VERSITA

Annales UMCS Informatica AI XI, 3 (2011) 87–100
DOI: 10.2478/v10065-011-0004-9

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Small prototype acquisition system with secure remote data access

Dariusz Rzońca*, Andrzej Stec†

Department of Computer and Control Engineering, Rzeszów University of Technology, W. Pola 2, 35-959 Rzeszów, Poland

Abstract

The paper describes Small Data Acquisition System, named PACQ, which can be used for collecting data from user-programmable I/O modules and sharing data through a web browser. All connections and data transfers between the client and the server are secure. Due to the resource restrictions only symmetric cryptography and challenge-response protocol could be used, but the solution should be resistant to most of common attacks.

1. Introduction

Development of science constantly increases requirements for technical devices. This rule can be observed in many fields, but especially in electronics and computer technology. Due to popularity of the Internet, many devices, even small ones, transfer data using the global network. For some people it creates temptation to steal private information. Unprotected or poorly protected network systems are exposed to a risk of losing data, or even control. Especially dangerous are cases in which an intruder in a tricky way wants to take control over the system and affect its performance. To prevent from such

*E-mail address: drzonca@kia.prz.edu.pl
†E-mail address: astec@kia.prz.edu.pl

situations, it is necessary to implement user authentication and encryption of transmitted data and to introduce other solutions which impede breaking of security protection. Although there is a large group of solutions guaranteeing a high degree of security, not all of them can be applied in every case. One of the reasons is insufficient computing performance of a processor. For devices based on small microcontrollers, with strongly limited resources, it is usually not possible to use asymmetric cryptography. Simpler solutions have to be used instead, but providing a similar level of security. The solution presented in the paper is implemented in a small data acquisition system (PACQ), in which data collected in control and acquisition systems are shared by a small server with secure remote access and data encryption.

For a few years, a team from the Department of Computer and Control Engineering at Rzeszów University of Technology, whose members are the authors of the paper, cooperates with the companies Lumel S.A. from Poland and Praxis Automation Technology B.V. from the Netherlands. The examples of cooperation results are CPDev engineering environment for programming control devices in the languages of IEC 61131-3 standard or algorithms for self-tuning of PID controllers. The solution, presented in the paper, is another proposition, this time for secure collection of data from remote monitoring stations as well as remote control of devices in small distributed systems. The authors hope that the proposed solution can be applied in future products.

2. PACQ system

PACQ is a prototype of small distributed control and measurement system developed at Rzeszów University of Technology. It is designed mainly for data acquisition, but simple control, through programmable I/O modules, is also possible. The main component of the system is a PACQ server, which is a microprocessor device for recording measurement data transmitted from I/O modules through a RS-232/485 converter and built-in a simple web server. Collected data, current or archived, can be viewed by the user via the web browser.

The system consists of several control modules and the main data acquisition with WWW server (Fig. 1). The control modules equipped with inputs and outputs are programmed in the CPDev environment in ST, FBD or IL languages [**1**]. Process variables are exposed by the modules to the PACQ server via a custom, Modbus-like communication protocol. The collected data is stored in the server database.

Communication with I/O modules is done within the industrial network, for which it is assumed that access to it is available only to authorized persons.

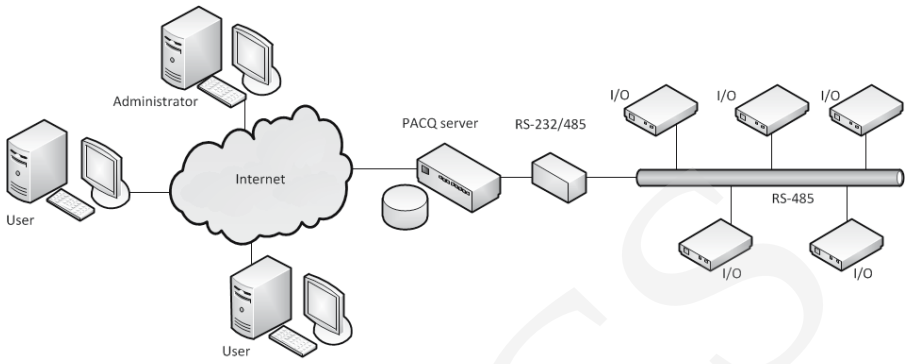

Fig. 1. Topology of PACQ system.

