Mística: Canales encubiertos sobre HTTP y DNS

(¡por ahora!)

Carlos Fernández
C0r0n4con 2020

CC BY NC SA

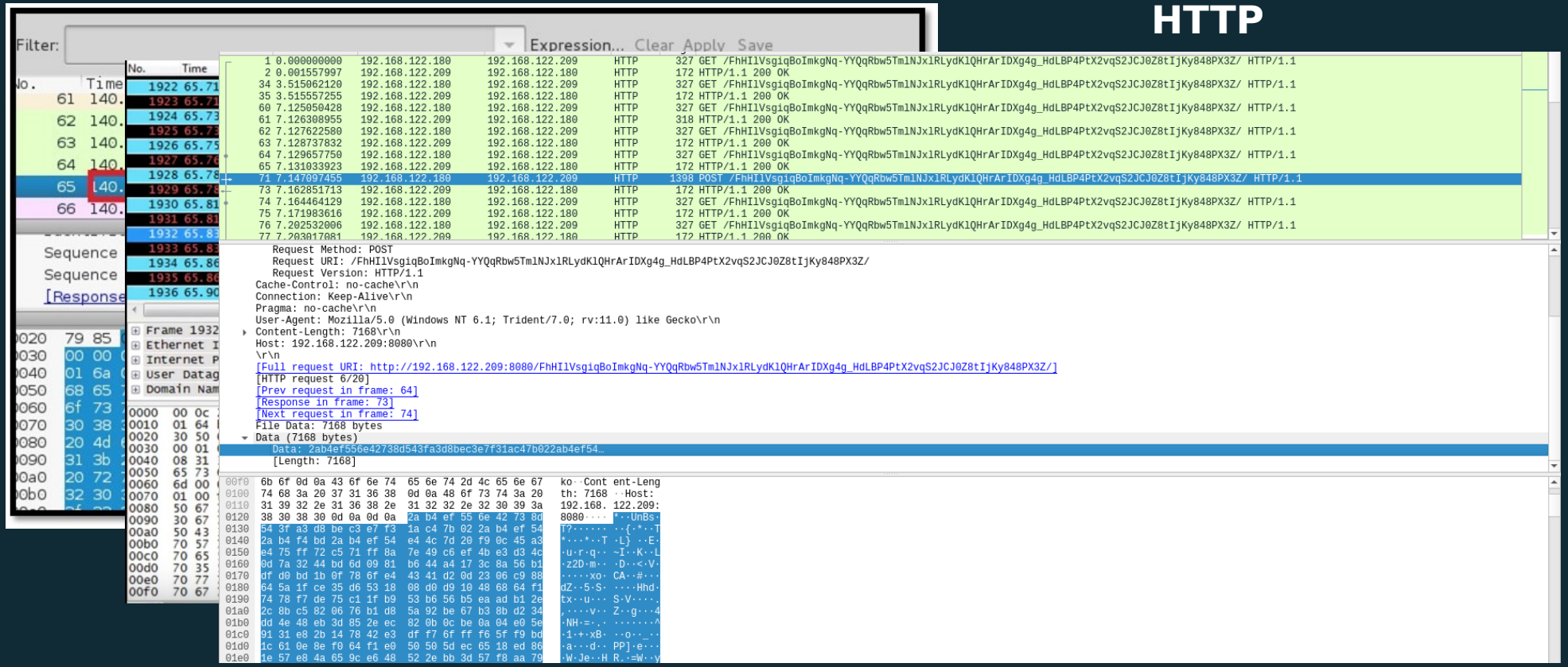
:~$ whoami

- **Carlos Fernández** (AKA Charlie)
- Coordinador Técnico de *Security Assessment* en INCIDE
- Coautor del libro “Resistencia Digital”, con Críptica (@CripticaOrg)
- Desarrollador de Mística, tu amigable contrabandista de datos, con **Raúl Caro** (AKA Secu)
- Twitter: @cfsgranda

Canales encubiertos

- Diseñados para transmitir información entre sistemas
- Usan medios que no habían sido diseñados para ese propósito
- El truco consiste en que emisor y receptor sepan cómo se codifica la información

Ejemplos de canales encubiertos



Lab: Identifying the use of Covert Channels - https://resources.infosecinstitute.com/lab-identifying-the-use-of-covert-channels/
Covert Command and Control over DNS with Beacon - https://www.youtube.com/watch?v=zAB5G-QOyx8

¿Por qué son interesantes?

- Los atacantes necesitan este tipo de canales para controlar sus implantes y extraer información de las organizaciones.
- **Perspectiva RED:** Debemos saber cómo crear estos canales para reproducir el comportamiento adversarial y camuflar nuestro tráfico en ejercicios de *Red Team*.
- **Perspectiva BLUE:** Debemos saber cómo funcionan estos canales para poder mejorar nuestra capacidad de detección y respuesta.

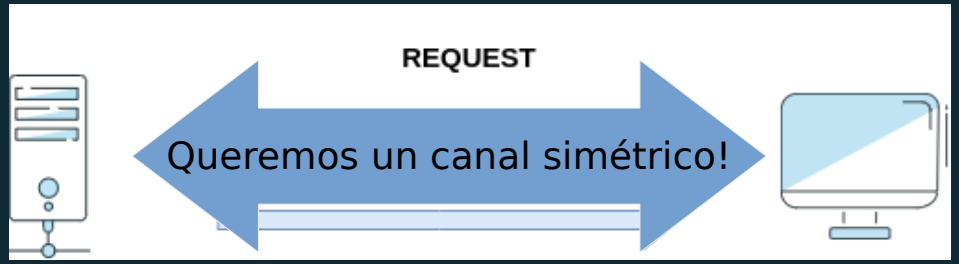
Diseñando canales encubiertos

Nos centraremos en protocolos que permiten intercambiar mensajes en dos direcciones: HTTP, DNS, ICMP, SMB, FTP, SMTP...

Son protocolos *Request-response.* El cliente solicita y el servidor responde

¿Qué necesita nuestro canal?

Poder transmitir en las dos direcciones de forma transparente



Los protocolos Request-Response son asimétricos: El servidor no puede enviar mensajes cuando quiera! Necesitamos solucionar este problema

¿Qué necesita nuestro canal?

<u>Poder transmitir información arbitraria</u>

- Comandos de C2
- Transferencia de ficheros
- Tráfico de red
- Independiente del protocolo subyacente (HTTP, DNS, ICMP…)
- ¡Estos protocolos han sido diseñados con otro propósito!

