Software tools communication process models for Modbus TCP/RTU for diagnostics using machine learning approaches

Yuri Sosnovskiy¹, Maria Lapina², Vitalii Lapin³, Massimo Mecella²,⁴

¹ Crimean Federal University named after V.I. Vernadsky, Simferopol, Russia
² North-Caucasus Federal University, Stavropol, Pushkin str., 1, 355017, Russia
³ Stavropol Regional Clinical Consulting and Diagnostic Center, Stavropol, Lenina str, 304, 355017, Russia
⁴ SAPIENZA Università di Roma, via Ariosto 25, I-00185 Roma, Italy

E-mail sosnovskiy.yv@cfuv.ru

Abstract. The work proposes a software solution to the problem of generating data sets that simulate information control processes in microprocessor control systems and automated process control system (APCS) at the lower and middle levels using the Modbus TCP/RTU protocol. Such datasets (dumps) with controlled characteristics, records of computer attacks and information and technical impact (ITI) of various types are advisable to use for training spacecraft recognition systems based on artificial intelligence and machine learning. Also, due to the possibility of introducing records of any types of attacks, it is possible to test intrusion detection systems, intelligent sensors of the spacecraft for their efficiency. The analysis of the existing software, the possibility of its combination to solve the problem is given. The structure of a data transmission system (DTS) fragment of a real VCE ACS and its representation as a set of device identifiers, addresses, registers and data types. The model of communication processes Modbus TCP/RTU was built, the results of modeling are presented. The features and limitations of the software used are discussed, as well as the prospects for improving the model.

Keywords: Software tools, Machine learning, Communication process models, Generating data sets

1. Introduction

Industrial network data transfer protocols are an integral part of systems that demonstrate a positive trend in the number of implementations - automated process control system (APCS), microprocessor control system (MCS), as well as intelligent devices included in the popular category of Internet of Things (IoT), Smart City, etc. [1]

It is worth noting that the set of options for the used data transfer protocols in a wide area of “smart” devices is quite large and, by no means, is not limited to the options for Modbus implementations. Even more “orthodox” automation systems allow the use of a variety of their own data transfer protocols. At the same time, most of these protocols have well-defined structures of packets or frames and sets of commands, which to some extent makes them similar to the category under consideration.
In connection with the massive use of such "intelligent" devices, the problem of monitoring traffic for potential errors in the operation of devices and various ITVs arises [2, 3]. In the case of using standard TCP/IP protocols, a wide range of monitoring, control, intrusion prevention, etc. programs are available [4]. However, when using the protocols of the lower (field) level of the automated process control system, standard solutions have low efficiency or are not applicable at all.

In this regard, the problem of developing intelligent sensors for the protocols of the lower level of APCS, MCS arises sharply [5, 6]. Accordingly, traffic sets are required, the so-called. "Dumps" that have predefined properties that allow, taking into account a priori information, to configure, test or evaluate the effectiveness of the corresponding solution. In addition, when using approaches based on artificial intelligence (AI) or machine learning (ML), large sets of real or model data are required to train the sensor system and test its effectiveness [7, 8, 9].

2. Setting goals, objectives

The aim of the work is to develop a solution that allows generating sets of traffic protocols Modbus RTU (serial interface, most often based on RS422 / 428 or 232) and Modbus TCP (based on Ethernet) with specified properties, such as the number of active devices, address ranges, sets of commands used and transmitted values, errors in data packets.

Additionally, it is required to generate a “man-in-the-middle” type ITV and record data exchange processes in a dump.

To achieve a given goal, it is required to solve a number of logically and functionally related tasks:

1. Creation of the structure of the simulated network of devices, selection of the master, endowing the slaves with the specified properties (number of inputs, ranges of values, options for changing them, number of outputs, options for controlling outputs);
2. Development of requirements for traffic dumps and, accordingly, for software components;
3. Analysis of existing software solutions;
4. Implementation of clauses 1 and 2 in the software environment of existing or own components;

In addition to purely practical and scientific tasks, the implementation of the corresponding solution will raise the educational process in the field of automation to a higher level by providing students with a tool for modeling and visualizing processes in the local self-government network and the possibility of studying the influence of certain types of ITV on the communication subsystem.