There is no possibility for outsiders to get uncontrolled access to the system resources. So there are no worries about data security from this point of view. But the web server is another case. The data is transmitted through the public network, and access to all information is permitted for each user who knows the username and password. What's more, it is assumed that the network infrastructure, in which the server operates, may be administrated by fraudulent company. The company all privileges and has full access to the local network. It is also assumed, that someone may try to connect to the network and break the security protection.

The PACQ server bases on a microcontroller with highly limited resources and a very simple operating system. It is not able to use advanced encryption solutions due to insufficient computing power of the microcontroller.

The website is available only to the registered users who previously received their own usernames and passwords. They may have administrator (advanced user) or common user privileges, depending on their functions. The first of them can read the recorded data, adjust the input and output of I/O modules and make changes to system configuration. The others can only read the recorded data. For security reasons, some functionalities of the system, such as defining users, changing passwords, and configuring system-critical settings are possible only through specialized application via RS-232 interface of the PACQ server.

Prototype of the PACQ server has been developed on the EVBnet02 evaluation board (Fig. 2) [**2**]. The board is equipped with Atmel AVR ATmega128 microcontroller, RAM and DataFlash memories, 10Mbit/s Ethernet controller, 2xUART, RTC, LCD display and 4 buttons. WWW and FTP services run under the Nut/OS operating system [**3**]. A dedicated master-slave protocol similar to Modbus is used to communicate with the modules.

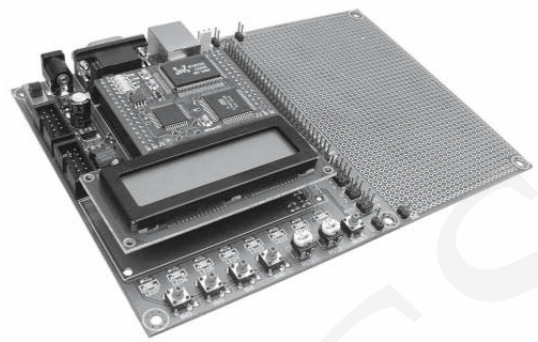

Fig. 2. Prototype board for the PACQ server (EVBnet-02 from Propox [**2**]).

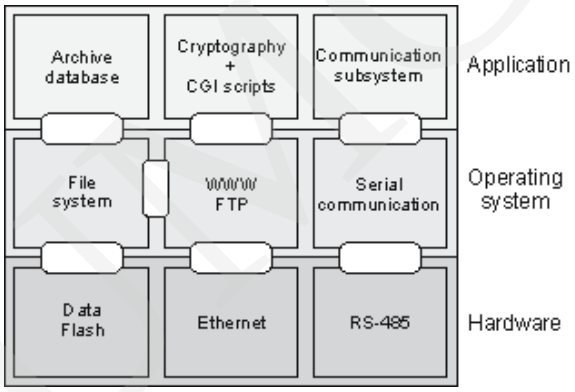

Fig. 3. Basic components of the server program.

Software for the PACQ server has been written in C. The application part consists of three major components (Fig. 3):

- serial communication subsystem for acquiring data from control modules,
- archiving service for storing the data in a database,
- web server CGI extensions and cryptography module for generating dynamic web pages.

Each of these components is serviced by a separate thread of the operating system.