¿Qué necesita nuestro canal?

Garantía de recepción: Si un mensaje no llega, se debe reenviar.

Acknowledgement Receipt with Company...

Company Name Address

Acknowledgement of Receipt of Good

March .....

To_____

The undersigned hereby acknowledge receipt of the goods described on the annexed list or invoice and further acknowledges that said goods have been inspected and are without defect.

Signed and Sealed_____

Purchasing Manager_____

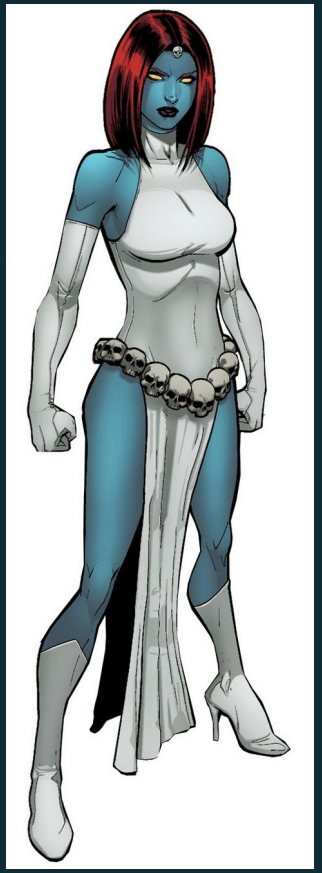
¿Qué necesita nuestro canal?

Confidencialidad: si alguien descubre el canal, no debe poder acceder a la información transmitida.

Todo esto se puede conseguir con Mística

¿Qué es Mística?

- Navaja suiza para el encapsulado de comunicaciones sobre protocolos de aplicación
- Basada en un protocolo de transporte custom, llamado SOTP
- Tiene dos tipos de módulos:
  - Wrappers: Encargados de codificar los paquetes SOTP en los protocolos (HTTP, DNS...)
  - Overlays: Encargados de proporcionar una funcionalidad al canal (Redirección de IO, shell, puertos...)
- Multi-plataforma y Software Libre



Simple Overlay Transport Protocol (SOTP)

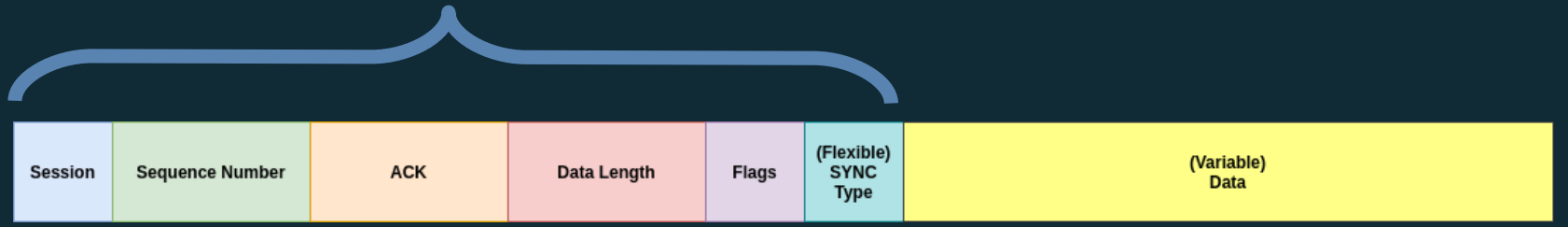
- Protocolo de transporte **minimalista:** Cabecera de 8 bytes. Tamaño de los datos variable
- Garantiza **recepción**
- La información se transmite en el transcurso de **sesiones**
- Etiquetas: permite multiplexar **varias sesiones por el mismo protocolo**, pudiendo conectar varios extremos

Simple Overlay Transport Protocol (SOTP)

- Mecanismo de ***polling*** para permitir bidireccionalidad: El canal se vuelve **simétrico**.
- Confidencialidad: El campo de datos viaja **cifrado** (de momento con RC4)
- La clave debe ser conocida previamente por emisor y receptor (de momento)

Simple Overlay Transport Protocol (SOTP)

Cabecera (en claro)



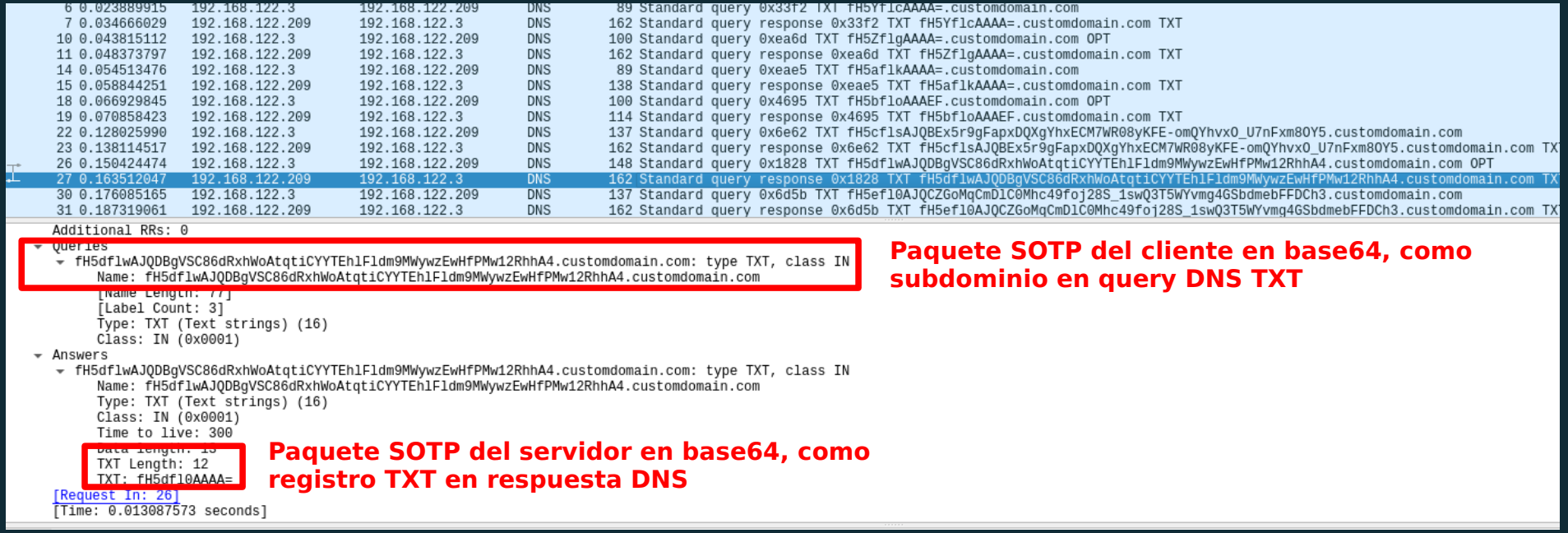
Cuerpo (cifrado)

