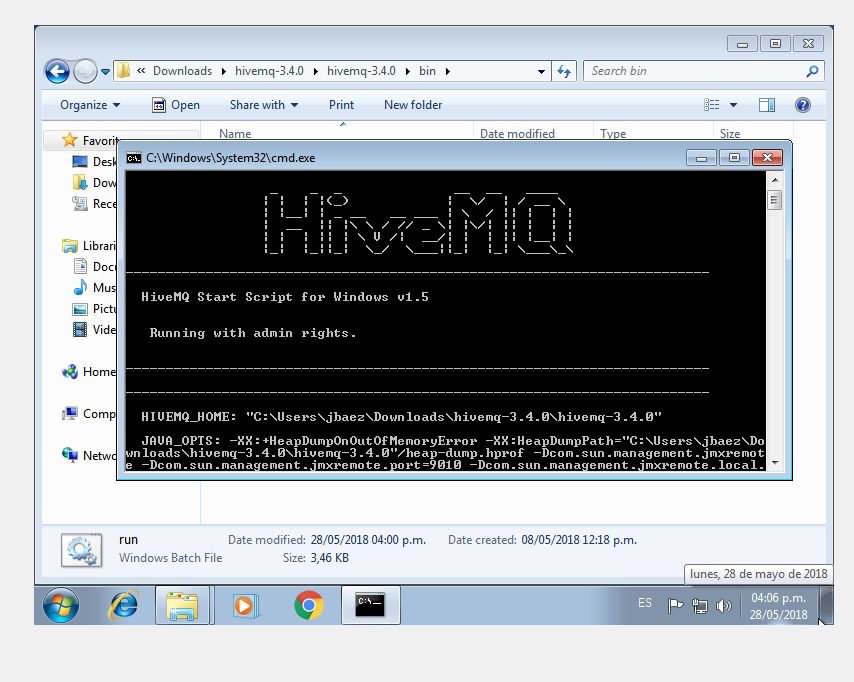
Nos registramos y descargamos HiveMQ:

<https://www.hivemq.com/downloads/>

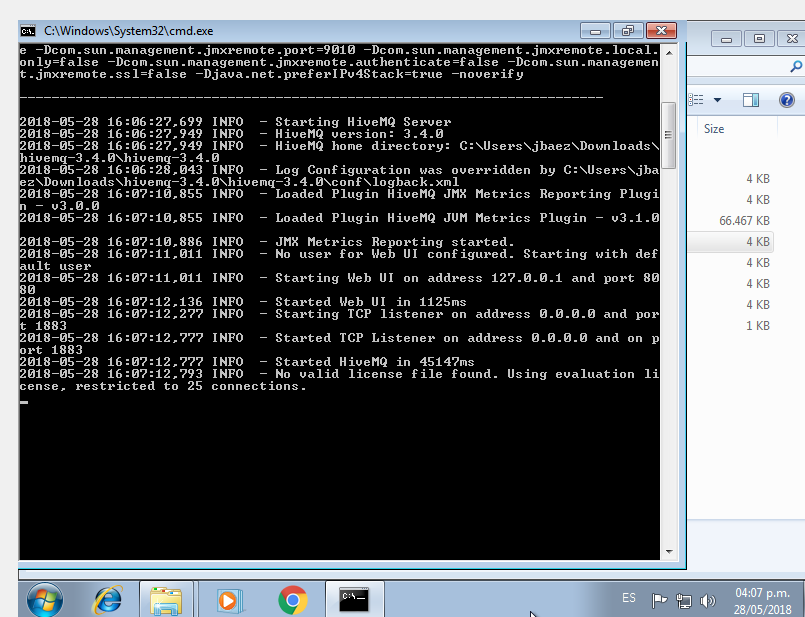
Instalamos JRE de Java:

<https://www.java.com/inc/BrowserRedirect1.jsp?locale=es>

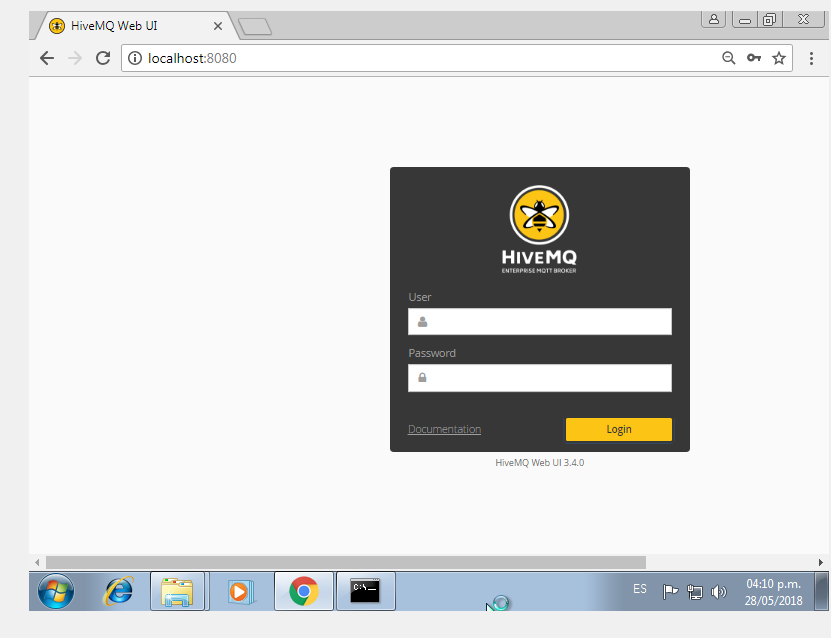
Ejecutamos el archivo /bin/run.bat en el sistema Windows (si estuvieramos en Linux, ejecutariamos /bin/./run.sh):



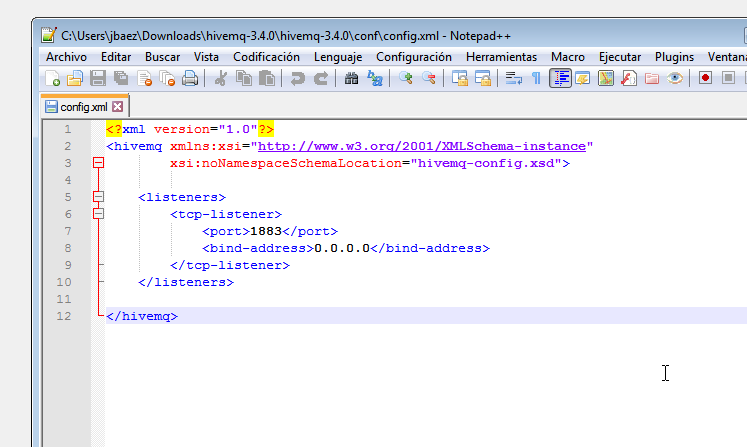
Vemos que inició el servidor HiveMQ:



Dicho server, al igual que APACHE posee una interfaz Web:



Sin embargo, aún no se configuró ningún tipo de parametría para la seguridad:



Configurar TLS:

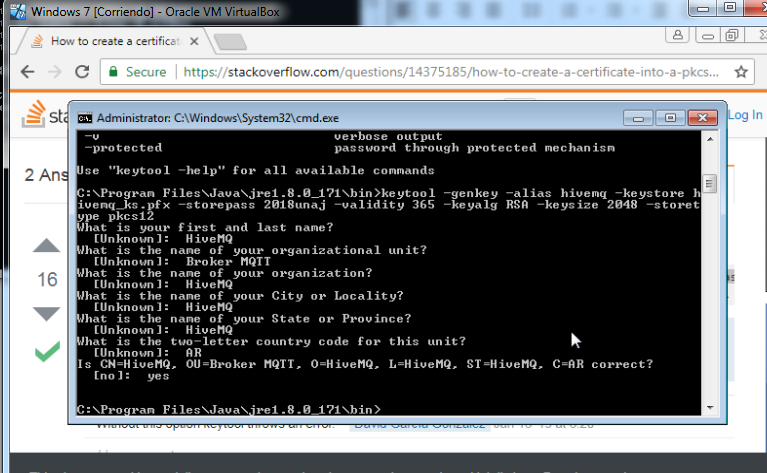
<https://www.hivemq.com/docs/hivemq/latest/#tls-chapter>

En la sección: “**Secure and Insecure Listeners**”

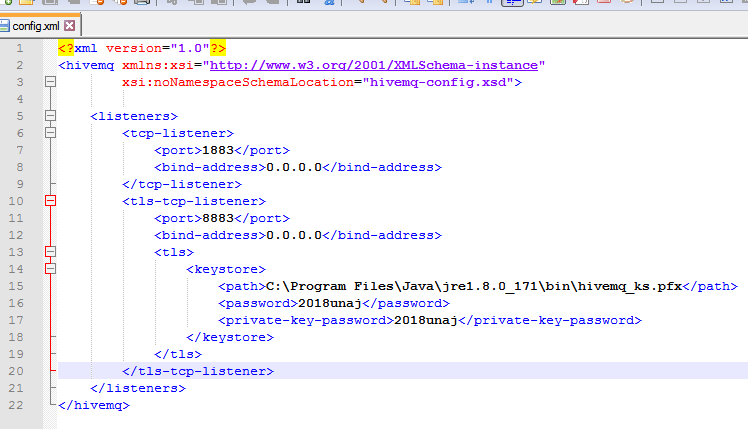
\*\*Para crear el par público-privado, seguir el tutorial en la documentación de la API Web.