Based on the configuration data stored in a file, the server program periodically polls the appropriate I/O modules. Details of the communication with the control modules as well as a hierarchical model in timed coloured Petri nets and simple performance analysis were presented in [**4**]. Each control module can service from a few to tens of data channels. The current version of the PACQ

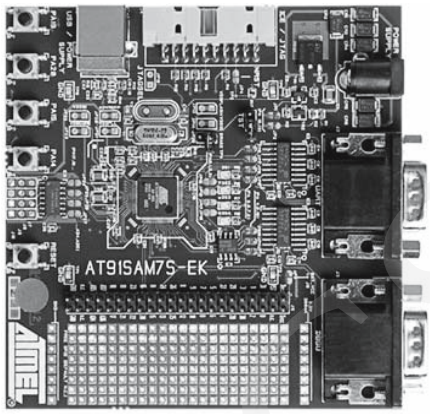

Fig. 4. Prototype board for I/O module (AT91SAM7S-EK from Atmel).

system supports up to 64 channels. Read values are stored in a current-data table and simultaneously archived in a database. These data are available for users in a form of dynamically generated web pages. This is accomplished by the CGI scripts implemented in the operating system and the POST method for data processing of forms. When the connection is established a session key is generated. This key is used for further data encryption. Due to the use of AES-128 algorithm in the CTR mode, each message differs from the previous one, even if it contains the same information. Details of the security algorithms are presented in sect. 3.

Control modules in the PACQ system are small programmable controllers equipped with analog or binary inputs and outputs. The module is based on the development board AT91SAM7S-EK from Atmel [**5**] (Fig. 4) with the ARM7 microcontroller. The board has been equipped with extra I/O hardware and RS-485 bus. Each module can be programmed to perform process control.

Programs are written in CPDev (Control Program Developer) engineering environment developed at Rzeszów University of Technology [**1**]. CPDev integrates tools for programming, simulation, hardware configuration, on-line testing and running control applications. Programs can be written in ST and IL textual languages (Structured Text, Instruction List) and in FBD graphical one (Function Block Diagram), according to the IEC 61131-3 standard [**6**]. Main window of CPDev environment with programs in ST and FBD is shown in Fig. 5. I/O module application runs user programs (from CPDev) under control of FreeRTOS operating system [**7**].

As it was mentioned before, the data collected from I/O modules can be observed by viewing a web page prepared by the PACQ server. More detailed description is provided in sect. 4.

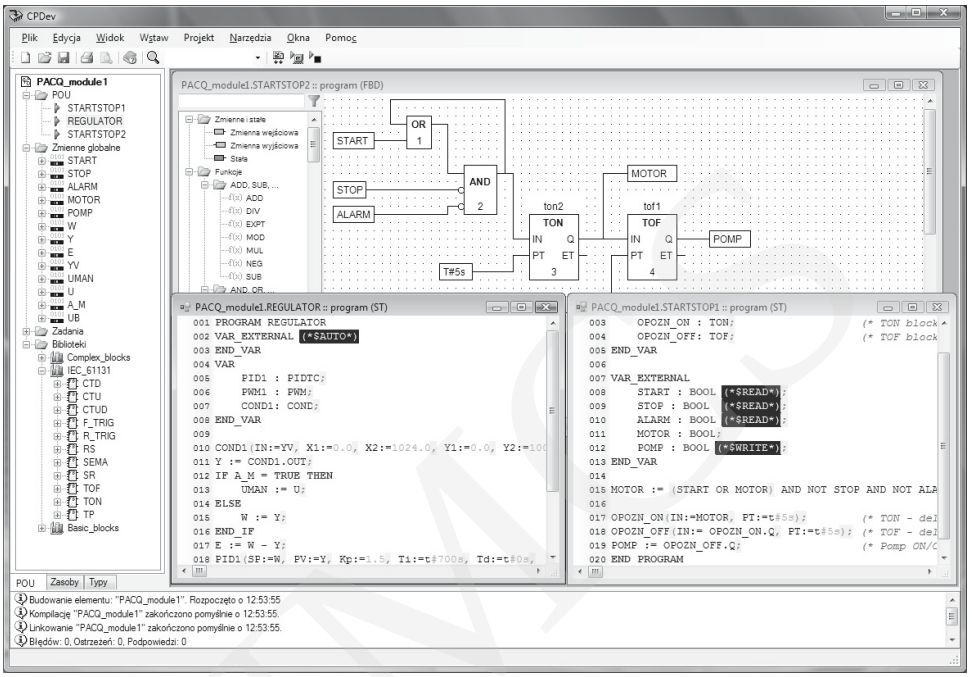

Fig. 5. Main window of CPDev programming environment.