Flags

- PUSH: Pasa datos al overlay
- SYNC: Tareas de sincronización
  - REQUEST_SESSION
  - RESPONSE_SESSION
  - POLLING_REQUEST
  - SESSION_TERMINATION
  - REINITIALIZATION

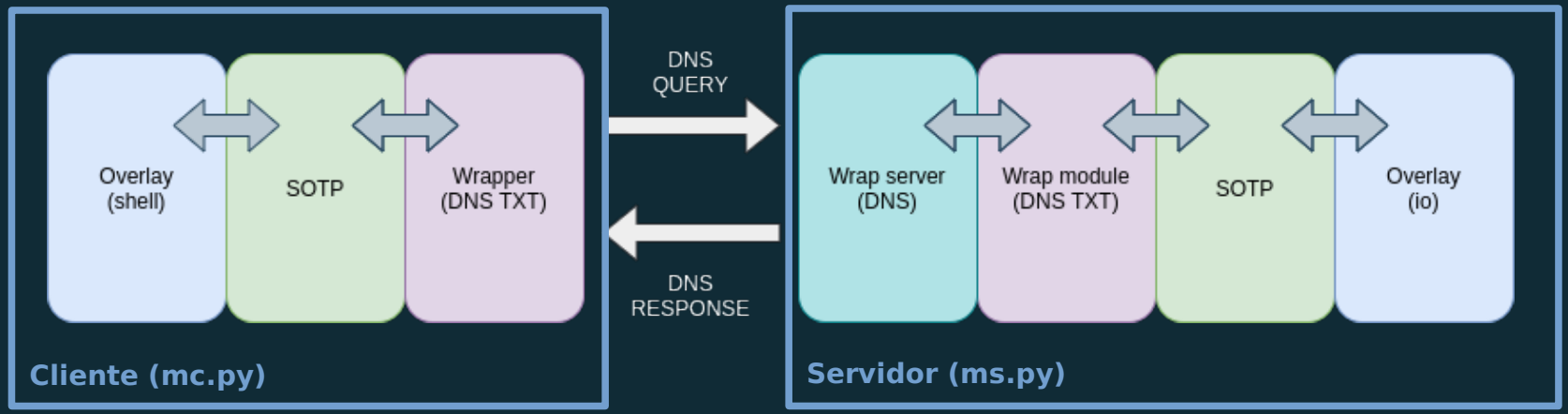
Simple Overlay Transport Protocol (SOTP)

Los paquetes SOTP son fácilmente encapsulables (ejemplo DNS):