**SSL y TSL:**

[**https://www.globalsign.com/es/blog/ssl-vs-tls-difference/**](https://www.globalsign.com/es/blog/ssl-vs-tls-difference/)

****

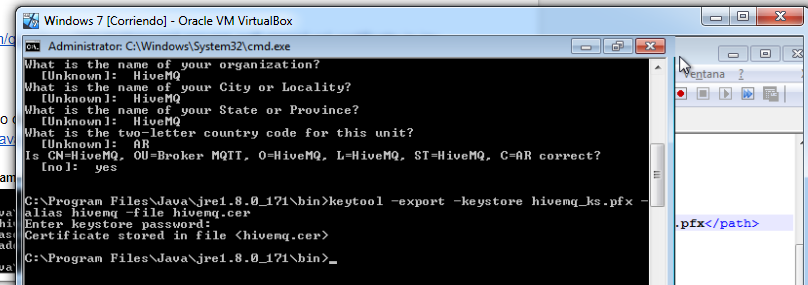
Configuramos Interceptores para el servidor y apuntamos al keystore que creamos:

****

Exportamos el Certificado desde la Keystore del server:

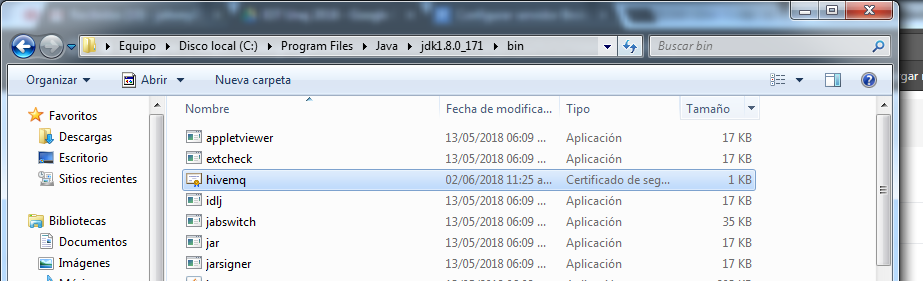
<https://docs.oracle.com/javase/tutorial/security/toolsign/step5.html>

**keytool -export -keystore examplestore -alias signFiles -file Example.cer**

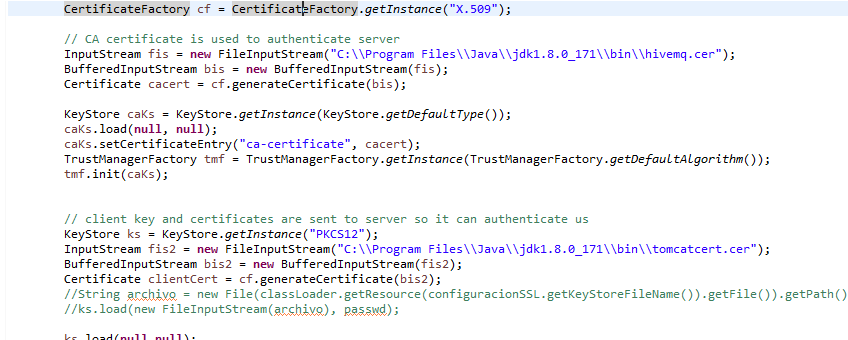
****

Copiamos el certificado al servidor de la API cliente, ya que es un certificado self-signed. De otro modo, si fuese legitimado por una entidad certificadora reconocida no tendríamos que hacerlo.