3. Selecting a fragment of the simulated system

As a real prototype for creating a model of Modbus TCP/RTU communication processes, domestic equipment for automation was chosen, manufactured by OWEN, consisting of:

- measuring-regulator OWEN TRM200. Contains two universal inputs for connecting a wide range of sensors, built-in RS-485 interface (OWEN protocol, Modbus ASCII / RTU) - 2 units;
- frequency converter of general industrial application PChV101-K18-A. Contains a built-in logic controller and a built-in RS-485 interface (OWEN protocol, Modbus ASCII / RTU) - 1 unit;
- programmable logic relay PR110 and interface module PR-MI485 for organizing work under the Modbus ASCII / RTU protocol - 1 set;
- touch panel controller OWEN SPK107, which also includes the Modbus ASCII / RTU protocol - 1 unit;

On the basis of the specified equipment, the simplest stand was assembled, in which, by means of Modbus, data are read from the meter-regulators, the frequency converter and the programmable relay are controlled. The structure of the simulated network fragment (slave devices) is revealed by the list of protocol identifiers. Table 1 provides a description of the address space and sets of transmitted parameters of the Modbus RTU data transmission network of the lower (field) level - the level of microprocessor control systems (MCS).

| Table 1. Description of the logical structure of the DTS MCS |
|-------------------------------------------------------------|
| **Device, id** | **Registers** | **Functions** | **The values** |
|-------------------------------------------------------------|---------------------------------|----------------|----------------|
|-------------------------------------------------------------|---------------------------------|----------------|----------------|

2
The restrictions transmitted to the communication subsystem of the MCS using physical devices. In addition, it also makes it possible to simulate various protocol and ITV errors with predetermined and known errors and interventions. The address space of devices, registers and variables described in Table 2 is the basis for modeling. It reproduces some variants of the address spaces of real devices and can be verified in a real model of the data transition system (DTS) MCS with predetermined and known errors and interventions.

Table 2. Description of the simplified logical structure of the DTS MCS model, taking into account the restrictions

| Device, id | Registers | Function          | The values                                |
|-----------|-----------|-------------------|-------------------------------------------|
| TRM1, 1   | ch1 0x0000, ch2 0x0001 | 0x03 | signed int16, static, 200 | signed int16, sinusoidal, [140,260], T=2 hours |
| TRM2, 2   | ch1 0x0000, ch2 0x0001 | 0x03 | signed int16, sinusoidal, [100,300], T=4 hours | signed int16, accidental, [180,190] |
| PChV1, 3  | CTW 0xC34F, REF 0xC359, STW 0xC417, MAV 0xC421, FREQ 0x3F01 | 0x10 | word (register command word) | word (job register RS-485) |
|           |           | 0x10 | word (job register RS-485, frequency) | word, static, 49392 (status word) |
|           | POW 0x3EE3, CURR 0x3F0B | 0x03 | word, static | word, static |
|           | var1, 0x200, var2, 0x202, var3, 0x204, var4, 0x206, var5, 0x208, var6, 0x20A, var7, 0x20C, var8, 0x20E | 0x03 | signed int16, sawtooth, [0,8640], T=100ч | signed int16, sinusoidal, [10,400], T=1 час |
|           |           | 0x03 | signed int16, sinusoidal, [200,500] | signed int16, static, 120, fluctuation 5% |
|           |           | 0x03 | signed int16, static, 255 | signed int16, static, 111, fluctuation 10% |
|           |           | 0x06 | signed int16 | signed int16 |
|           |           | 0x06 | signed int16 | signed int16 |

The address space of devices, registers and variables described in Table 2 is the basis for modeling. It reproduces some variants of the address spaces of real devices and can be verified in a real model using physical devices. In addition, it also makes it possible to simulate various protocol and ITV errors for the communication subsystem of the MCS. As a result, it allows to obtain test sequences of data transmitted to the data transition system (DTS) MCS with predetermined and known errors and interventions.
The most difficult, from the point of view of communication interaction, is the PChV device, which has more than 40 registers available for reading and writing through the communication interface. Key registers and their relationship in the model are shown in Table 3.