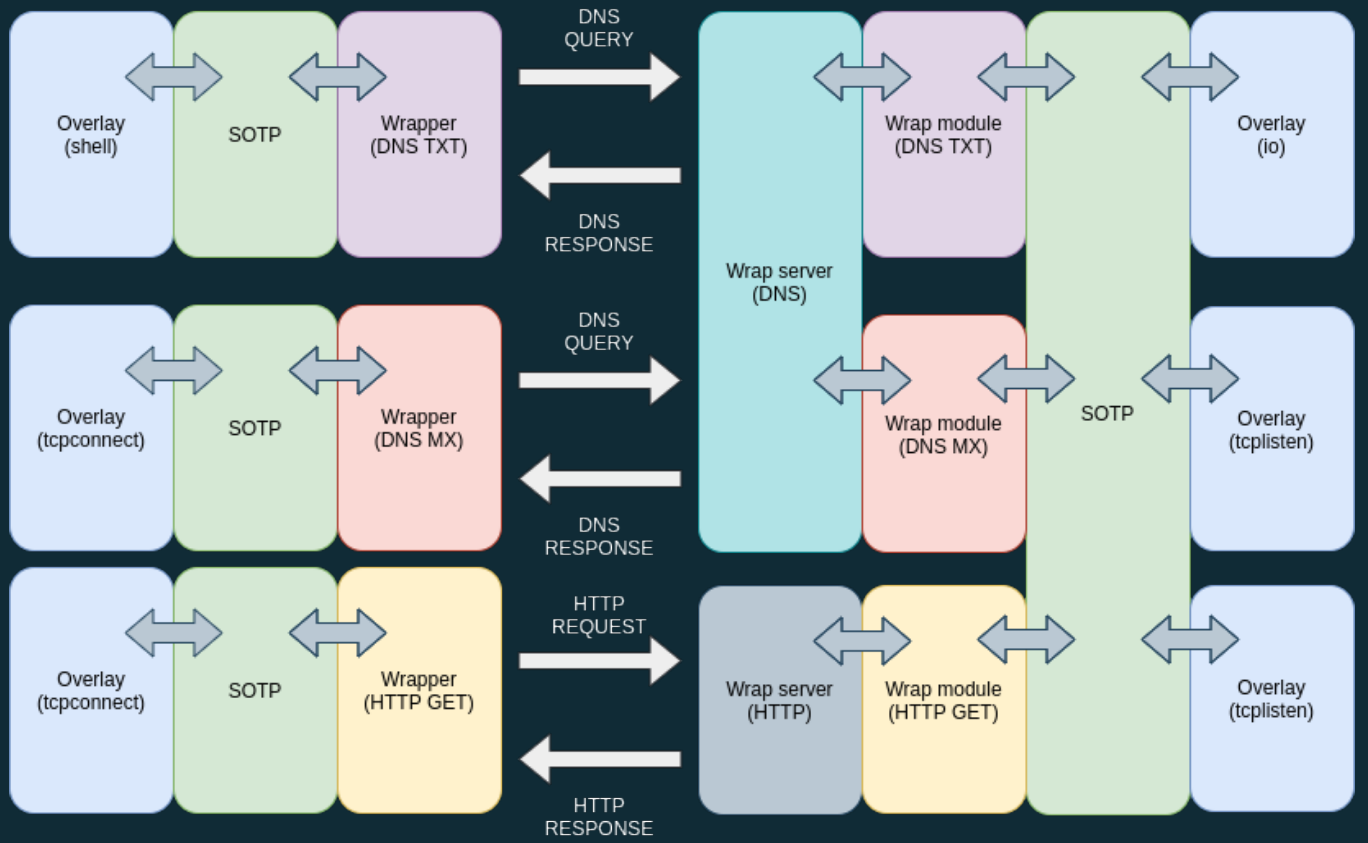
Mística: Diseño

Ejemplo de interacción modular: Shell reversa sobre DNS



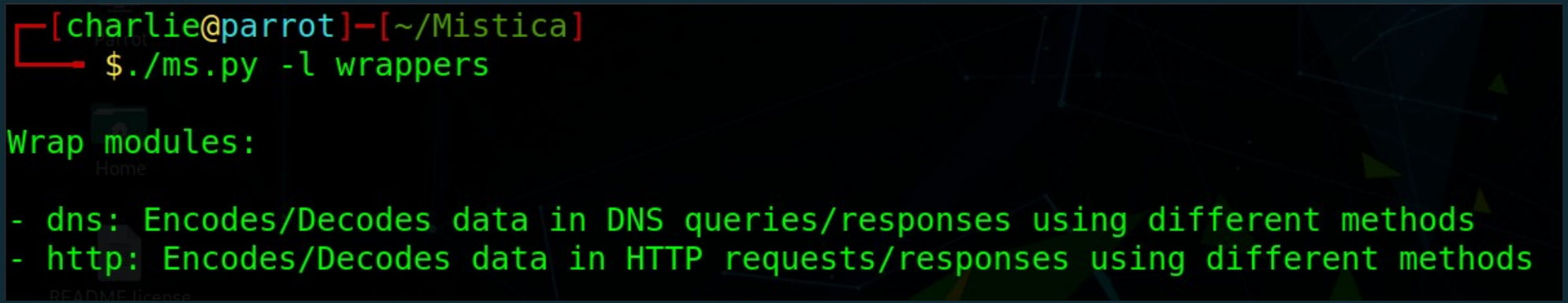
Mística: Diseño

Modo Multi-Handler (no implementado plenamente)



