Trabajo Práctico 1 - Sockets

En la Provincia de Córdoba existen aproximadamente 400 estaciones hidro-meteorológicas automáticas (AWS por sus siglas en inglés), de distintas marcas, modelos y configuraciones, que pertenecen a distintas redes, instituciones y organismos, desparramadas por todo el territorio provincial. Cada AWS consta de una serie de sensores (barómetro, termómetro, sensor de radiación solar, etc.), conectado a un sistema de adquisición de datos, que toma medidas de los sensores (telemetría) un período determinado de tiempo.

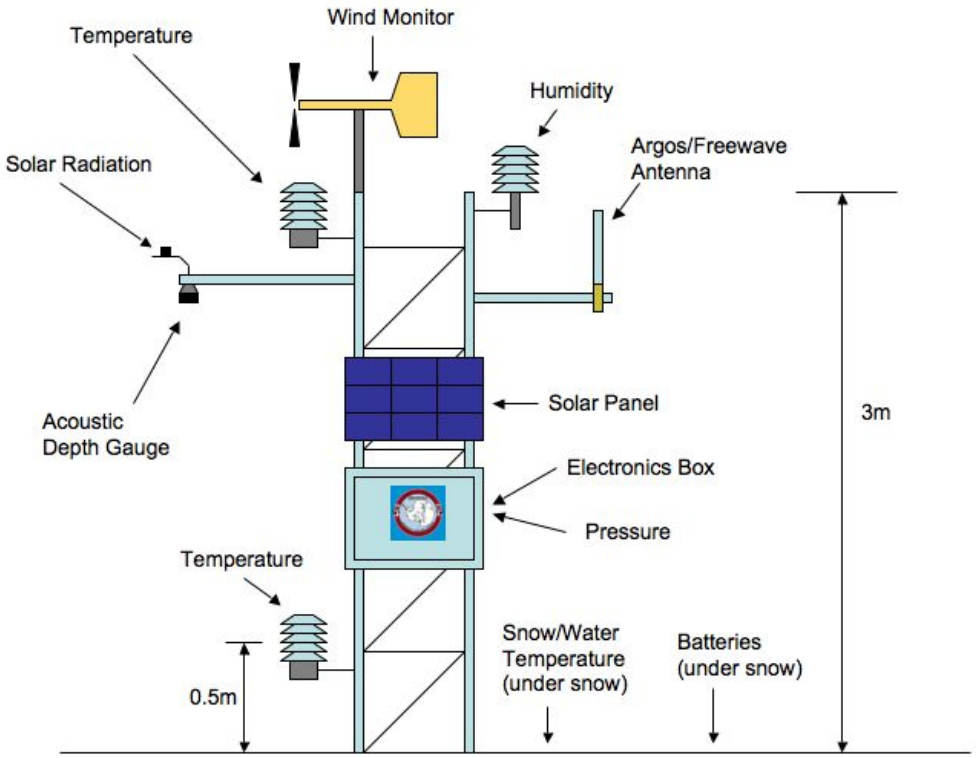

Ilustración 1 - Diagrama de una Estación Meteorológica

Estos datos se guardan en formato CVS, con el siguiente formato:

Número,Estación,ID Localidad,Fecha,Temperatura [ºC],Humedad [%HR],Punto de Rocío [ºC],Precipitación [mm],Velocidad Viento [Km/h],Dirección Viento,Rafaga Maxima [Km/h],Presión [hPa],Radiación Solar [W/m2],Temperatura Suelo 1 [ºC],Temperatura Suelo 2 [ºC],Temperatura Suelo 3 [ºC],Humedad Suelo 1 [%grav],Humedad Suelo 2 [%grav],Humedad Suelo 3 [%grav],Humedad de Hoja [%],

Se le solicita a la Escuela de Ingeniería en Computación que diseñe, implemente y testee un software (desarrollado en C), que permita realizar la conexión, control y transferencia de datos telemétricos entre el un cliente y el servidor que contiene los datos de las AWS (arquitectura cliente servidor).