3. Security aspects

During remote access to the device (via Ethernet), there is a risk of disclosing confidential data or introducing unauthorized changes of configuration. It is necessary to use appropriate mechanisms to ensure transmission security. In particular, such mechanisms shall provide authentication, confidentiality and integrity. Unfortunately, limited resources of the microserver force us to develop a specific solution, different from those typically used in large systems, however, providing a sufficient level of security, despite limited hardware. In particular, algorithms based on the asymmetric key should be avoided, and only symmetric key cryptography can be used due to low microprocessor performance.

Authentication of the user is based on username (ID) and password. The password has to be defined earlier, via secure channel (in service mode during local serial port connection by a special program). This password is stored in the PACQ in a hashed form, together with random salt. This way there is no possibility to read the password from the device, it is known only to the user. Common attack on the hashed passwords involves using a precomputed lookup table of hashes for typical passwords to reverse hash function. Usually *rainbow tables* instead simple lookup tables are used as a time-memory tradeoff.

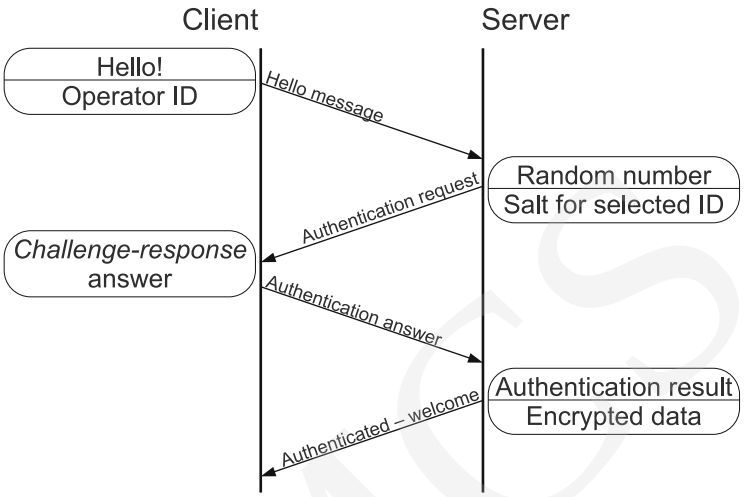

Fig. 6. Authentication process in the PACQ server.

Employing the random salt and running the hash function multiple times (*key stretching*), according to PBKDF2 [**8**], make such an attack infeasible.

The authentication process and format of the transmitted messages are shown in Fig. 6.

To avoid transmitting the password in a plain text during the authorization process we decided to use the *challenge-response* protocol [**9**]. The protocol is used to authenticate the user, and to generate a unique session key used later to encrypt the transmission. The *challenge-response* protocol used in PACQ is shown in Fig. 7.

Authorization and encryption are performed by a special intermediary application which makes a secure tunnel between PACQ and the web viewer. The operator enters the ID (transmitted to the device in a plain text) and password (not transmitted). PACQ answers with a generated random number and the salt connected with the ID. On the client side, the password is hashed together with salt making so-called secret key. The identical secret key is read from the database on the server side. On both sides, independently the secret key is hashed together with the random number. One part of this calculation is transmitted to the PACQ – the correct value means successful authentication. The remaining part of the hash makes a session key, generated on both sides, used later to encrypt the transmission, and MAC key.

Commonly AES-128 [**10**] is used as a cipher and SHA-256 [**11**] as a hash function are used, however, we decided to choose the CTR (Counter) mode (Fig. 8) as a cipher mode of operation.