| Register, modbus address | Description and set of register values (discrete or interrelation of output and input) |
|--------------------------|--------------------------------------------------------------------------------------|
| CTW, 0xC34F              | Control word register. Command set: stop 2C 63, start straight 2E 63, start reverse: 2E 62. |
| REF, 0xC359               | RS-485 reference register - setpoint. Value from 0 to 1000 (operational from 0 to 400). |
| FREQ, 0x3F01             | If the commands start forward / reverse and (REF> 2), then FREQ = REF-2; otherwise FREQ = 0. |
| STW, 0xC417              | Status output word. Enabled / disabled is determined by command from CTW. Set of values - based on REF setpoint value: enabled, no errors: 0F 03 (FREQ <= 100); on, overheating: 8F 03; on, current limit: 4F 03; on, overheating, current limit: CF 03; on, out of frequency range: 0B 03 (FREQ > 102); on, out of frequency range, overheating: 8B 03; on, out of frequency range, current limit: 4B 03 (FREQ > 200) AND (CURR> 5); on, out of frequency range, overheating, current limit: CF 03; on, out of frequency range, overheating, current limit: CB 03 (FREQ > 200) AND (CURR> 6); off, no errors: 07 03; off, overheating: 87 03 (CTW = 11363) AND (REF> 400). |
| CURR, 0x3F0B             | If (FREQ < 76) then CURR = (FREQ * 0.1) + Random (0..5 * 0.2); If (FREQ > 75) AND (FREQ < 201) then CURR = (FREQ * 0.06) + Random (0..5) * 0.2); If (FREQ > 400) then CURR = 0 (residual current circuit simulation) |
| POW, 0x3EE3              | POW= CURR * 380 |
are introduced into the model, three of which are available for reading with the corresponding range of values indicated in Table 2, and two variables available for records. However, the values in them do not affect the values in the var1-var3 variables.

4. Overview of existing software components
Specialists in the field of automation use various software solutions to configure and control the operation of the lower (field) level of the APCS. A practically relevant overview of the most famous of them as of 2016 is given in the material "Software for testing and setting up devices and networks based on MODBUS" at the link [10]. However, below is a number of software tools, with which it is advisable to supplement the material [10].

To simulate the master in the Modbus network, with some restrictions, primarily concerning the processes of writing to the registers of the slave device, you can use the OPC-server program (manufactured by NPF Krug). Terms of use - "freeware" limited to 30 tags. It provides a convenient interface for configuring and monitoring the status of tags, provides statistics on the use of the interface, and allows operation both in the case of a serial interface and via the TCP protocol.

The Modbus TCP/RTU emulator from ardSoft is extremely convenient for simulating the communication component of the slave devices. Simulates groups of slave devices, makes it possible to operate with an unlimited number of devices and tags. Provides access to groups of slave devices via a built-in proxy server via a serial port or Ethernet (including using localhost). Has a convenient organization of slave groups and tags. A convenient control tool is the built-in data stream sniffer.

Weprex is a program of a domestic author that simulates a Modbus master device within the RTU and TCP interfaces. Has the ability to implement all basic Modbus functions, graphing of the recorded values is available. It can read both one register and a group in one request.

Termite (Modbus RTU terminal) - fully functional terminal, free. Transfer function, templates for all Modbus functions. There is a possibility of automatic scanning according to a predefined template. Supports only serial ports.

The "EAT Console" program is designed to control remote devices from a personal computer via the RS-232 / RS-485 serial interface (USB – RS-485 converter) based on the Modbus RTU protocol. Requires Eclipse.

Multiway (Omron) - Modbus TCP, RTU terminal, Modbus functions. Auto poll according to a pre-written template. Shows response delay. Has a built-in protocol sniffer, but it is only available for serial connection.

Conclusions on the use of existing software. In general, for the task at hand – obtaining a dump of Modbus TCP/RTU protocol traffic (namely, part of Modbus traffic, excluding the TCP protocol overhead), existing software is sufficient. In particular, the program emulator Modbus devices version 2.34.63.950 has a free unlimited license and allows you to create virtual servers for connecting slave devices, select the type of protocol (Modbus RTU or TCP) and specify the port through which you can use third-party software to interact with slave devices. For each slave device, the creation of so-called "Tags" – in fact, a description of the Modbus device registers, their addresses, valid Modbus function codes, and values. It is worth noting the possibility of programatically generating values in such registers both based on typical laws (random, harmonic, sawtooth), and by writing scripts that can calculate the value to the current register based on other values, including from other virtual registers slave device.

It is also advisable to use software of domestic authors – Weprex as a model of the master device. It allows you to connect both serial and TCP to real hardware and, in this case, to a model slave server. Provides a basic set of Modbus functions for reading and writing registers. At the same time, the number of readable registers within one configuration set does not exceed 12. Thus, for large models it is required to create several configuration sets and manually control the reading of registers.

5. A Modbus TCP/RTU Communication Process Model for Diagnostics Using Machine Learning Approaches
To train a model with the subsequent forecasting of its characteristics, data are needed; they can be obtained by conducting a simulation experiment with the registration (recording) of dumps of communication activity. These dumps will be the basis for building a training dataframe.

To build the model, we used the “ardSoft” program-emulator of Modbus devices. The registers of the emulated Modbus network devices are interrogated using the Werpex software, the transmitted data via the TCP interface is recorded by the sniffer built into the ardSoft software. The structure of the network model of communication processes is shown in Figure 1.

