguridad frente al cibercrimen.

1.2. Motivación y Objetivo

La motivación para desarrollar HoneypotSEC nace, principalmente, de nuestra pasión por la ciberseguridad, un ámbito en el que aspiramos a desarrollarnos a nivel profesional. Como estudiantes de segundo curso del CFGS de Administración de Sistemas Informáticos en Red (ASIR) en Santa Coloma de Gramenet, contamos con una base técnica sólida en redes y sistemas. Por ello, consideramos que este Proyecto de Fin de Ciclo representa el escenario ideal para profundizar en la seguridad defensiva y aplicar nuestros conocimientos sobre infraestructuras reales y complejas.

A esta inquietud técnica se suma una motivación de carácter estratégico y social: la creciente sofisticación de los ciberataques y la constante exposición de las pymes frente a estas amenazas. En la gran mayoría de las brechas de seguridad, el eslabón más débil sigue siendo el factor humano. Los empleados que carecen de una formación continuada a menudo son víctimas de técnicas de phishing e ingeniería social, logrando comprometer la red de toda su organización.

En este contexto, el proyecto persigue un doble objetivo fundamental:

- **Objetivo Técnico:** Diseñar una arquitectura robusta y escalable que permita migrar un entorno local de *honeypots* hacia un servidor VPS en la nube (**Hostinger KVM2**). Se busca gestionar múltiples servicios simulados (SSH, SMB, HTTP, etc.) mediante contenedores **Docker**, garantizando la seguridad mediante accesos perimetrales estrictos y claves criptográficas.

- **Objetivo Comercial y Educativo:** Transformar la inteligencia de amenazas generada por los señuelos en un producto bajo modelo **SaaS**. El fin es ofrecer a las empresas una herramienta que no solo audite ataques, sino que utilice esos datos reales en un módulo de gamificación para educar a su personal de manera práctica y efectiva.

1.3. Alcance y Limitaciones

Alcance del proyecto:

- **Infraestructura y Hardening:** Configuración, securización y aislamiento de un servidor en la nube (Hostinger KVM2). Esto incluye la implementación de políticas de cortafuegos (firewall) y la restricción de acceso administrativo de forma exclusiva mediante pares de claves criptográficas SSH, garantizando un entorno base seguro para alojar el sistema.

- **Despliegue de Sensores (Honeypots):** Instalación y orquestación de seis tipos de sensores especializados (Cowrie, Dionaea, Glastopf, Conpot, Honeytrap y Honeyd). Estos contenedores emulan servicios informáticos reales (como SSH, bases de datos, aplicaciones web y protocolos industriales) con el fin de atraer la atención de los atacantes y capturar su huella digital y comprobar sus rastros.

- **Procesamiento de Datos (Backend):** Desarrollo de un motor de recolección automatizado en Python (Collector) y una API robusta que se encargan de monitorizar, unificar, geolocalizar y organizar los registros de actividad maliciosa almacenándolos en una base de datos relacional (PostgreSQL).

- **Plataforma Web y Dashboard:** Creación de una interfaz interactiva con control de acceso por roles. Dispone de un panel para el Administrador o Gerente (orientado al análisis avanzado y la visualización de ataques en tiempo real mediante un mapa 3D) y un perfil para el Empleado (enfocado en rutas de formación dinámica y resolución de cuestionarios basados en las intrusiones capturadas).

Limitaciones:

- **Naturaleza Pasiva del Sistema:** HoneypotSEC actúa como una plataforma de engaño e inteligencia de amenazas, no como un Sistema de Prevención de Intrusos (IPS) ni un cortafuegos activo corporativo. Su función es aislar, observar y registrar lo que ocurre dentro de los servidores trampa, no bloquear ni mitigar el tráfico de red general de la empresa.

- **El factor humano:** Para que la formación en ciberseguridad funcione, es clave que el equipo se implique. La plataforma ofrece el entorno de simulación y los casos reales, pero el aprendizaje real depende de que los empleados participen de forma activa.

- **Restricciones de Hardware y Red: La disponibilidad del sistema está directamente supeditada a los recursos del plan VPS contratado (memoria, CPU, ancho de banda y disco). Dado que la plataforma registra cada interacción maliciosa, ante un escenario extremo como un ataque masivo de Denegación de Servicio (DDoS), el sistema se enfrenta a dos riesgos principales:**

    1. **Saturación de almacenamiento:** Una cantidad colosal de ataques puede generar gigabytes de logs en cuestión de minutos, pudiendo agotar el espacio del disco NVMe y detener la base de datos PostgreSQL.

    2. **Colapso del ancho de banda:** Si se supera el límite de red entrante de la máquina, no solo se perdería la captura de nuevos ataques, sino que la interfaz web (Dashboard) dejaría de ser accesible temporalmente para los administradores y empleados.

2. Estado del Arte y Tecnologías

2.1. Conceptos de Honeypots (Baja vs Alta interacción)

En el ámbito de la ciberseguridad, los honeypots (sensores) se clasifican principalmente según el nivel de libertad y profundidad de interacción que ofrecen al atacante:

- **Honeypots de Baja Interacción:** Simulan únicamente ciertos servicios o protocolos de forma superficial (por ejemplo, responden a un saludo de conexión o muestran un mensaje de bienvenida). No existe un sistema operativo real detrás. Son muy seguros y fáciles de desplegar, pero la información que capturan es limitada, centrándose principalmente en direcciones IP y puertos atacados.

- **Honeypots de Alta Interacción:** Son sistemas operativos reales con servicios completos. El atacante puede entrar, ejecutar comandos reales y modificar archivos. Aunque ofrecen información de un valor incalculable sobre el comportamiento del intruso, son peligrosos: si el atacante logra romper el aislamiento, podría utilizar vuestra infraestructura para atacar a terceros.

- **Nuestra elección (Media-Alta Interacción):** En este proyecto utilizamos soluciones como **Cowrie**, que se sitúan en un punto intermedio. Estos sistemas emulan un entorno de comandos tan convincente que el atacante cree estar en un sistema real, lo que nos permite capturar sus acciones y los archivos que intenta descargar sin poner en riesgo el núcleo (*kernel*) del servidor principal.

2.2. Tecnologías utilizadas:

2.2.1. Cloud Computing: Hostinger - KVM2

Para asegurar la disponibilidad 24/7, se evaluaron distintas opciones de infraestructura en la nube. Aunque la implementación final se realizó en Hostinger, se llevó a cabo una fase previa de despliegue exhaustiva en **Oracle Cloud**.

- **Investigación en Oracle Cloud:** Se desarrolló una guía técnica completa sobre la creación y configuración de instancias gratuitas en Oracle Cloud (Nivel Siempre Gratis), que incluye el aprovisionamiento de recursos A1 con arquitectura ARM.
- **Incidencias y Migración:** Durante la fase de administración avanzada de los honeypots, la reconfiguración agresiva de puertos y las políticas de seguridad estrictas provocaron una pérdida de acceso al terminal del servidor (SSH).
- **Justificación del cambio:** Ante la necesidad de una gestión de red más flexible y un soporte técnico directo para la recuperación de instancias en un entorno de producción de ASIR, se decidió migrar la infraestructura al plan **KVM2 de Hostinger**.
- **Documentación Adjunta:** Todo el proceso de configuración inicial de Oracle Cloud, los scripts de despliegue específicos para dicha plataforma y el análisis del error que forzó la migración se encuentran detallados para su consulta en el [Free Tier Oracle Cloud - HoneypotSEC].

Documento para crear instancia en Oracle Cloud

COMPARACIÓN DE PROVEEDORES VPS

Datos actualizados en Mayo del 2026.

Proveedor	Modelo / Plan	Recursos	Precio	Ventajas	Desventajas
Oracle Cloud	Free Tier (A1)	4 vCPU / 24 GB RAM (ARM)	**0€/mes** Permanente sin limite: Créditos prueba: 300$	-Recursos gratuitos muy altos. -10 TB de ancho de banda incluido -IP propia	Registro extremadamente lento, rechazo de datos reales y alta burocracia.
AWS	Free Tier (t2.micro)	1 vCPU / 1 GB RAM	**0€/mes** Solo primer año para nuevas cuentas	-Estándar de la industria -Muy escalable -Ecosistema masivo de servicios	Recursos insuficientes para 6 honeypots y gestión compleja de facturación.
Google Cloud	Free Trial	2 vCPU / 1 GB RAM	**0€/mes** 1 instancia e2-micro siempre gratis. 300$ créditos 90 días	-Integración excelente con servicios GCP -Red global de alta calidad -Descuentos por uso	Panel muy complejo; tras el periodo de prueba los costes son elevados.
Hostinger	**KVM2 (Pago)**	**2 vCPU / 8GB RAM**	**7’99€/mes** Renovación: 11’99€ – 16/mes	**-Hardware de alto rendimiento** **-IP propia** **-Backups semanales incluidas** **-Despliegue en un click**	**Coste mensual (aunque bajo para sus prestaciones).**

2.2.2 Virtualización de Contenedores: Docker y Docker Compose

Para la gestión de los honeypots, hemos descartado la instalación directa sobre el SO. En su lugar, utilizamos **Docker** por:

- **Aislamiento:** Cada honeypot corre en su propio contenedor. Si una vulnerabilidad del honeypot es explotada, el atacante queda atrapado en el contenedor sin afectar al resto del ecosistema ni al servidor principal.

- **Orquestación:** Utilizamos Docker Compose para levantar toda la infraestructura (12 servicios interconectados) mediante un único archivo de configuración estructurado en YAML, facilitando un despliegue rápido y reproducible.

2.2.3. Honeypots, Backend y Base de datos

Hemos seleccionado una red de sensores conformada de herramientas especializadas que nos permite cubrir los servicios más utilizados en las empresas:

- **Red de Sensores:** Empleamos Cowrie y Dionaea para capturar accesos por consola y recolección de malware; Glastopf y Conpot para simular vulnerabilidades en aplicaciones web y entornos industriales (SCADA) respectivamente; y Honeytrap junto con Honeyd para la monitorización de escaneos de puertos y emulación de topologías de red.

- **Procesamiento (Python y FastAPI):** Desarrollamos un motor Collector en Python encargado de normalizar y geolocalizar los registros en tiempo real. La información se sirve a través de una API RESTful construida con FastAPI, que destaca por su altísimo rendimiento y gestión asíncrona de peticiones.

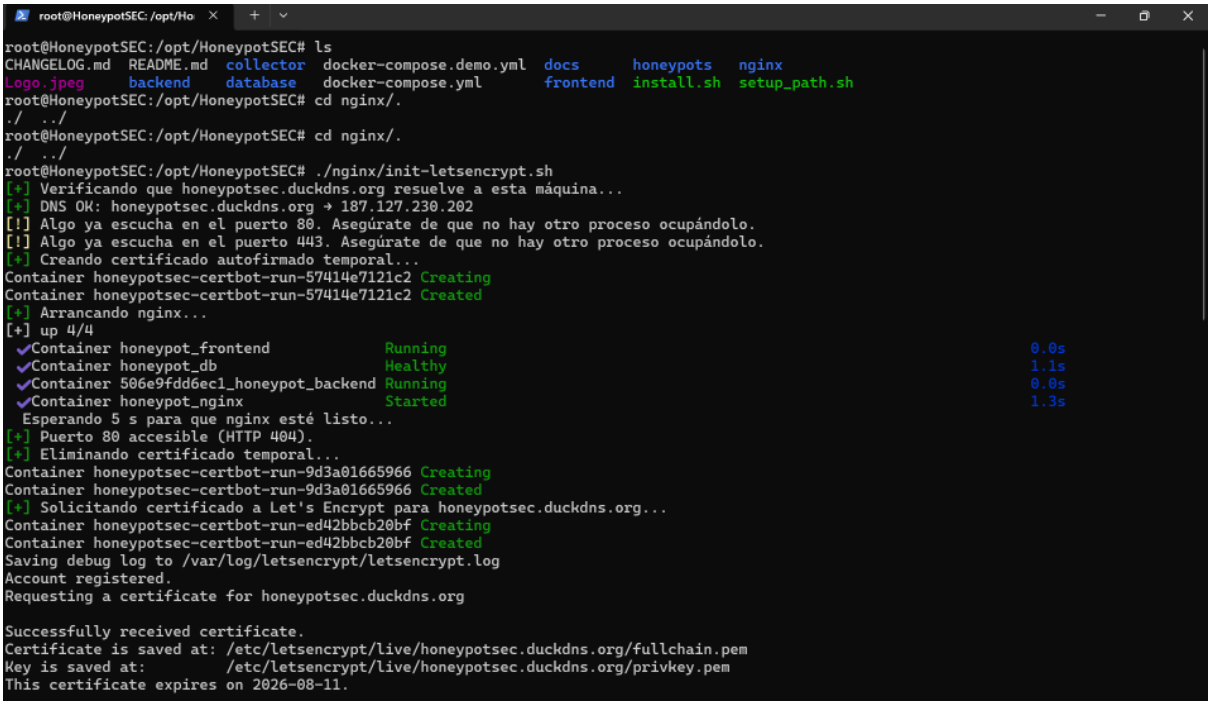
- **Almacenamiento (PostgreSQL):** Toda la inteligencia recolectada se centraliza en una base de datos relacional PostgreSQL, garantizando la persistencia e integridad de los logs y del progreso de los usuarios.

2.2.4. Securización Perimetral (Hardening)

Dado que estamos exponiendo trampas directamente a Internet, la seguridad del servidor principal es nuestra prioridad absoluta. Para evitar que un atacante tome el control real del sistema, hemos aplicado las siguientes medidas:

- **Autenticación Asimétrica:** Se ha deshabilitado por completo el acceso administrativo por contraseña, permitiendo el login exclusivamente mediante pares de claves criptográficas SSH (RSA/Ed25519).

- **Control Perimetral (UFW / iptables):** Se han establecido estrictas reglas de cortafuegos para negar cualquier conexión entrante por defecto (Drop All), abriendo exclusivamente los puertos lógicos que corresponden a los honeypots y al tráfico web (HTTP/HTTPS cifrado mediante Nginx y Let's Encrypt).



2.2.5. Frontend y Experiencia de Usuario

Para que **HoneypotSEC** sea un producto atractivo para otras empresas, la web debe ser rápida, moderna y fácil de usar. Para ello, hemos utilizado:

- **React y Vite:** Garantizan un entorno reactivo donde la información (como las alertas de intrusión mediante WebSockets) fluye en tiempo real sin recargar la página. Vite acelera el tiempo de empaquetado y carga de la web.

- **Visualización 3D (COBE):** Para representar el origen geográfico de los ataques de forma inmersiva, hemos integrado COBE, una librería de renderizado WebGL de alto rendimiento que genera un globo terráqueo interactivo a 60 FPS.

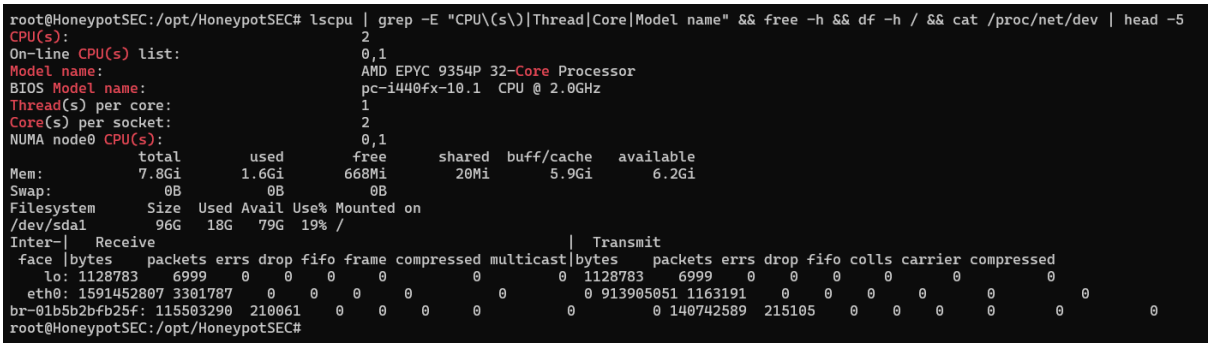
- **TailwindCSS:** Como framework de diseño, permite aplicar una estética profesional, responsive y moderna, utilizando utilidades de clase y esquemas de color adaptados a interfaces de ciberseguridad (modo oscuro).