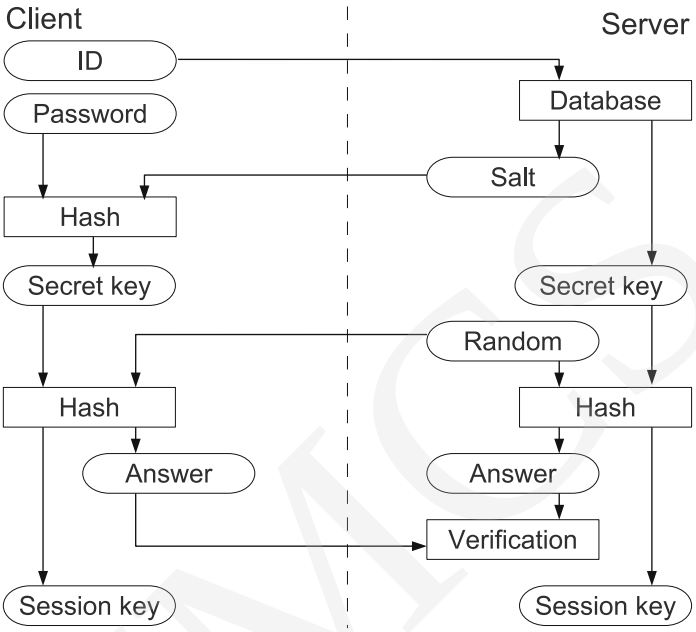

Fig. 7. *Challenge-response* protocol.

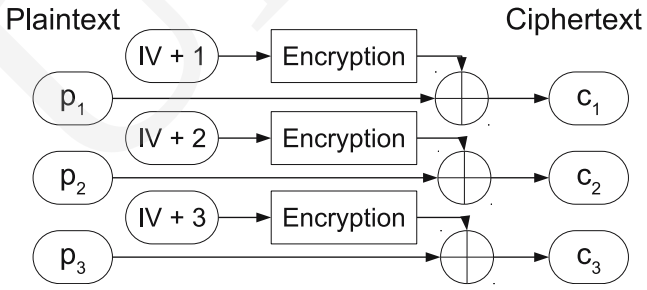

Fig. 8. Block cipher in the Counter (CTR) mode.

The CTR mode, in fact, turns the block cipher (AES) into a stream cipher. The sequential numbers (counter) together with the initialization vector (IV) are encrypted by AES using the session key. The AES output is XOR-ed with a plaintext message to create a ciphered message. This way identical plaintext messages are encrypted to different ciphertexts, unlike in the simple ECB (Electronic codebook) mode.

This mode provides confidentiality, but integrity of the message is not guaranteed. An attacker during *man-in-the-middle* attack is not able to decrypt the message, however he can blindly modify the encrypted message in a random way and change its meaning (see Fig. 9). Moreover, if the attacker can guess

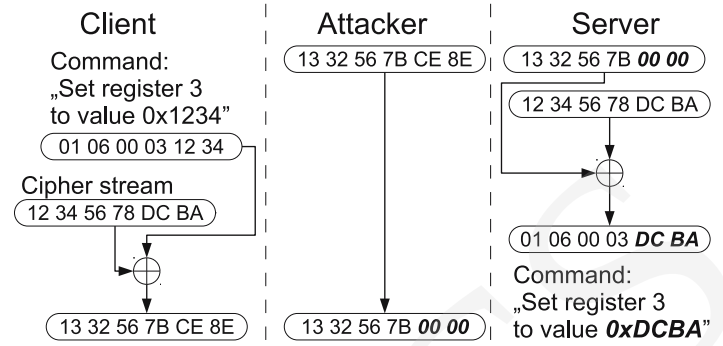

Fig. 9. Blind man-in-the-middle attack.

