The communication unit of measuring device in power engineering

PETR MLYNEK, MARTIN KOUTNY, JIRI MISUREC
Department of Telecommunications, Faculty of Electrical Engineering and Communication,
Brno University of Technology, Purkynova 118, 612 00 Brno, CZECH REPUBLIC
xmlyne01@phd.feec.vutbr.cz, koutny.martin@phd.feec.vutbr.cz, misurec@feec.vutbr.cz
http://www.utko.feec.vutbr.cz

Abstract: Various devices, for example electrometers, can be connected to the Internet by means of the communication unit. Thanks to the communication unit, remote acquisition of power consumption data is enabled. This article focuses on the design of the communication unit for remote data acquisition. The article also deals with the design and implementation of the authentication, secure data transmission, and transmission block integrity. Finally describes the system for testing robustness of the communication unit.

Key-Words: communication unit, remote data acquisition, RCM3700, Internet, measuring device, authentication, transmission block, Dynamic C, attack

1 Introduction

The Internet is now the widest network used for data communication. The tendency to make maximum use of it for data transmission from various devices requires creating appropriate communication units which allow connecting various devices to the Internet, such as electrometer. An electrometer can be connected to the Internet by means of a communication unit, enabling remote acquisition of data on power consumption.

An advantage of remote data acquisition is the possibility of frequent readings without physical presence at the electrometers. Data transmission over the Internet can be the subject of various attacks, which is a disadvantage. For this reason it is necessary to design and include authentication tools in the communication unit and have the possibility of encrypted data transmission.

Communication for network control, monitoring and power consumption metering is mainly running on communication channels based on the RS-232 standard. The construction of new communication channels is expensive and impractical and thus current communication channels will be used, in our case the Internet. For this reason, the communication unit is designed as an RS232/Ethernet transducer and vice versa.

The communication unit enables connecting single-purpose devices for measuring electric quantities of the power network to the Internet via the TCP/IP protocol. Information from these measuring devices is transmitted to a telemetric acquisition system.

2 Communication unit

The communication unit consists of the RCM 3700 module and the motherboard for this module.

2.1 RCM3700

RCM3700 is a cryptographic module (see Fig. 1) assuring cryptographic operations connected with the protection of data transmission. Together with the encryptor it implements authentication and secure transmission of data measured. This is the main purpose of the proposed communication unit from the viewpoint of secure data transmission. The RCM3700 development module includes a number of program libraries, which were not available in the other development modules of similar nature when searching for suitable modules. Moreover, the two required interfaces are already implemented in this module. At this stage, RCM modules are mainly designed for the development and implementation of embedded control systems. The manufacturer, Rabbit Semiconductors, offers a number of such kits, which differ in storage size, communication interfaces or processor type. The core of the module selected is an 8-bit Rabbit3000 microprocessor operating at a frequency of 22.1 MHz, which has sufficient power reserve for implementing also more complicated security algorithms. Currently, the module software is being designed, with the 512KB Flash memory configuration being considered for the program and 512KB SRAM for the data. [1]

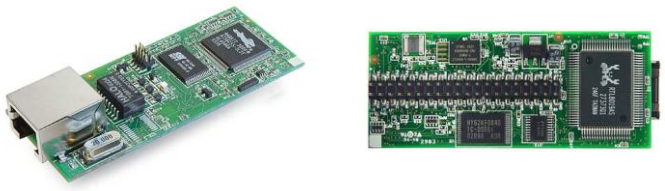

Fig. 1: RCM 3700

2.2 Motherboard for RCM module

The block diagram of the motherboard is shown in Fig 2.

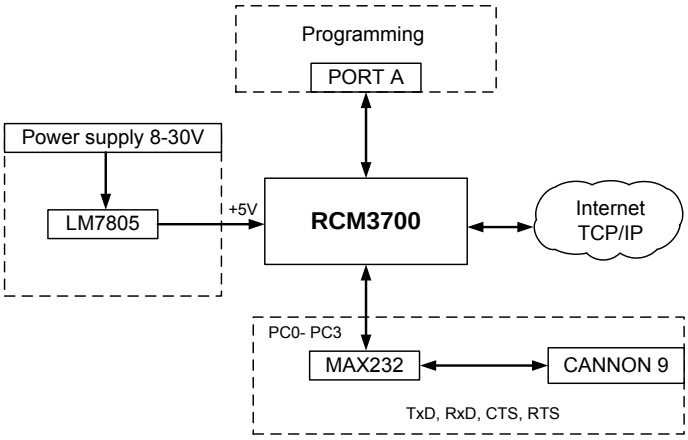

Fig. 2: Block diagram of the motherboard

The module connection was realized via a 40-pin connector. Table 1 shows the description of individual pins, but only of those which were used in the communication unit. A more detailed description detail can be found in [1].