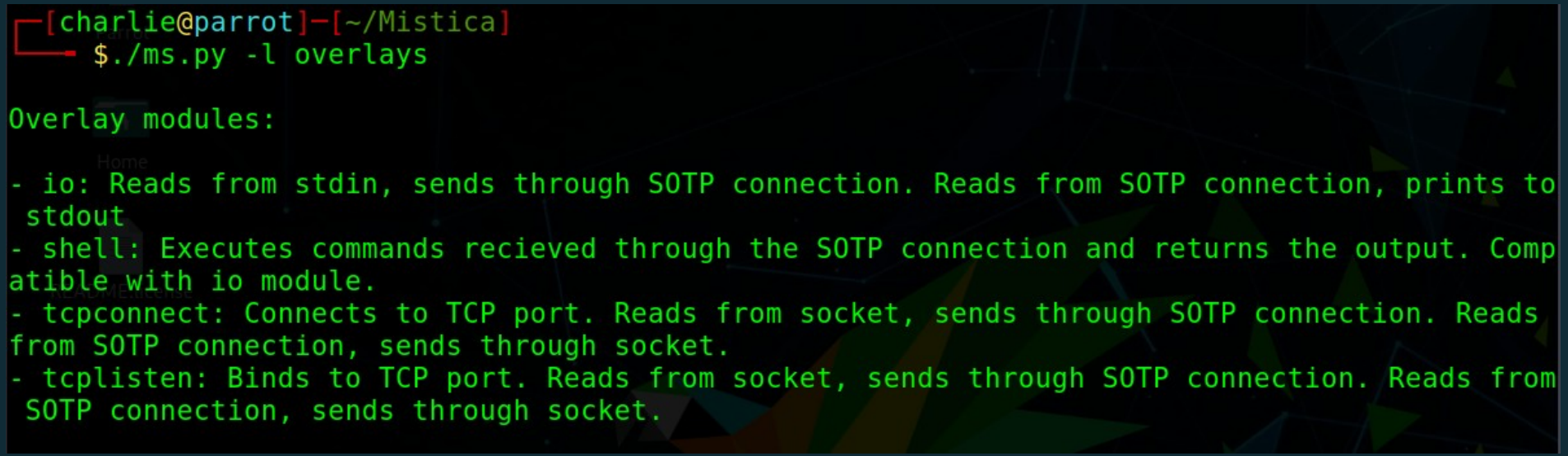
Mística: Wrappers

Los Wrappers son los encargados de encapsular los paquetes SOTP

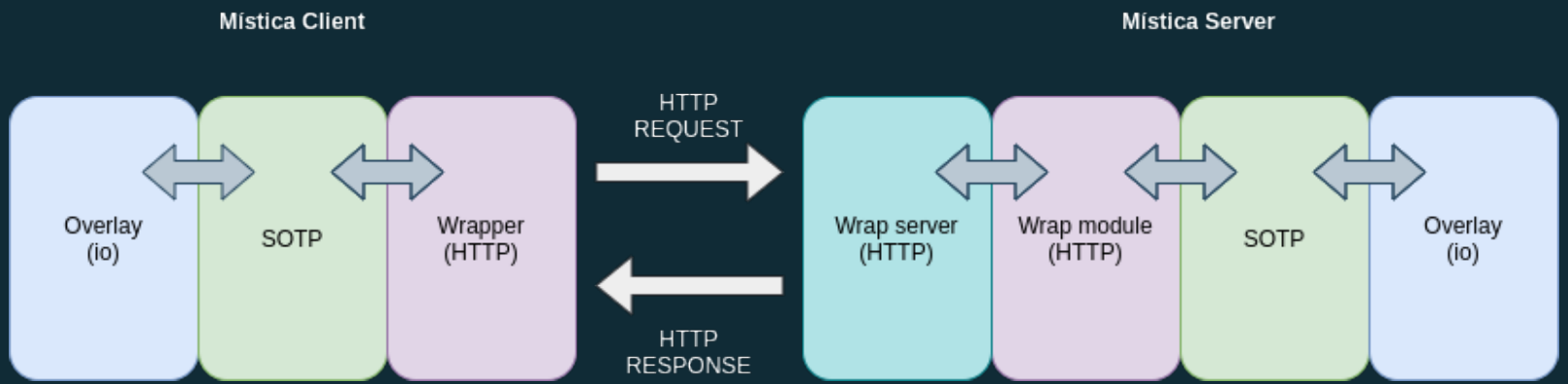


Mística: Overlays

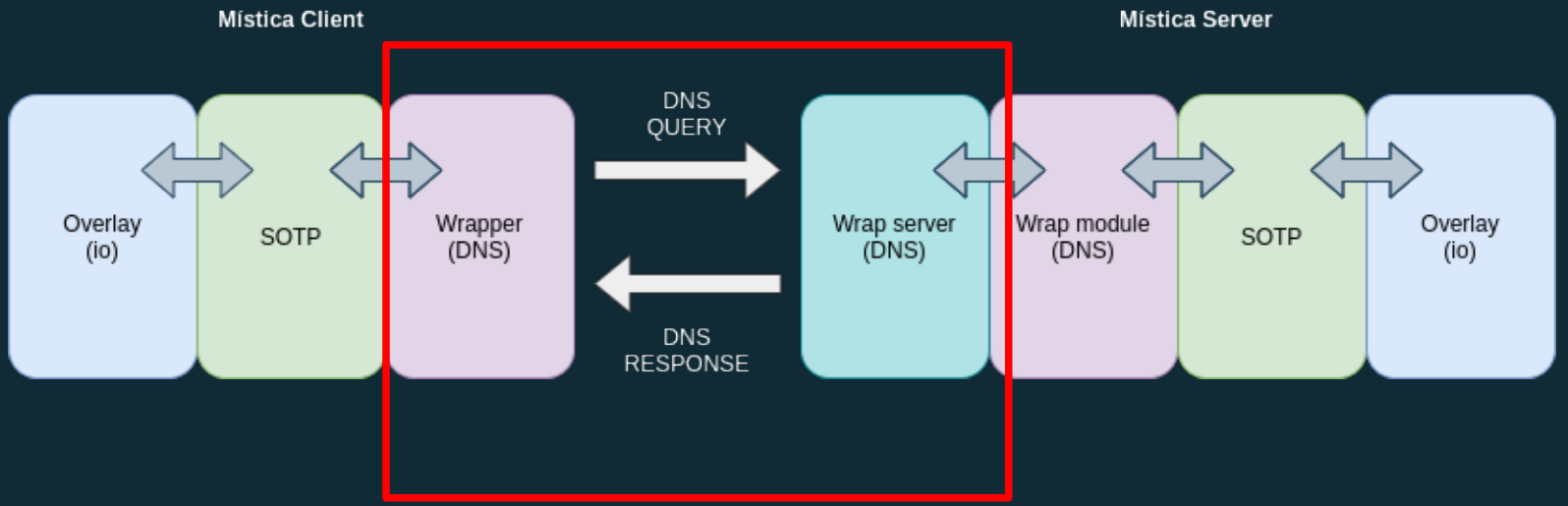
Los Overlays son los encargados de usar el canal SOTP