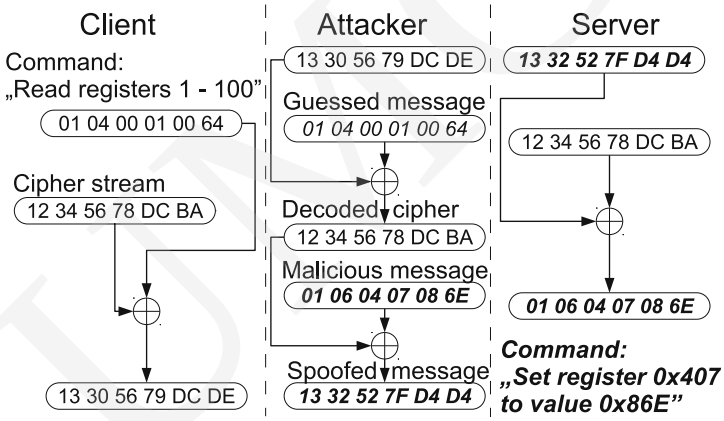

Fig. 10. Man-in-the-middle attack with guessed message.

the plaintext message he can use it to obtain the cipher stream and prepare a spoofed message to perform malicious actions (Fig. 10). Such guess might be possible because industrial applications often behave in a predictable way, e.g. issue the ”read configuration” command at the beginning of the connection.

To avoid spoofing integrity of the received messages must be confirmed. In our case integrity of the transmission is guaranteed by appropriate MAC – Message Authentication Code (HMAC – Hash-based Message Authentication Code [**12**]). HMAC calculation is shown in Fig. 11. The MAC key is XOR-ed with inner and outer paddings. The keyed inner padding is concatenated with the message and hashed. Hash result is concatenated with the keyed outer padding and hashed again to produce HMAC of the message.

To minimize probability of a successful dictionary attack, the password has to fulfil requirements about the length (minimum eight characters). Moreover, the

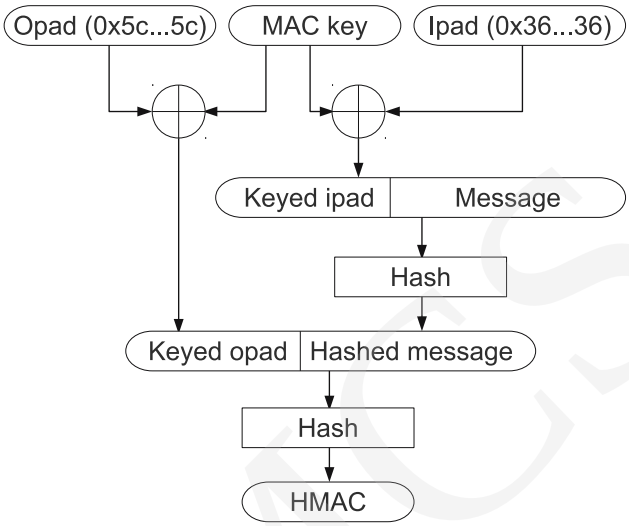

Fig. 11. HMAC calculation.

password must be a combination of small and capital letters as well as digits, it may also contain special characters. After every unsuccessful login attempt (entering the wrong password) the additional delay before the PACQ answers is increased.

The following transmission is encrypted by the generated session key. The operator can perform a remote configuration of the device, watch values of the variables (current and archived ones) as well as read and set remote inputs and outputs.

4. Web page

Any popular web browser (e.g. IE, FireFox, Opera, Chrome) in a current version is required for page browsing of the PACQ server. Additionally, there is also a specialized intermediary application, which must be run before. The application is needed for data encryption/decryption and to protect against unauthorized redirection to another page for password phishing.

The web pages stored on the server are written in HTML using JavaScript and CSS. Because of low computing performance of the PACQ server, a big part of calculations is made by the browser (on the client side). To reduce the size of transmitted messages from the server, only raw data (as tables) are sent out, and the rest of actions are made by the browser.