El programa que corre en el Cliente debe permitir conectarse a al Servidor, de forma segura (orientado a conexión), utilizando un puerto fijo (6020) y tomar un número de un puerto libre (aleatorio) de su sistema operativo.

Debe implementarse un sistema de autentificación (usuario y password) de acceso al servidor.

A los fines de establecer la conexión, el usuario debe ejecutar el programa cliente, el cual debe proporcionar un prompt, que aceptará únicamente el siguiente comando:

- ***connect usuario@numero_ip:port***: debe conectarse a la AWS que posea esa *ip* y a ese *puerto y loguearse con ese usuario. Luego debe solicitarle el password de dicho usuario.*

Al llegarle esta información al servidor, de no existir el usuario o de haber ingresado un password incorrecto, debe notificarse al cliente, con leyenda “nombre de usuario y/o contraseña incorrecto”. En caso de haber sido positivo, el servidor le envía un mensaje de autorización.

Una vez autorizado, el cliente muestra un menú de opciones a ejecutar. Luego la aplicación proveerá de un "prompt" que identifica al usuario (*usuario@servidor:*). En el contexto de este "prompt" se podrán ejecutar comandos propios de la aplicación. Los comandos de la aplicación son:

- ***listar***: muestra un listado de todas las estaciones que hay en la “base de datos”, y muestra de que censores tiene datos.

- **descargar nº_estación**: descarga un archivo con todos los datos de nº_estación.

- ***diario_precipitacion nº_estación***: muestra el acumulado diario de la variable precipitación de nº_estación (*nº_día: acumnulado mm*).

- ***mensual_precipitacion nº_estación***: muestra el acumulado mensual de la variable precipitación (*nº_día: acumnulado mm*).

- ***promedio variable***: muestra el promedio de todas las muestras de la variable de cada estación (nº_estacion: promedio.

- ***desconectar***: termina la sesión del usuario.

Se pide que la transferencia (descarga) de archivos se realice con utilizando conexión no segura (sin conexión).

Durante la operación de transferencia del archivo, se debe bloquear la interfaz de usuario del programa, de tal forma que no se puedan ingresar nuevos comandos en la aplicación hasta que no haya finalizado la transferencia.

Debe incluirse un mecanismo de control y manejo de errores por parte del servidor con comunicación al cliente.

Todos los procesos deben ser mono-thread.

A fines de llevar a cabo su implementación, se provee de una plataforma de desarrollo, INTEL Galileo V1, sobre la cual desarrollar el prototipo de ingeniería. A su vez también hace entrega de un archivo con los datos de una estación meteorológica (telemetría), con el cual realizar las pruebas. El estudiante puede utilizar su propia placa

de desarrollo, siempre y cuando soporte MMU y posea un arquitectura compatible con GNU/Linux.

Tanto la especificación del protocolo de red en la capa de aplicación, así como la elección de la interfaz de usuario del programa cliente, quedan liberadas a criterio del alumno.

Se recomienda el uso de Cppcheck y la compilación con el uso de las flags de warning *-Werror*, *-Wall* y *-pedantic*. Se pide utilizar el estilo de escritura de código de GNU [1] o el estilo de escritura del kernel de Linux [2].

Se deberá proveer los archivos fuente, así como cualquier otro archivo asociado a la compilación, si existiera (archivos de proyecto "Makefile", por ejemplo) y el código correctamente documentado. También se debe entregar un informe, con el formato adjunto. Se debe asumir que las pruebas de compilación se realizarán en un equipo que cuenta con las herramientas típicas de consola para el desarrollo de programas (Ej: gcc, make), y **NO** se cuenta con herramientas "GUI" para la compilación de los mismos (Ej: eclipse).

FECHA DE ENTREGA: 16 de Abril de 2017 a las 23:59

Referencias:

[1] GNU, "*5 Making The Best Use of C*", https://www.gnu.org/prep/standards/html_node/Writing-C.html, [Marzo 2017]
[2] Trovalds, L., Et al., https://github.com/torvalds/linux/tree/master/Documentation/process, [Marzo, 2017]