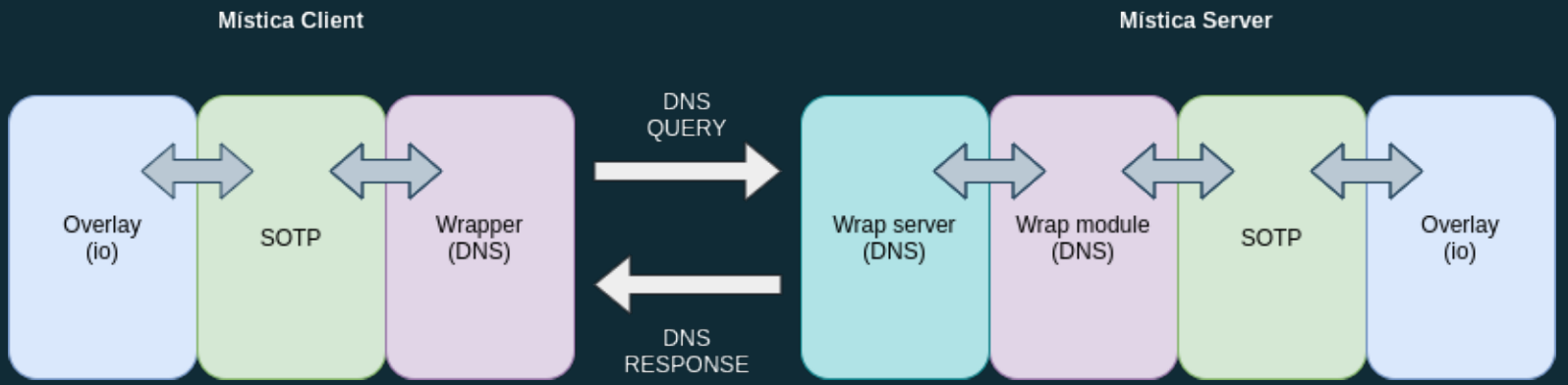
Demo 1: Canalizando por HTTP



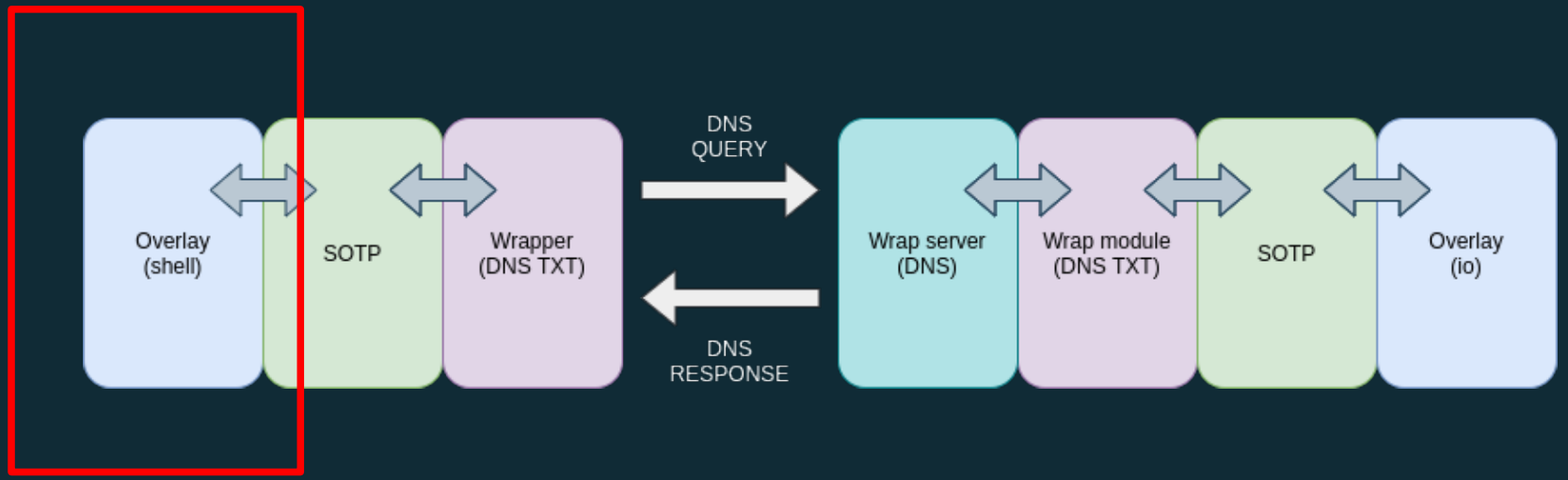
Demo 2: Canalizando por DNS



Demo 3: Exfiltración de ficheros DNS



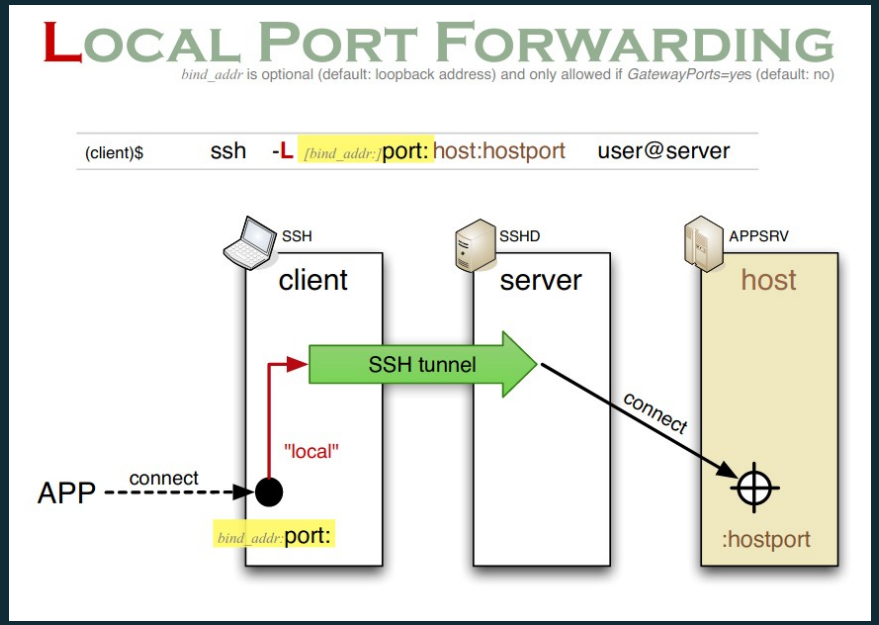
Demo 4: Ejecución remota de comandos

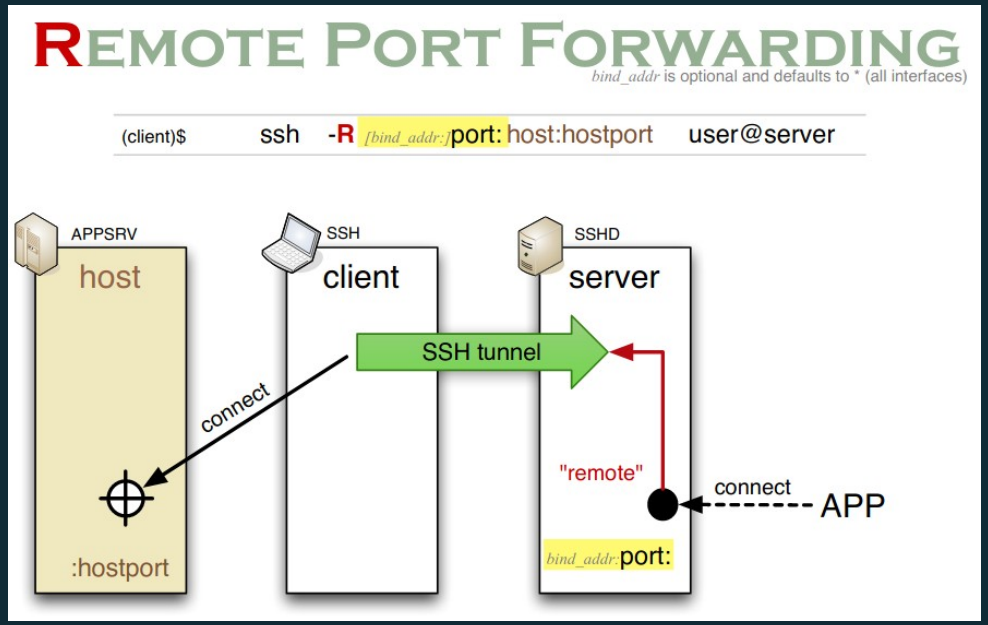


Revisitando la redirección de puertos

- Redirección de puertos (port forwarding): Sistema de reenvío de información recibida por el puerto A al puerto B y viceversa.
- Local Port Forwarding: Las conexiones que se reciben en el puerto local A se conectan a través del tunel con el puerto remoto B.
- Remote Port Forwarding: Las conexiones que se reciben en el puerto remoto B se conectan a través del tunel con el puerto local A.
- En ambos casos se mapea un puerto local con un puerto remoto