The devices included in the model are described above in the "Selecting a Fragment of the Simulated System" section. The formation of a training sample is performed by setting a data change model, the basis for which is Table 2. Table 3 shows data describing the types, values of registers and their relationship for modeling a Modbus device with an integrated PLC - a vector frequency converter (PChV1). Using the built-in scripting language in the Modbus TCP/RTU emulator program, it is possible to describe such behavioral features. An example of a script for describing the Modbus STW parameter is shown in Figure 2.

```
begin
    if ((CTW = 11874) OR (CTW = 11875)) then
        begin
            self := 3843;
            if (FREQ > 100) then self := 2819;
            if (FREQ > 200) AND (CURR > 5) then self := 19203;
            if (FREQ > 400) then self := 34563;
        end;
    if ((CTW = 11363) AND (REF > 400)) then
        begin
            self := 34563;
        end;
    if ((CTW = 11363) AND (REF < 400)) then
        begin
            self := 1795;
        end;
    end;
```

Figure 2. The script of the Modbus TCP / RTU emulator program

Screenshots of an example of a given configuration, polling and data fragments are shown in Figures 3 and 4.
Figure 3. Setting the configuration in the emulator of Modbus devices and an example of data transfer.

Figure 4. Polling registers.

Figure 4 shows the interface of the Weprex program, which simulates the master, which polls the Modbus registers and performs read operations (code 0x03) and write operations on registers (0x06). In this case, the read functions are set at a specified interval, and for writing, manual activation of the register write mode is required.

6. Conclusion
The result of the work is a set of software tools and configuration files for slaves and masters of the Modbus TCP/RTU communication process model. The created model takes into account the specifics of the data transmission system of the simulated equipment. It can be used to generate training and test sequences for diagnosing information security problems using approaches based on machine learning and artificial intelligence.

The model is verified by comparing data sets obtained from the developed model and obtained by registering data from a real system with the specified components. In this case, the service part of the TCP packets was discarded, only Modbus frames were left. The obtained data, with some simplifications, correspond to the data circulating in a real MCS with the specified equipment.
The presented model of Modbus TCP/RTU communication processes for diagnostics using approaches based on machine learning allows:

- supplement the model with new devices, describe their registers and ranges of variable values (with some peculiarities due to Weprex restrictions, described above);
- simulate complex Modbus devices and registers, calculate values in them taking into account the values of other registers of the virtual device;
- generate training datasets for AI systems with pre-introduced distortions, variants of various computer attacks and facilitate the process of marking up such data for later use;
- generate test data sets with specified deviations and spacecraft parameters.

References

[1] "Smart" environments, "smart" systems, "smart" production: a series of reports (a series of green books) within the project "Industrial and technological foresight of the Russian Federation" 2012 / Team of authors; Center for Strategic Research North-West Foundation SPb Issue 4 62 p

[2] Tsapko G P, Verigo A A, Katashev A S 2016 Analysis of security risks of automated control systems for technological processes // Internet-journal "Science" Volume 8 No.5 http://naukovedenie.ru/PDF/55TVN516.pdf (free access). Title from the screen. Yaz. rus., eng.

[3] Khraisat A, Gondal I, Vamplew P et al. 2019 Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecur 2, 20 https://doi.org/10.1186/s42400-019-0038-7

[4] Tiwari Mohit, Kumar Raj, Bharti Akash, Kishan Jai. 2017 INTRUSION DETECTION SYSTEM. International Journal of Technical Research and Applications 5 pp 2320-8163

[5] Makeenko N I, Novozhilov I O, Korabeynikov D N 2017 Intrusion detection system // Modern scientific research and innovations No 5 [Electronic resource]. URL: http://web.snauka.ru/issues/2017/05/82889 (date accessed: 09/14/2020)

[6] Straub Jeremy 2017 Testing automation for an intrusion detection system. 1-6. https://doi.org/10.1109/AUTEST.2017.8080473

[7] Kang M J, Kang J W 2016 Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security PLOS ONE 11(6): e0155781. https://doi.org/10.1371/journal.pone.0155781

[8] Tsaia C, Hsub Y, Line C, Lin W 2009 Intrusion detection by machine learning: A review. Expert Systems with Applications

[9] Zhuk A, Orel D 2019 Improved Method for Estimating Noise Immunity of Global Navigation Satellite Systems Proceedings of the 2nd international conference on modelling, simulation and applied mathematics (MSAM2017) https://doi.org/10.2991/ITIDS-19.2019.54

[10] Software for testing and adjusting devices and networks based on MODBUS [Electronic resource]. URL: https://habr.com/ru/post/281430/ (date accessed: 09/10/2020)