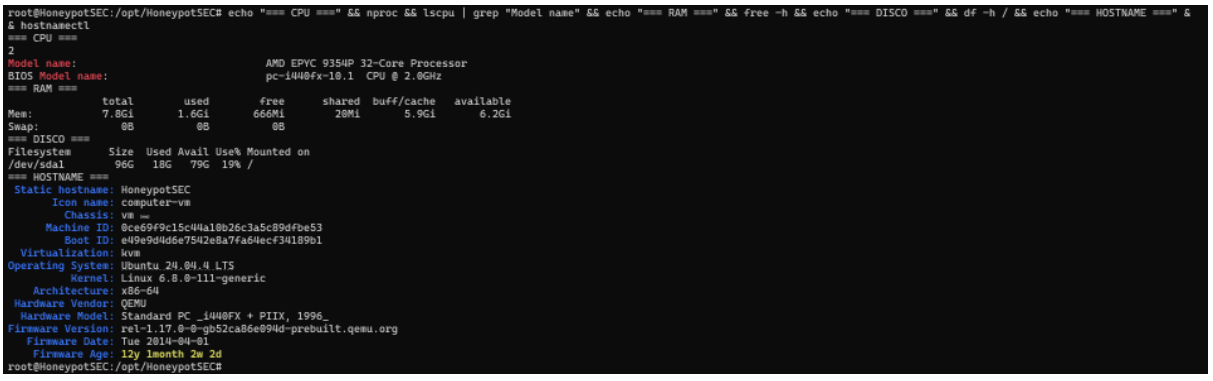
3. Análisis y Diseño

3.1. Requisitos del sistema (Hardware y Software)

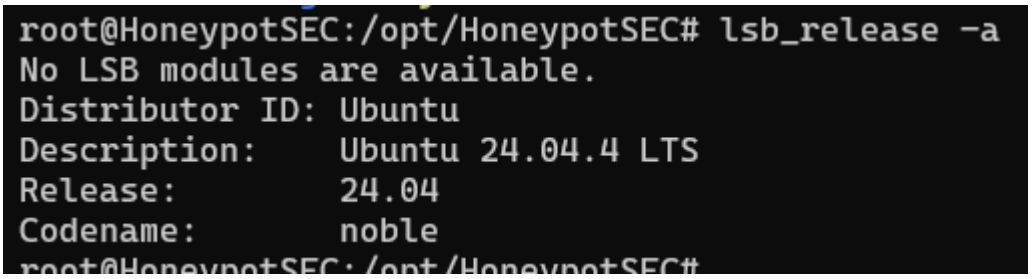
Para garantizar el correcto funcionamiento de HoneypotSEC en su entorno de producción en la nube, se han dimensionado los recursos basándose en los requisitos mínimos para la orquestación simultánea de 12 contenedores:

- **Hardware (VPS KVM 2):** El sistema dispone de 2 núcleos vCPU, 8 GB de memoria RAM, 100 GB de almacenamiento NVMe y 2 TB de ancho de banda. Estos recursos garantizan la fluidez necesaria para ejecutar la recolección, el motor de base de datos y la interfaz web sin cuellos de botella, incluso si los sensores reciben ataques masivos y concurrentes.

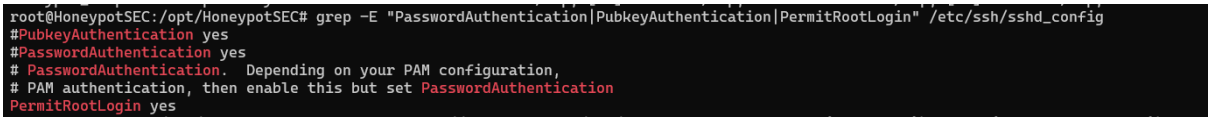


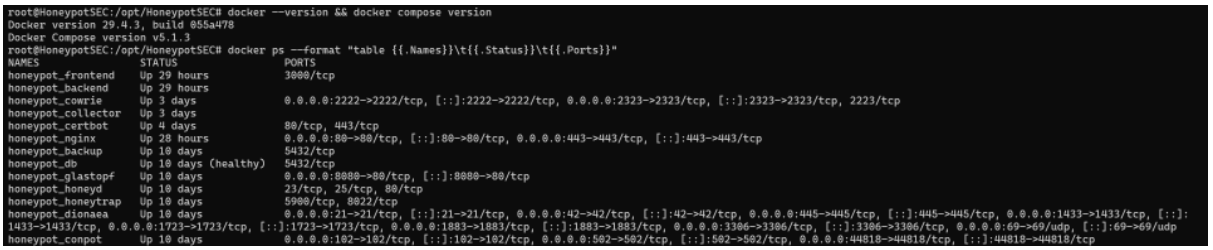


Software Base: Sistema operativo Linux (Ubuntu Server 22.04 LTS), motor de virtualización Docker Engine y Docker Compose para la orquestación de servicios.

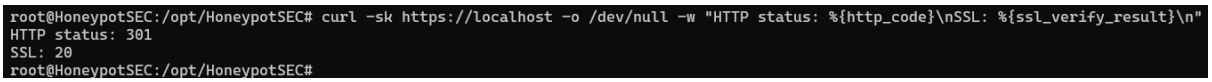


- **Seguridad de Administración:** Acceso mediante cliente SSH con soporte exclusivo para autenticación por claves criptográficas asimétricas.





- **Acceso de Cliente (SaaS):** Cualquier navegador web moderno para visualizar el Dashboard interactivo y el contenido educativo.



3.2. Casos de Uso

La plataforma contempla diferentes actores que interactúan con el sistema con propósitos distintos:

- **Actor Atacante (Explotación):** Entidad maliciosa (humano o botnet automatizada) que escanea Internet buscando vulnerabilidades. Al detectar los puertos lógicos abiertos por nuestros sensores (como el puerto 2222 de Cowrie o el 21 de Dionaea), interactúa con ellos inyectando comandos o malware.

- **Actor Administrador/Gerente (Supervisión):** Usuario con privilegios de gestión en la interfaz B2B. Su función es revisar la telemetría de seguridad corporativa (volumen de ataques, orígenes geográficos, credenciales vulneradas) y supervisar el cumplimiento de la formación de su plantilla.

- **Actor Empleado (Formación):** Usuario estándar que accede a la plataforma para adquirir concienciación cibernética. Su interacción se centra en visualizar una versión adaptada del dashboard con las amenazas dirigidas a su empresa y en superar los módulos de aprendizaje continuo.

3.3. Diseño de la Arquitectura de Red y Datos

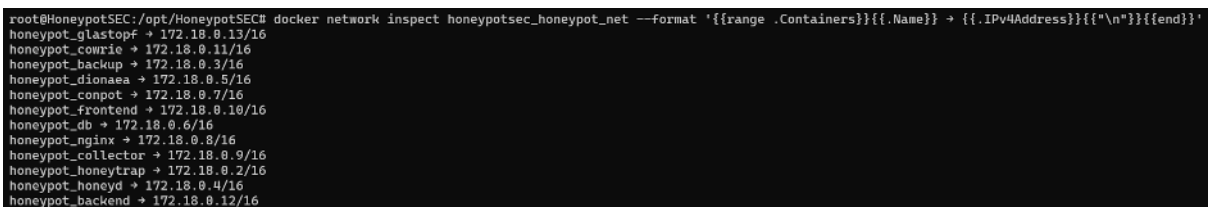
La arquitectura de **HoneypotSEC** se basa en un ecosistema de microservicios orquestados mediante contenedores, donde cada componente cumple una función específica dentro de la cadena de captura, procesamiento y formación.

3.3.1. Topología de Red y Componentes

La infraestructura está diseñada para separar el tráfico "sucio" de los atacantes del tráfico "limpio" de los usuarios mediante la siguiente estructura:

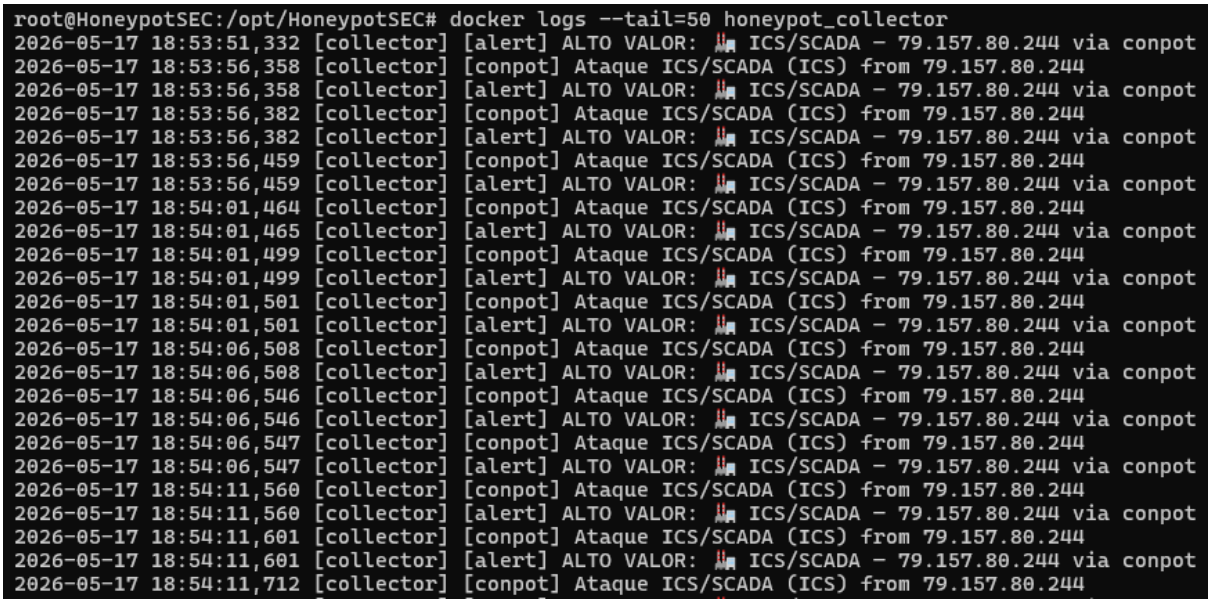
- **Proxy Inverso (Nginx):** Actúa como puerta de enlace perimetral. Gestiona la terminación SSL/TLS (cifrado HTTPS) y enruta las peticiones web de los clientes hacia el frontend y las peticiones de datos hacia la API.

- **Red de Sensores (Honeypots):** Seis contenedores aislados (Cowrie, Dionaea, Glastopf, Conpot, Honeytrap y Honeyd) que emulan servicios vulnerables como SSH, aplicaciones web, bases de datos y protocolos industriales. Registran su actividad maliciosa en volúmenes compartidos de solo lectura.

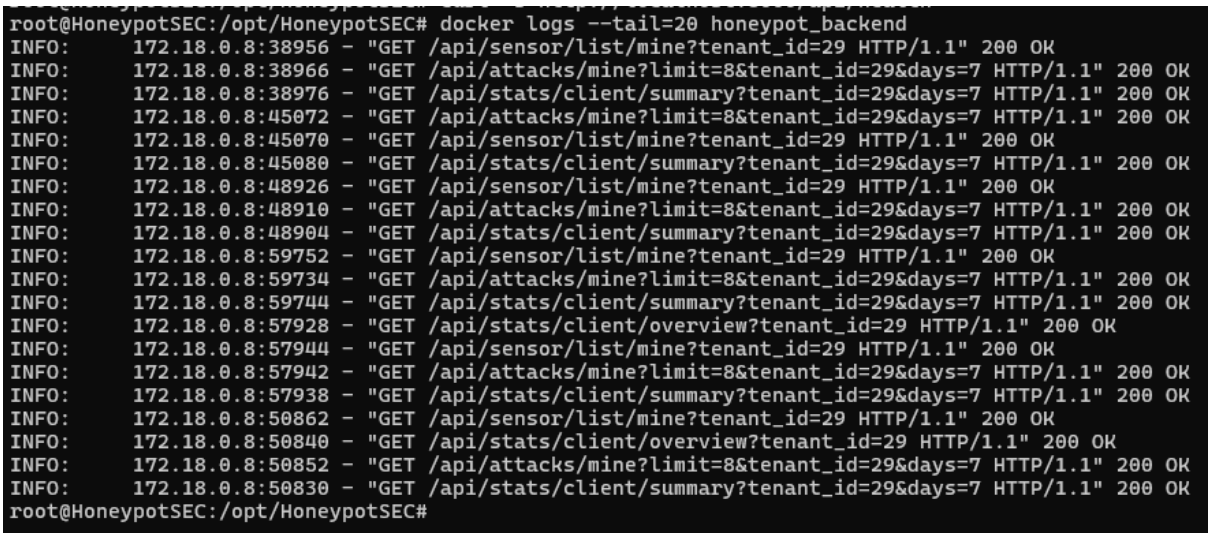


- **Motor de Recolección (Collector):** Un servicio en segundo plano (demonio) desarrollado en Python que monitoriza en tiempo real los

volúmenes de logs de los sensores. Su función es extraer, normalizar y geolocalizar la información relevante de cada ataque.



- **Núcleo del Sistema (Backend & DB):** Una API RESTful desarrollada con FastAPI que aplica la lógica de negocio y una base de datos relacional (PostgreSQL) que asegura la persistencia transaccional e integridad de los registros capturados.



3.3.2. Modelo de Datos y Entidades

1. **Registro de Intrusiones (Logs):** Almacena la telemetría del atacante. Incluye la IP de origen, el Timestamp (momento exacto), el sensor vulnerado, las coordenadas geográficas y los detalles técnicos del payload (como las contraseñas probadas en ataques de fuerza bruta).

2. **Gestión de Usuarios y Roles:** Segmenta el acceso entre Administradores (con acceso total a métricas corporativas) y Empleados (perfil enfocado a la formación). Almacena el progreso, las métricas de retención y la puntuación de cada usuario.

```
root@HoneypotSEC:/opt/HoneypotSEC# docker exec honeypot_db psql -U honeypot_user -d honeypot_db -c "\d+ usuarios"
                                                    Table "public.usuarios"
      Column       |           Type           | Collation | Nullable |              Default               | Storage  | Compression | Stats target | Description
-------------------+--------------------------+-----------+----------+------------------------------------+----------+-------------+--------------+-------------
 id                | integer                  |           | not null | nextval('users_id_seq'::regclass)  | plain    |             |              |
 nombre_usuario    | character varying(100)   |           | not null |                                    | extended |             |              |
 email             | character varying(255)   |           |          |                                    | extended |             |              |
 contrasena_hash   | character varying(255)   |           | not null |                                    | extended |             |              |
 rol               | character varying(20)    |           | not null | 'employee'::character varying      | extended |             |              |
 activo            | boolean                  |           | not null | true                               | plain    |             |              |
 creado_en         | timestamp with time zone |           |          | now()                              | plain    |             |              |
 nombre_empresa    | character varying(255)   |           |          |                                    | extended |             |              |
 sector_empresa    | character varying(100)   |           |          |                                    | extended |             |              |
 plan              | character varying(20)    |           |          | 'basico'::character varying        | extended |             |              |
 token_instalacion | text                     |           |          |                                    | extended |             |              |
 token_creado_en   | timestamp with time zone |           |          |                                    | plain    |             |              |
Indexes:
    "users_pkey" PRIMARY KEY, btree (id)
    "idx_usuarios_nombre_usuario" btree (nombre_usuario)
    "idx_usuarios_rol" btree (rol)
    "users_email_key" UNIQUE CONSTRAINT, btree (email)
    "users_username_key" UNIQUE CONSTRAINT, btree (nombre_usuario)
Check constraints:
    "usuarios_rol_check" CHECK (rol::text = ANY (ARRAY['admin'::character varying, 'empleado'::character varying, 'cliente'::character varying]::text[]))
Referenced by:
    TABLE "resultados_quiz" CONSTRAINT "quiz_results_user_id_fkey" FOREIGN KEY (id_usuario) REFERENCES usuarios(id) ON DELETE CASCADE
Access method: heap

root@HoneypotSEC:/opt/HoneypotSEC#
```

3. **Módulos de Formación:** Entidades que estructuran los seis módulos interactivos (Fuerza Bruta, SQL Injection, Phishing, XSS, Contraseñas débiles y Simulaciones en vivo).

```
root@HoneypotSEC:/opt/HoneypotSEC# docker exec honeypot_db psql -U honeypot_user -d honeypot_db -c "\dt"
              List of relations
 Schema |         Name          | Type  |     Owner
--------+-----------------------+-------+---------------
 public | articulos_educativos  | table | honeypot_user
 public | ataques               | table | honeypot_user
 public | preguntas_quiz        | table | honeypot_user
 public | resultados_quiz       | table | honeypot_user
 public | sensores              | table | honeypot_user
 public | usuarios              | table | honeypot_user
(6 rows)

root@HoneypotSEC:/opt/HoneypotSEC# docker exec honeypot_db psql -U honeypot_user -d honeypot_db -c "\d+ ataques"
                                                    Table "public.ataques"
      Column       |           Type           | Collation | Nullable |              Default               | Storage  | Compression | Stats target | Description
-------------------+--------------------------+-----------+----------+------------------------------------+----------+-------------+--------------+-------------
 id                | integer                  |           | not null | nextval('attacks_id_seq'::regclass) | plain    |             |              |
 marca_tiempo      | timestamp with time zone |           | not null | now()                              | plain    |             |              |
 honeypot          | character varying(50)    |           | not null |                                    | extended |             |              |
 ip_origen         | character varying(45)    |           | not null |                                    | extended |             |              |
 puerto_origen     | integer                  |           |          |                                    | plain    |             |              |
 puerto_destino    | integer                  |           |          |                                    | plain    |             |              |
 protocolo         | character varying(20)    |           |          |                                    | extended |             |              |
 pais              | character varying(100)   |           |          |                                    | extended |             |              |
 codigo_pais       | character(2)             |           |          |                                    | extended |             |              |
 ciudad            | character varying(100)   |           |          |                                    | extended |             |              |
 latitud           | double precision         |           |          |                                    | plain    |             |              |
 longitud          | double precision         |           |          |                                    | plain    |             |              |
 tipo_ataque       | character varying(100)   |           |          |                                    | extended |             |              |
 usuario           | character varying(255)   |           |          |                                    | extended |             |              |
 contrasena        | character varying(255)   |           |          |                                    | extended |             |              |
 payload           | text                     |           |          |                                    | extended |             |              |
 id_sesion         | character varying(255)   |           |          |                                    | extended |             |              |
 datos_brutos      | jsonb                    |           |          |                                    | extended |             |              |
 id_sensor         | integer                  |           |          |                                    | plain    |             |              |
Indexes:
    "attacks_pkey" PRIMARY KEY, btree (id)
    "idx_ataques_codigo_pais" btree (codigo_pais)
    "idx_ataques_honeypot" btree (honeypot)
    "idx_ataques_ip_origen" btree (ip_origen)
    "idx_ataques_marca_tiempo" btree (marca_tiempo DESC)
    "idx_ataques_tipo_ataque" btree (tipo_ataque)
    "idx_attacks_sensor_id" btree (id_sensor)
Access method: heap
```