En el cliente:

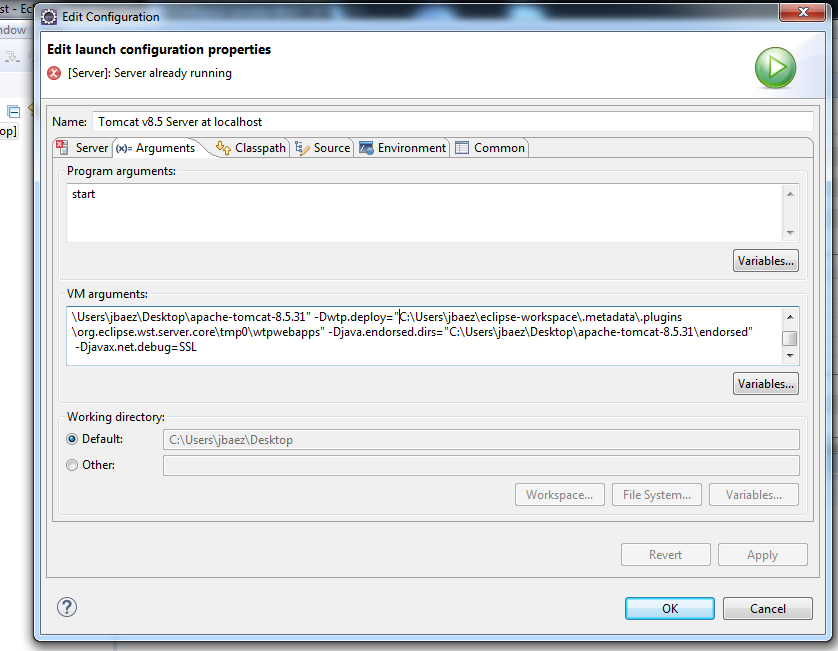


Apuntamos desde el código a ambos certificado (cliente y servidor):



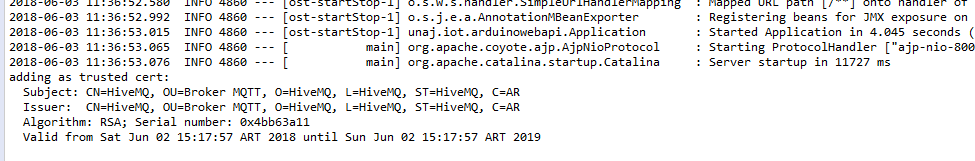
Para poder debuggear la conexión de nuestro Tomcat con el servidor HiveMQ agregamos el siguiente argumento al arranque del Tomcat:

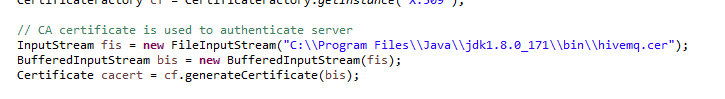
***-Djavax.net.debug=SSL***

******

***Comunicación TLS:***

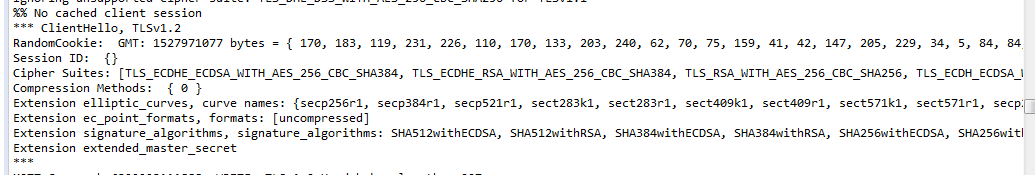
Inicialmente, agregamos en el código el certificado del servidor:



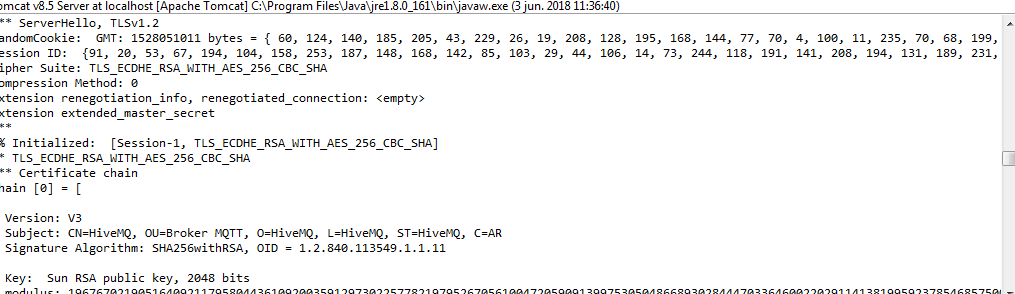


Handshake:

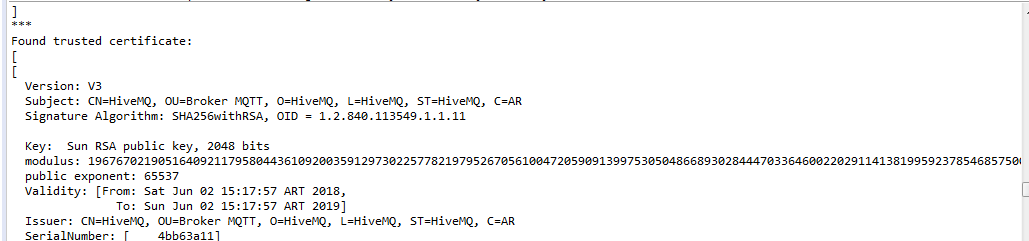
Cliente saluda al servidor:



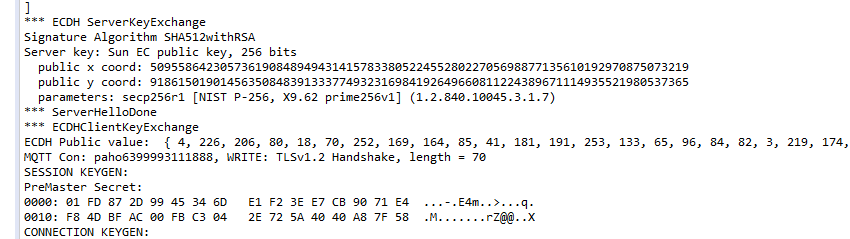
Servidor saluda al cliente enviando su certificado público:



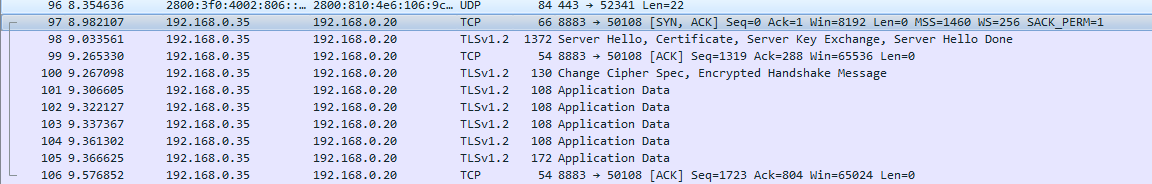
El cliente, encuentra un certificado validado en su almacén de certificados (lo especificamos al principio):



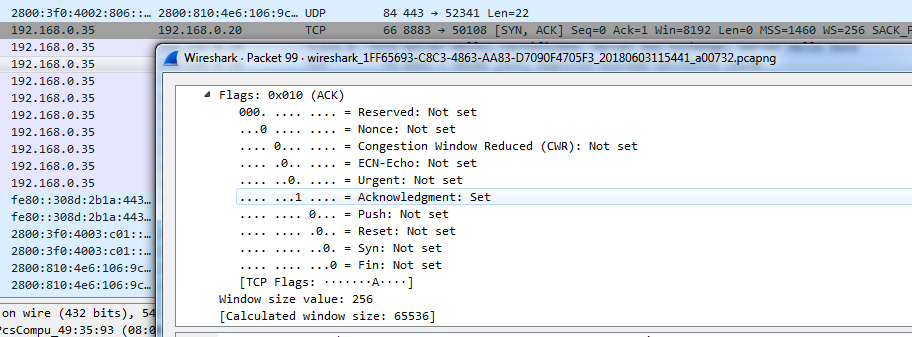
Se completa la fase de saludo, y comienza la sesión:



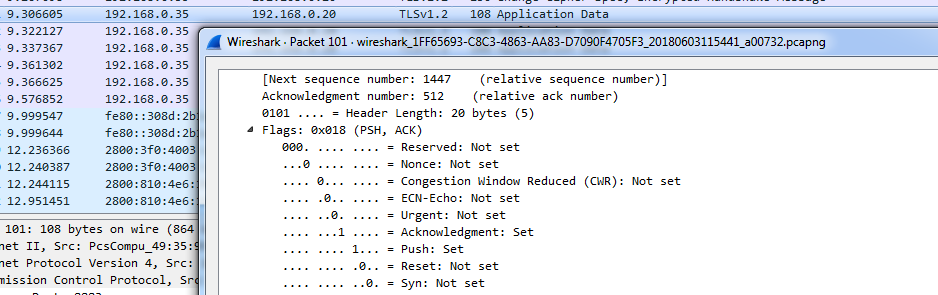
Podemos observar con el Wireshark la comunicación TCP que se produce para los paquetes de control y cómo se encriptan con TLSv1.2 los paquetes de datos (flag push seteado):



Paquete de control:



Paquete de datos:



Referencia conexión:

<https://gist.github.com/sharonbn/4104301>