Mística

Canales encubiertos sobre
HTTP/s, DNS e ICMP
(por ahora)

whoami

Raúl Caro (AKA Secury)

Analista de Seguridad en INCIDE

Staff de Bitup Alicante plataforma de jornadas de ciberseguridad creada por y para estudiantes (@bitupalicante)

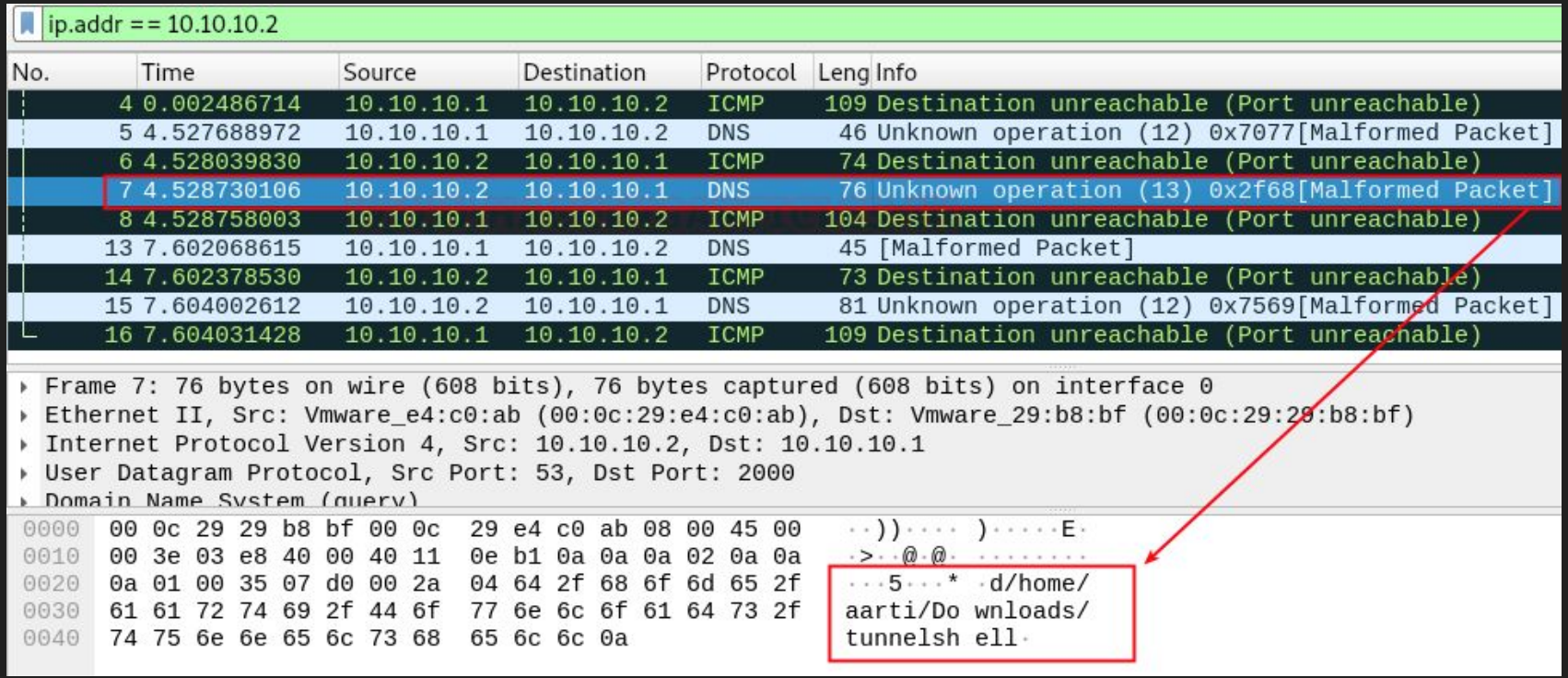
Desarrollador de Mística, tu amigable contrabandista de datos, con Carlos Fernández (AKA Charlie)

Twitter: @secu_x11

CANALES ENCUBIERTOS

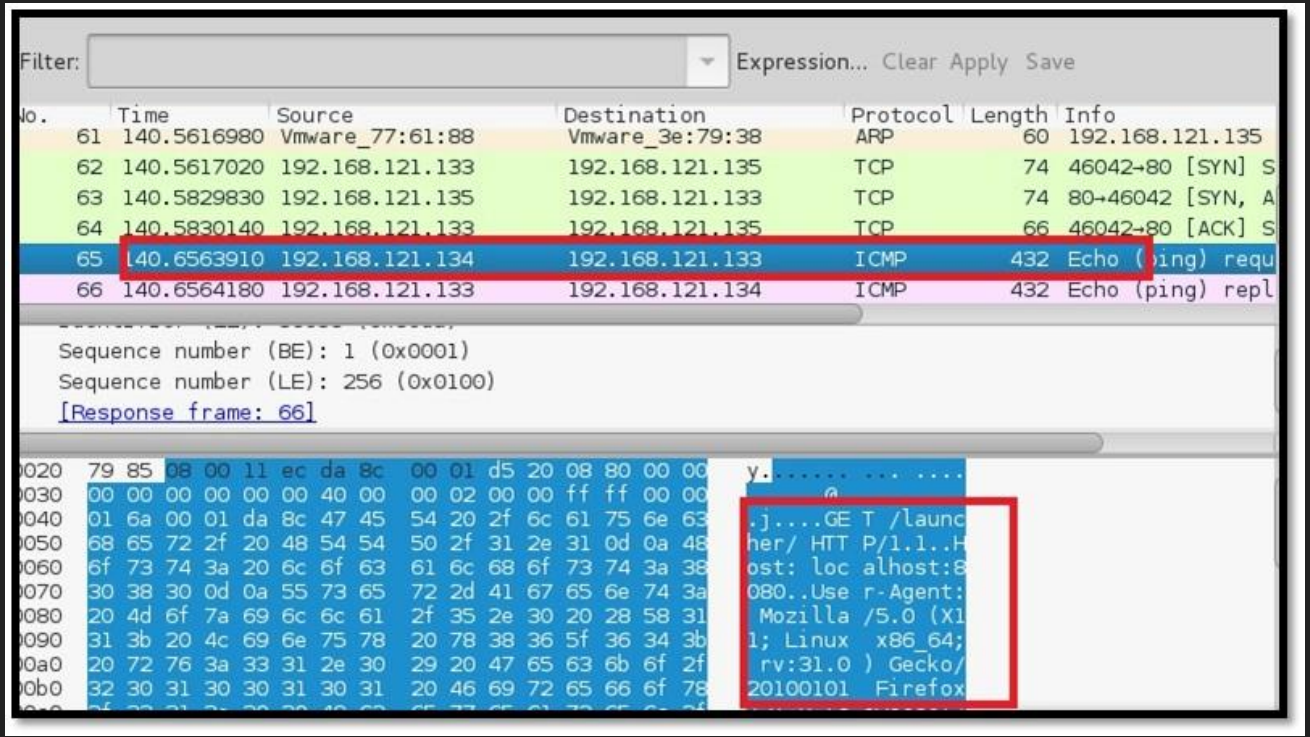
- Se utilizan para transferir información entre máquinas
- Usan medios que no habían sido diseñados para ese propósito.
- Los participantes de la comunicación tienen que acordar cómo van a codificar los datos.

Algunos ejemplos (dns)



Covert Command and Control over DNS with Beacon - https://www.youtube.com/watch?v=zAB5G-QOyx8

Otros ejemplos (icmp)



https://resources.infosecinstitute.com/lab-identifying-the-use-of-covert-channels/

Usos de los Canales Encubiertos

Perfil RED:

- Ocultar la comunicación con el C2
- Posibilidad de establecer comunicación hacia el exterior en entornos restrictivos
- Evadir las medidas de seguridad perimetrales

Perfil BLUE y GRAY:

- Capacidad de extracción de artefactos en entornos muy restrictivos
- Proteger el canal de comunicación ante agentes externos
- Medidas para evitar la censura y mejorar la privacidad del usuario

Funcionalidades Deseadas

Algunas de las funcionalidades a conseguir:

- Transferencia de Ficheros a nivel binario
- Comandos a ejecutar en cualquier extremo
- Información de otros protocolos que se van a encapsular

No importa el protocolo a utilizar, sino cómo se encapsulan los datos y se procesan.

Problemas en canales encubiertos

Se presentan algunos problemas en la comunicación:

- Protocolos Unidireccionales (Request/Response)
- Pérdida de paquetes
- Desincronización
- Protocolos inseguros que no protegen la información
- Espacio datos disponible por el protocolo limitado
- Codificación de la información a transmitir

Protocolos y Encapsulación (http)

```
POST / HTTP/1.1
Host: example.com
Accept: */*
User-Agent: mistica-dev
Content-Length: 21
Content-Type: application/x-www-form-urlencoded

data=test&data2=test2
```

URI PATH (GET & POST)

HEADER (GET & POST)

POST FIELD (POST)

Protocolos y Encapsulación (dns)

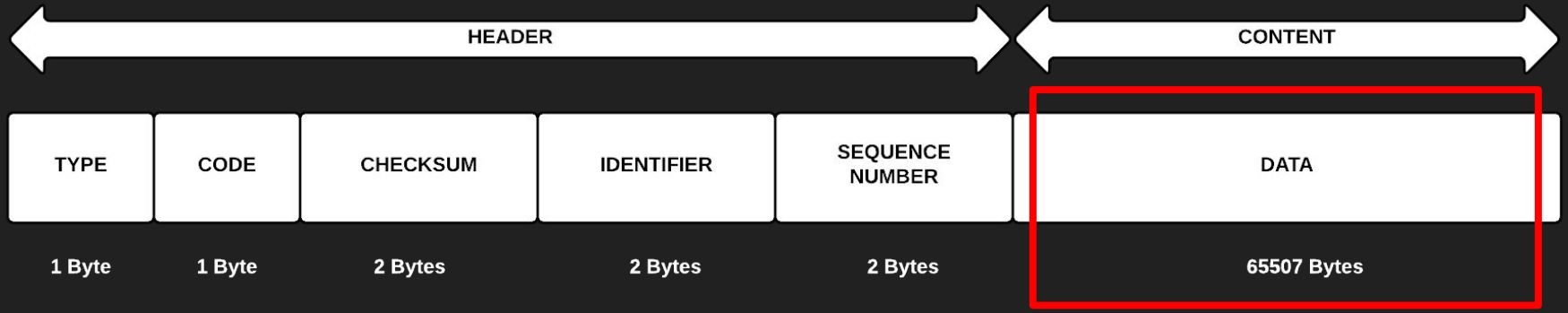
mistica.dev

another.example.mistica.dev

ZGF0b3Mgc2VjcmV0b3M.mistica.dev

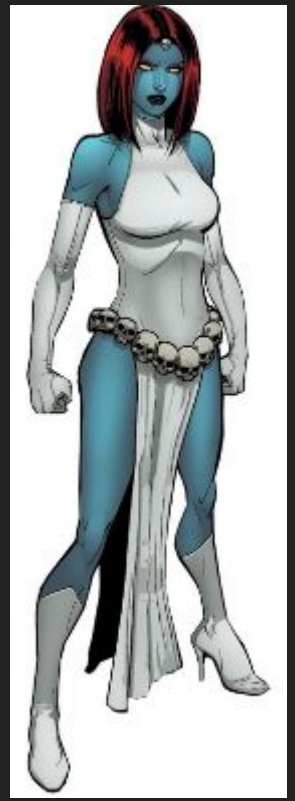
¿Alguien ha dicho DNSSEC?

Protocolos y Encapsulación (icmp)



¿Qué es Mística?

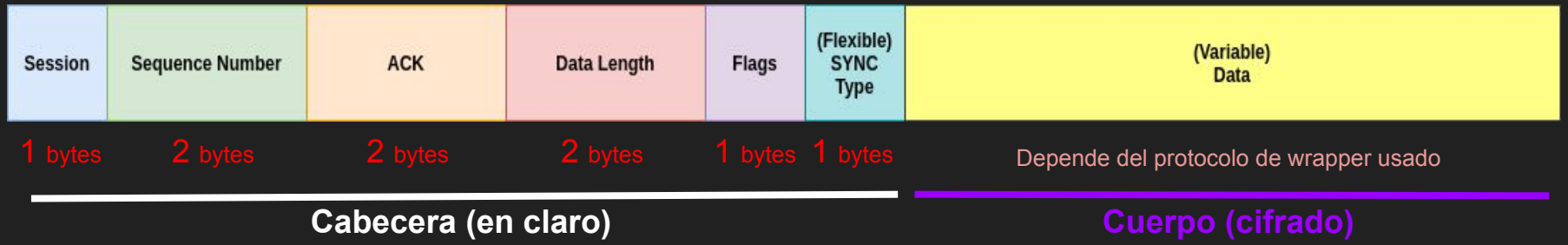
- Navaja suiza para el encapsulado de comunicaciones sobre protocolos de aplicación
- Basada en un protocolo de transporte custom llamado SOTP (Simple Overlay Transport Protocol)
- Tiene dos tipos de módulos:
  - Wrappers: encargados de codificar los paquetes SOTP en el protocolo que se utilice para la comunicación (HTTP, DNS, ICMP, etc)
  - Overlays: encargados de proporcionar una funcionalidad al canal (redirección IO, Shell, Port Forwarding, etc)
- Multiplataforma y Software Libre



¿Qué es 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
- Cifrado de los datos por defecto

Estructura Paquete SOTP



Flags:

- PUSH (pasar datos al overlay)
- SYNC (acciones de sync)

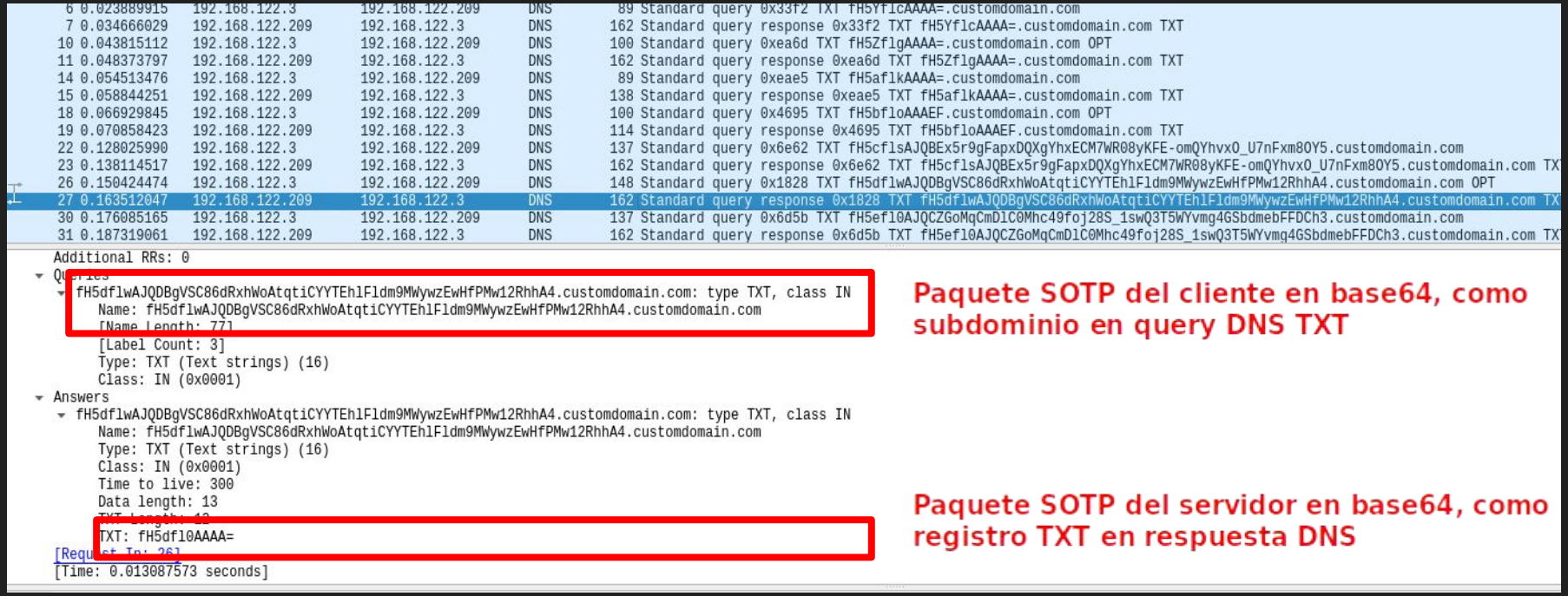
- Request Session
- Response Session
- Polling Request
- Session Termination
- Session Re-initialization

Características de SOTP

- Bidireccionalidad en la comunicación (Polling -> canal simétrico)
- Confirmación de paquetes (ACKs)
- Numeración de paquetes (Sequence Number)
- Cifrado de los datos (por ahora RC4)
- Extensión del canal de datos dependiendo del protocolo en uso (DNS*)
- Codificación de la información de los overlays (por ahora Base64)

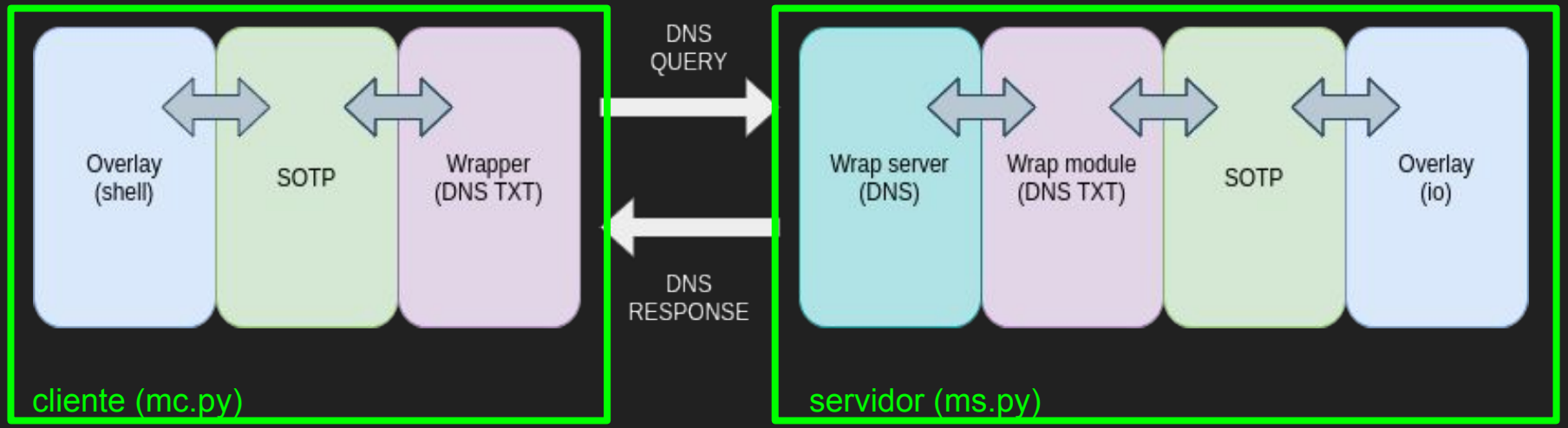
Algunas de sus características resuelven los problemas mencionados.

Apariencia de un paquete SOTP

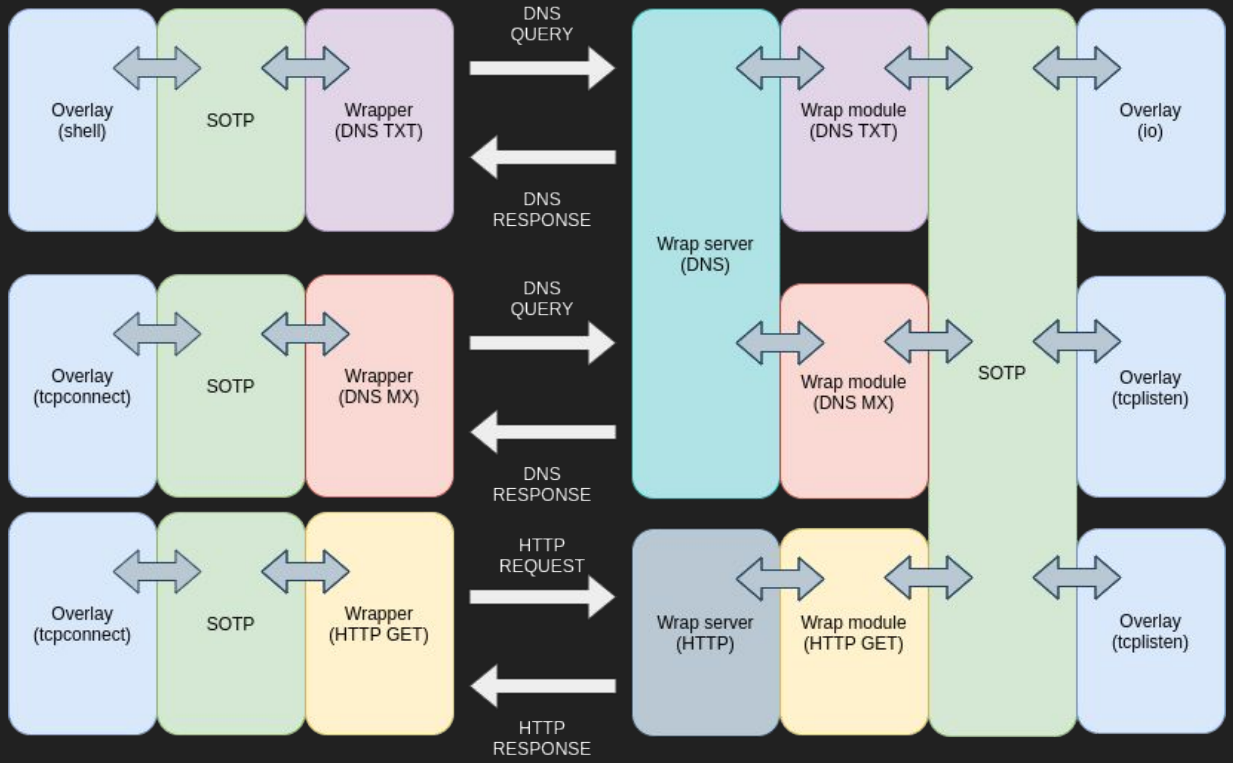


Arquitectura de Mística

Mística está estructurado en módulos que permiten una interacción circular.
La siguiente diapositiva muestra el modo Single Handler (1 cliente, 1 servidor)



Modo Multi-Handler (no implementado completamente)



Mística - Wrappers

Los Wrappers son los encargados de comunicar cliente y servidor de mística

```
┌─[rcaro@atlantis]─[~/github/Mistica]
└──╼ $./ms -l wrappers

Wrap modules:

- dns: Encodes/Decodes data in DNS queries/responses using different methods
- http: Encodes/Decodes data in HTTP requests/responses using different methods
- icmp: Encodes/Decodes data in ICMP echo requests/responses on data section

┌─[rcaro@atlantis]─[~/github/Mistica]
└──╼ $
```

Mística - Overlays

Los Overlays son los que proporcionan la funcionalidad al canal, es decir, los que realizarán el paso de datos entre la aplicación final y el módulo Sotp.

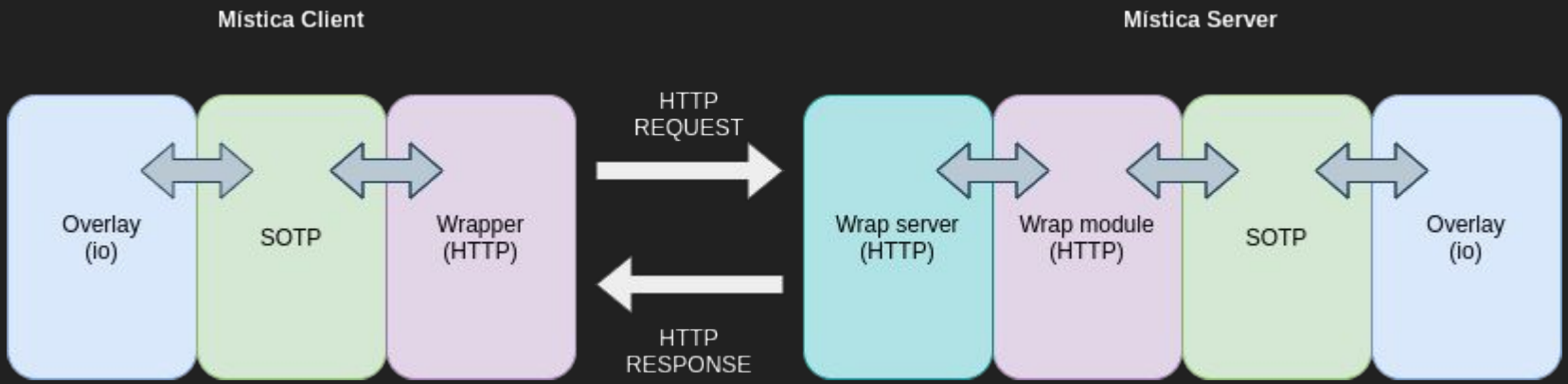
```
┌─[rcaro@atlantis]─[~/github/Mistica]
└──╼ $./ms -l overlays

Overlay modules:

- io: Reads from stdin, sends through SOTP connection. Reads from SOTP connection, prints to stdout
- shell: Executes commands recieved through the SOTP connection and returns the output. Compatible with io module.
- tcpconnect: Connects to TCP port. Reads from socket, sends through SOTP connection. Reads from SOTP connection, sends through socket.
- tcplisten: Binds to TCP port. Reads from socket, sends through SOTP connection. Reads from SOTP connection, sends through socket.

┌─[rcaro@atlantis]─[~/github/Mistica]
└──╼ $
```

Mística - Compilación

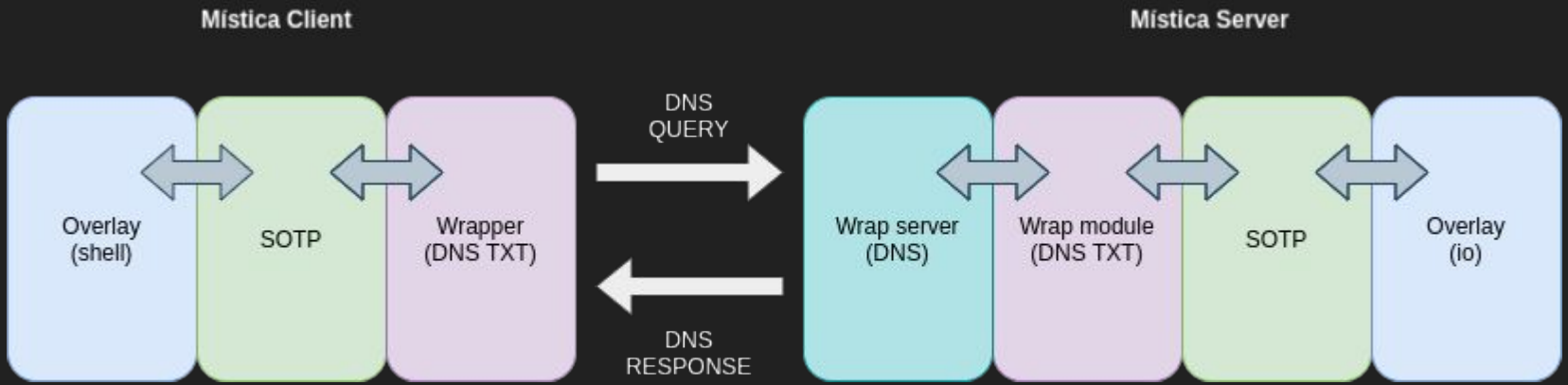
Al estar desarrollado en Python, Mística puede ser compilado utilizando ***Pyinstaller*** para generar un binario dependiente del Sistema Operativo.

```
pyinstaller --onefile \
  --hiddenimport overlay.client.io \
  --hiddenimport overlay.client.shell \
  --hiddenimport overlay.client.tcpconnect \
  --hiddenimport overlay.client.tcplisten \
  --hiddenimport wrapper.client.http \
  --hiddenimport wrapper.client.dns \
  --hiddenimport wrapper.client.icmp \
  --hiddenimport overlay.server.io \
  --hiddenimport overlay.server.shell \
  --hiddenimport overlay.server.tcpconnect \
  --hiddenimport overlay.server.tcplisten \
  --hiddenimport wrapper.server.wrap_module.http \
  --hiddenimport wrapper.server.wrap_module.dns \
  --hiddenimport wrapper.server.wrap_module.icmp \
  --hiddenimport wrapper.server.wrap_server.httpserver \
  --hiddenimport wrapper.server.wrap_server.dnsserver \
  --hiddenimport wrapper.server.wrap_server.icmpserver \
  mc.py
```

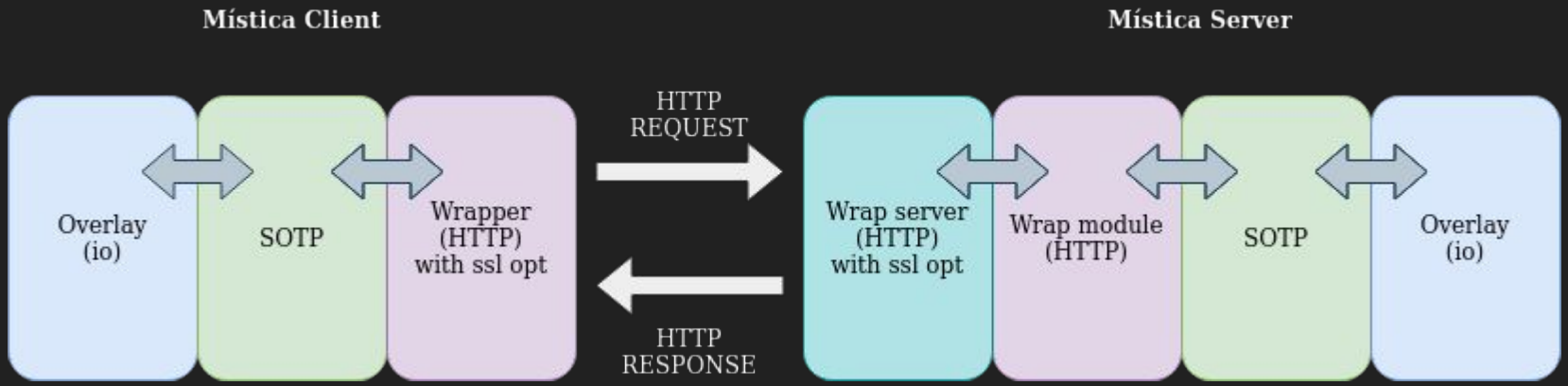
Demo I - Comunicación vía HTTP



Demo II - Shell vía DNS



Demo III - Exfiltración de archivos vía HTTPs



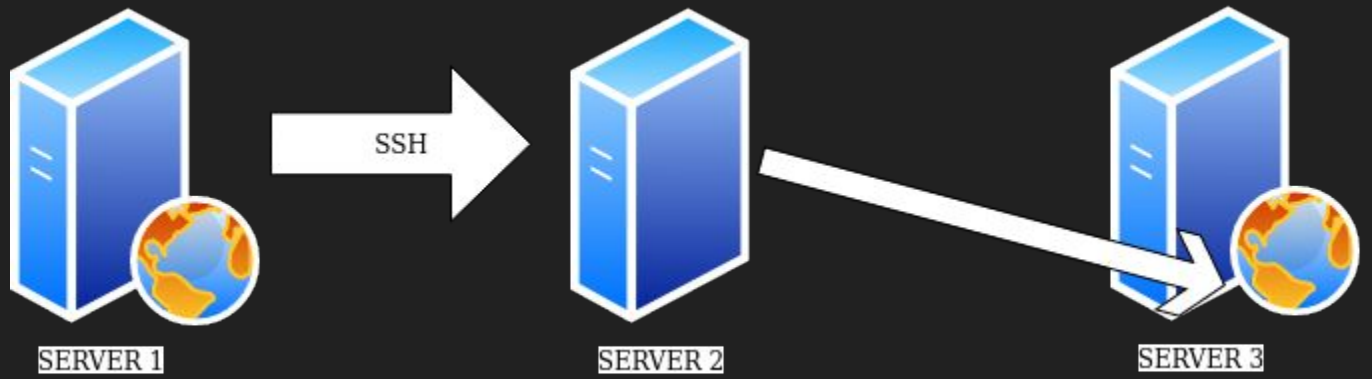
Review Port Forwarding

El **Port Forwarding** o reenvío de puertos es una técnica de reenvío de información entre dos puertos (uno local y otro remoto).

Tipos de Port Forwarding

- **Local Port Forwarding:** Las conexiones que se reciben en el puerto local A se conectan a través del túnel con el puerto remoto B.
- **Remote Port Forwarding:** Las conexiones que se reciben en el puerto remoto B se conectan a través del túnel con el puerto local A.

Review Port Forwarding



Port Forwarding con Mística

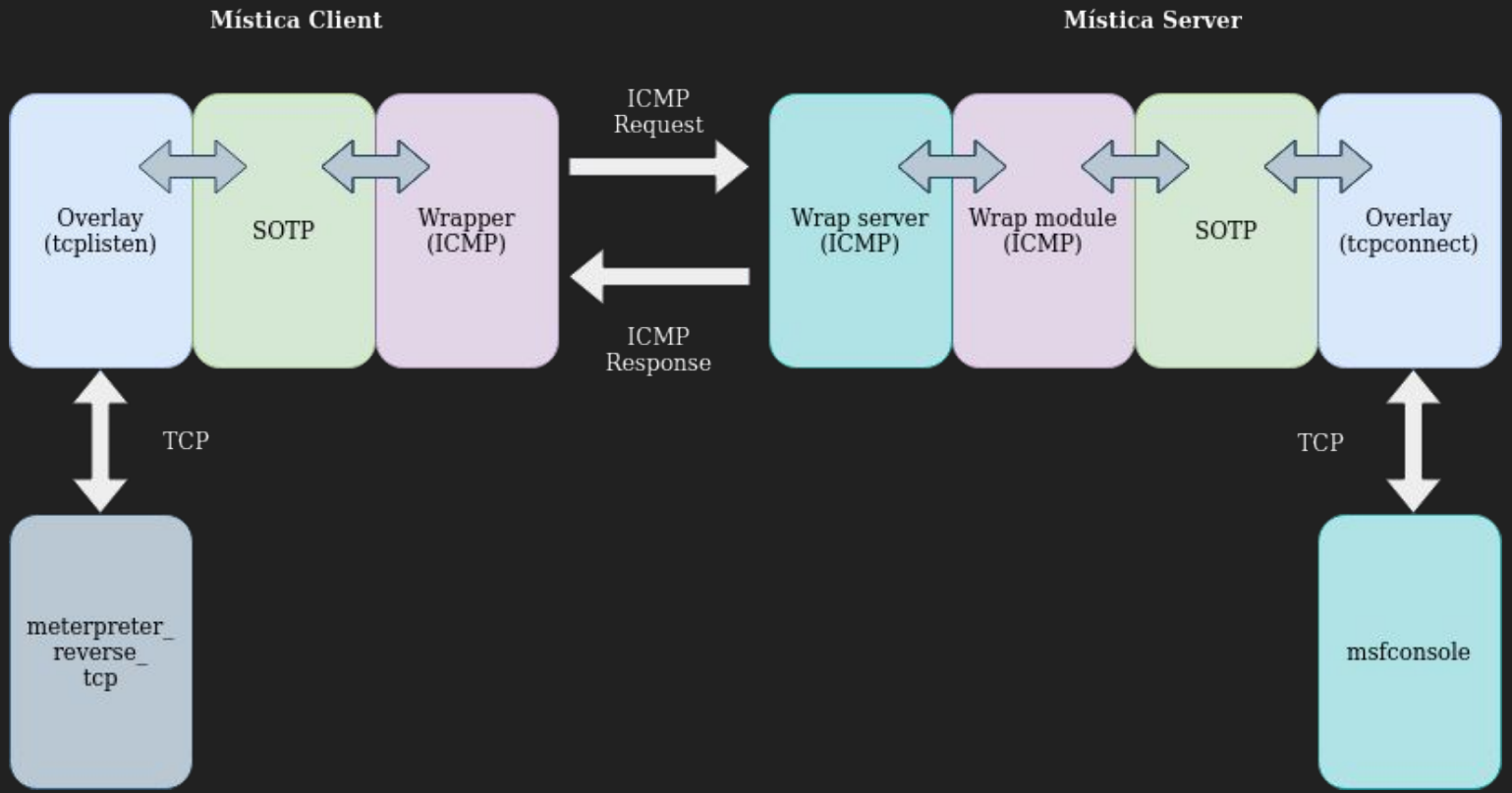
Módulos de Mística de Port Forwarding:

- **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.

Que equivaldría a:

- tcplisten local + tcpconnect remoto: **Local port forwarding**
- tcpconnect local + tcplisten remoto: **Remote port forwarding**

Demo IV - Meterpreter vía ICMP



Status

- Status: PoC / Alpha ¡Queremos que lo probéis y lo rompáis!
- (Y si lo rompéis, decírnoslo :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: SMB, DNSoH, DNSoT, FTP…
  - 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, romperlo y abrid issue!

Agradecimientos

A Carlos Fernández, por aportar diseño, coherencia y funcionamiento. Este proyecto no sería nada sin él (¡Seguidle en Twitter! @cfsgranda)

A INCIDE, por potenciar el Software Libre y de Código Abierto.

A Bitup Alicante y todos esos grupos que se preocupan por llevar el mundo de la seguridad informática y el hacking a estudiantes y todas las personas del entorno.

A esa personita que siempre me da su apoyo (;

MUCHAS GRACIAS

Y esto es todo amigos!

Mail: rcaro [at] incide.es

Twitter: @secu_x11

Mística: https://github.com/IncideDigital/Mistica