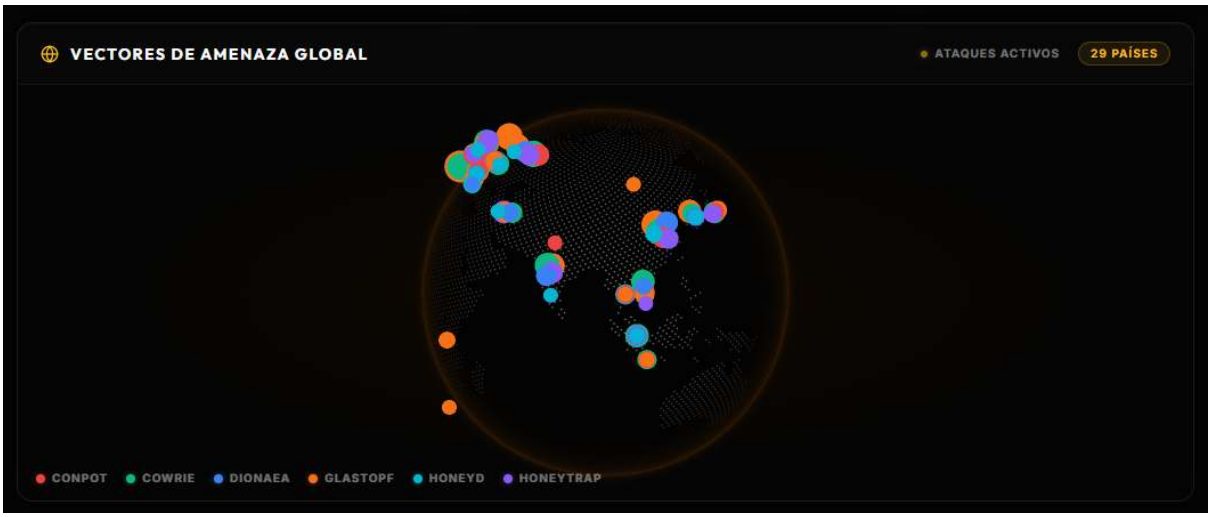
3.3.3. Visualización Geoespacial (COBE Globe)

Para la representación visual táctica de las amenazas, hemos implementado el componente **COBE**, una librería de renderizado basada en WebGL. Esta tecnología nos permite:

Renderizado acelerado por hardware: Al ejecutarse directamente en la tarjeta gráfica del cliente, el globo terráqueo rota a 60 FPS constantes sin impactar el rendimiento del Dashboard, soportando la representación simultánea de cientos de intrusiones.

Canal de Datos en Tiempo Real (WebSockets): Se ha configurado una conexión persistente para que, cada vez que la API notifica un nuevo ataque, se genere instantáneamente un marcador con efecto de expansión (ripple effect) en las coordenadas exactas del atacante sobre el globo.

Diseño e Identidad UI: La estética con el tema claro o oscuro y minimalista del globo refuerza la identidad visual de un "Centro de Operaciones de Seguridad" (SOC), esencial para que el producto SaaS sea atractivo en el mercado profesional B2B.



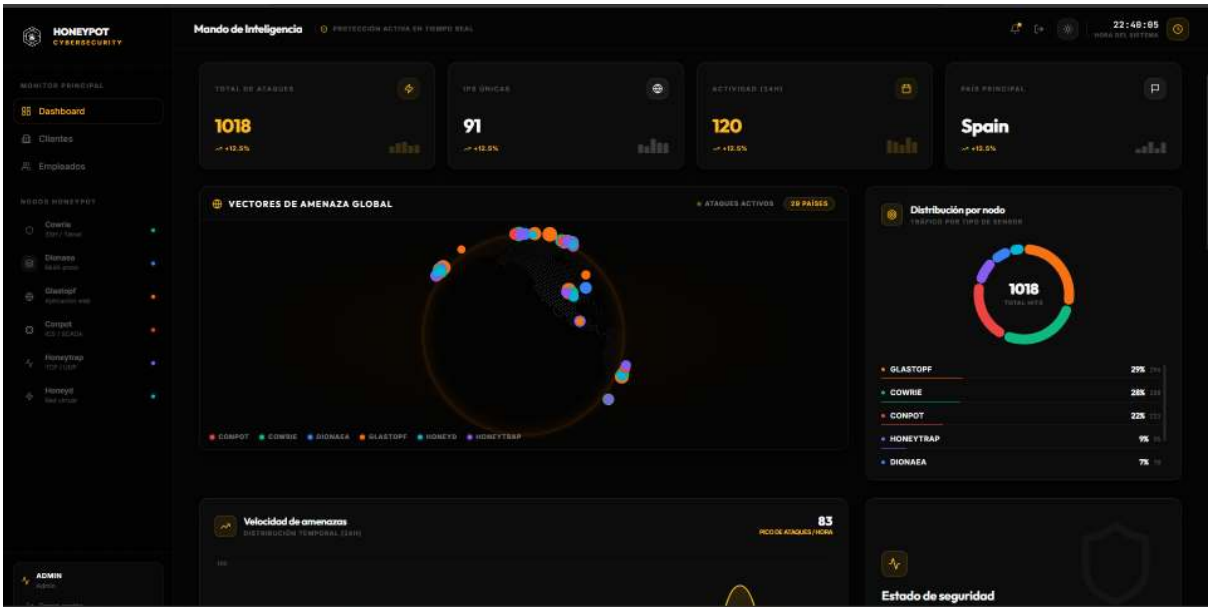
3.4. Mapa de Navegación y Flujo de Usuario

Para asegurar una experiencia de usuario (UX) fluida y coherente con la arquitectura de microservicios, la interfaz de **HoneypotSEC** utiliza un sistema de enrutamiento dinámico. La interfaz se estructura en rutas protegidas que restringen las vistas y funciones basándose en la validación de los tokens (JWT) del rol del usuario autenticado.

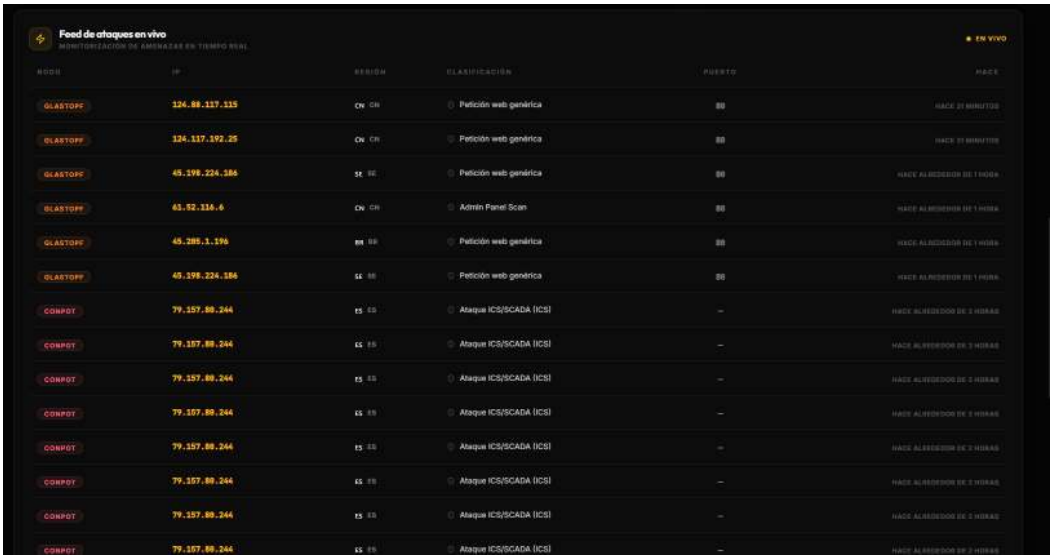
3.4.1. Flujo del Administrador (Análisis de Seguridad)

El rol de administrador o gerente dispone de una visión técnica, estratégica y global del sistema:

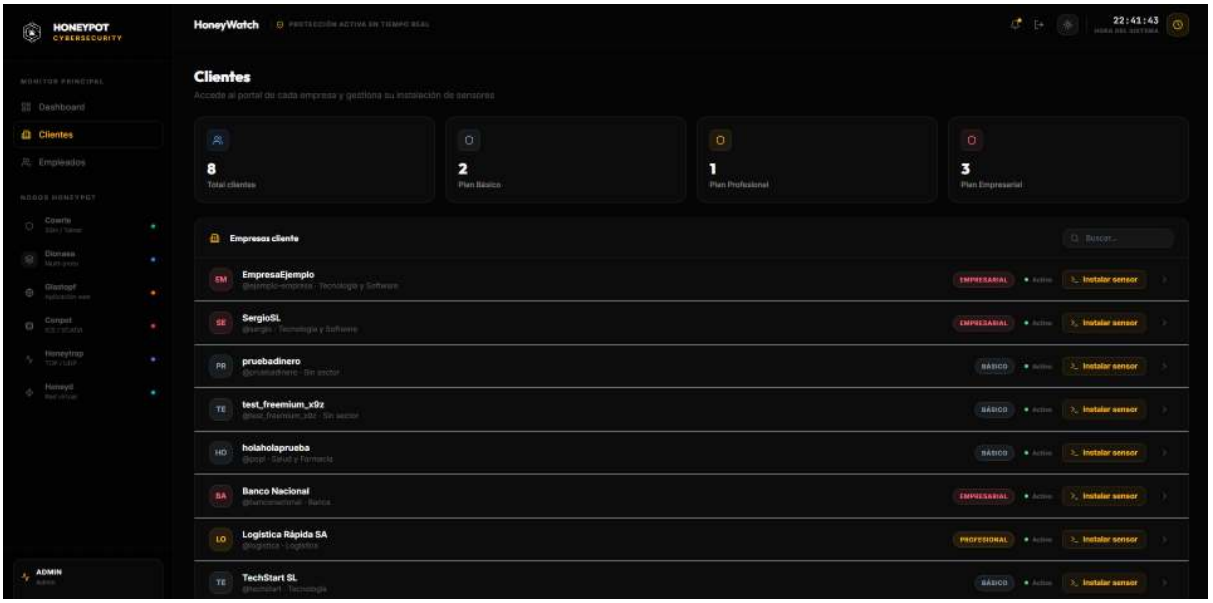
- **Panel de Control (Dashboard Central):** Es la pantalla principal donde reside el renderizado interactivo del globo COBE. Permite visualizar de un vistazo la actividad global y los indicadores clave (KPIs) críticos, como el Top de IPs atacantes, tendencias de tráfico o los sensores que sufren mayor asedio.



- **Explorador de Amenazas (Auditoría):** Una vista tabular detallada y conectada directamente a la base de datos PostgresSQL. Permite búsqueda avanzada para realizar auditorías forenses, revisando la carga útil (payload) o los comandos ejecutados por los atacantes (funcionalidad especialmente crítica para el sensor SSH).



- **Centro de Gestión B2B:** Espacio administrativo (interfaz CRUD) dedicado a la gestión de identidades de la organización. Permite dar de alta a nuevos empleados, gestionar contraseñas y monitorear el cumplimiento individual de los objetivos de formación.



3.4.2. Flujo del Empleado (Rutas de Aprendizaje)

El flujo del empleado está diseñado para transformar el miedo a las ciberamenazas en conocimiento práctico:

- **Visualización de Exposición:** Visualización de Exposición Contextual: El empleado visualiza una versión simplificada del globo terráqueo que muestra exclusivamente los ataques dirigidos hacia su empresa. Esta evidencia visual aumenta la percepción de riesgo y la relevancia del aprendizaje.

- **Módulos Interactivos Guiados:** Acceso secuencial a los 6 bloques formativos. La plataforma utiliza los logs reales capturados en el backend para construir dinámicamente ejercicios prácticos y contextualizados (ej. “¿Cuál de estas direcciones IP detectadas ayer crees que realizó un ataque de inyección SQL?”).

- **Sistema de Gamificación:** Tras superar las simulaciones prácticas, el flujo redirige al usuario a su perfil personal. En esta pantalla, el empleado puede consultar su progresión, las certificaciones internas obtenidas y los puntos de experiencia acumulados en comparación con el ranking corporativo del resto de la organización.

4. Implementación y Desarrollo

4.1. Aprovisionamiento y Configuración del Servidor Cloud

El despliegue de la plataforma comenzó con la puesta en marcha de la infraestructura en la nube. Siguiendo las directrices marcadas en la fase de análisis previo, se procedió a la contratación y configuración inicial del VPS en Hostinger.

4.1.1. Despliegue de la instancia VPS

El proceso se realizó seleccionando el plan **KVM 2**, asegurando que la ubicación del centro de datos fuera óptima para minimizar la latencia.

Imagen de los planes de Hostinger:

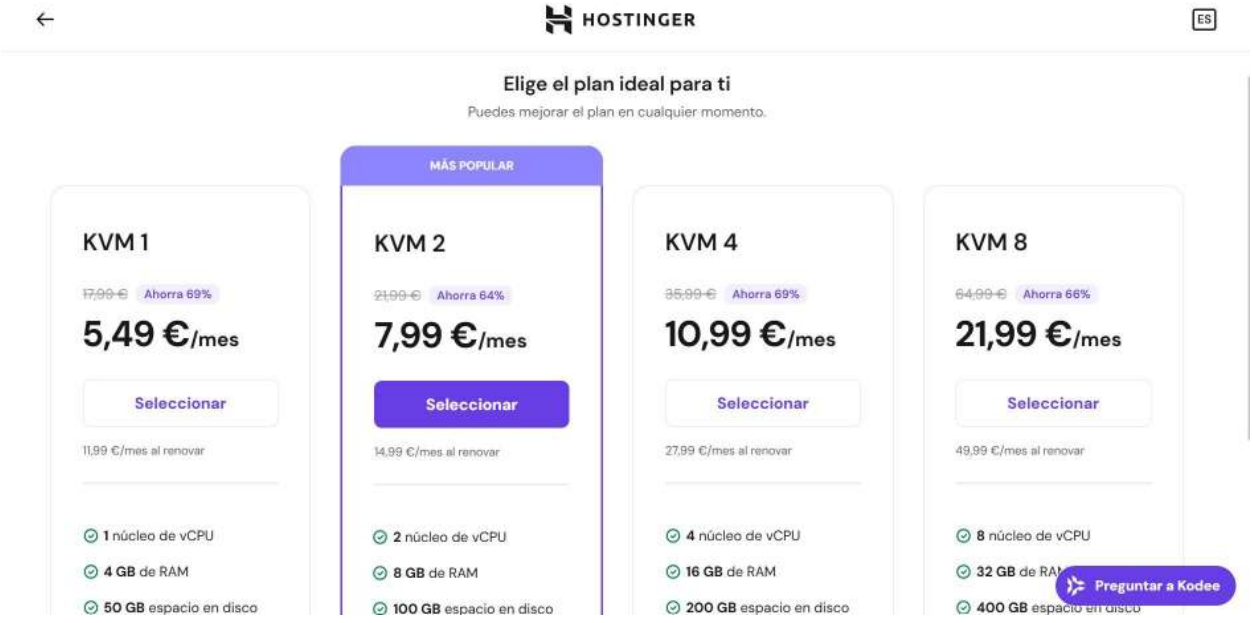


Imagen de la ubicación de nuestro VPS de Hostinger:

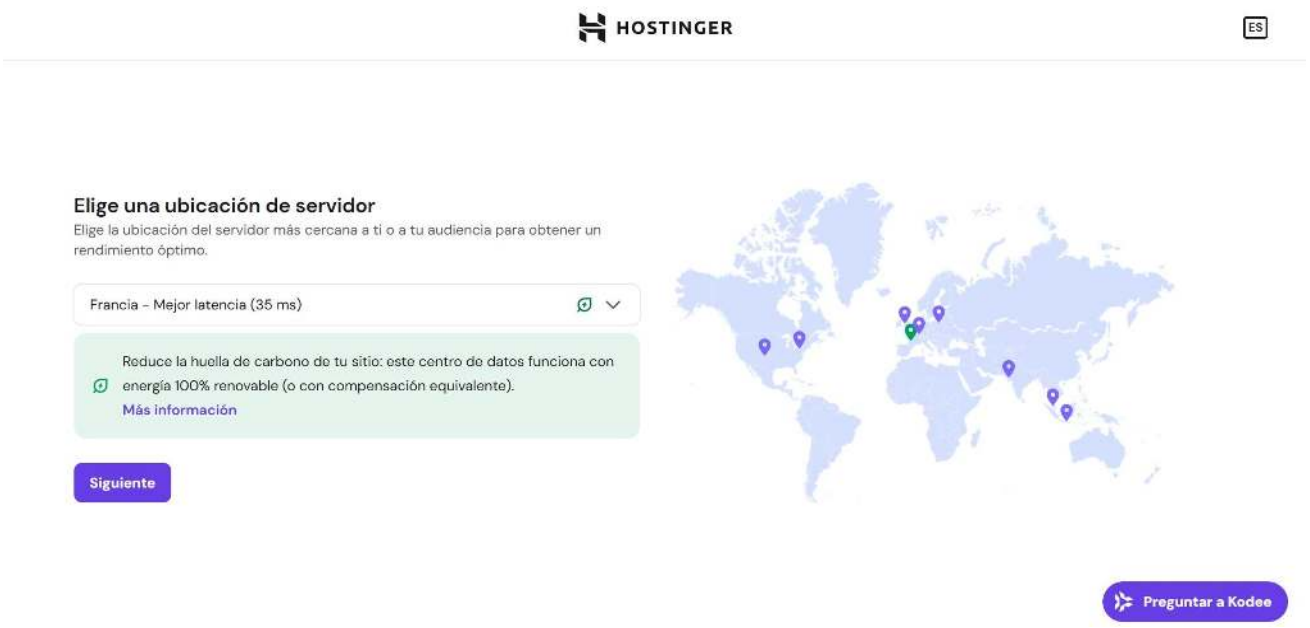
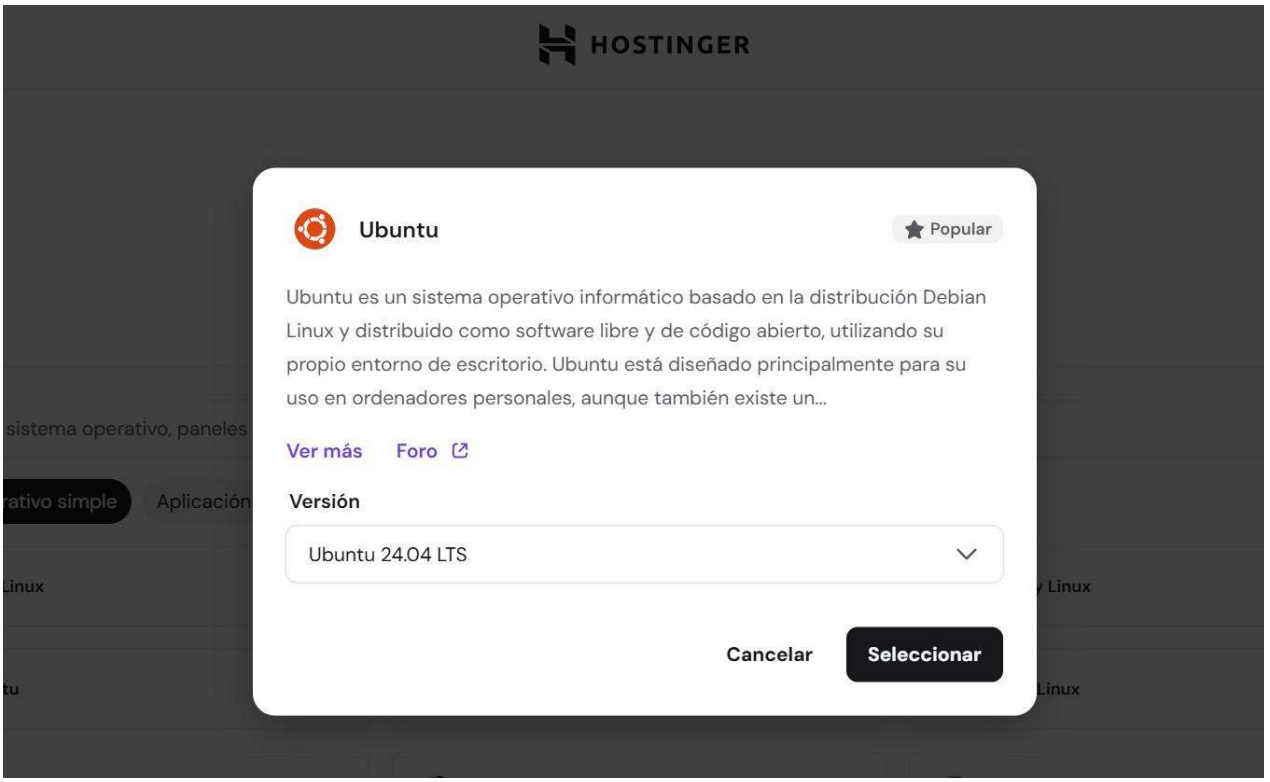


Imagen del sistema operativo seleccionado para nuestro VPS:



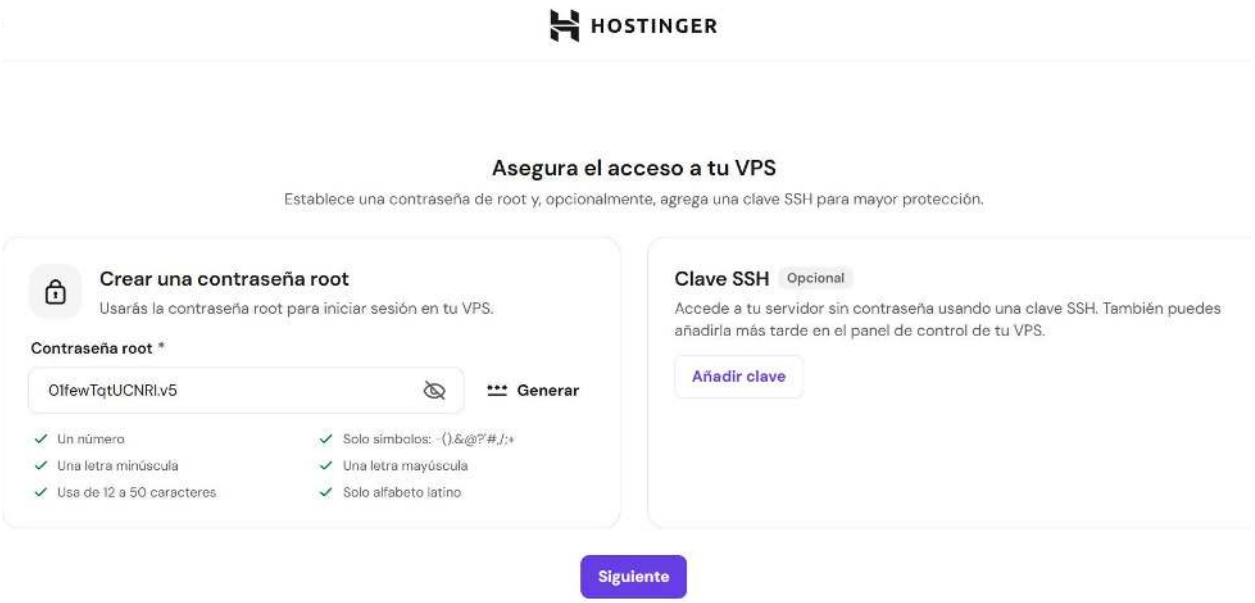
4.1.2. Securización del acceso (Hardening inicial)

Como paso crítico antes de desplegar y exponer los honeypots a Internet, se aplicaron medidas de hardening para proteger el acceso administrativo al servidor host:

1. **Actualización del sistema base:** Ejecución de los comandos *apt update && apt upgrade* para parchear posibles vulnerabilidades del kernel de Linux y los paquetes base.

2. **Configuración de SSH por clave pública:** Se deshabilitó por completo el acceso mediante contraseña modificando el archivo /etc/ssh/sshd_config, permitiendo de forma exclusiva las conexiones autenticadas mediante pares de claves asimétricas (RSA/Ed25519).

3. **Gestión de credenciales:** Se configuró una contraseña robusta para el usuario root y se creó un usuario estándar de administración con privilegios de sudo.

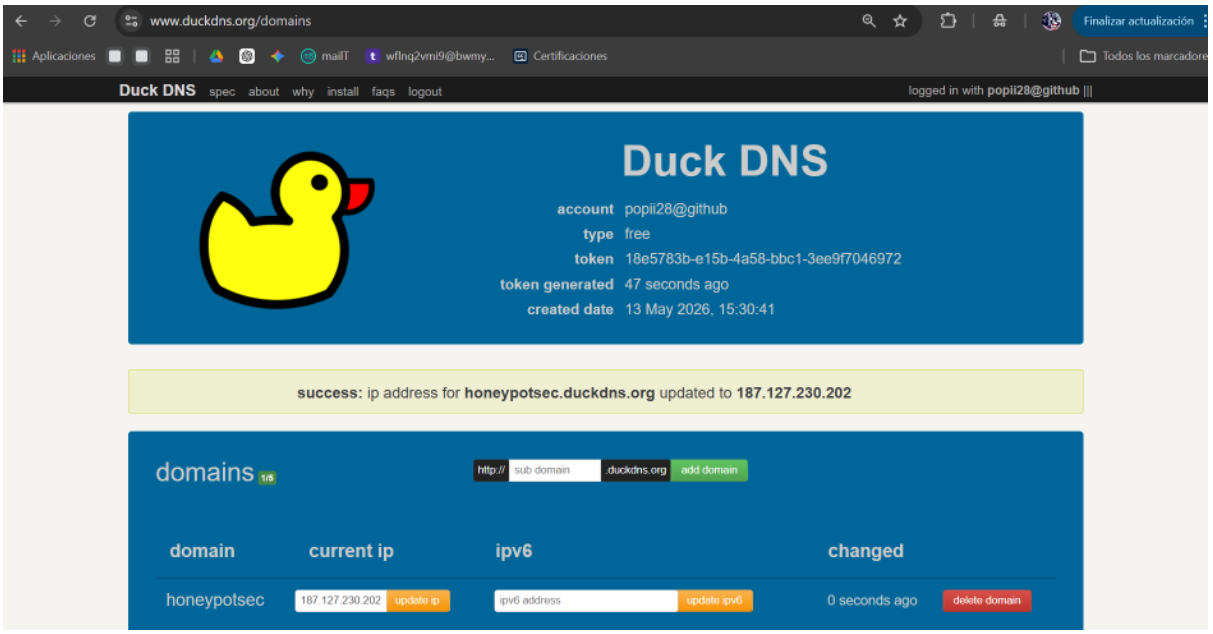


4.1.3. Resolución DNS y Dominio Público

Para facilitar el acceso al Dashboard y dotar a la plataforma de una identidad corporativa, se ha evitado el acceso directo por dirección IP mediante la configuración de una resolución de nombres.

- Proveedor de DNS Dinámico: Se ha utilizado el proveedor Duck DNS, gestionando el registro y la autenticación de forma segura vinculando nuestra cuenta de desarrollador de GitHub.

- URL de Producción: Todo el ecosistema web es accesible de forma global a través del dominio honeypotsec.duckdns.org. Este dominio resuelve directamente contra la IP pública del servidor VPS en Hostinger y es, además, la pieza clave que permite al contenedor de Certbot validar el dominio y emitir los certificados SSL/TLS, garantizando que el acceso al panel B2B se realice siempre bajo un entorno seguro (HTTPS).



4.2. Instalación del Entorno de Ejecución (Docker)

Una vez securizado el servidor base, se procedió a instalar las herramientas de contenedorización necesarias para ejecutar y aislar la red de sensores.

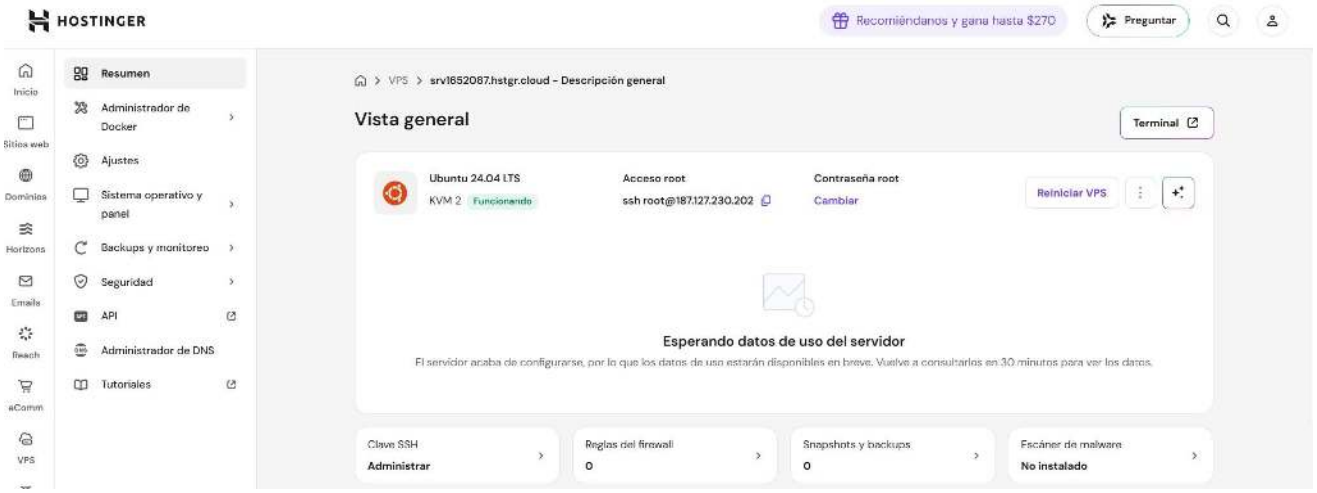
Para automatizar este proceso y asegurar que cualquier administrador de sistemas pueda replicar el entorno de producción sin margen de error, la tarea se integró en un script de despliegue (install.sh) que automatiza los siguientes pasos:

- Instalación del motor Docker Engine y la herramienta de orquestación Docker Compose.

- Configuración del demonio (daemon) de Docker en systemd para asegurar su arranque automático tras un reinicio físico del VPS.

- Verificación y asignación de permisos al usuario de administración para ejecutar contenedores sin necesidad de invocar sudo de forma constante.

```
root@HoneypotSEC:~# docker --version
Docker version 29.4.3, build 055a478
root@HoneypotSEC:~# docker compose version
Docker Compose version v5.1.3
root@HoneypotSEC:~# sudo systemctl is-active docker
```



4.3. Automatización y Orquestación de Servicios

Una de las mayores ventajas técnicas de **HoneypotSEC** es su capacidad de despliegue automatizado, lo que reduce errores humanos y asegura que todos los componentes se comuniquen correctamente desde el primer segundo.

4.3.1. Script de Instalación Automatizada (install.sh)

Para facilitar la puesta en marcha, se desarrolló un script en Bash que centraliza las tareas de preparación del sistema. Este script se encarga de:

- **Verificación de Dependencias:** Comprueba la existencia y correcta versión de Docker y Openssl.

- **Generación de Seguridad TLS:** Ejecuta un sub-script de Nginx para generar certificados de seguridad, garantizando que toda comunicación entre el cliente B2B y nuestro servidor viaje cifrada mediante HTTPS.

- **Configuración del Entorno:** Gestiona la inicialización de los archivos de configuración dinámicos a partir de las plantillas .env, inyectando las credenciales necesarias (como contraseñas de bases de datos o tokens JWT) de forma segura.

4.3.2. Orquestación con Docker Compose

La infraestructura completa se despliega mediante el archivo docker-compose.yml, el cual define la jerarquía, mapeo de puertos y dependencias de la red virtual. Hemos configurado el despliegue utilizando directivas de dependencia (depends_on con comprobaciones healthcheck) para que los servicios sigan un orden lógico y seguro de encendido:

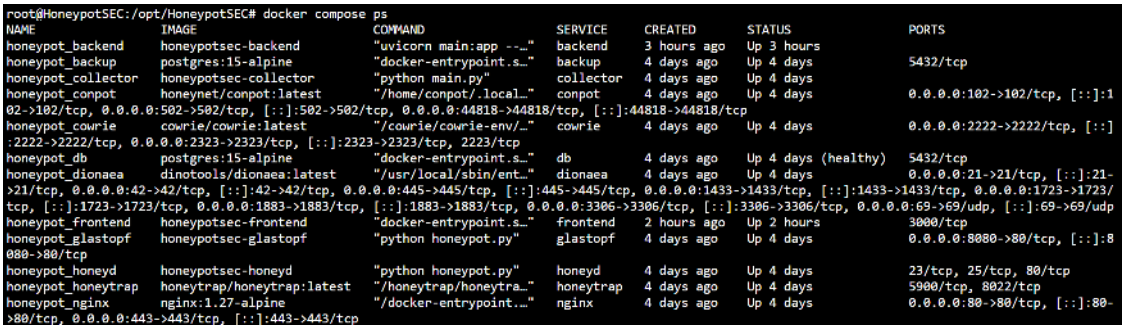
1. **Base de Datos (db):** En primer lugar se levanta el contenedor de PostgreSQL, asegurando que su puerto interno esté disponible para recibir conexiones.