To be able to connect to the website, it is necessary to launch the intermediary application which transmits all messages from theb web browser to the

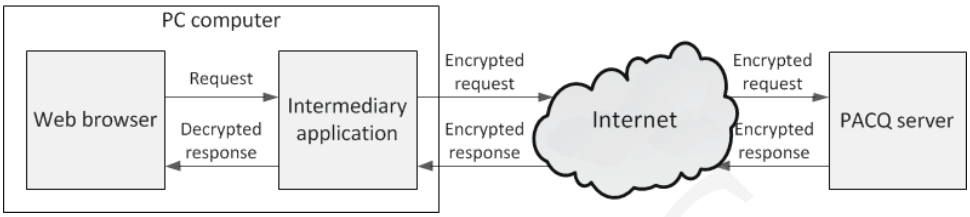

Fig. 12. Dataflow between the web browser and the PACQ server.

PACQ server. This application encrypts and decrypts messages, and also protects against password phishing by redirection the user to a fake website (web spoofing). If you try a direct connection to the server you will only see the information that the intermediary application is needed and it must be launched before. The application can be downloaded from a website, but it is important to download it from a trusted website, preferably a digitally signed one (like the application itself.) Depending on the user's access rights to the resources of PC or personal preferences, the application can be permanently installed on the computer and launched when the computer starts up, or be activated on demand.

When the user starts a web browser he has to connect to the intermediary application, instead of directly to the PACQ server. The application is available at a localhost (127.0.0.1) and a selected port number, e.g. 32000. After the connection is established a welcome page is presented. The user enters the website address of the PACQ server, his username and password. The data are sent to the intermediary application, which tries to generate a session key and to establish connection with the PACQ server. If successful, further communication is continued in secure mode. All requests from intermediary application and responses from the PACQ server are encrypted (Fig. 12). After decryption by the intermediary application web pages are transmitted to the web browser. At the end of the work the user must logout to close connection. The connection can also be closed automatically if no user activity is detected for a declared period of time.

Intermediary application is mainly involved in encryption and decryption. It does not interpret data transmitted from the server and does not store web pages (besides welcome page). It can be treated as an universal solution. Content changes on the web server do not affect intermediary application, so it can be used without modification in other cases.

Depending on the user privileges, the browser page appears in a common user mode (U), which allows us to view only the data, or an advanced user mode (A), which also permits to modify some parameters of the system. The

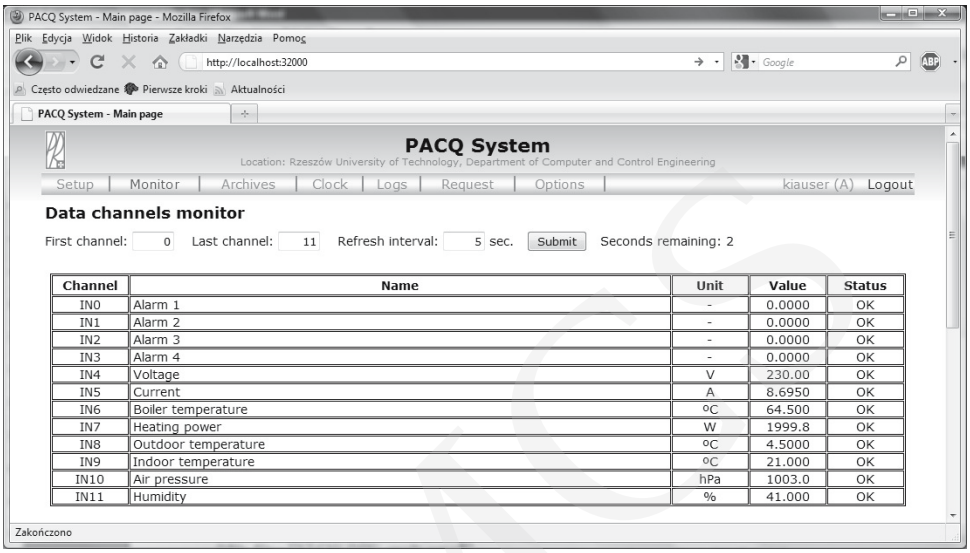

Fig. 13. Sample page for data channels monitoring. The user can choose channels for monitoring and page refresh time.

website allows current data reading (Fig. 13) as well as archive browsing. The size of database mostly depends on the size of DataFlash memory mounted on the prototype board.

