¿Qué es un Honeytoken o Canary Token?

Un honeytoken es un sistema falso que se implementa junto a tus activos digitales genuinos. Está diseñado para resultar atractivo a un atacante, y cuando el atacante cae en la trampa, no solo desperdicia sus recursos en un sistema inútil o datos falsos, sino que también revela información crucial sobre la naturaleza de su estrategia de ataque. No cumplen un rol real en el sistema, únicamente existe para alertar de potenciales accesos no autorizados.

No todos los honeytokens hacen call home por sí mismos de la misma forma: algunos son activos (realizan una petición automática al abrirse o ejecutarse — ej. PDF con recurso remoto, binario, página web con <img>), mientras que otros son pasivos/indirectos y solo generan señal cuando un atacante intenta usarlos (p. ej. credenciales, emails o API keys falsas que son probadas contra un servicio).

Bibliografía consultada:
https://developer.nvidia.com/blog/defending-ai-model-files-from-unauthorized-access-with-canaries/
https://www.crowdstrike.com/en-us/cybersecurity-101/identity-protection/honeytokens/
https://www.sentinelone.com/cybersecurity-101/cybersecurity/honeytokens/
https://www.fortinet.com/resources/cyberglossary/honey-tokens

Implementaciones de tipos de Honey Token:

1. **QR**

Lenguaje: Python

Idea: Generar un código QR que apunte a una URL única del servidor (la de nuestro TokenSnare). Cuando alguien lo escanee y se abra el link, el servidor registra la activación con la IP y la hora.

Fuente consultada:
https://stackoverflow.com/questions/66672150/how-to-generate-qr-code-with-python-and-when-scanned-make-it-open-a-url-defined

2. **Binario**

Lenguaje: Python / C

Idea: Crear un ejecutable que al correrse haga una petición HTTP (o DNS) a una URL única del servidor (la de nuestro TokenSnare). Cuando alguien lo ejecute, el backend registra la activación (IP, timestamp y id del token). Para pruebas se puede generar fácilmente con un pequeño script (requests/curl) y empaquetarlo con pyinstaller o compilar en C para obtener un binario independiente.

3. **Mail**

Lenguaje: Python / JavaScript

Idea: Usar el email pixel tracking. Es una imagen cuadrada de 1 × 1 px creada mediante una línea de código insertada en un mensaje de correo electrónico. El destinatario no percibe la presencia de los píxeles de seguimiento, ya que suelen ser transparentes y estar ubicados discretamente en el encabezado o pie de página del correo electrónico. mandar un mail en HTML con una imagen remota 1×1 (<img src="https://mi-servidor/token/<id>"/>). Cuando alguien abre el email y el cliente carga la imagen, tu servidor recibe la URL única y registra la activación

Fuente consultada:
https://www.nutshell.com/blog/email-tracking-pixels-101-how-do-tracking-pixels-work

4. **Excel**

Lenguaje: Python

Idea: Utilizar la función WEBSERVICE, que devuelve datos de un servicio web en Internet o Intranet. problemática: necesita algún mecanismo para que recalcule cada vez que se accede (posible solución utilizar un timestamp en la función). Otra idea es hacer una consulta powerQuery a una URL única en nuestro servidor Token Snare, que esté programado para que se refresque cada vez que se abra. Problemática: puede pedir refrescar credenciales en ciertas corporaciones. Se puede implementar con pandas o openpyxl.

Fuente consultada:

https://www.geeksforgeeks.org/python/working-with-excel-files-using-pandas

https://openpyxl.readthedocs.io/en/stable

https://support.microsoft.com/en-us/office/webservice-function-0546a35a-ecc6-4739-aed7-c0b7ce1562c4

https://community.fabric.microsoft.com/t5/Power-Query/Refresh-Query-automatically-Excel-PowerQuery/td-p/2886464

5. **Página web**

Lenguaje: JavaScript y HTML + alguna forma de clonar la página.

Idea: La URL o un directorio local y replicar la página (por ejemplo usando pywebcopy). Después, procesar todos los archivos HTML descargados e inyectar en cada <head> o al final del <body> un JavaScript que haga un GET a una URL única del servidor de Token Snare. Cuando alguien abra el sitio clonado en su navegador, el script realizará la petición y el backend registrará la activación (IP, timestamp, user-agent, id).

6. **Docs**

Lenguaje: Python

Idea: Usar una imagen linkeada (no embebida) a https://mi-servidor-token-snare/token/<id>. Al abrir el DOCX, Word intenta cargar la imagen remota y el backend registra la activación (IP, hora, user-agent, id). Nota: depende de “Actualizar vínculos automáticos al abrir” y de políticas de contenido externo.

Fuente consultada:
https://superuser.com/questions/38870/in-microsoft-word-how-can-i-link-to-an-image-from-the-web-which-updates

https://python-docx.readthedocs.io/en/latest

7. **PDF**

Lenguaje: -

Idea: Crear un PDF que intente “llamar a casa” al abrirlo, usando JavaScript (app.launchURL("https://mi-servidor/token/<id>", true)) como OpenAction. Como muchos visores bloquean el JS o piden confirmación, además agregamos un hipervínculo (visible o invisible) con acción /URI al mismo endpoint para registrar clics. Cuando el JS se ejecuta o alguien hace clic, el servidor recibe la URL única y registra la activación (IP, hora, id). Nota: la ejecución automática de JS depende del visor (Acrobat/Reader suele preguntar; Chrome/Edge suelen bloquear).

Fuente consultada:
https://stackoverflow.com/questions/5501876/link-a-pdf-to-open-in-a-new-tab-from-a-pdf
https://pypdf.readthedocs.io/en/latest/user/add-javascript.html

https://opensource.adobe.com/dc-acrobat-sdk-docs/library/jsapiref/index.html#overview

https://opensource.adobe.com/dc-acrobat-sdk-docs/library/jsapiref/JS_API_AcroJS.html#launchurl

8. **Mysql Dump**

Lenguaje: Python

Idea: Cuando un atacante intenta restaurar un dump, hacer que su servidor mysql intente conectarse con el nuestro. Esto sucede a través de las instrucciones relacionadas a la replicación, al intentar comunicarse con el servidor réplica. Se puede realizar con un dump de cero o recibiendo un dump de entrada.

Fuente consultada:

https://blog.thinkst.com/2021/09/a-mysql-canarytoken.html