2. **Lógica y Recolección (backend & collector):** Una vez que el sistema detecta que la base de datos está lista y saludable (healthy), se inician los motores de procesamiento y la API.

3. **Sensores (Honeypots):** Se despliegan simultáneamente los 6 contenedores trampa (Cowrie, Dionaea, Glastopf, Conpot, Honeytrap y Honeyd), simulando servicios como SSH, HTTP, FTP, bases de datos y protocolos SCADA, mapeando estratégicamente sus puertos hacia la IP pública.

4. **Puerta de Enlace (nginx):** Nginx actúa como escudo perimetral (Proxy Inverso) y distribuidor de tráfico, exponiendo únicamente los puertos seguros 80 y 443 hacia los clientes.

Este comando te mostrará una tabla con el nombre de los servicios, su estado, los puertos que están usando, cuando se creó y su estado.



```
root@HoneypotSEC:/opt/HoneypotSEC# docker compose ps
NAME                IMAGE                          COMMAND                  SERVICE     CREATED       STATUS                   PORTS
honeypot_backend    honeypotsec-backend            "uvicorn main:app --…"   backend     3 hours ago   Up 3 hours
honeypot_backup     postgres:15-alpine             "docker-entrypoint.s…"   backup      4 days ago    Up 4 days                5432/tcp
honeypot_collector  honeypotsec-collector          "python main.py"         collector   4 days ago    Up 4 days
honeypot_conpot     honeynet/conpot:latest         "/home/conpot/.local…"   conpot      4 days ago    Up 4 days                0.0.0.0:102->102/tcp, [::]:1
02->102/tcp, 0.0.0.0:502->502/tcp, [::]:502->502/tcp, 0.0.0.0:44818->44818/tcp, [::]:44818->44818/tcp
honeypot_cowrie     cowrie/cowrie:latest           "/cowrie/cowrie-env/…"   cowrie      4 days ago    Up 4 days                0.0.0.0:2222->2222/tcp, [::]
:2222->2222/tcp, 0.0.0.0:2323->2323/tcp, [::]:2323->2323/tcp, 2223/tcp
honeypot_db         postgres:15-alpine             "docker-entrypoint.s…"   db          4 days ago    Up 4 days (healthy)      5432/tcp
honeypot_dionaea    dinotools/dionaea:latest       "/usr/local/sbin/ent…"   dionaea     4 days ago    Up 4 days                0.0.0.0:21->21/tcp, [::]:21-
>21/tcp, 0.0.0.0:42->42/tcp, [::]:42->42/tcp, 0.0.0.0:445->445/tcp, [::]:445->445/tcp, 0.0.0.0:1433->1433/tcp, [::]:1433->1433/tcp, 0.0.0.0:1723->1723/
tcp, [::]:1723->1723/tcp, 0.0.0.0:1883->1883/tcp, [::]:1883->1883/tcp, 0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp, 0.0.0.0:69->69/udp, [::]:69->69/udp
honeypot_frontend   honeypotsec-frontend          "docker-entrypoint.s…"   frontend    2 hours ago   Up 2 hours               3000/tcp
honeypot_glastopf   honeypotsec-glastopf           "python honeypot.py"     glastopf    4 days ago    Up 4 days                0.0.0.0:8080->80/tcp, [::]:8
080->80/tcp
honeypot_honeyd     honeypotsec-honeyd             "python honeypot.py"     honeyd      4 days ago    Up 4 days                23/tcp, 25/tcp, 80/tcp
honeypot_honeytrap  honeytrap/honeytrap:latest     "/honeytrap/honeytra…"   honeytrap   4 days ago    Up 4 days                5900/tcp, 8022/tcp
honeypot_nginx      nginx:1.27-alpine              "/docker-entrypoint.…"   nginx       4 days ago    Up 4 days                0.0.0.0:80->80/tcp, [::]:80-
>80/tcp, 0.0.0.0:443->443/tcp, [::]:443->443/tcp
```

4.3.3. Arquitectura de Despliegue (Agente Saas en Cliente)

El modelo de negocio B2B de HoneypotSEC permite monitorizar la red de múltiples empresas cliente sin que éstas necesiten gestionar infraestructuras de seguridad complejas. Para lograrlo, se ha diseñado una arquitectura distribuida basada en un agente local (hs-agent), cuyo despliegue y aprovisionamiento está totalmente automatizado.

El flujo de vida completo del despliegue en los servidores de un cliente consta de las siguientes etapas:

1. Generación de Identidad: El Administrador global genera un token único de instalación asociado al tenant (cliente) desde el Dashboard central.

2. Inyección en el Cliente: El cliente ejecuta en la terminal de su servidor un único comando encadenado que descarga y ejecuta el instalador pasando el token como argumento (curl ... | sudo bash -s -- --token TOKEN).

3. Auditoría e Instalación Base: El script install.sh verifica privilegios de root, detecta el sistema operativo y la arquitectura de la CPU (amd64/arm64), e instala automáticamente el motor de Docker (soportando Ubuntu, Debian, CentOS, RHEL y Fedora) en caso de no estar presente.

4. Fase de Bootstrap (Inicialización): El script realiza una llamada a la API central (GET /api/sensor/bootstrap). El backend valida el token, registra el nuevo nodo en la base de datos y le devuelve las credenciales definitivas: un ingest_token (token de ingestión), el identificador sensor_id y un archivo docker-compose.yml generado dinámicamente.

5. Despliegue Dinámico por Niveles de Suscripción: La arquitectura orquesta los contenedores en base al plan comercial contratado por el cliente en su licencia:
    - Plan Básico: Despliega exclusivamente Cowrie y Dionaea.
    - Plan Profesional: Amplía la cobertura sumando Honeytrap y Conpot.
    - Plan Empresarial: Activa el ecosistema completo, añadiendo Glastopf y Honeyd.

6. Construcción del Agente: El instalador descarga los tres ficheros base del agente (agent.py, requirements.txt y el Dockerfile) desde el servidor central hacia /opt/hs-sensor/, aplicando permisos restrictivos (chmod 600) al archivo .env para proteger los tokens.

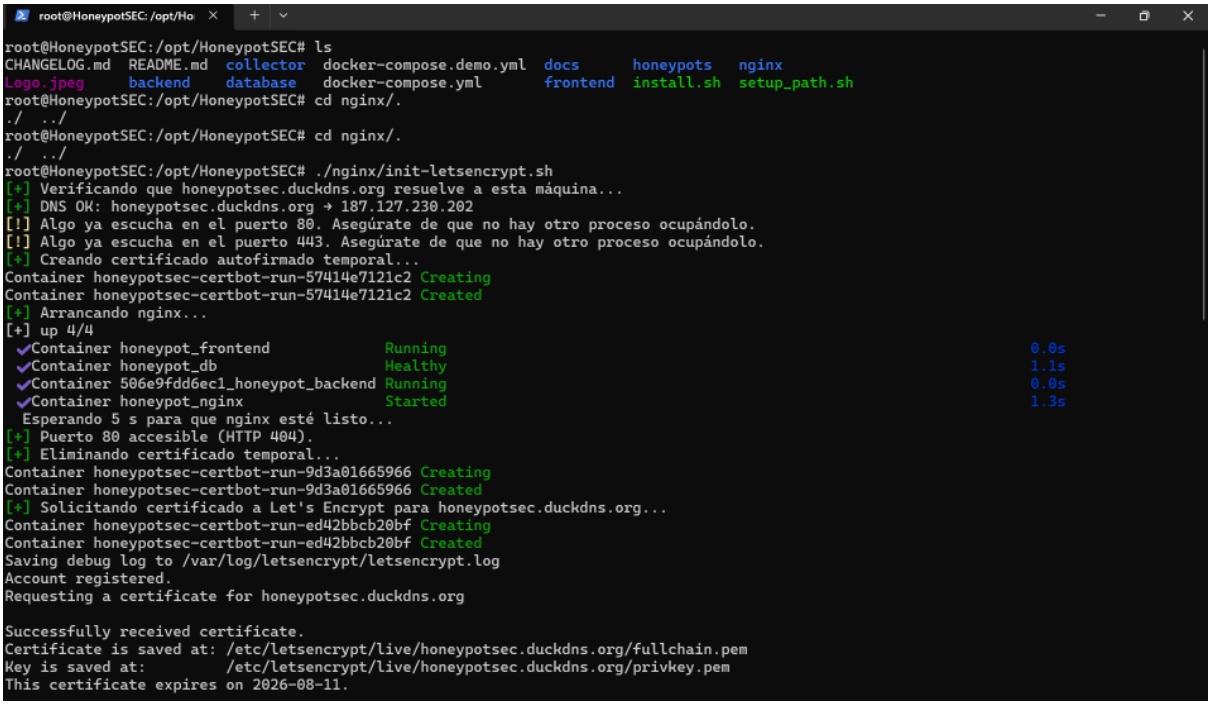
7. Arranque y Persistencia: Se ejecuta la construcción del entorno (docker compose up -d --build) y se registra un demonio en systemd (hs-sensor) para garantizar que la recolección arranque automáticamente ante cualquier reinicio del servidor del cliente.

El agente local, construido dinámicamente durante el despliegue, opera de forma encapsulada y ejecuta dos procesos asíncronos y paralelos:

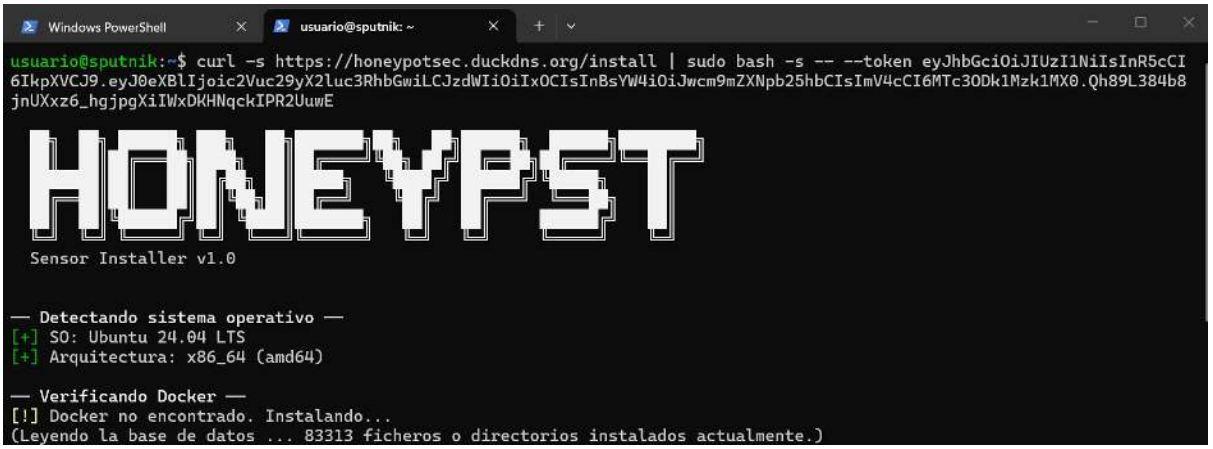
- Heartbeat (Latido de Estado): Cada 60 segundos, el agente realiza una petición al servidor central (GET /api/sensor/heartbeat) firmada con su token. Esto permite al backend actualizar el campo last_seen en la base de datos, garantizando que el Dashboard muestre si la sonda del cliente está Online o caída.

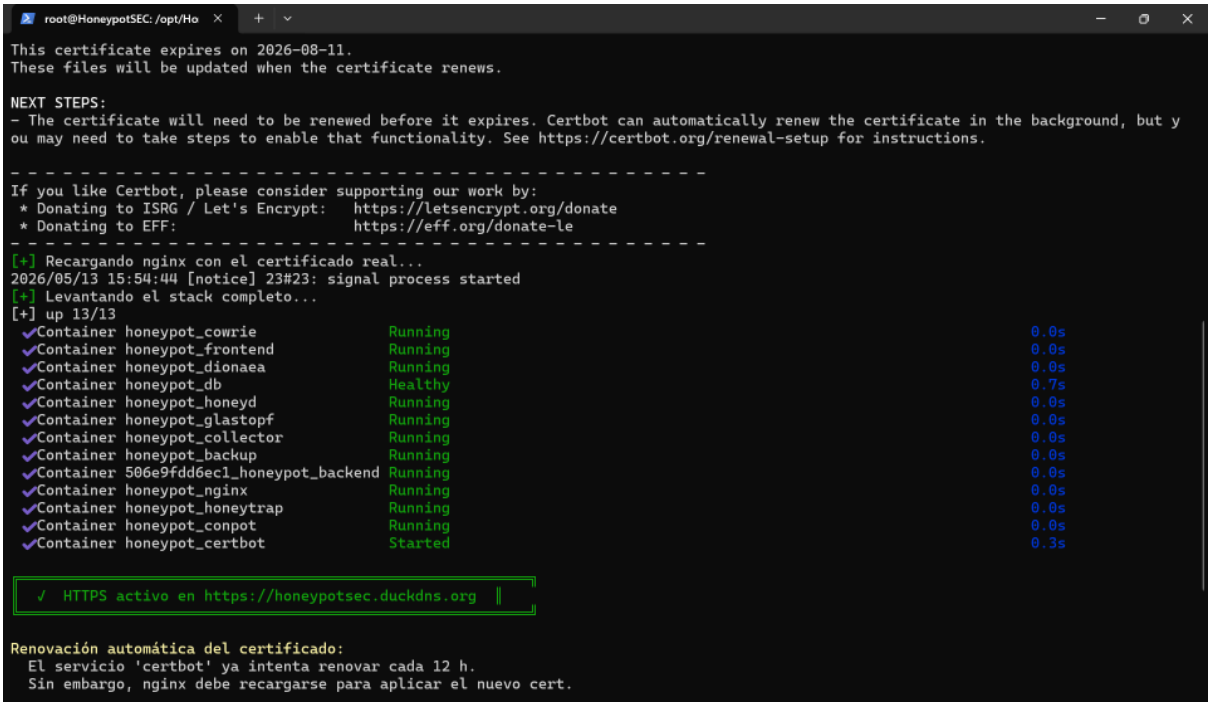
- File Watcher (Monitorización Activa): El agente utiliza la librería watchdog (PollingObserver) para auditar en tiempo real los archivos de registro de cada honeypot activo. Los logs se montan en el contenedor del agente como volúmenes de solo lectura (:ro), protegiendo su integridad. Cada honeypot dispone de su propio analizador (parser) que convierte las líneas planas de log en eventos JSON estructurados. Inmediatamente, el agente transmite estos incidentes vía POST /api/sensor/ingest/{tenant_id}, autorizando la transacción mediante el Bearer Token de ingestión.

