

Honeytokens: La Trampa Invisible para Detectar Ciberataques

¿Qué son los Honeytokens?

1

Datos o credenciales falsas

Son señuelos estratégicamente ubicados en una infraestructura.

2

Cuando un atacante usa un honeypot, se activa una alerta inmediata y silenciosa, notificando la intrusión.

3

Son una forma **sencilla y efectiva** de detectar intrusiones y movimientos dentro de la red.



Honeytokens vs Honeypots: Diferencias Clave

1 Honeypots: Sistemas Señuelo Completos



Los **Honeypots** son sistemas o servidores falsos que simulan activos reales, como bases de datos, aplicaciones o redes enteras. Su propósito es **atraer y engañar a los atacantes**, convenciéndolos de que están interactuando con un objetivo legítimo.

2 Honeytokens: Datos Falsos Incrustados



Los **Honeytokens** son datos falsos específicos incrustados en sistemas reales, como archivos, credenciales API o claves. Son **más fáciles de desplegar y escalar**, ya que no requieren la creación de infraestructuras completas.

¿Por qué usar Honeytokens?

Impacto y Ventajas

1 Detección Ultra-Rápida

Detectan violaciones en **minutos**.

2 Rastreo Detallado del Atacante

Permiten rastrear al atacante a través de indicadores como su dirección IP, el tiempo de la acción (**timestamp**) y las acciones específicas realizadas.

3 Detectan lo que normalmente no se ve

Firewalls, IDS y antivirus buscan *patrones conocidos*.

Un honeytoken detecta algo distinto:

Uso indebido de información que nunca debería usarse.

Si un token se activa, **no hay falsos positivos**: alguien accedió a algo que no debía.





Cómo esperamos que funcione nuestro sistema ?



Usuario

El **usuario** hace uso de una aplicación de línea de comandos que le permita generar diferentes tipos de honeytokens.



CLI

Debe interactuar con el servidor de alertas para registrar un nuevo token y obtener una URL única de **callback** para incrustar en el archivo pedido.

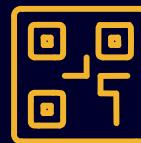


Servidor

Un servidor web simple. Cuando un token es activado, el servidor debe **registrar el evento** en un log o en la consola, incluyendo información como la fecha/hora, la dirección IP de origen y el tipo de token.



¿Qué HoneyTokens Implementamos?



URL/QR



WebImage



Fake Credentials



Archivo Microsoft Word (.docx)

URL/QR

Es una **dirección web controlada por nosotros** que **no debería usarse nunca** en condiciones normales.

- La ponemos en un lugar “tentador” (un PDF, un repo privado, un mail interno).
- Si alguien hace clic o la abre:
 - El servidor **registra el acceso** (IP, time-stamp).
 - Sabemos que **alguien no autorizado** llegó ahí.

Es lo mismo, pero **embebido en un código QR**.

- El QR apunta a un **URL honeypot**.
- Lo imprimimos o lo dejamos en un documento.
- Si alguien lo escanea:
 - Se accede al URL.
 - Se genera la **alerta automática**.



Weblimage



Una **web image honeypot** es una **imagen señuelo** incrustada en una página web, correo, documento o sistema, cuyo **único propósito es detectar accesos no autorizados**.

La imagen apunta a una **URL controlada por nosotros**. Si alguien intenta cargarla, el servidor registra el evento como **actividad sospechosa**.

- Se crea una imagen (real o ficticia).
- Esa imagen tiene una **URL única y secreta**
- Se puede incrustar en:
 - HTML ()
 - Emails
 - PDFs
 - Documentación interna
- **Nadie debería acceder a esa URL naturalmente.**
- Si se accede:
 - Se registra IP, hora,...
 - Se genera una **alerta**.

Fake Credentials

Credenciales deliberadamente falsas, plantadas para que nadie legítimo las use. Pueden ser cualquier tipo de credencial:

- Usuario / contraseña (la que implementamos nosotros)
- API keys
- Tokens JWT

¿Dónde se “plantan”?

Las fake credentials se colocan **intencionalmente** en lugares donde:

- ✗ No deberían usarse
- ✓ Un atacante las buscaría

Por ejemplo:

- Repositorios Git

¿Cómo funcionan?

1. Se crean credenciales falsas **con formato real**
2. Se registran para monitoreo (ej. sistema propio)
3. Se “filtran” o se dejan accesibles
4. Un atacante las prueba
5. El sistema detecta el uso
6. ⚡ Alerta inmediata

Un usuario legítimo **no tiene motivo** para usarlas.



Microsoft Word Token (.docx)

Es un archivo de Microsoft Word (.docx) aparentemente normal y legítimo, que contiene un **elemento de rastreo oculto**.

¿Cómo Funciona Técnicamente?

El Mecanismo: Se basa en cómo Word maneja las **Imágenes Vinculadas**.

Explicación Técnica:

En lugar de *incrutar* una imagen dentro del archivo (guardar los bytes en el zip), le decimos a Word que la imagen está en una **URL externa**.