Revisitando la redirección de puertos



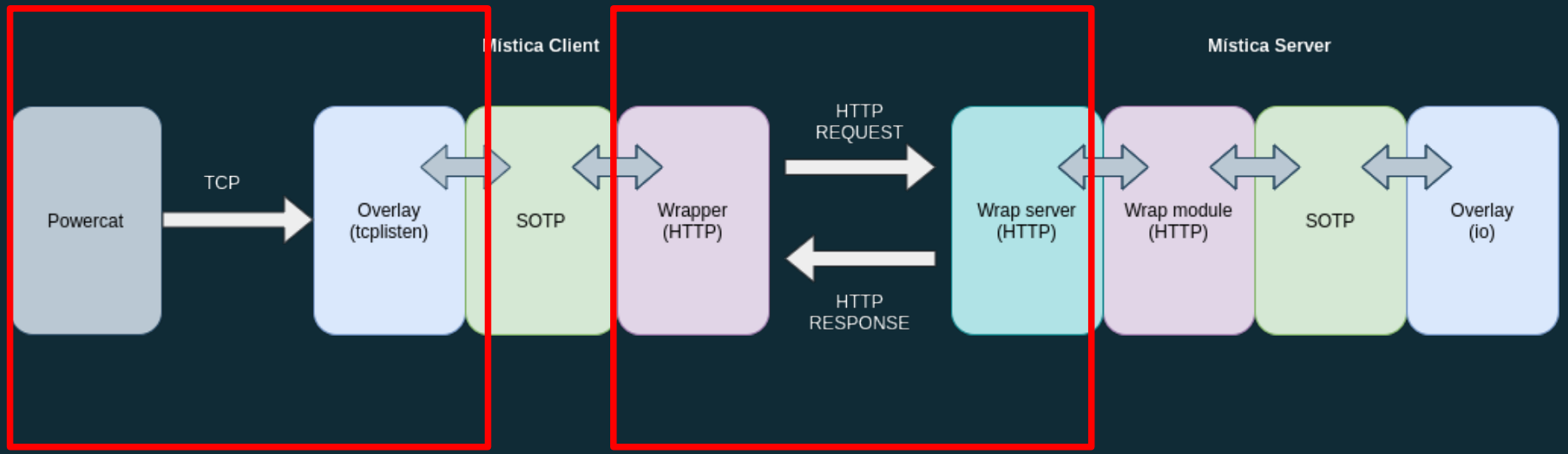


Dirk Loss - http://dirk-loss.de/ssh-port-forwarding.pdf

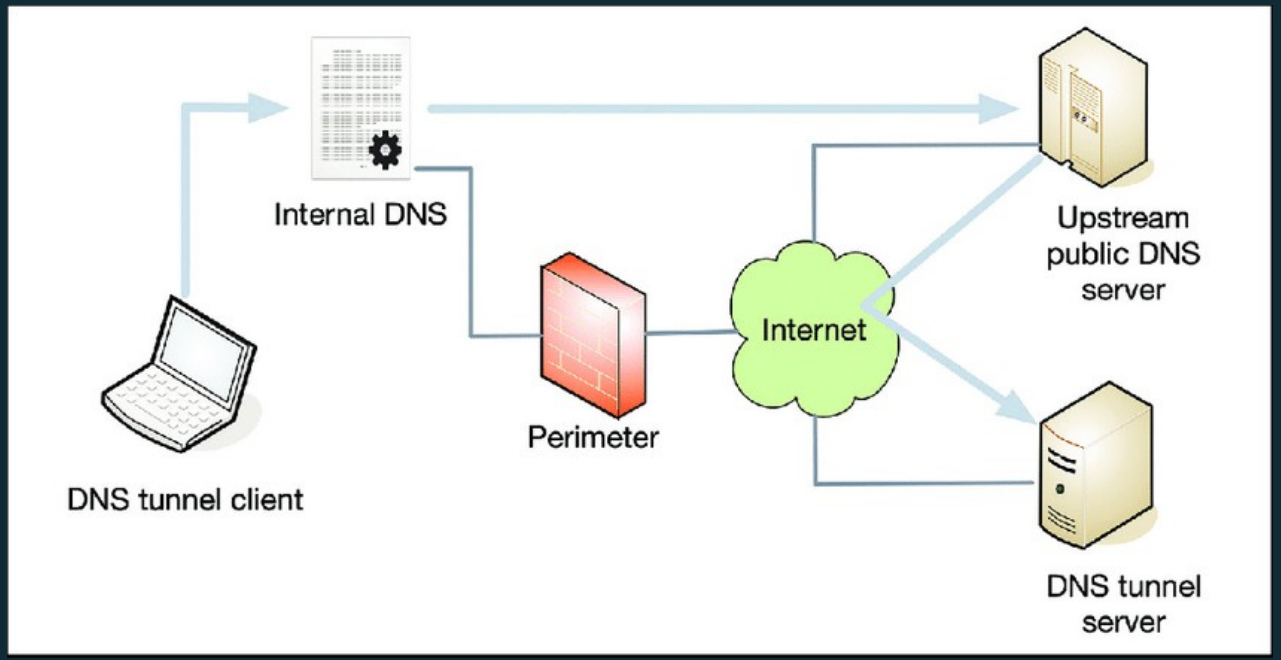
Redirección de puertos en Mística

- **tcplisten**: Escucha en un puerto. Lee del socket, envía por canal SOTP. Lee de canal SOTP, envía por socket.
- **tcpconnect**: Se conecta a un puerto. Lee del socket, envía por canal SOTP. Lee de canal SOTP, envía por socket.
- **tcplisten local + tcpconnect remoto**: Local port forwarding
- **tcpconnect local + tcplisten remoto**: Remote port forwarding
- En mística se pueden realizar más combinaciones (**io local + tcplisten remoto, etc.)**