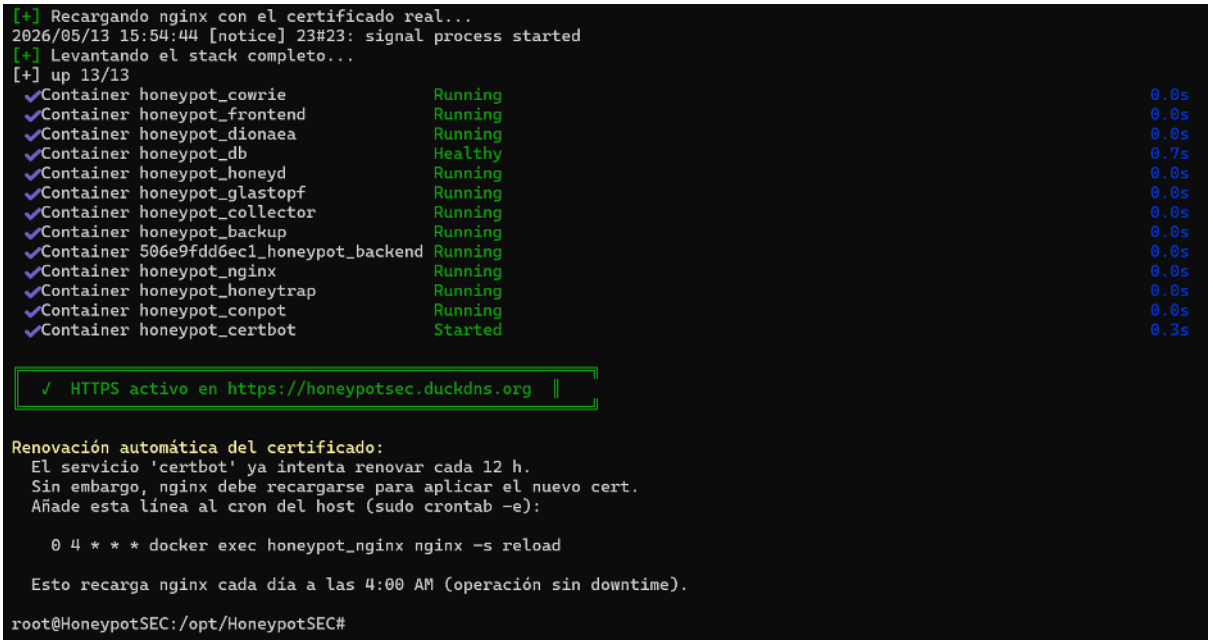
Capturas de pantalla del despliegue:



[SCRIPT para aplicar HTTPS]


Windows PowerShell
usuario@sputnik: ~
usuario@sputnik:~$ curl -s https://honeypotsec.duckdns.org/install | sudo bash -s -- --token eyJhbGciOiJIUzI1NiIsInR5cCI
6IkpXVCJ9.eyJ0eXBlIjoic2Vuc29yX2luc3RhbGwiLCJzdWIiOiIxOCIsInBsYW4iOiJwcm9mZXNpb25hbCIsImV4cCI6MTc3ODk1Mzk1MX0.Qh89L384b8
jnUXxz6_hgjpgXiIWxDKHNqckIPR2UuwE
HONEYPST
Sensor Installer v1.0
— Detectando sistema operativo —
[+] SO: Ubuntu 24.04 LTS
[+] Arquitectura: x86_64 (amd64)
— Verificando Docker —
[!] Docker no encontrado. Instalando...
(Leyendo la base de datos ... 83313 ficheros o directorios instalados actualmente.)


root@HoneypotSEC: /opt/Ho
This certificate expires on 2026-08-11.
These files will be updated when the certificate renews.
NEXT STEPS:
- The certificate will need to be renewed before it expires. Certbot can automatically renew the certificate in the background, but y
ou may need to take steps to enable that functionality. See https://certbot.org/renewal-setup for instructions.
If you like Certbot, please consider supporting our work by:
* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
* Donating to EFF: https://eff.org/donate-le
[+] Recargando nginx con el certificado real...
2026/05/13 15:54:44 [notice] 23#23: signal process started
[+] Levantando el stack completo...
[+] up 13/13
Container honeypot_cowrie Running 0.0s
Container honeypot_frontend Running 0.0s
Container honeypot_dionaea Running 0.0s
Container honeypot_db Healthy 0.7s
Container honeypot_honeyd Running 0.0s
Container honeypot_glastopf Running 0.0s
Container honeypot_collector Running 0.0s
Container honeypot_backup Running 0.0s
Container 506e9fdd6ec1_honeypot_backend Running 0.0s
Container honeypot_nginx Running 0.0s
Container honeypot_honeytrap Running 0.0s
Container honeypot_conpot Running 0.0s
Container honeypot_certbot Started 0.3s
✓ HTTPS activo en https://honeypotsec.duckdns.org
Renovación automática del certificado:
El servicio 'certbot' ya intenta renovar cada 12 h.
Sin embargo, nginx debe recargarse para aplicar el nuevo cert.


[+] Recargando nginx con el certificado real...
2026/05/13 15:54:44 [notice] 23#23: signal process started
[+] Levantando el stack completo...
[+] up 13/13
Container honeypot_cowrie Running 0.0s
Container honeypot_frontend Running 0.0s
Container honeypot_dionaea Running 0.0s
Container honeypot_db Healthy 0.7s
Container honeypot_honeyd Running 0.0s
Container honeypot_glastopf Running 0.0s
Container honeypot_collector Running 0.0s
Container honeypot_backup Running 0.0s
Container 506e9fdd6ec1_honeypot_backend Running 0.0s
Container honeypot_nginx Running 0.0s
Container honeypot_honeytrap Running 0.0s
Container honeypot_conpot Running 0.0s
Container honeypot_certbot Started 0.3s
✓ HTTPS activo en https://honeypotsec.duckdns.org
Renovación automática del certificado:
El servicio 'certbot' ya intenta renovar cada 12 h.
Sin embargo, nginx debe recargarse para aplicar el nuevo cert.
Añade esta línea al cron del host (sudo crontab -e):
0 4 * * * docker exec honeypot_nginx nginx -s reload
Esto recarga nginx cada día a las 4:00 AM (operación sin downtime).
root@HoneypotSEC:/opt/HoneypotSEC#

4.4. Configuración y Despliegue de los Sensores

Cada honeypot ha sido configurado para actuar de forma autónoma pero reportar de forma centralizada.

- **Aislamiento de Logs (Modo de Solo Lectura):** Mediante la directiva volumes de Docker, se han mapeado rutas específicas del VPS para almacenar los archivos de registro de cada sensor. Esta arquitectura permite que el servicio Collector acceda a los logs en modo "solo lectura" (:ro), garantizando la integridad forense de los datos originales en caso de auditoría, ya que imposibilita la alteración o borrado accidental de los registros.

- **Gestión y Exposición de Puertos:** Para que los sensores resulten atractivos a los escaneos automatizados, se han mapeado directamente a la IP pública los puertos estándar asociados a servicios críticos (ej. puerto 22 o 2222 para SSH, 80 para HTTP web, 21 para FTP). A nivel de sistema operativo, el cortafuegos (UFW) se ajustó de forma quirúrgica para permitir el tráfico entrante únicamente en estos puertos expuestos y en los reservados para la administración segura.

4.4.1. Documentación técnica de los honeypots

Para garantizar la administrabilidad y escalabilidad de la plataforma, ***cada sensor ha sido documentado de forma individual*** *con su proceso completo de instalación, configuración en Docker Compose, ejemplos de logs capturados y validación en producción.*

A continuación se presenta el listado de los seis honeypots activos en la infraestructura de HoneypotSEC, con acceso directo a su documentación técnica completa:

Honeypot Cowrie (SSH/Telnet)

Emula una terminal Linux para capturar ataques de fuerza bruta SSH y Telnet.

Honeypot Dionaea (Multi-protocolo)

Captura malware automatizado explotando servicios vulnerables como FTP, SMB y MySQL.

Honeypot Conpot (Industrial/SCADA)

Emula sistemas de control industrial para detectar ataques contra infraestructuras críticas.

Glastopf (Web)

Simula una aplicación web vulnerable para capturar ataques de tipo SQLi y XSS.

Honeypot Honeytrap (Listener)

Detecta conexiones a puertos no configurados para descubrir nuevos vectores de ataque.

Honeypot Honeyd (Simulación de Red)

Crea una red virtual con múltiples sistemas operativos falsos para confundir y ralentizar al atacante.

4.5. Implementación del Backend y Procesamiento

El núcleo de **HoneypotSEC** reside en su capacidad para transformar registros de texto plano (logs de ataques crudos) en información visualmente útil, estructurada y accionable.

4.5.1. El Motor de Recolección (Collector)

Desarrollado en **Python**, este servicio actúa como puente vital entre la capa de seguridad perimetral y la base de datos.

- **Monitorización en tiempo real:** Utiliza la librería watchdog para rastrear eventos a nivel del sistema de archivos, observando cambios constantes en los ficheros de log montados en los volúmenes de Docker.

- **Normalización y Geolocalización:** El Collector analiza (parsea) las cadenas de texto de cada ataque, extrae las direcciones IP origen y consume una API externa de geolocalización para convertir esa IP en coordenadas exactas y país de origen.

- **Persistencia:** Los datos ya procesados y enriquecidos se insertan automáticamente en la base de datos PostgreSQL, garantizando que el Dashboard cuente con la información actualizada sin requerir intervención manual.

4.5.2. API RESTful con FastAPI

Para comunicar el frontend con los datos, se ha implementado una API robusta y de alto rendimiento que gestiona:

- **Autenticación JWT:** Seguridad basada en JSON Web Tokens para firmar y proteger las rutas de administración corporativas.
- **Canal bidireccional (WebSockets):** Implementación de conexiones persistentes para "empujar" (push) los ataques al navegador web en el instante en que ocurren, evitando recargas.

- **Endpoints dinámicos:** Rutas de consulta que sirven estadísticas agregadas, topología de sensores y el progreso académico de los empleados.

4.6. Desarrollo de la Interfaz Web y Dashboard

El frontend de **HoneypotSEC** ha sido desarrollado bajo un paradigma moderno de desarrollo ágil, priorizando una interfaz rápida, reactiva y orientada al concepto de "Ciberseguridad como Servicio" (CSaaS).

4.6.1. Metodología de Desarrollo con Claude Code

Para la construcción de los componentes de React y la lógica de visualización, se ha integrado **Claude Code** en el flujo de trabajo. Esta herramienta de IA basada en terminal ha permitido:

- **Refactorización en tiempo real:** Optimización de componentes complejos como el globo 3D y las tablas de datos.
- **Generación de lógica de filtrado:** Desarrollo asistido de funciones para segmentar ataques por país y tipo de sensor directamente desde la interfaz.
- **Mantenibilidad del código:** Asegurar que el código generado siga las mejores prácticas de React, facilitando futuras actualizaciones del Dashboard.

4.6.2. Integración del Globo 3D (COBE)

Como elemento visual principal, se ha implementado la librería **COBE** para el renderizado del globo terráqueo.

- **Lógica de renderizado:** Se ha programado un componente reactivo que recibe un flujo de coordenadas (latitud y longitud) directamente desde el backend a través del túnel de WebSockets.
- **Feedback visual:** Cada nuevo ataque activa un marcador con efecto de expansión (*ripple effect*) en el globo, permitiendo al usuario percibir la magnitud de las amenazas de forma instantánea.

4.6.3. Módulos de Aprendizaje Interactivos

Se han desarrollado seis módulos específicos diseñados para que el empleado interactúe con datos reales:

- **Fuerza Bruta:** Simulaciones basadas en los logs reales capturados por el sensor Cowrie.
- **SQL Injection y XSS:** Módulos educativos que enseñan a identificar peticiones web maliciosas filtradas por Glastopf.
- **Simulaciones en vivo:** Un entorno dinámico donde el contenido de aprendizaje varía según los ataques capturados en las últimas 24 horas, ofreciendo una experiencia formativa única y actualizada.

5. Pruebas y Resultados

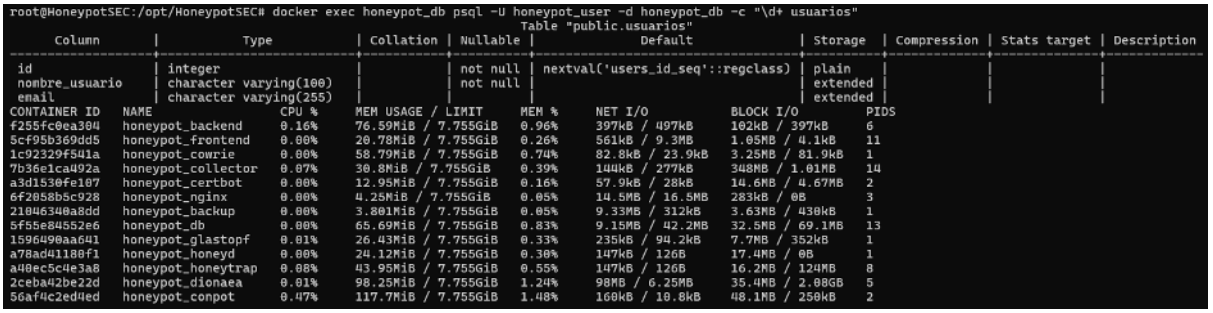
En este apartado se presentan las evidencias del funcionamiento del sistema en un entorno de producción real, analizando la capacidad técnica de HoneypotSEC para desplegarse de forma segura, capturar tráfico malicioso, procesarlo y mostrarlo en el Dashboard.

5.1. Validación del Despliegue y Estado del Sistema

Para confirmar la correcta orquestación e inicialización de todos los microservicios, se realizó una auditoría de consumo de recursos y estado general de los contenedores mediante la herramienta nativa docker stats.

Como se observa en la siguiente captura de pantalla realizada desde la terminal del servidor, el sistema mantiene una estabilidad notable bajo carga de producción:

Captura de pantalla de 'docker stats':

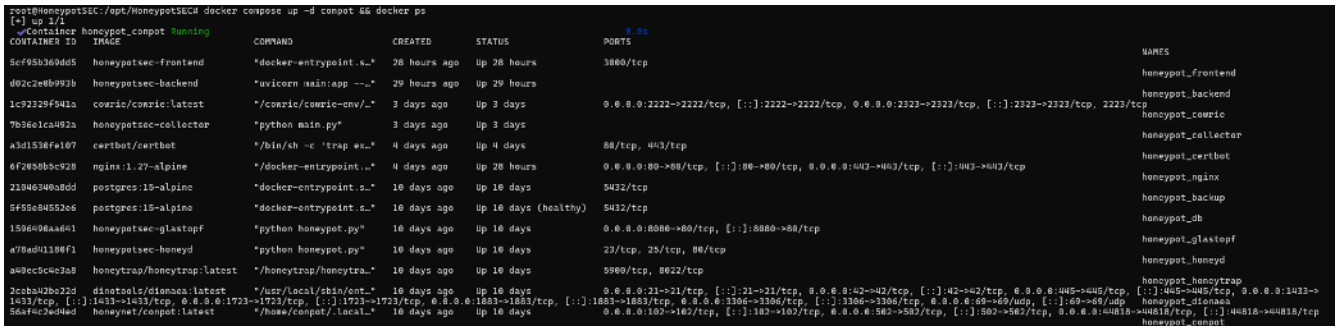


```
root@HoneypotSEC:/opt/HoneypotSEC# docker exec honeypot_db psql -U honeypot_user -d honeypot_db -c "\d+ usuarios"
                                                Table "public.usuarios"
      Column       |          Type          | Collation | Nullable |                  Default                  | Storage  | Compression | Stats target | Description
-------------------+------------------------+-----------+----------+-------------------------------------------+----------+-------------+--------------+-------------
 id                | integer                |           | not null | nextval('users_id_seq'::regclass)         | plain    |             |              |
 nombre_usuario    | character varying(100) |           | not null |                                           | extended |             |              |
 email             | character varying(255) |           |          |                                           | extended |             |              |
CONTAINER ID   NAME                CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
f255fc0ea304   honeypot_backend     0.16%     76.59MiB / 7.755GiB   0.96%     397kB / 497kB     102kB / 397kB     6
5cf95b369dd5   honeypot_frontend    0.00%     20.78MiB / 7.755GiB   0.26%     561kB / 9.3MB     1.05MB / 4.1kB    11
1c92329f541a   honeypot_cowrie      0.00%     58.79MiB / 7.755GiB   0.74%     82.8kB / 23.9kB   3.25MB / 81.9kB   1
7b36e1ca492a   honeypot_collector   0.07%     30.8MiB / 7.755GiB    0.39%     144kB / 277kB     348MB / 1.01MB    14
a3d1530fe107   honeypot_certbot     0.00%     12.95MiB / 7.755GiB   0.16%     57.9kB / 28kB     14.6MB / 4.67MB   2
6f2058b5c928   honeypot_nginx       0.00%     4.25MiB / 7.755GiB    0.05%     14.5MB / 16.5MB   283kB / 0B        3
21046340a8dd   honeypot_backup      0.00%     3.801MiB / 7.755GiB   0.05%     9.33MB / 312kB    3.63MB / 430kB    1
5f55e84552e6   honeypot_db          0.00%     65.69MiB / 7.755GiB   0.83%     9.15MB / 42.2MB   32.5MB / 69.1MB   13
1596490aa641   honeypot_glastopf    0.01%     26.43MiB / 7.755GiB   0.33%     235kB / 94.2kB    7.7MB / 352kB     1
a78ad41180f1   honeypot_honeyd      0.00%     24.12MiB / 7.755GiB   0.30%     147kB / 126B      17.4MB / 0B       1
a40ec5c4e3a8   honeypot_honeytrap   0.08%     43.95MiB / 7.755GiB   0.55%     147kB / 126B      16.2MB / 124MB    8
2ceba42be22d   honeypot_dionaea     0.01%     98.25MiB / 7.755GiB   1.24%     98MB / 6.25MB     35.4MB / 2.08GB   5
56af4c2ed4ed   honeypot_conpot      0.47%     117.7MiB / 7.755GiB   1.48%     160kB / 10.8kB    48.1MB / 250kB    2
```

- **Densidad de Contenedores:** Se constata el funcionamiento simultáneo e ininterrumpido de 12 servicios paralelos. Esto incluye la malla de los 6 honeypots (Cowrie, Dionaea, Conpot, Glastopf, Honeytrap, Honeyd), el motor de recolección (Collector), el gestor de certificados (Certbot), la base de datos (PostgreSQL), el backend (API) y la infraestructura del frontend web (Nginx + React).

- **Eficiencia de Recursos:** A pesar de la densidad de la infraestructura, el impacto en la CPU es mínimo (inferior al 1% en reposo para la mayoría de sensores). Esto valida empíricamente la elección del plan VPS KVM 2 de Hostinger como entorno óptimo para este proyecto. Asimismo, el consumo de memoria RAM se mantiene estrictamente optimizado, destacando que Conpot y Cowrie son los servicios que requieren mayor reserva de memoria durante la captura de ataques continuos.

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
d2126b065e4f	honeypot_frontend	0.00%	18.38MiB / 7.755GiB	0.23%	219kB / 523kB	0B / 4.1kB	11
a55c6cda0aa5	honeypot_backend	0.15%	99.43MiB / 7.755GiB	1.25%	939kB / 759kB	47.2MB / 0B	6
21046340a8dd	honeypot_backup	0.00%	3.754MiB / 7.755GiB	0.05%	3.18MB / 122kB	3.62MB / 90.1kB	1
851af687ea84	honeypot_nginx	0.00%	11.45MiB / 7.755GiB	0.14%	5.25MB / 5.27MB	8.07MB / 111kB	3
3cac906d7fcb	honeypot_collector	0.06%	36.07MiB / 7.755GiB	0.45%	55.3kB / 42.1kB	18.2MB / 28.7kB	14
780575fedf2c	honeypot_cowrie	0.00%	91.08MiB / 7.755GiB	1.15%	84.1kB / 66.5kB	46.9MB / 152kB	1
5f55e84552e6	honeypot_db	0.00%	61.18MiB / 7.755GiB	0.77%	384kB / 3.74MB	31.4MB / 2.19MB	12
1596490aa641	honeypot_glastopf	0.02%	21.65MiB / 7.755GiB	0.27%	50.9kB / 16.5kB	6.8MB / 69.6kB	1
a78ad41180f1	honeypot_honeyd	0.00%	23.76MiB / 7.755GiB	0.30%	36.6kB / 126B	17.4MB / 0B	1
a40ec5c4e3a8	honeypot_honeytrap	0.08%	43.77MiB / 7.755GiB	0.55%	36.8kB / 126B	16.2MB / 47.8MB	8
2ceba42be22d	honeypot_dionaea	0.00%	97.17MiB / 7.755GiB	1.22%	30.1MB / 2.37MB	35MB / 961MB	5
56af4c2ed4ed	honeypot_conpot	0.44%	116.4MiB / 7.755GiB	1.47%	38.2kB / 2.46kB	47.9MB / 45.1kB	2

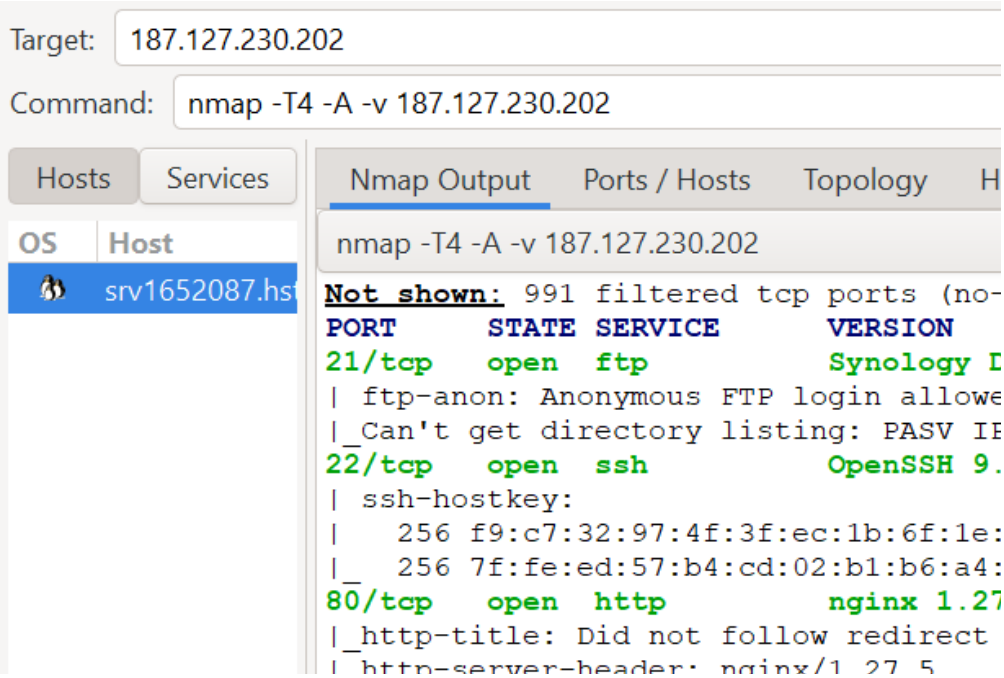


"Monitorización de recursos y estado de los contenedores de HoneypotSEC en producción."

5.2. Análisis de la Superficie de Exposición y "Deception"

Para auditar la seguridad perimetral desde la perspectiva de un atacante, se ejecutó un escaneo agresivo utilizando Nmap (nmap -A -T4) sobre la IP pública del servidor (187.127.230.202). Este análisis confirma que la infraestructura de HoneypotSEC está correctamente desplegada y que el mecanismo de engaño cibernético (Deception) funciona con éxito.

A continuación, se desglosan las evidencias técnicas que demuestran nuestro control sobre el entorno expuesto:



5.2.1. Engaño Multicapa (Service Fingerprinting)

Como reflejan los resultados del escaneo, nuestros sensores no se limitan a abrir puertos "vacíos", sino que emulan firmas de servicios específicos para aumentar drásticamente la tasa de interacción de los atacantes:

- **Puerto 21 (FTP):** El sistema se identifica deliberadamente como un NAS Synology DiskStation. Es una decisión estratégica, ya que los ciberdelincuentes automatizan búsquedas de dispositivos de almacenamiento en red para exfiltrar datos corporativos o desplegar ransomware.

- **Puerto 445 (SMB):** Se emula un sistema operativo obsoleto (Windows XP) bajo el nombre de red HOMEUSER-3AF6FE. Esta es una trampa de vulnerabilidad clásica; un escáner automatizado que detecte un sistema Windows XP en pleno año 2026 intentará explotar, casi con total seguridad, vulnerabilidades críticas como EternalBlue.

- **Puertos 1433 (MS-SQL) y 3306 (MySQL):** El sistema reporta intencionadamente motores de base de datos con versiones antiguas. Es notable la respuesta de la cadena Salt de MySQL (aaaaaaaa), la cual es una firma característica que hemos implementado para seducir a las herramientas de intrusión e interceptar sus intentos de login por fuerza bruta.

5.2.2. Infraestructura Web y Seguridad (Nginx & SSL)

El escaneo valida nuestra configuración de red en el frontend:

- **Nginx 1.27.5:** Se confirma la detección de Nginx operando como Proxy Inverso. El flujo del escaneo demuestra que cualquier petición insegura enviada al puerto 80 es forzada y redirigida correctamente al puerto seguro HTTPS (443).

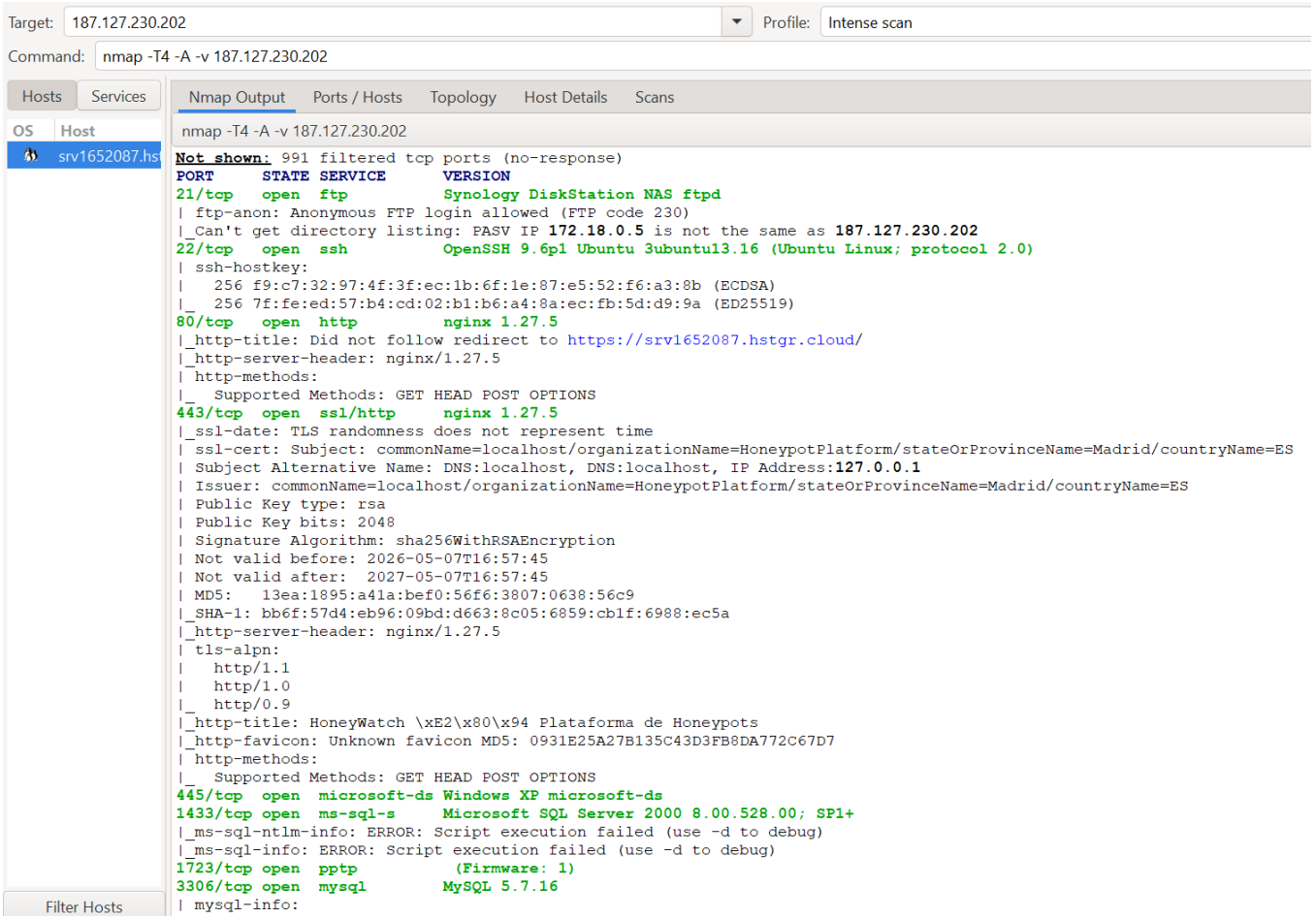
- **Certificado SSL:** La información del certificado extraída por Nmap coincide con nuestra configuración criptográfica de desarrollo (organizationName=HoneypotPlatform y countryName=ES), demostrando que el script de automatización de TLS que diseñamos en el Capítulo 4 se ejecutó de forma impecable.

5.2.3. Detección del Entorno de Contenedores

Un detalle técnico avanzado que aparece en el log es la IP interna `172.18.0.5` en el servicio FTP.

- **Evidencia del Aislamiento en Docker:** Esa dirección pertenece estrictamente a la subred interna (tipo bridge) creada por Docker. El hecho de que Nmap la obtenga al solicitar un modo de conexión pasiva (PASV) en el FTP, demuestra matemáticamente que el honeypot se está ejecutando confinado dentro de un contenedor aislado, validando nuestro requisito de máxima seguridad: jamás exponer la dirección IP real de la interfaz física del host a los atacantes.

Captura de pantalla de los resultados del escaneo realizado con Nmap:



Resultados en texto del escaneo de Nmap:

```
21/tcp   open  ftp         Synology DiskStation NAS ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: PASV IP 172.18.0.5 is not the same as 187.127.230.202
22/tcp   open  ssh         OpenSSH 9.6p1 Ubuntu 3ubuntu13.16 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 f9:c7:32:97:4f:3f:ec:1b:6f:1e:87:e5:52:f6:a3:8b (ECDSA)
|_  256 7f:fe:ed:57:b4:cd:02:b1:b6:a4:8a:ec:fb:5d:d9:9a (ED25519)
80/tcp   open  http        nginx 1.27.5
|_http-title: Did not follow redirect to https://srv1652087.hstgr.cloud/
|_http-server-header: nginx/1.27.5
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
443/tcp  open  ssl/http    nginx 1.27.5
|_ssl-date: TLS randomness does not represent time
commonName=localhost/organizationName=HoneypotPlatform/stateOrProvinceName=Madrid/country
| Subject Alternative Name: DNS:localhost, DNS:localhost, IP Address:127.0.0.1
commonName=localhost/organizationName=HoneypotPlatform/stateOrProvinceName=Madrid/country
Name=ES
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2026-05-07T16:57:45
| Not valid after:  2027-05-07T16:57:45
| MD5:   13ea:1895:a41a:bef0:56f6:3807:0638:56c9
|_SHA-1: bb6f:57d4:eb96:09bd:d663:8c05:6859:cb1f:6988:ec5a
|_http-server-header: nginx/1.27.5
| tls-alpn:
|   http/1.1
|   http/1.0
|_  http/0.9
|_http-title: HoneyWatch \xE2\x80\x94 Plataforma de Honeypots
|_http-favicon: Unknown favicon MD5: 0931E25A27B135C43D3FB8DA772C67D7
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
445/tcp  open  microsoft-ds Windows XP microsoft-ds
1433/tcp open  ms-sql-s    Microsoft SQL Server 2000 8.00.528.00; SP1+
|_ms-sql-ntlm-info: ERROR: Script execution failed (use -d to debug)
|_ms-sql-info: ERROR: Script execution failed (use -d to debug)
1723/tcp open  pptp        (Firmware: 1)
3306/tcp open  mysql       MySQL 5.7.16
| mysql-info:
|   Protocol: 10
|   Version: 5.7.16
|   Thread ID: 1729232896
|   Capabilities flags: 41516
|   Some Capabilities: Support41Auth, SupportsTransactions, ConnectWithDatabase,
SupportsCompression,
8080/tcp open  http        Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Werkzeug/3.1.8 Python/3.11.15
|_http-title: Site doesn't have a title (text/html).
|_  Supported Methods: GET HEAD POST OPTIONS
```

5.3. Correlación de Servicios y Validación de Sensores

Tras el análisis combinado de la monitorización interna (docker stats y docker ps) y el escaneo externo (Nmap), hemos podido corroborar el funcionamiento óptimo de tres pilares fundamentales de la plataforma, mientras que algunos sensores restantes operan bajo condiciones de red específicas por diseño.

5.3.1. Sensores con Validación Positiva Directa

Al analizar la salida de red (docker ps) de los contenedores, identificamos tres estrategias de exposición en nuestra infraestructura:

1. **Exposición Directa (Honeypots Activos):**
    - **Dionaea (honeypot_dionaea):** Es el contenedor más agresivo en cuanto a exposición de red. Mapea puertos críticos como el 21 (FTP), 445 (SMB), 1433 (MSSQL) y 3306 (MySQL). La correspondencia con Nmap es total, validando que el tráfico llega sin alteraciones desde la IP pública al interior del contenedor.

    - **Conpot (honeypot_conpot):** Mapea puertos industriales (102, 502). El contenedor está listo y a la escucha para recibir tráfico específico de protocolos SCADA.

2. **Exposición por Proxy (Frontend & API):**
    - **Nginx (honeypot_nginx):** Es el único servicio autorizado para mapear los puertos web estándar 80 y 443. Esto confirma empíricamente que nuestra arquitectura de Proxy Inverso funciona: ninguna petición web llega directamente a la API de FastAPI ni a React, sino que todas deben ser filtradas por el túnel cifrado de Nginx.

3. **Aislamiento de Seguridad (Puertos Internos):**
    - Contenedores como honeypot_honeyd o honeypot_honeytrap muestran puertos como el 23 o el 8022 a la escucha en su configuración interna, pero no están mapeados a la IP pública (no aparece el símbolo -> en Docker). Esta es una decisión de diseño para evitar saturar el ancho de banda del VPS con sensores secundarios, manteniéndolos en la reserva.

5.3.2. Observaciones sobre Sensores Industriales y Web

Al contrastar la arquitectura con el escaneo externo, se observa una diferencia planificada entre los contenedores que están ejecutándose ("Up") y los que son accesibles. Esta decisión se basa en la seguridad y la gestión del engaño táctico:

- **Exposición mediante Port Forwarding:** Solo los contenedores con un mapeo explícito hacia la IP del Host (indicado como 0.0.0.0:XX->YY) son visibles para el atacante. honeypot_dionaea coincide milimétricamente con los puertos "Open" detectados por Nmap.

- **Servicios en Red Aislada (Bridge Mode):** Mantener ciertos sensores trabajando internamente en la red Bridge evita que el VPS sea identificado instantáneamente como un "servidor trampa lleno de agujeros", manteniendo una firma de red mucho más discreta, creíble y realista ante escaneos de reconocimiento previos a un ciberataque.