Table 1: Pins used in the RCM3700 module connector

Power part		**Serial interface**	
Pin	Used as	Pin	Used as
GND	Ground	PC1/PG2	RxD
GND	Ground	PC3/PG3	RTS
+5V	power +5V	PC0	TxD
VBAT	power +5V	PC2	CTS

2.2.1 Power supply

The microprocessor needs for its functionality both the CMOS and the TTL power supply. For the function of the RCM 3700 module the TTL level is sufficient. The module itself converts the TTL level to the CMOS level. The LM7805 Stabilizer is used for the TTL level on the input of power supply.

The LM7805 Stabilizer requires for its functionality a DC power supply of 8 - 30V. The output of power supply is a voltage of 5 V, designed for the RCM module.

2.2.2 Serial interface

Another integral part of the motherboard is elements for operating the serial interface. The module contains the serial interface, but it does not contain elements that provide RS232 communication, which needs for its functionality a voltage that is higher than the usual value of TTL.

For this reason, the Maxim MAXS3232CPE driver of serial interface was used. This driver enables mutual conversion of the RS232 and the TTL levels. Signals from RS232 were connected to a standard CANNON 9 connector, which is used in most measuring devices. Only RxD, TxD, RTS, and CTS signals are used, because these signals are sufficient for the majority of applications.

3 Communication string

Fig. 3 shows the communication string of the system of telemetric data acquisition. The station initializes communication via the encryptor, which is then connected to the appropriate cryptography module of the communication unit. After successful mutual authentication, communication can start between the acquisition station and the measuring device.

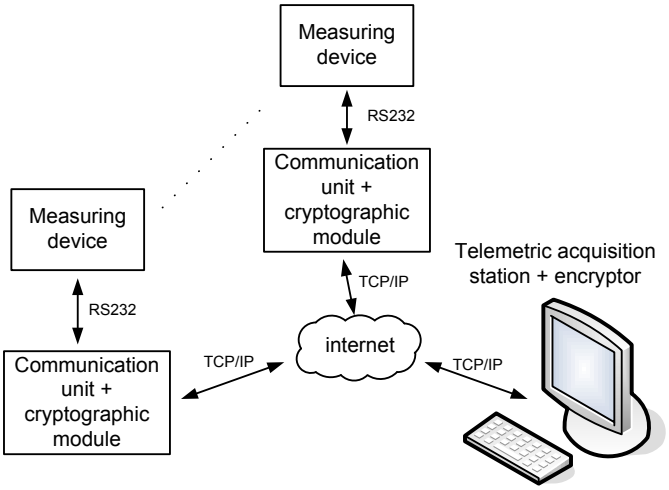

Fig. 3: Communication string

The telemetric acquisition station represents the central point of the network, assuring data collection in its sub-domain. The encryptor is implemented in the station and ensures cryptographic operations in the telemetric acquisition station. Its task is dual authentication while connecting to the cryptographic module, i.e. decrypting the data stream running from the metering facility or encrypting the data stream running towards the metering facility.

The communication unit contains an RCM3700 module. The RCM3700 module is a cryptographic module assuring cryptographic operations connected with the protection of data transmission. Together with the encryptor it implements authentication and secure transmission of the data measured.

4 Communication unit testing using a measuring device

Testing the functionality of the communication unit of remote data acquisition was realized with the PQ monitor MEg33 and the CU-E21 communication module of electrometer.

4.1 PQ monitor MEg33

The PQ monitor is designed for metering voltage quality parameters according to the EN 50160 Standard. More detailed information about the PQ monitor MEg 33 can be found in [2]. The communication unit is connected directly to the PQ monitor, because it contains the RS-232 interface. The PQ-monitor software was not primarily designed for remote data acquisition via the TCP/IP protocol and therefore a mediator was used, which enabled the simulation of serial port. This virtual port was then adjusted, as if it was physically present. Incoming requirements on this port were sent by the TCP protocol to a preconfigured IP address of the unit. The simulation environment is shown in the following figure:

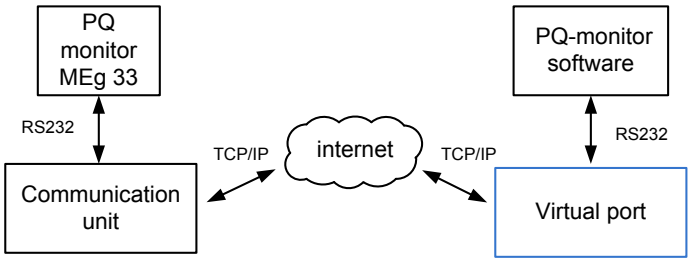

Fig. 4: Simulation environment

4.2 Communication module CU-E21

The electrometer ZMD 310 [3] does not enable communication over the RS-232 interface, but via the communication module CU-E21 it is possible. Module CU-E21 [4] contains the RS-232 serial interface. CU-E21 represents the interface between our communication unit and the electrometer.