Cuando el documento se abre, el motor de renderizado de Word intenta descargar esa imagen automáticamente para mostrarla, activando así el mecanismo call-Home.



Desafíos y limitaciones encontradas

1. Funcionamiento solo local

- ✖ Limitación: solo funciona en localhost:9999
- ⚠ No accesible desde dispositivos externos o internet

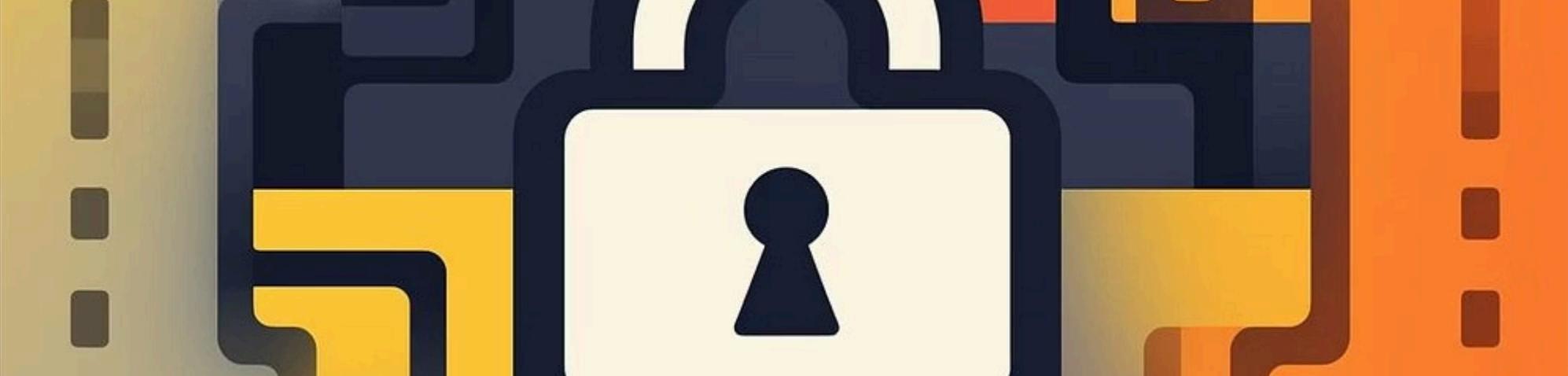
2. PDF: compatibilidad limitada

- ✖ JavaScript automático solo funciona en Adobe Acrobat Reader
- ⚠ Otros lectores (Firefox, Chrome) no ejecutan JS embebido

3. Word

- ✖ Funciona únicamente con Microsoft word
- ⚠ Al intentar buscar la imagen y no tener una respuesta, Microsoft word intenta muchas veces ([retries](#)), activando el mismo token y mandando muchos mails a la vez.





Cómo implementamos las responsabilidades de
cada componente ? PoC



Server (Controllers)

✓ TokenCreationController

Recibe una petición desde el CLI para crear un honeypoint, elige la estrategia correcta según el tipo, lo crea y devuelve la URL única del nuevo token.

✓ TokenDetectionController

Define un endpoint que detecta cuando alguien activa un honeypoint, lo registra y responde según el tipo.

✓ FakeLoginController

Implementa un login falso (honeypot) que detecta cuando alguien intenta usar como contraseña un *Credential Honeytoken*.

> CLI (Commands).

A partir de un **comando de Spring Shell** (generate):

✓ URL

```
generate url --mail usuario@mail.com --message "texto"
```

✓ QR

```
generate qr --mail usuario@mail.com --message "texto"
```

✓ WebImage (**webImageTokenService** genera el token basado en una imagen)

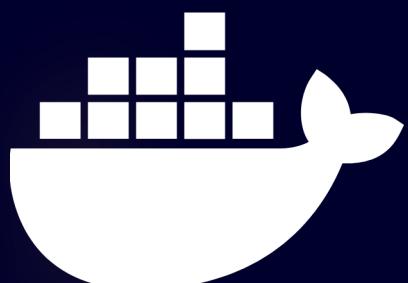
```
generate webImage --mail usuario@mail.com --message "texto" --image foto.png
```

✓ Word (.docx)

```
generate word --mail alguien@example.com --message "texto"
```

✓ Credentials (**credentialsTokenService** genera un **token de credenciales falsas**)

```
generate Credentials --mail usuario@mail.com --nameFile secret.json
```



Cómo hacemos que it-works-on- my-machine no sea un problema ?

1. Hacemos el build desde el directorio raiz.

```
sudo docker-compose up --build
```

*En caso de no tener el docker-compose ->

```
sudo apt install docker-compose
```

2. Cuando termina el build, vamos a tener el server corriendo, tenemos que hacer el attach desde otra terminal

```
sudo docker attach honey-cli
```

Para luego poder hacer uso de la CLI.

Integrantes



Milagros Lucía Peris



Marco Romano Fina



Tomás F Melli



Victoria Espil