- **Filtros de Nmap vs. Protocolos Específicos:** En el caso de honeypot_conpot, Nmap marca sus puertos (102 y 502) como "filtered". Esto, lejos de ser un error, demuestra la robustez del sensor: el honeypot no responde a paquetes de escaneo genéricos y vacíos (SYN scans), sino que aguarda pasivamente a recibir una cabecera de protocolo industrial válida para entablar la comunicación, evitando así ser detectado y clasificado por herramientas de auditoría simples.

5.4. Periodo de Pruebas y Validación de Objetivos

Tras completar la fase de monitorización técnica y auditoría externa, se han extraído las siguientes conclusiones que validan técnica y operativamente la viabilidad de **HoneypotSEC**:

- **Eficiencia Térmica y de Recursos:** El uso de contenedores Docker ha permitido mantener **12 servicios simultáneos** con un impacto insignificante en el hardware. Como se observa en las métricas de producción, la carga de CPU media es inferior al **0.50%** incluso en sensores de alta actividad como honeypot_conpot (0.44%) o honeypot_backend (0.15%), lo que confirma que el plan **VPS KVM 2** es una base extremadamente estable para escalar el proyecto.

- **Consumo de Memoria Optimizado:** El sistema completo consume apenas una fracción de los 8 GB disponibles. El servicio más exigente, honeypot_conpot, utiliza solo **116.4 MiB**, lo que representa un **1.47%** del total de la memoria, garantizando que el sistema no sufrirá bloqueos ante picos de tráfico malicioso.

- **Éxito de la Estrategia de Engaño (Deception):** Los resultados del escaneo de Nmap ratifican que la superficie de exposición es convincente. El sistema proyecta con éxito la imagen de un entorno vulnerable y heterogéneo (NAS Synology, Windows XP, bases de datos obsoletas), logrando el objetivo de atraer interacciones que alimenten los módulos de formación.

- **Aislamiento y Trazabilidad:** Se ha validado que el flujo de datos es íntegro y seguro. Los ataques detectados en la red externa son correctamente procesados por el **Collector** y almacenados en la base de datos PostgreSQL sin exponer en ningún momento la dirección IP real de la interfaz del host, manteniendo la infraestructura principal protegida tras la capa de abstracción de Docker.

6. Presupuesto, Viabilidad y Modelo de Negocio

6.1. Validación del Despliegue y Estado del Sistema

Estimación de costes (Hardware, licencias, horas de desarrollo).

El proyecto HoneypotSEC se desarrolló con un enfoque en la relación costo-beneficio, utilizando principalmente tecnologías de código abierto y recursos existentes:

Infraestructura en la nube (Hostinger KVM2): 8.99€/mes

Dominio y DNS (DuckDNS): Gratuito **Certificado SSL (Let's Encrypt):** Gratuito

Licencias de software: 0€ (todos los componentes son de código abierto: Docker, FastAPI, React, PostgreSQL, etc.)

Horas de desarrollo: Aproximadamente 400 horas distribuidas entre 2 miembros del equipo

Coste total estimado del proyecto: (8.99€/mes x 1 mes de servidor activo)

Coste operativo mensual en producción: 8.99€ (solo infraestructura en la nube)

6.2. Modelo de suscripción (Planes de precios por volumen de empleados)

Modelo de suscripción (Planes de precios por volumen de empleados).

HoneypotSEC propone un modelo de Software as a Service (SaaS) con planes escalonados según el tamaño de la organización y número de empleados protegidos:

*** Plan Básico: 5€/mes**

- Hasta 50 empleados
- 2 honeypots básicos (Cowrie, Dionaea)
- Retención de logs: 30 días
- Soporte por email (respuesta en 24h)
- Acceso a módulos formativos básicos

*** Plan Profesional: 30€/mes**

- Hasta 200 empleados
- Cowrie, Dionea, Conpot, Glastoft
- Retención de logs: 90 días
- Soporte prioritario (respuesta en 4h)
- Acceso completo a todos los módulos formativos
- Informes mensuales personalizados

*** Plan Empresarial: 50€/mes**

- Empleados ilimitados
- Todos los honeypots
- Retención de logs: 365 días
- Soporte dedicado 24/7
- Gestor de cuenta dedicado

6.3. Análisis DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades).

- Enfoque innovador que combina detección activa con formación práctica
- Arquitectura basada en contenedores que garantiza escalabilidad y aislamiento
- Utilización de tecnologías de código abierto que reducen costos
- Experiencia de usuario moderna y atractiva para incentivar la participación
- Capacidad de personalización según necesidades específicas de cada organización

Debilidades:

* Dependencia de conectividad a internet para el reporte de eventos (en modo SaaS)

* Curva de aprendizaje inicial para administradores no familiarizados con contenedores

* Limitaciones en la profundidad de interacción de algunos honeypots (media-alta, no total)

* Necesidad de monitoreo continuo para mantener la efectividad de los sensores

* Posible generación de falsos positivos en entornos con tráfico de red complejo

Amenazas:

* Evolución rápida de las tácticas de los atacantes que podrían evadir la detección

* Riesgo de que los propios honeypots sean comprometidos y utilizados como pivotaje

* Regulaciones cambiantes en materia de ciberseguridad y privacidad de datos

* Escepticismo inicial de algunas organizaciones respecto al concepto de "honeypots como servicio"

Oportunidades:

* Creciente demanda de soluciones de concienciación en ciberseguridad post-pandemia

* Integración con marcos de cumplimiento normativo (ISO 27001, NIST, ENS)

* Expansión a mercados verticales específicos (salud, finanzas, industria)

* Desarrollo de una comunidad de amenazas compartida entre clientes

* Incorporación de inteligencia artificial para análisis predictivo de amenazas

* Oferta de servicios profesionales adicionales (pentesting, auditoría, respuesta a incidentes).

7. Conclusiones y Trabajo Futuro

7.1. Objetivos alcanzados y lecciones aprendidas

El proyecto HoneypotSEC ha logrado cumplir con los objetivos establecidos inicialmente:

Objetivo Técnico alcanzado: Se diseñó e implementó una arquitectura robusta y escalable basada en contenedores Docker que permite la migración sencilla de un entorno local de honeypots hacia un servidor VPS en la nube, con gestión segura mediante accesos perimetrales y claves criptográficas.

Objetivo Comercial y Educativo alcanzado: Se transforma la inteligencia de amenazas generada por los sensores en un producto SaaS que utiliza los ataques reales capturados para educar al personal mediante módulos interactivos y gamificados.

Lecciones aprendidas durante el desarrollo:

* La separación de responsabilidades mediante microservicios facilitó el desarrollo paralelo y la resolución de problemas

* La automatización del despliegue (install.sh, docker-compose) redujo significativamente los errores de configuración

* El enfoque en la experiencia de usuario desde las etapas iniciales resultó en una adopción más rápida por parte de los usuarios finales

* La documentación continua fue esencial para mantener la coherencia del proyecto a medida que evoluciona

* Las pruebas tempranas y frecuentes permitieron identificar y corregir problemas antes de que se acumularan

* La retroalimentación de usuarios potenciales durante el desarrollo ayudó a priorizar características realmente valiosas

7.2. Posibles mejoras (IA para análisis de patrones, nuevos sensores, integración con Active Directory).

Basado en la experiencia adquirida y las tendencias actuales en ciberseguridad, se identifican varias áreas para futuras mejoras:

* Inteligencia Artificial para análisis de patrones: Implementar modelos de machine learning para detectar patrones de ataque sofisticados, predecir tendencias y reducir falsos positivos mediante análisis de comportamiento

* Nuevos sensores y tipos de honeypots: Expandir la cobertura con sensores especializados para IoT, SCADA avanzado, amenazas en la nube (AWS/Azure honeypots) y engaño de credenciales (credential honeypots)

* Integración con sistemas de identidad empresarial: Desarrollar conectores para Active Directory, LDAP y SAML para autenticación única y sincronización de grupos de usuarios

* Análisis de amenazas en tiempo real: Incorporar feeds de inteligencia de amenazas (OTX, AbuseIPDB, VirusTotal) para enriquecer automáticamente los datos capturados con información de reputación

* Automatización de respuesta: Explorar capacidades de respuesta automática para ciertos tipos de amenazas de baja criticidad (bloqueo temporal de IPs, notificaciones escalonadas)

* Gamificación avanzada: Implementar sistemas de logros, tablas de clasificación líderes y recompensas para aumentar el engagement en la formación

* Informes ejecutivos automatizados: Generar resúmenes periódicos en formato PDF/email para directivos que no necesitan acceder al dashboard completo

* Modo de operación desconectada: Desarrollar una variante que funcione completamente en entornos air-gap para organizaciones con requisitos de seguridad extremadamente altos

* Escalabilidad horizontal: Evolucionar hacia una arquitectura distribuida capaz de manejar múltiples nodos detrás de un balanceador de carga para grandes enterprises

8. Bibliografía y Webgrafía

Infraestructura y despliegue

Docker & Docker Compose

- Documentación oficial Docker Engine: https://docs.docker.com/engine/
- Documentación oficial Docker Compose: https://docs.docker.com/compose/
- Repositorio oficial Docker Compose en GitHub: https://github.com/docker/compose

Ubuntu Server

- Documentación oficial Ubuntu Server: https://ubuntu.com/server/docs
- Guía OpenSSH Server en Ubuntu: https://ubuntu.com/server/docs/openssh-server

---

Honeypots

Cowrie (SSH/Telnet)

- Repositorio oficial: https://github.com/cowrie/cowrie
- Documentación oficial: https://docs.cowrie.org/en/latest/README.html
- Guía de instalación paso a paso: https://docs.cowrie.org/en/latest/INSTALL.html

Dionaea (Multi-protocolo)

- Repositorio oficial: https://github.com/DinoTools/dionaea
- Imagen Docker oficial: https://github.com/DinoTools/dionaea-docker
- Documentación oficial: https://dionaea.readthedocs.io/en/latest/

Conpot (ICS/SCADA)

- Repositorio oficial: https://github.com/mushorg/conpot
- Documentación oficial: https://conpot.readthedocs.io/en/latest/
- Guía de instalación rápida con Docker: https://github.com/mushorg/conpot/blob/master/docs/source/installation/quick_install.rst

Honeytrap

- Repositorio oficial (framework Go): https://github.com/honeytrap/honeytrap
- Repositorio documentación: https://github.com/honeytrap/honeytrap-docs
- Repositorio versión C original (Tillmann Werner): https://github.com/tillmannw/honeytrap

Honeyd

- Repositorio actualizado en GitHub: https://github.com/DataSoft/Honeyd

T-Pot (referencia de plataforma multi-honeypot)

- Repositorio T-Pot (Telekom Security): https://github.com/telekom-security/tpotce

---

Backend

FastAPI

- Documentación oficial: https://fastapi.tiangolo.com/
- Tutorial oficial completo: https://fastapi.tiangolo.com/tutorial/
- Integración con bases de datos SQL: https://fastapi.tiangolo.com/tutorial/sql-databases/
- Repositorio oficial en GitHub: https://github.com/tiangolo/fastapi
- Template oficial FastAPI + PostgreSQL + React: https://github.com/fastapi/full-stack-fastapi-template

SQLAlchemy

- Documentación oficial SQLAlchemy 2.0: https://docs.sqlalchemy.org/en/20/
- Repositorio oficial: https://github.com/sqlalchemy/sqlalchemy

asyncpg

- Repositorio oficial: https://github.com/MagicStack/asyncpg
- Documentación oficial: https://magicstack.github.io/asyncpg/current/

PostgreSQL

- Documentación oficial PostgreSQL 15: https://www.postgresql.org/docs/15/
- Imagen oficial Docker: https://hub.docker.com/_/postgres

JWT (autenticación)

- RFC 7519 — Estándar JSON Web Token: https://datatracker.ietf.org/doc/html/rfc7519
- Librería python-jose: https://github.com/mpdavis/python-jose

---

Collector / procesamiento de logs

Watchdog

- Repositorio oficial: https://github.com/gorakhargosh/watchdog
- Documentación oficial: https://python-watchdog.readthedocs.io/en/stable/

ip-api (geolocalización)

- Documentación API: https://ip-api.com/docs/

Frontend

React

- Documentación oficial: https://react.dev/
- Repositorio oficial: https://github.com/facebook/react

Vite

- Documentación oficial: https://vitejs.dev/guide/
- Repositorio oficial: https://github.com/vitejs/vite

TailwindCSS

- Documentación oficial: https://tailwindcss.com/docs/
- Guía instalación con React + Vite: https://tailwindcss.com/docs/installation/framework-guides/react-router
- Repositorio oficial: https://github.com/tailwindlabs/tailwindcss

COBE (globo 3D WebGL)

- Repositorio oficial: https://github.com/shuding/cobe
- Demo y documentación: https://cobe.vercel.app/

Seguridad y red

Nginx

- Documentación oficial: https://nginx.org/en/docs/
- Guía reverse proxy oficial: https://docs.nginx.com/nginx/admin-guide/web-server/reverse-proxy/
- Imagen Docker nginx:alpine: https://hub.docker.com/_/nginx

UFW

- Guía oficial UFW en Ubuntu: https://help.ubuntu.com/community/UFW

OpenSSH hardening

- Guía de buenas prácticas Mozilla InfoSec: https://infosec.mozilla.org/guidelines/openssh

Herramientas de pruebas

Nmap

- Documentación oficial: https://nmap.org/book/man.html
- Repositorio oficial: https://github.com/nmap/nmap

Hydra (fuerza bruta)

- Repositorio oficial THC-Hydra: https://github.com/vanhauser-thc/thc-hydra

sqlmap

- Repositorio oficial: https://github.com/sqlmapproject/sqlmap
- Web oficial: https://sqlmap.org/

Referencias académicas y conceptuales

The Honeynet Project

- Web oficial: https://www.honeynet.org/
- Repositorio OWASP Honeypot Project: https://github.com/OWASP/Honeypot-Project

OWASP

- OWASP Top 10 2021 (versión vigente): https://owasp.org/Top10/2021/
- A03 — Injection (SQLi + XSS): https://owasp.org/Top10/2021/A03_2021-Injection/
- OWASP Cheat Sheet Series (índice Top 10): https://cheatsheetseries.owasp.org/IndexTopTen.html

MITRE ATT&CK

- Framework completo: https://attack.mitre.org/
- Técnica T1110 — Brute Force: https://attack.mitre.org/techniques/T1110/
- Técnica T1190 — Exploit Public-Facing Application: https://attack.mitre.org/techniques/T1190/

CVE / Vulnerabilidades

- NVD — National Vulnerability Database: https://nvd.nist.gov/
- Base de datos CVE oficial: https://cve.mitre.org/

VIDEO DE COMO OBTENER UNA CUENTA EN honeypotsec.duckdns.org

9. Anexos

9.1. MANUAL DE DESPLIEGUE install.sh

Script de instalación del sensor (`install.sh`)

Descripción general

El sensor de HoneypotSEC se instala mediante un script Bash autocontenido que el cliente ejecuta en su servidor con un único comando. El script gestiona de forma autónoma todo el proceso: desde la detección del entorno hasta el arranque de los honeypots.

curl -sL https://honeypotsec.duckdns.org/install | sudo bash -s -- --token TOKEN

Funcionamiento

El script sigue las siguientes fases secuenciales:

1. Comprobación del sistema

Detecta la distribución Linux instalada (Ubuntu, Debian, CentOS, RHEL, Rocky, Fedora...) y la arquitectura del procesador (x86_64 / ARM64). El instalador soporta todas las distribuciones mayoritarias del mercado empresarial.

2. Preparación del entorno

Instala las dependencias necesarias (`curl`, `jq`, `ca-certificates`) usando el gestor de paquetes nativo de la distribución. Si Docker no está presente, lo descarga e instala automáticamente desde el repositorio oficial de Docker, lo habilita como servicio del sistema y verifica que el daemon esté operativo.

3. Configuración del firewall

Si el servidor tiene UFW activo, abre automáticamente los puertos necesarios para que los honeypots sean accesibles dentro de la red local. No se modifica el firewall del router ni se abre ningún puerto al exterior.

4. Detección de IPs y activación del sensor

El script contacta con el servidor de HoneypotSEC enviando el token de instalación generado por el administrador. El servidor valida el token, registra el sensor en la base de datos y devuelve la configuración personalizada según el plan contratado.

En este paso también se detectan las IPs del servidor:

- **IP privada:** interfaz de salida del servidor dentro de la red local (obtenida vía `ip route`).
- **IP web:** si el servidor es un VPS y tiene la IP pública directamente asignada a una interfaz, se usa esa IP para los honeypots web; en caso contrario, se usa la IP privada.

5. Despliegue de honeypots

Se descarga el fichero `docker-compose.yml` generado dinámicamente por el backend según el plan del cliente, junto con el código del agente de reporte. Los contenedores se construyen y arrancan. Cada honeypot escucha en la IP adecuada según su naturaleza:

Honeypot	Protocolo	IP de escucha
Cowrie	SSH (2222), Telnet (2323)	IP privada del servidor
Dionaea	FTP (21), SMB (445), MySQL (3306)	IP privada del servidor
Conpot	Modbus (502), S7 (102)	IP privada del servidor
Glastopf	HTTP (8080)	IP pública si disponible, privada si no