In our experimental network, data exchange between the electrometer and the telemetric acquisition station is implemented by the Energy Data Collection (EDC) program [5]. This program provides its services in cooperation with the iMEGA meter2cash and iMEGA Device Driver modules [5]. The task of iMEGA meter2cash is to establish connection with the electrometer and then to share this connection via a virtual serial port in the Windows system. Via the iMEGA Device Driver the virtual port is used to provide communication between the meter2cash program and the virtual serial port. The simulation environment is shown in the following figure:

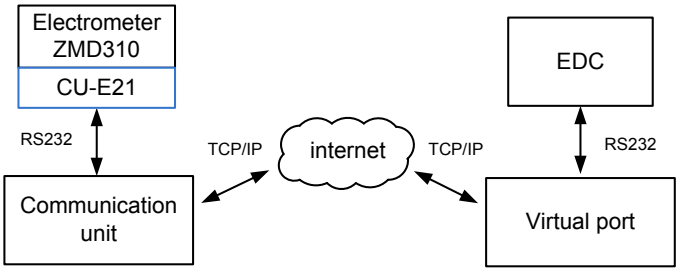

Fig. 5: Simulation environment

5 Model of authentication

Communication between the communication unit and the acquisition station is divided into authentication and the data transmission itself. The communication process is based on the principle of symmetric cryptography, specifically on the block cipher with ECB and CBC modes and AES algorithm [6]. The authentication algorithms are based on the ECB mode. The data transfer is based on the CBC mode.

A precondition for the functioning of authentication is that the value of distribution key, DK, is known to both sides. The distribution key is used only in authentication transmissions. Using all the time one and the same key for secure data transmission entails the risk of potential cryptanalytical attack on the basis of periodization. For this reason, the key must be changed from time to time. [7]

More detailed information about the choice of cryptography can be found in the literature [8]. Fig. 6 illustrates the design of authentication and establishment of keys for transmission.

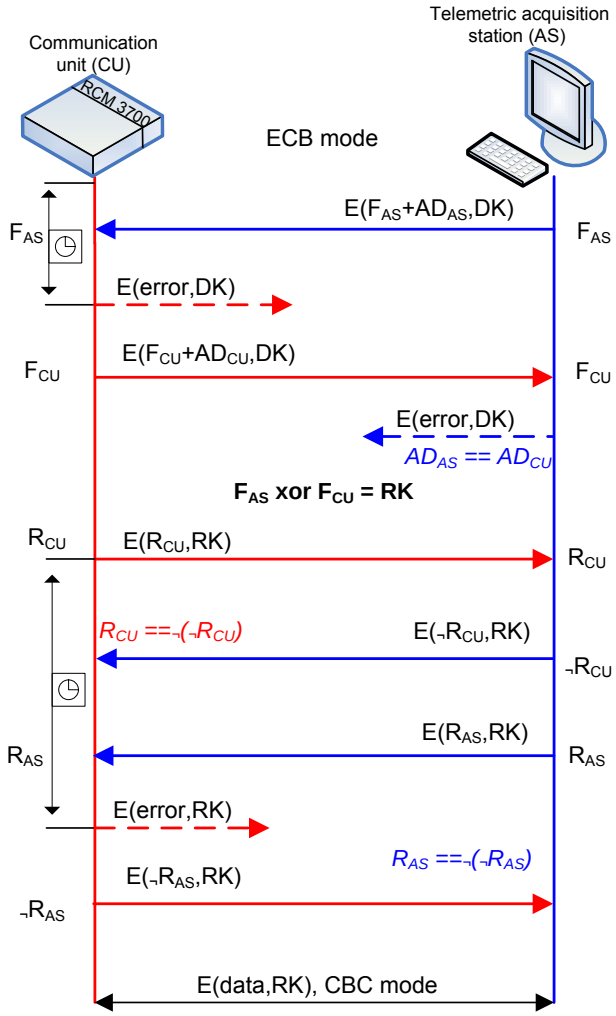

Fig. 6: Model of authentication

Where:

DK is the distribution key for *CU;*
R_{CU}, R_{AS}, F_{CU}, F_{AS} are random numbers;
$\neg R_{AS}$, $\neg R_{CU}$ are negated R_{AS} and R_{CU};
AD_{AS}, AD_{CU} are the addresses of *AS* and *CU;*

Description of authentication [8]:

1) When establishing the connection, *AS* generates a random number F_{AS}. This number F_{AS} together with the address of *AS* is encrypted with a common distribution key *DK* for a given cryptographic module of *CU*. The resulting cryptogram $E\,(F_{AS}+AD_{AS},DK)$ is then sent to the respective cryptographic module.

2) The cryptographic module of *CU* waits for a definite period of time to receive the cryptogram. If the cryptogram does not arrive, the cryptographic module

sends an error message to the telemetric acquisition station *AC* and *CU* returns to the initial state. When the unit has registered an authentication attempt within a defined period of time, it generates a random number F_{CU}. This number F_{CU} together with the address of *CU* is then encrypted with the distribution key. The resulting cryptogram E (F_{CU} + AD_{CU}, DK) is sent to the telemetric acquisition station *AS*.

3) At both ends, the random numbers and addresses received are decrypted by the distribution key. After that, the addresses are checked. If the sender address and the receiver address are different, then the exclusive *or* operation with the two random numbers is performed. The result is a key $RK = F_{AS} \oplus F_{CU}$.