Demo 5: Powercat sobre HTTP con tcplisten + io

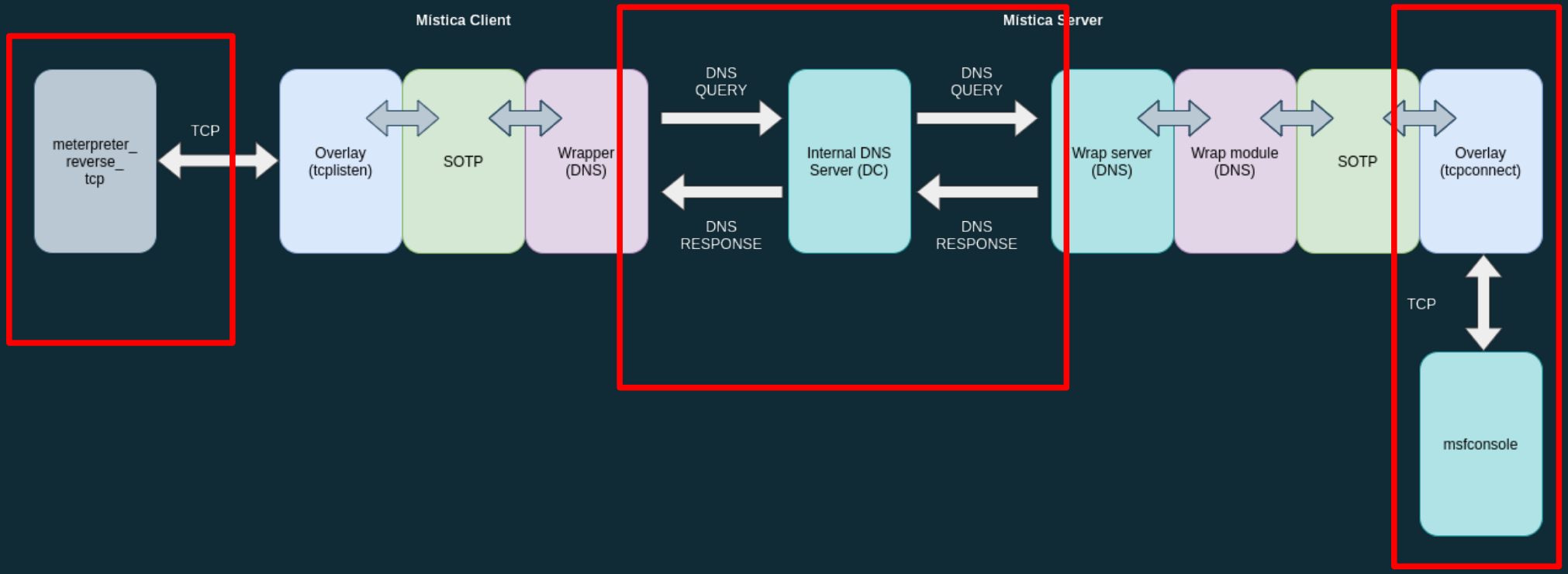


Resolución DNS, tu mecanismo de salida para firewalls quisquillosos

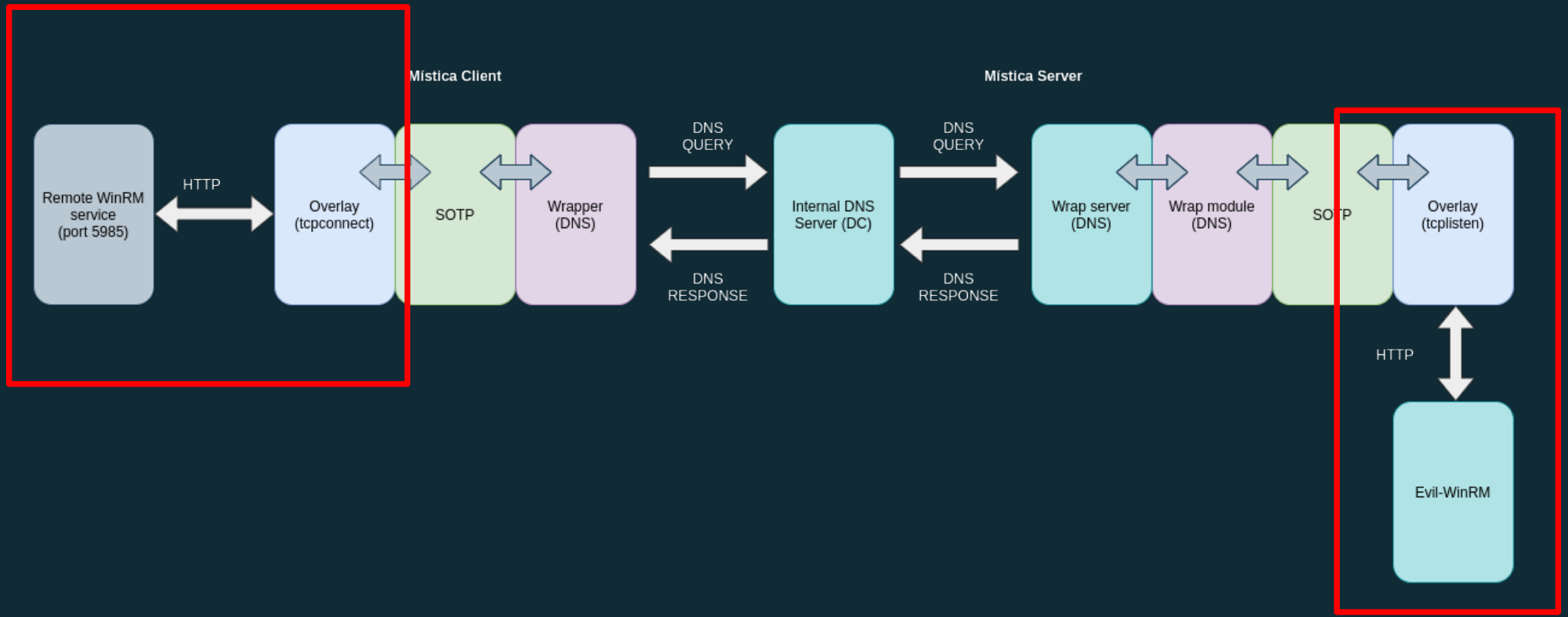


https://www.researchgate.net/figure/Basic-DNS-tunnel-components_fig1_338000308

Demo 6: Meterpreter sobre DNS, usando el DC como proxy para salir.



Demo 7: Evil-WinRM sobre DNS, usando el DC como proxy + canales persistentes



Status

- Status: PoC / Alpha ¡Queremos que lo probéis y lo rompáis!
- (Y si lo rompéis, decídnoslo :D)
- Siguientes pasos:
  - Protocolo de generación de claves en SOTP
  - Generador de payloads ejecutables multi-plataforma.
  - Modo multi-handler / interactivo
  - Módulo de ofuscación de paquetes SOTP: Padding, cifrado, stego...
  - Nuevos Wrappers: ICMP, HTTPS, SMB, DNSoH, DNSoT...
  - Nuevos Overlays: Dynamic port forwarding, RAT, FileTransfer, VPN...
  - Especificación de SOTP y documentación de desarrollo

Conclusiones

- Mística aporta un enfoque distinto a los canales encubiertos
- Permite al **Blue Team** mejorar su capacidad de detección y respuesta.
- Permite al **Red Team** construir canales únicos y flexibles, integrados con otras herramientas
- Es un proyecto emocionante y tenemos muchas ganas de hacerlo crecer
- ¡Probadlo, rompedlo y abrid issue!

Agradecimientos

- A **Raúl Caro**, por poner todo su empeño y talento en el proyecto (¡Seguidle en Twitter! **@secu_x11**)
- A **INCIDE**, por potenciar el Software Libre y de Código Abierto.
- A **C0r0n4con**, por la magnifica iniciativa, esfuerzo y coordinación

¡Hasta pronto!

- Mail: cfernandez [at] incide.es
- Twitter: @cfsgranda
- Mística: https://github.com/IncideDigital/Mistica