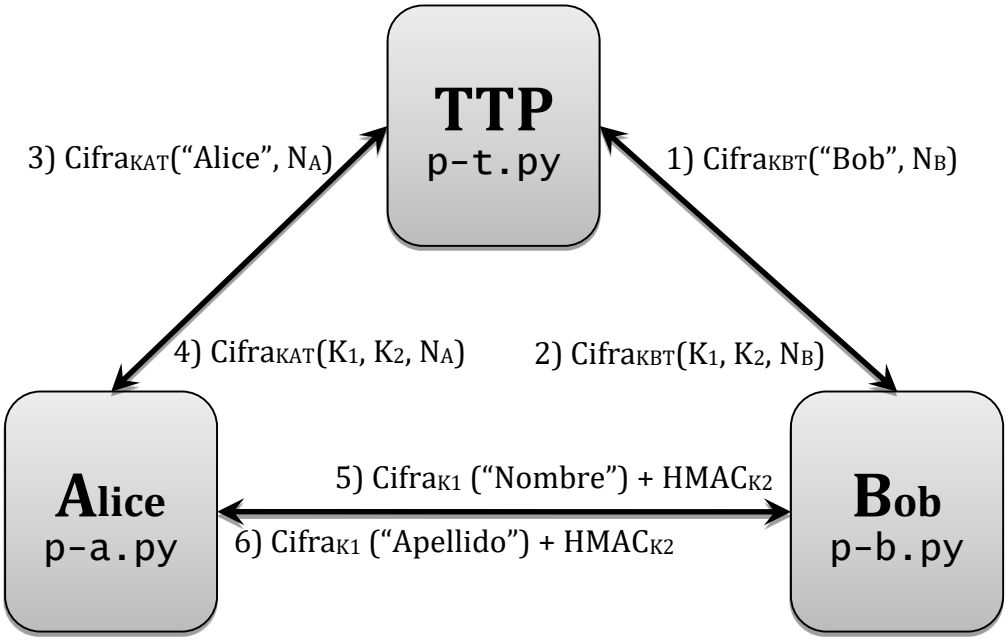
# PRÁCTICA 3: Protocolos (Parte A)

Seguridad de la Información
Curso 2019-2020

Lenguajes y Ciencias de la Computación.
E.T.S.I. Informática, Universidad de Málaga

## RELACIÓN DE EJERCICIOS:

1. Se pide implementar el siguiente protocolo entre **A**lice y **B**ob y la tercera parte confiable **TTP** indicado en la figura de abajo, donde tanto **B** como **A** contactan con el **TTP** y reciben dos claves simétricas; y posteriormente **A** envía a **B** el nombre del alumno/a, y **B** responde a **A** con el apellido del alumno/a. El código fuente parcial del **TTP** (y las claves $K_{AT}$ y $K_{BT}$, las cuales se generarán automáticamente al ejecutar TTP) y **B** ya se proporciona en el campus virtual.




El alumno/a deberá tener en cuenta los siguientes aspectos:

a. En este ejercicio, se utilizará la clase SOCKET_SIMPLE_TCP del campus virtual, que permite crear un cliente y un servidor TCP básicos. Es por ello que las comunicaciones entre **A**, **B**, y el **TTP** se realizarán no a través de ficheros (como hemos venido realizando hasta ahora), sino a través de sockets.
   i. TTP actuará como servidor de las conexiones de **A** y **B**, mientras que **B** actuará como servidor de las conexiones de **A**.
b. El mecanismo de cifrado a utilizar en los pasos 1), 2), 3) y 4) será AES GCM. Sin embargo, el mecanismo de cifrado en los pasos 5) y 6) será AES CTR. Es por eso que para asegurar la integridad del mensaje será necesario el uso de HMAC (SHA256).
c. Para construir los mensajes entre **A**, **B**, y **TTP**, se utilizará el formato JSON.
d. Antes de ejecutar los pasos 5 y 6, **A** y **B** deben comprobar que los nonces recibidos de T son los que se enviaron anteriormente.

El formato JSON, o JavaScript Object Notation, es una forma de representar objetos en forma de diccionarios, donde **cada valor debe ser una cadena**. Para el envío de mensajes, es posible guardar cada campo dentro de un array, de la siguiente forma:

```
mensaje = [] # Array vacio
mensaje.append(cadena_de_caracteres)
mensaje.append(array_de_bytes.hex()) # Conversion en Base64
json = json.dumps(mensaje)
```

…y recuperarse de la siguiente forma:

```
mensaje = json.loads(json)
cadena_de_caracteres, array_de_bytes_como_cadena = msg_ET
array_de_bytes = bytearray.fromhex(array_de_bytes_como_cadena) # De Base64 a Array
```