4) The cryptographic module of *CU* generates a random number R_{CU} and encrypts it with the key *RK*. The resulting cryptogram E (R_{CU}, RK) is sent to the telemetric acquisition station *AS*. The cryptographic module starts the timer and waits for a specific period of time for the response from *AS*. The telemetric acquisition station *AS* receives the cryptogram E(R_{CU}, RK), decrypts it and obtains a number R_{CU}. It negates this number, encrypts it with the key *RK* and sends it back as a cryptogram E ($\neg R_{CU}$, RK). The cryptographic module receives the cryptogram E ($\neg R_{CU}$, RK), decrypts it, and compares $\neg R_{CU}$ with R_{CU}. Then it decides on the success of authenticating the telemetric acquisition station, which brings the process of authenticating the *AS* to an end.

5) The telemetric acquisition station *AS* generates a random number R_{AS}, encrypts it with the key *RK* and sends the cryptogram E (R_{AS}, RK) to the cryptographic module of *CU*. If the cryptogram comes within a definite period of time, it is decrypted by the cryptographic module and a number R_{AS} is obtained. This number is negated and then, encrypted with the key *RK*, sent in a cryptogram E ($\neg R_{AS}$, RK) back to the telemetric acquisition station *AS*. The telemetric acquisition station *AS* decrypts the cryptogram and compares the two numbers, R_{AS} and $\neg R_{AS}$. If the numbers are identical, the process of authenticating the cryptographic module of *CU* is completed.

6) The establishment of authentication process is followed by the transmission of encrypted data using the CBC mode and the computed key *RK*, which thus becomes different for each connection.

Time limitations on the communication unit side are designed for the case that the packet gets lost or has a major delay in the network.

The verification of addresses is designed for the protection against reflection attack (see Fig. 7). An attacker can catch the first cryptogram from *AS* and send it back to the *AS* as a first cryptogram from *CU*. *AS* will calculate the key $RK = 0$, because the exclusive *or* operation with two identical numbers is zero (see Table 3).

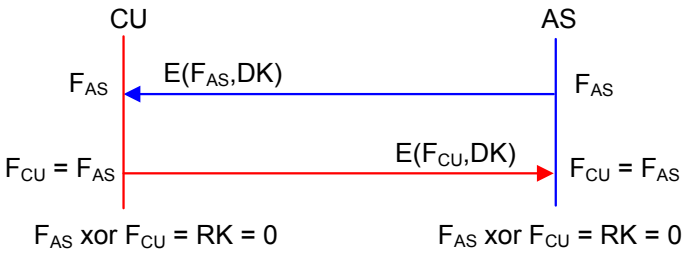

Fig. 7: Authentication mirroring

A	B	A⊕B
0	0	0
1	0	1
0	1	1
1	1	0

Table 3: Exclusive *or* operation

6 Transmission block

Fig. 8 shows the format of an encrypted message. The CBC mode is used for the process of secure data transmission. It is therefore necessary to fill the data with padding bits for the sake of preserving the required size of data block. Since we use PAD bits, it is also necessary to know for decrypting the message how many bits from the block are taken up by the message and how many bits are used for filling. For this reason, 16 bits of the message have been reserved for information about the block length LEN. Another 16 bits have been reserved for message check sum CRC [9], which together with the CBC mode guarantees data integrity. The application of initialization vector follows from the principle of CBC mode. This

vector will be generated for each transmission randomly in order to assure a dissimilarity of the messages transmitted.

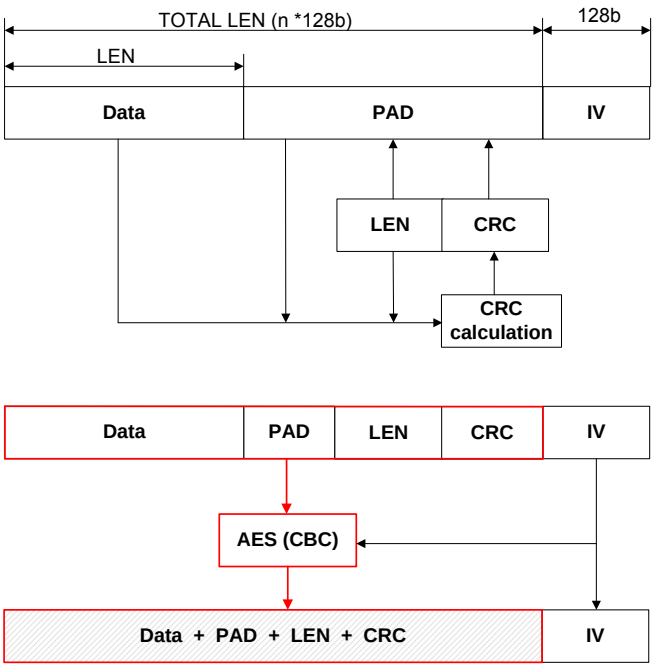

Fig. 8: Transmission block

Communication with the PQ monitor works on the instruction-response principle. In an experimental network testing procedure it was established that the PQ monitor software divides instructions into several blocks. These instructions are encrypted and sent to the communication unit over Ethernet.

These blocks are received by the communication unit, but these blocks came in one stream. These blocks are decrypted by the communication unit and the check sum is calculated. The check sum is not correct, because it was calculated from several blocks (see Fig.9).

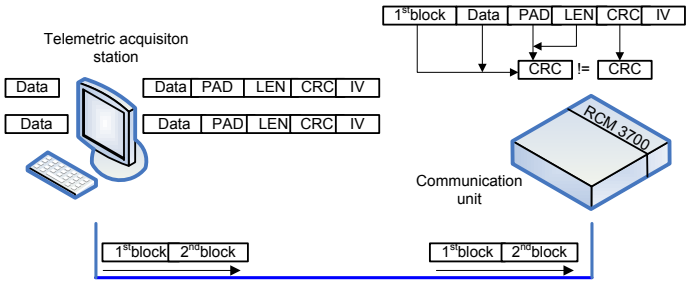

Fig. 9: CRC calculation

For this reason a new transmission block was designed (see Fig.10). The new block contains, in addition, the total length of the block. The total length is calculated without an initialization vector. In the case of an intentional change of the total length, the attacker is not able to ensure the corresponding check sum for this block.

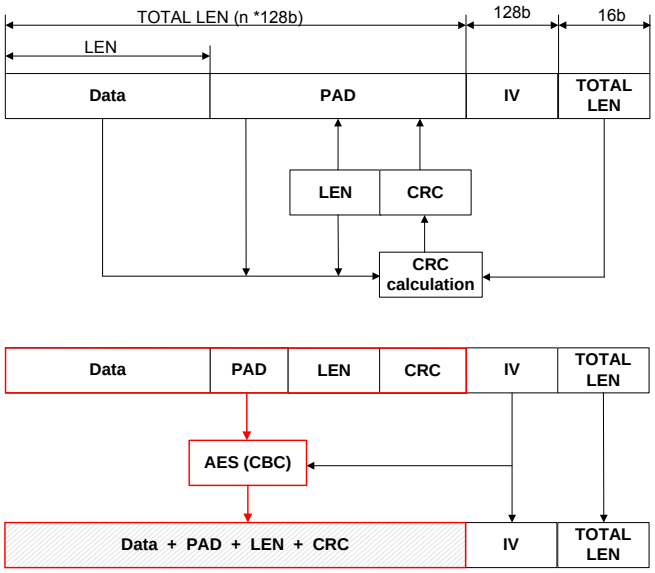

Fig. 10: Transmission block

An algorithm is designed on the communication unit side (see Fig. 11) that reads the total length value from the data received. This value is compared with the real length of received data. If the values are the same, only one block was sent. If the values are different, then the total length of next block was read. If the sum of the two total lengths is the same as the real length of received data, two blocks were sent. If not, the algorithm will continue running.

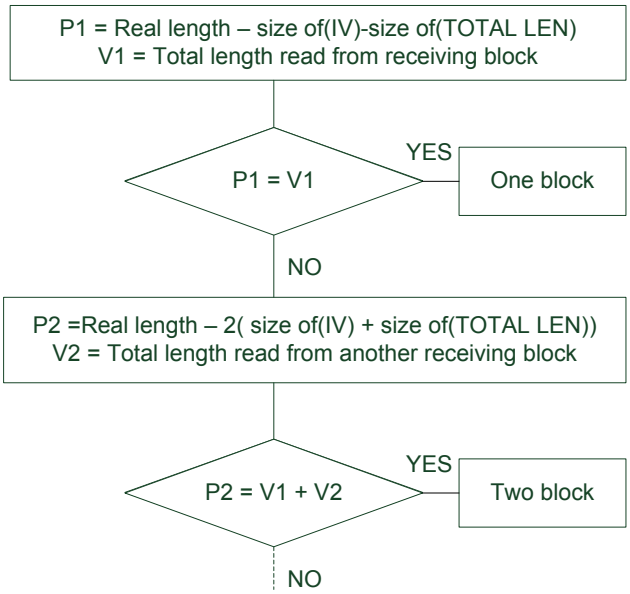

Fig. 11: Flow diagram

7 Implementation on RCM3700

The designed communication process was implemented on an RCM3700 development kit of Rabbit Semiconductors. Programs for the Rabbit 3000 microprocessor are written in the Dynamic C [10] development environment of Z-World, which has been derived from the classical ANSI C standard and complemented with functions for working with the module.

The programming of secure communication transmission is based on the TCP/IP state machine of a protocol [11] that in a shortened form has three states:

- Establishment of connection – handshake, creation of tunnels, etc.,
- Transmission of data – confirmed transmission of TCP packets,
- Termination of connection – after the last received packet and call for termination.

For the purpose of implementing authentication, which is an extra state, the basic three-state system has been extended by further states. The principle is shown in Fig. 12.

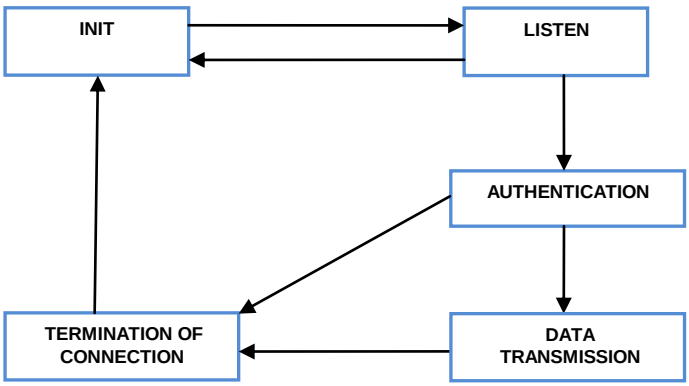

Fig. 12: State machine designed

This state machine was implemented by an infinite-loop function, which constantly tests the network interface of the module: *my_handler(&my _state)*.

The generated function represents the state machine of the TCP protocol and is complemented with authentication mechanisms implemented by a classical switch:

```
switch(state->state)
{
case INIT :
…
break;
case LISTEN:
…
break;
…
}
```

The input states and their brief characteristic are given in Table 3. The first column gives the state that can occur in the machine while the second column gives its brief characteristic.

Table 3: Designed scheme of communication

State	Description
INIT	opening the port, waiting for the beginning of connection with encryptor
LISTEN	TCP handshake
AUTHENTICATION	dual authentication, establishing a new key
DATA TRANSMISSION	mutual data transmission begins; encryption of sent and decryption of received messages
TERMINATION OF CONNECTION	erroneous authentication, another error or termination of transmission terminates the connection

7.1 AES standard and its implementation into module

AES (Advanced Encryption Standard) is a public standard for whose application no license fees are paid [12]. The Rijndael algorithm was officially adopted by the American NIST (National Institute of Standards and Technology) institution as the encryption standard. The new AES standard was to replace the outdated DES (Data Encryption Standard) algorithm, which had been used till then.

It is a block cipher with a key length that can acquire three values: 128, 192 and 256 bits. The length of blocks is in all cases 128 bits but, depending on the key length, the number of rounds changes (both AES and DES make use of repeated applications of parts of algorithm). Ten rounds are sufficient for the shortest key, twelve for the medium key, and fourteen for the 256-bit key. In individual rounds the substitution is first performed and this is followed by two special transposition steps. The block is arranged in a matrix, with individual rows being rotated as the first. In the next step, the columns are mixed up via multiplication by a special matrix. In the end, the data are combined with the encryption key. Just as with DES, the key changes for each round.

In view of the basic operation used, the Rijndael algorithm can be implemented effectively in both software and hardware products.

For the implementation of AES encryption with the 128-bit key the Dynamic C environment contains the AES_CRYPT.LIB library with the necessary functions. Prior to the beginning of message encryption and decryption it is necessary to generate an expanded key, which uses the functions to set the keys of individual rounds:

```
void AESexpandKey(char *expanded, char
*key, int nb, int nk, int rounds)
```

The function used for data encryption:

```
void AESencrypt(char *data, char
*expandedkey, int nb, int nk)
```

The function used for data decryption:

```
void AESdecrypt(char *data, char
*expandedkey, int nb, int nk)
```

For the generation of random numbers necessary for authentication and key establishment, the libraries were extended to include the functions:

```
void random (unsigned char *key)

unsigned char irand ()
```

Generating a random number is based on global non-initiated variable *ranSeed*, on the *xor* operation and on bit operations with this variable. Since a software-implemented generator of pseudo-random numbers is concerned here, certain periodization can be expected. So far, it has not been proved.

The AES library contains the proposed modes of block ciphers. For the ECB mode, functions were used that initiate the expanded key and can encrypt and decrypt data in the ECB mode. The functions are:

```
void AESinitStream(AESstreamState
*state, char *key, char *init_vector)

void AESencryptStream(AESstreamState
*state, char *data, int count)

void AESdecryptStream(AESstreamState
*state, char *data, int count)
```

Similarly for the CBC mode, functions were used that initiate the expanded key and can encrypt and decrypt data in the CBC mode. The functions are:

```
void AESinitStream(AESstreamState
*state, char *key, char *init_vector)

void
AESencryptStream_CBC(AESstreamState
*state, char *data, int count)

void
AESdecryptStream_CBC(AESstreamState
*state, char *data, int count)
```

8 System for testing the robustness of the communication unit

Testing the security of the communication unit for remote data acquisition was realized on a test facility. The block diagram is shown in Fig. 13.

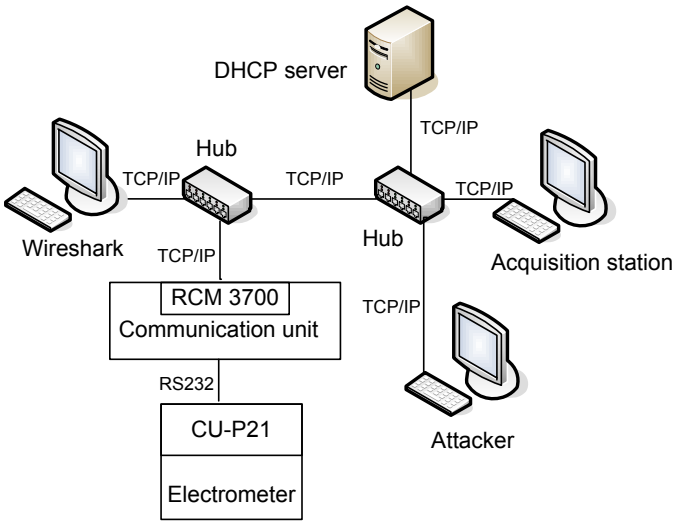

Fig. 13: Block diagram of test facility

The system for testing the communication security and robustness of the communication unit was created. This system contains potential attacks from the Internet. The system for testing the communication unit was created in the Bourne Again Shell. The purpose of the system is to merge all the methods and the attacks into one system.

Example of system:

```
System for testing communication unit
Scanning
   1.Scanning for hosts
   2.Scanning for ports
Denial of services
   3.SYN flood
   4.Reset of connection
Man in the middle attack
   5.ARP spoofing
   6.DHCP spoofing
   7.MAC flooding
```

8.1 Scanning

Nmap program is used for scanning. *Nmap* serves to finding live systems and services.

8.1.1 Scanning for hosts

A multiple ping is used for scanning for hosts, which are potential victims. During multiple ping, *nmap* sends an ICMP echo request packet to the destination system. If an ICMP echo reply is received, the system is up, and ICMP packets are not blocked. If there is no response to the ICMP ping, *nmap* will try a ”TCP Ping”, to determine whether ICMP is blocked or whether the host is really not online. A TCP Ping sends either a SYN or an ACK packet to any port (80 is the default) on the remote system. If RST or a SYN/ACK is returned, then the remote system is online. If the remote system does not respond, it is either offline or the chosen port is filtered and thus not responding to anything. [13]

8.1.2 Scanning for ports

TCP SYN scan is used for finding ports, on which network services are running. This information is necessary for a successful attack. TCP SYN scan sending a SYN packet and looking at the response. If SYN/ACK is sent back, the port is open and the remote end is trying to open a TCP connection. If the port is closed, an RST will be sent. [13]

8.2 Denial of services

8.2.1 SYN flood

The SYN flood attack can be realized with the *hping* program. The SYN flood attacks overload network resources. During a SYN flood attack, the attacker sends a large number of SYN packets without the corresponding ACK packet responses to the victim’s SYN/ACK packets. The victim’s connections table rapidly fills with incomplete connections, crowding out the legitimate traffic.

8.2.2 Reset of connection

For the realization of resetting connection must be the attacker on the same network as the victim. The host machine the victim is communicating with can be

anywhere. The first step is to use a sniffing technique to sniff the victim's connection, which allows the attacker to find out the sequence numbers. Then the attacker sends a spoofed RST packet with correct sequence number to the host machine. Only packets with an ACK flag can be reset and therefore we create filters only for ACK packets [14]. We realized an attack using the *tcpdump*, *awk* programs and the *nemesi*s packet-injection tool. *Tcpdump* is used to sniff for established connections by filtering for packets with the ACK flag turned on. *Awk* is a scripting tool that is used to parse through the *tcpdump* output to extract the source and destination IP address, ports, MAC addresses, and acknowledgment and sequence numbers. This information can be used to create a spoofed RST packet using the *nemesis* program. This spoofed packet is then sent out, and all connections that are captured by *tcpdump* will be reset. This process is realized in the script.

8.2.3 Protection

For the realization of the attacks the attacker must be on the same network as the victim.

The basic protection against the SYN flood consists in shortening the period during which the server is waiting for the continuation of the relation which was started by the SYN packet. Another protection is to block the network traffic coming from false addresses. This protection can be realized by a firewall [15], which is situated on the communication unit side and on the acquisition station side. The protection in the form of SYN and RST Cookie is also possible. Software for network analysis can recognize incoming SYN flood attacks too.

To reset the connection, the sequence numbers must be known. Protection against connection reset is provided by reducing the size of connection windows and the source port must be chosen completely randomly.

8.3 Man in the middle attack

8.3.1 ARP spoofing

ARP spoofing [16] can be realized by *arpspoof* tool. The *arpspoof* tool falsifies all ARP responses, so that network traffic designed for the victim finishes at the attacker and vice versa. The *arpspoof* tool constantly sends to the victim ARP answers telling it that the MAC address belonging to the IP address of the opposite computer is an attacker MAC address. The victim will believe and make a wrong entry in their ARP cache. Next time the victim wants to send an IP packet to the opposite computer it sends the packet to the attacker MAC address.

8.3.2 DHCP spoofing

DHCP spoofing can be realized by the *dhcpx* program. *Dhcpx* program use up all IP addresses of the regular DHCP server. After that we can start own DHCP server. For this can be use *Ettercap* tool.

8.3.3 MAC flooding

MAC flooding attack can be realized by *macof* tool. *Macof* floods the local network with random MAC addresses ant therefore the CAM table of the switch is consumed.

8.3.4 Protection

With ARP spoofing, DHCP spoofing or MAC flooding we can manipulate communication between the electrometer and the acquisition station, but this communication is encrypted with symmetric cipher algorithm AES. For decryption we would have to perform 2^{128} operations, this is not possible to accomplish in real time.

For the realization of these attacks must be the attacker on the same network as the victim.

9 Conclusion

Great accessibility of Internet offers a lot of benefits, but also brings many security risks. Great accessibility predestinates it for data transmissions and remote data acquisition from measuring equipment.

The communication unit serves to secure data transmission between energetic devices and the power network operator. The communication unit contains a serial interface, which is included in most energetic devices for local data acquisition. This serial interface is converted to the Ethernet in the unit. This transducer enables connection to the Internet and remote data acquisition.

The paper describes two possible methods of testing the functionality of communication unit. In one method, the PQ monitor MEg 33 is used, which is designed for metering the voltage quality parameters. In the other method, the CU-E21 communication module of electrometer is used.

Data transmission over the Internet can be the subject of various attacks and therefore secure authentication and secure data transmission are described in this article. The

new transmission block, which ensures a secure and reliable data transmission, is described too.

The paper suggests a possible implementation of a simple authentication algorithm. Authentication works on the principle of two secret random numbers, which are exchanged in the authentication mechanisms. From these random numbers the encryption key for data transmission is derived.

10 Acknowledgement

The paper was written within the Czech Academy of Sciences project No. 1ET 110530523.

References:

[1] Rabbit Semiconductor Inc.: *RabbitCore RCM3700: User's Manual*. 2005. 166p.

[2] MEgA Měřicí Energetické Aparáty: *PQ monitor: MEg30, MEg31, MEg32 a MEg33*. 2006. Online: <http://e-mega.cz/doc/pqmonitor_mail.pdf>.

[3] Landis + Gyr: *ZMD310AT/CT - Technical data*. Online: <www.landisgyr.eu/files/pdf2/LandisGyr_ZXD300ATCT_TechData_EN1.pdf>.

[4] Landis + Gyr: *Communication unit CU-E20, E21, E22 - Technical data*. Online: <www.landisgyr.eu/files/pdf2/7102000320_en1.pdf>.

[5] Landis + Gyr: *Central station DGC300 - User manual*.

[6] HOMER WU, CHONG-YEN LEE, WUU-YEE CHEN, TSANG-YEAN LEE, TENG-SENG SHIH. Keyless Cryptology for Cipher Text Transmitted in Network. *WSEAS Transactions on Communications*. Issue 4, Volume 6, 2007, pp.636-643. ISSN 1109-2742.

[7] MOLLIN, Richard A. *An introduction to cryptography.* 2007. 413 p. ISBN 1-58488-618-8.

[8] KOUTNY, M. *Design of secure communications for measuring equipment networks*. In 8-th International Conference - Research in Telecommunication Technology RTT - 2007. 1. Zilina, Slovakia, ZILINSKÁ UNIVERZITA. 2007. p. 1 - 4. ISBN 978-80-8070-735-4.

[9] WACKER H. D., BOERCSOEK J., HILLMER, H. Redundant Data Transmission and Nonlinear Codes. *WSEAS Transactions on Communications*. Issue 6, Volume 7, 2008, pp.594-604. ISSN 1109-2742.

[10] Rabbit Semiconductor Inc.: *Dynamic C : User's Manual*, 2006. 356p.

[11] COMER, Douglas E. *Internetworking with TCP/IP Vol. I : Principles, Protocols, and Architecture (Hardcover)*. [s.l.] : [s.n.], 2006. 650s. ISBN 0-13-187671-6.

[12] DAEMEN, Joan, RIJMEN , Vincent.: *The Design of Rijndael : AES - The Advanced Encryption Standard (Information Security and Cryptography) (Hardcover)*. [s.l.] : [s.n.], 1997. 238 s. ISBN 3-540-42580-2.

[13] MCCLURE, S., SCAMBRAY, J., KURTZ, G.: *Hacking Exposed*, McGraw-Hill Osborne Media, 2005, ISBN 9780072260816

[14] ERICSSON, J.: *Hacking: The Art of Exploitation*, No Starch Press, 2003, ISBN 1- 59327-007-0

[15] TUGKAN TUGLULAR. Location Awere Self-Adapting Firewall Policiies. *WSEAS Transactions on Communications*. Issue 6, Volume 7, 2008, pp.563-573. ISSN 1109-2742.

[16] YANG LIU, KAIKUN DONG, LAN DONG, BIN LI. Research of the ARP Spoofing Principle and a Defensive Algorithm. *WSEAS Transactions on Computers*. Issue 5, Volume 7, 2008, pp. 516-520. ISSN 1109-2742.