Basic functionalities of the website:

- Setup
  Configuration of measuring data channels (name, unit, conversion, interval between I/O module readings, etc.).
- Monitor
  Monitoring of current values of data channels, selection of channels range to be monitored, auto-refresh interval, and displaying the data of monitored channels as a table.
- Archives
  Selection of channels range and the period for searching data, the results type (table, file – TXT/CSV/DBF), results presentation as a table or file download.
- Clock
  Reading or setting the RTC.
- Logs
  Viewing the events (errors, warnings, user activity, etc.).

- Request
  Special commands for I/O control, queries for specific data, service or diagnostic tests.
- Options
  Setup with saving of user preferences for browser settings (the default channels range, date range, the size of browser window, etc.).

Changing channels configuration (Setup), and Real-Time-Clock (Clock) are available only for the advanced user (A). Due to security issues, system-critical settings are set only by using a specialized application, which communicates with the PACQ server via the RS-232 interface. It is possible to configure the system using FTP, but because of lack of encryption, this possibility is blocked by default. It is permissible to unlock the feature using the buttons on the prototype board, but it works only for a preset time (e.g. 30 minutes) or until server reset.

5. Conclusions

So far commonly used small industrial devices connected to the Ethernet network have not supported data encryption. They must be connected to the network via PC computer to ensure data security. However, it is possible to provide user authentication as well as data confidentiality and integrity, despite limited hardware capabilities. An example is the new data acquisition system with secure remote access described in the paper. The central unit of the system is the PACQ server with the data acquisition and WWW capability. The control modules can be programmed in IEC-61131 languages in the CPDev engineering environment. The PACQ system has been developed in a way to provide secure connection between the PACQ server and the web browser. Of course, small devices can not service many users simultaneously, usually only a few, but in many applications it is just enough. Such systems are more economical because they do not need additional equipment and consume less electricity. It seems that the presented solution can be also applied in commercial products.

References

[1] Rzońca, D., Sadolewski, J., Stec, A., Świder, Z., Trybus, B., Trybus, L.: Mini-DCS System Programming in IEC 61131-3 Structured Text. Journal of Automation, Mobile Robotics & Intelligent Systems 2(3) (2008): 48.

[2] EVBnet02 User's Manual; http://www.propox.com/download/docs/EVBnet02_en.pdf

[3] Nut/OS Software Manual; http://www.ethernut.de/pdf/enswm28e.pdf

[4] Rzońca D., Stec A., Trybus B.: Data Acquisition Server for Mini Distributed Control System, in: A. Kwiecień, P. Gaj, P. Stera (Eds.): Computer Networks 2011, Communications in Computer and Information Science 160 (2011): 398.

[5] AT91SAM7S-EK Evaluation Board User Guide;
http://www.atmel.com/dyn/resources/prod_documents/doc6112.pdf
[6] IEC 61131-3 Standard: Programmable Controllers. Part 3. Programming Languages, IEC 2003.
[7] Barry, R.: Using the FreeRTOS Real Time Kernel – A Practical Guide.
[8] Turan M. S., Barker E., Burr W., Chen L. : Recommendation for Password-Based Key Derivation Part 1: Storage Applications. National Institute of Standards and Technology Special Publication 800-132 (2010).
http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf
[9] Kutyłowski M., Strothmann W.: Kryptografia. Teoria i praktyka zabezpieczania systemów komputerowych. Oficyna Wydawnicza Read Me, Warszawa (1999) (in Polish).
[10] Advanced Encryption Standard (AES). Federal Information Processing Standards Publications 197, National Institute of Standards and Technology (2001).
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
[11] Secure Hash Standard (SHS). Federal Information Processing Standards Publications 180-3, National Institute of Standards and Technology (2008).
http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf
[12] The Keyed-Hash Message Authentication Code (HMAC). Federal Information Processing Standards Publications 198-1, National Institute of Standards and Technology (2008).
http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf