On the practical implementation of Russian protocols for low-resource cryptographic modules

Alexey Yu. Nesterenko & Aleksandr M. Semenov

Journal of Computer Virology and Hacking Techniques

e-ISSN 2263-8733

J Comput Virol Hack Tech
DOI 10.1007/s11416-020-00362-y

Springer

Journal of Computer Virology and Hacking Techniques
https://doi.org/10.1007/s11416-020-00362-y

ORIGINAL PAPER

On the practical implementation of Russian protocols for low-resource cryptographic modules

Alexey Yu. Nesterenko[1] · **Aleksandr M. Semenov**[1]

Received: 20 October 2019 / Accepted: 14 July 2020

Abstract
In 2018, the authors of this article developed a cryptographic mechanism, which was adopted in 2019 as a recommendations on standardization R 1323565.1.028-2019 "Cryptographic mechanisms for secure interaction of control and measuring Devices" by Technical Committee "Cryptographic Information Protection". These recommendations contain a description of the family of cryptographic protocols designed to produce key information, as well as for the exchange of encrypted information with integrity protection. The article describes the cryptographic mechanisms used in the protocol, their difference from the existing solutions, peculiarities of the key system and methods of authentication of participants in secure interaction. The results of the program implementation developed by the authors will be presented.

Keywords Cryptographic protocol · Secure interaction · Implementation of cryptographic primitives

1 Introduction

Modern low-resource devices are a large class of technical devices built on the basis of microcontrollers with different architecture. Combining such devices into large heterogeneous networks, existing differences in technical characteristics lead to the need to use channels for communication, which differ both in their properties and in the environment of information distribution. Cryptographic mechanisms used to ensure secure interaction of low-resource devices should not depend on the physical level of information transfer and, in particular, on the presence or absence of a guaranteed message delivery property.

Further we will discuss a family of secure cryptographic protocols which were standardized as recommendations on standartization R 1323565.1.028-2019 [1]. Since these protocols have more general application area we use an unformal title "Secure protocols for the Internet of Things" or SP FIOT. It should be noted that some of the important cryptographic features of SP FIOT were not described earlier. The main goal of this paper is to fill the gap.

The principles of SP FIOT are based on a two-level model of an organization of protected interaction:

- A transport protocol is implemented at an lower level to send/receive messages containing both open and encrypted parts, as well as message authentication code (MAC), allowing to ensure the integrity of transmitted data. Thus the open information is used both for formation an initialization vector, and for protection against replay attacks at an reception of messages. The transport-level protocol uses cryptographic mechanisms, but does not control aspects of generating the key information;
- At a higher level, there is a session protocol, the main task of which is to implement mechanisms for authentication of protocol participants, as well as mechanisms for generation, agreement and modification of key information. Session protocol considers the underlying transport protocol as a cryptographic tunnel through which the information coming from the application level is transmitted.

The reported study was funded by RFBR, Project Number 19-37-90155.

✉ Alexey Yu. Nesterenko
nesterenko_a_y@mail.ru

Aleksandr M. Semenov
semenov_am@tc26.ru

[1] Department of Computer Security, Higher School of Economics, Tikhonov Moscow Institute of Electronics and Mathematics (MIEM HSE), Tallinskaya Street, 34, Moscow, Russia 123458

Published online: 27 July 2020

The given model allows to separate aspects of participants authentication and generating of the key information, from mechanisms of transmission of the encrypted information through a communication channel. The same mechanism of key information generation can be implemented both for channels with and without guaranteed delivery of messages. Examples of such a model may be:

- Transport Layer Security (TLS) protocol implemented over TCP or Datagram Transport Layer Security (DTLS) protocol implemented over UDP;
- The pair of IKEv2 & ESP protocols as a part of IPSec infrastructure; these protocols are implemented over IP;
- The pair of protocols IEEE 802.1AE (MACSec) as a data link layer with IEEE 802.1X-2010 used as key management protocol,

As we can see, similar cryptographic properties are provided at each level differently. Recommendations [1] can be considered as one possible universal mechanism. The use of SP FIOT to secure the transmission of information over the Internet has shown good results. The application of this protocol over the physical layer of the data channel which is used by IoT devices ensures the achievement of the same level of security.

2 Cryptographic primitives used in SP FIOT

SP FIOT uses the following Russian standardized cryptographic solutions:

- Streebog hash function regulated by GOST R 34.11-2012, see [2], [3], with 512-bit output;
- block cyphers with block lengths of 64 and 128 bits regulated by GOST R 34.12-2015 [4];
- the confidentiality and data integrity can be provided by the conventional "encryption-then-mac" principle regulated by GOST R 34.13-2015 [5];
- alternatively, the newest authenticated encryption mode (Authenticated Encryption with Associated Data - AEAD) regulated by R 1323565.1.026-2019 [6] can be used;
- for authentication of protocol participants the cryptographic mechanism is used, allowing to link unique identifiers and keys of authentication of devices;
- by default preshared secret keys may be used for mutual authentication;
- PKI infrastructure can also be used for mutual or one-way authentication of protocol participants; in this case, the support of the infrastructure regulated by Russian Federal law [7] is provided; digital signature generation and verification algorithms are regulated by GOST R 34.10-2012 [8];
- the key agreement protocol is based on the "Echinacea" scheme regulated by R 1323565.1.004-2017 [9];
- HMAC-based key derivation function, regulated by the recommendations R 50.1.113-2016 [10];
- the algorithms that are regulated by R 1323565.1.017-2018 [11], [12] are chosen for generating derivative encryption and MAC keys.

It is easy to see that these algorithms can be easily replaced by foreign analogues.

3 Connection establishment, authentication and identification mechanisms used in SP FIOT

On the session layer SP FIOT can be presented as a sequential work of two subprotocols:

- Key Generation Protocol (KGP) — key agreement and participant authentication protocol;
- Application Data Transmission Protocol (ADTP) – protocol for application data transmission.

KGP describes the client-server interaction between protocol participants. The client and the server must have their identifiers ID_c and ID_s respectively. Identifiers can be defined by one of the following methods:

- are assigned during the production phase of the device or installed at the moment of initialization of the device and change/installation of the ePSK (external pre-shared key) authentication key, which is common to the client and server;
- by public key certificate (the identifier is the value of the owner's field of the public key certificate in the BER coding).

A key-agreement protocol allows one-way or mutual authentication of participants. In the case of one-way authentication, the client always performs server authentication. Authentication can be implemented using the following cryptographic mechanisms:

- digital signature verification – this mechanism allows both one-sided and mutual authentication;
- use of PSK (pre-shared key) for authentication – this mechanism allows to implement only mutual authentication on the fact of confirmation of knowledge of this key by the protocol participant;

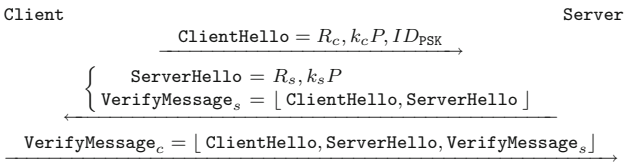

Fig. 1 Scheme of authentication using pre-shared key. Session layer

Fig. 2 Scheme of authentication using pre-shared key. Transport layer

The communication process begins with the connection establishment and the execution of the key generation protocol. We consider the connection establishment process on the example of the PSK-based authentication scheme (Fig. 1).

The protocol is implemented in a group of elliptic curve points. On Fig. 1 we define nonces (random numbers) R_c, R_s, random points of some fixed elliptic curve (k_cP, k_sP), an identifier of pre-shared key PSK (ID_{PSK}). `VerifyMessage` transfers the hash value of all messages generated and sent so far.

At the session level, messages are treated as serialized representations of data structures. From the point of view of session level messages are transferred in the course of performance of the key generation protocol in an unencrypted kind and without the integrity confirmation of transferred messages.

Encryption and integrity protection of transmitted messages is performed at the transport layer, see Fig. 2.

In this figure symbol $\lfloor M \rfloor_K$ – designation of MAC for message M using the K key, symbol $\{M\}_K$ – designation of encryption of message M using the key K, symbols $eSHTK$ and $eCHTK$ designate encryption keys, and symbols $iSHTK$ and $iCHTK$ designate MAC keys which are used for maintenance of confidentiality and integrity of the information transferred from a server to the client and, accordingly, from the client to a server.

Since we use two different keys for encryption and integrity, the AEAD algorithm used also must have two different keys. In our opinion, this is the only problem for the application of foreign cryptographic algorithms in SP FIOT.

The device authentication method used determines the protocol workflow. The KGP specification provides for other connection schemes, but the process of forming key information and keys remains unchanged. For example, establishing a connection using another authentication mechanism uses

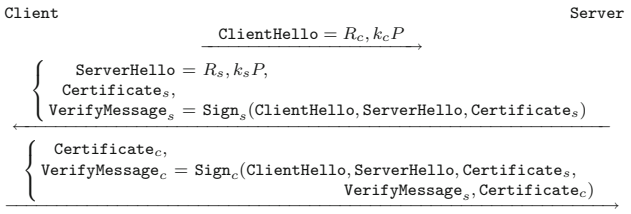

Fig. 3 Scheme of authentication using digital signature and PKI. Session layer

public key certificates, see Fig. 3. Detailed description of PKI infrastructure authentication schemes can be found in [1].

4 Session layer in SP FIOT: key system and management

On session layer SP FIOT consistently uses key information of three types:

- authentication keys (such as PSK or digital signature keys);
- secret information and derived keys used to encrypt traffic when establishing a connection. This information is different for different data transmission directions – client handshake transport secret (CHTS) and server handshake transport secret (SHTS). These secrets are binary vectors of a fixed length of 512 bits used to generation of derived encryption (eSHTK, eCHTK) and MAC (iSHTK, iCHTK) keys;
- secret information used to generate derivative keys during application data transfer. This information – client application traffic secret (CATS) and server application traffic secret (SATS) are also two binary vectors of a fixed length of 512 bits and different for different data transmission directions.

The use of SHTS and CTHS secrets allows one to hide the information transmitted during the execution of KGP and, in particular, the client's identifier. The KGP execution results are CATS and SATS secrets being transmitted to ADTP. The latter is responsible for the periodic transformation of CATS and SATS.

During the KGP the client and the server form sequences of octets $H_1, \ldots, H_5$, which are concatenations of messages and extensions that are successively transmitted. The diagram of the process of octet sequence generation $H_1, \ldots, H_5$ is shown on Fig. 4 (Since the sending of extensions is optional [1], the sequences of the octets H_1 and H_2, as well as H_3 and H_4 may coincide).

Let $x(Q)$ be a common secret – x-coordinate of elliptic curve point $Q = k_ck_sP$. Further sequences of octets R_1 and

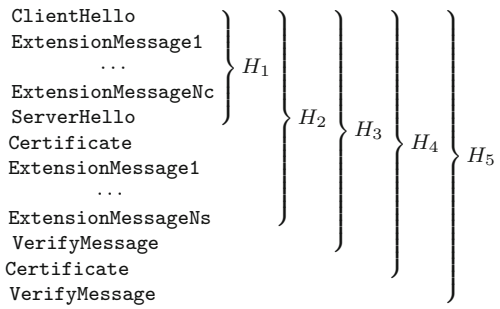

Fig. 4 Octet sequence $H_1, \ldots, H_5$ formation

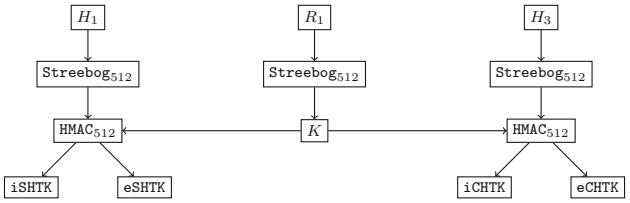

Fig. 5 Key generation scheme for the connection establishment protocol

R_2 are formed:

$$R_1 = x(Q)||PSK^*,$$
$$R_2 = ID_s||ID_c^*||PSK^*,$$

where symbol "*" means that this sequence of octets is optional and its inclusion in the sequence of octets R1 and/or R2 depends on the actions of the client and server when creating `ClientHello` and `ServerHello`.

The derived encryption and MAC keys represents as sequences of octets of 512 bits length, as a output of hash function Streebog, and is formed as follows (Fig. 5):

After the KGP is finished and the connection has been established, the client and server move on to the application data exchange using the ADTP. On this stage, ADTP accepts secrets $CATS$ and $SATS$ which generation procedure is shown in Fig. 6.

The secret value of T is also being transmitted to ADTP and used later for the transformation of CATS and SATS.

In the process of data transfer, ADTP changes the received secrets independently for the client and server. Modification of SATS and CATS can be presented as a two-level procedure. On the first level there is a rare and slow transformation, on the second level there is a fast and frequent transformation.

In [13] we have shown that the number of slow CATS and SATS transformations should not exceed 2^8. At the same time, the number of fast transformations shouldn't exceed 2^{16}. Since the pair of encryption and message authentication keys are used to encrypt no more than 2^{16} data frames, it allows one to ultimately transfer terabytes of information without interrupting the connection and using KGP for new secrets creation. More specially, the maximal amount of transferred data is

$$1500 * 2^{40} \text{ bytes} > 2^{10} \text{ TB} = 1 \text{ petabyte},$$

where 1500 is an average size of the frame. This means that for devices with small amounts of transmitted information, the KGP can be executed only once - when the device is initialized.

Total amount of secret information used to transfer application data in both directions is 2.5 Kb. This value defines the minimum memory size used by a low-resource module and isn't critical for most microprocessors.

The slow transformation of secrets CATS and SATS is performed by the following rule

$$\begin{aligned} CATS_1 &= CATS, \\ SATS_1 &= SATS, \\ CATS_{n+1} &= HMAC_{512}(T, CATS_n||n+1), \\ SATS_{n+1} &= HMAC_{512}(T, SATS_n||n+1), \end{aligned}$$

where T is the shared secret key for the client and the server and defined when the KGP is executed, see fig. 6.

The fast transformation is a transformation of derived keys. Derived keys are generated from secrets $CATS_n$ and $SATS_n$ for each counter value n and are used directly for encryption and integrity control of frames transmitted via communication channels.

The algorithm of client's generation of derivative keys – encryption keys $eCFK_{n,m}$ and MAC keys $iCFK_{n,m}$ is schematically shown in the Fig. 7.

The Ctr and J values depend on the selected encryption algorithm – for "Magma" algorithm $J = 2$ and $Ctr = FFFFFFFF00000000_{16}$, for "Kuznechik" algorithm $J = 4$ and

$$Ctr = FFFFFFFFFFFFFFFF0000000000000000_{16}.$$

The Ser function encodes an integer value as a byte sequence.

The generation scheme for the server's derivative keys is similar to the one shown in the Fig. 7.

To generate integrity keys $iCFK_{n,m}$ we use simple one block encryption and applying this to encryption of sequence of unique values within fixed secret $CATS_n$ value and secret key CK_n.

To generate encryption keys $eCFK_{n,m}$ we also use simple one block encryption but applying this to encryption of fixed constant with a previous encryption key, i.e.

$$\begin{aligned} \text{eCFK}_{n,m} &= ACPKM(\text{eCFK}_{n,m-1}) = \\ &= \text{E}(\text{eCFK}_{n,m-1}, D_1)||\cdots||\text{E}(\text{eCFK}_{n,m-1}, D_J), \end{aligned}$$

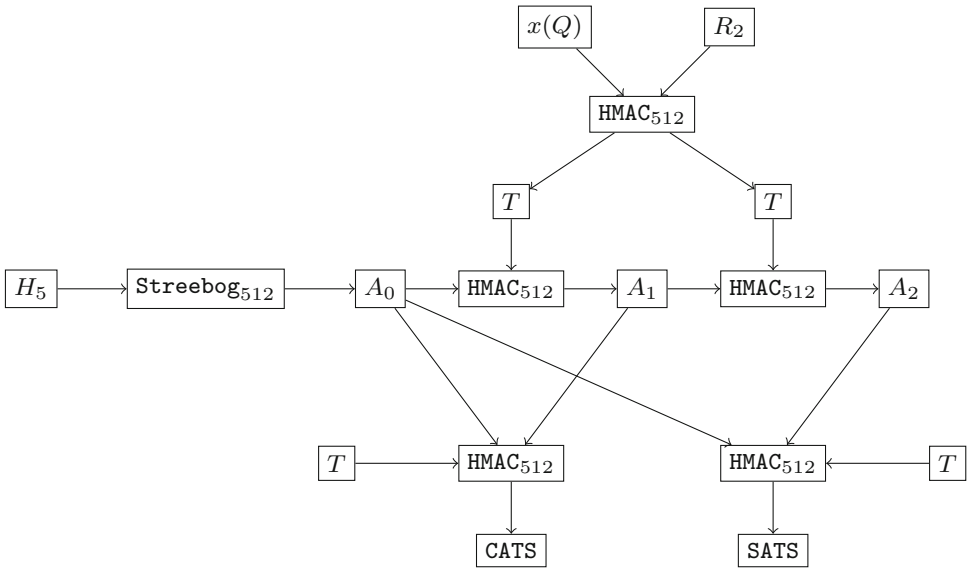

Fig. 6 Key information generation scheme for the application data transmission protocol

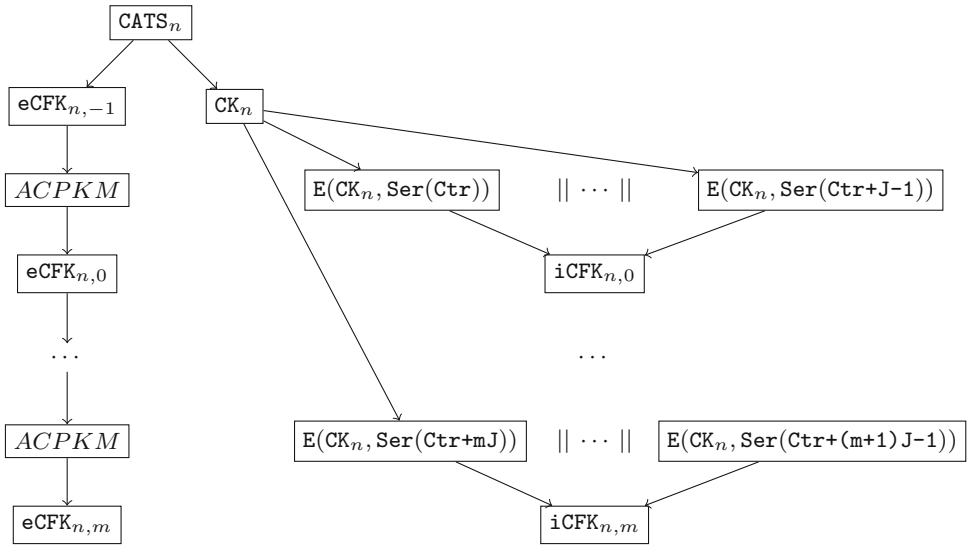

Fig. 7 Key generation scheme for the application data transmission protocol for the client side

where E is a used block cypher and $D_1||\cdots||D_J$ – fixed constant

$$D = (\mathtt{0x80}||\mathtt{0x81}||\mathtt{0x82}||\mathtt{0x83}||\mathtt{0x84}||\mathtt{0x85}|| \mathtt{0x86}||\mathtt{0x87}||\ \mathtt{0x88}||\mathtt{0x89}||\mathtt{0x8A}||\mathtt{0x8B}|| \mathtt{0x8C}||\mathtt{0x8D}||\mathtt{0x8E}||\mathtt{0x8F}||\mathtt{0x90}||\mathtt{0x91}|| \mathtt{0x92}||\mathtt{0x93}||\mathtt{0x94}||\mathtt{0x95}||\mathtt{0x96}||\mathtt{0x97}|| \mathtt{0x98}||\mathtt{0x99}||\mathtt{0x9A}||\mathtt{0x9B}||\mathtt{0x9C}||\mathtt{0x9D}|| \mathtt{0x9E}||\mathtt{0x9F}),$$

The ACPKM has been standardized in R 1323565.1.017-2018 [11] and RFC 8645 [12].

The considered KGP have a number of distinctive features:

- during the generation of common secret Q in the KGP only groups of points of the elliptic curve are used, which providing a higher level of protection compared to the usage of multiplicative groups of the prime fields;
- using 0-RTT (zero round trip time resumption) mode reduces of security properties ([14], appendix E 5.), this mode is not supported by SP FIOT protocol, unlike TLS 1.3 where there the 0-RTT mode is available;
- application data transfer is only allowed after the connection process is completed, unlike TLS 1.3, which allows sending application data until the connection is established;
- the possibility of deferred authentication not supported by SP FIOT, in contrast to TLS 1.3, where such a possibility exists [15];
- the formation of common key information is always done via the Diffy-Hellman protocol. PSK only mode is not

available. This provides ephemerality of the encryption and integrity protection keys;
- regardless of the authentication scheme the PFS[1] property is always provided;
- during the generation of encryption and integrity protection keys of application data the binding to all handshake messages is performed. This mechanism provides additional protection against impersonation attacks;
- use of identifiers in key generation makes it possible to link them to specific participants. This is one of the requirements of [17];
- the SP FIOT allows one to control the amount of information encrypted on one key according to the requirements of each particular system. The higher the security level, the less information is encrypted on one key;
- PSK-based authentication is useful when used with devices for which unique key information can be placed in the device during production;
- PKI-based authentication is useful for devices that are designed to provide access to secure communications to individuals (such as cash registers).

Cryptographic survey of these features was carried out in [13]. We hope that the proposed SP FIOT session layer's protocols will be cryptographically stable and in demand for a long time.

5 Transport protocol and Implementation Issues

In SP FIOT the transport layer protocol (TLP) is responsible for the transmission of real data. The frame format, which is used to incapsulation and transmission of application data, is following.

The frame has a variable-length header, consists of frame length, unique frame number and optional data. The unique frame number is used for initial vector generation and defence against replay attacks. In most cases, the optional data are omitted.

The variable-length header may contain participant IDs, which allows to separate the information flows processed by one device. This is analogous to the addressing mechanism on the Internet and allows easy integration of new devices in IoT networks.

The application data stored in the frame's body as a `message`. When the confidentiality of stored data is provided, the body of the frame is encrypted. The length of the stored message may be hidden by random padding. This feature is needless when TLP is used only for data authentication.

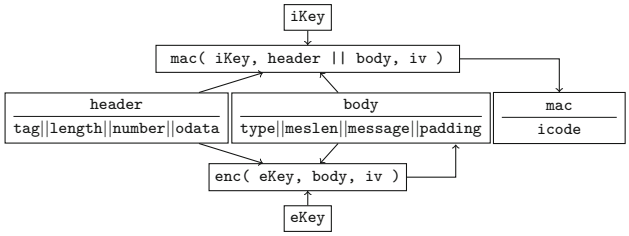

Fig. 8 TLP frame format

Also, frame has an authentication code which is used to protect the integrity of the whole frame, including body and header. As mentioned above, the encryption key `eKey` and the authentication key `iKey` are different.

From a theoretical point of view, TLP can be implemented over any abstraction layer of OSI model [18]. For example, consider the data link layer. We can use optional data in the TLP header to place a standard Ethernet packet header. Since TLP provides encryption and message authentication, we easily get an analogous MACSec protocol [19]. Implementation of this approach requires its own hardware implementation of the network stack. We are planning to implement this approach in the future.

The first experience of real SP FIOT implementation was gained during the development of client/server applications for the virtual network of low-resource mobile modules. We implemented TLP over IP using the mechanism of raw sockets and a random protocol number not assigned by the Internet Assigned Numbers Authority (IANA), see [20].

Since our network also uses NAT, we had to implement TLP over UDP and TCP. All our implementations belong to the user's space and do not use the kernel's cryptographic API. The source codes can be found at [21].

Despite the compound description of the ADTP their implementation is not complicated. The key point is a realization of special procedure of gaining defence against replay attacks while accepting frames. This procedure is similar to ESP from IPSec infrastructure and based on "flying window" method of checking unique frame numbers (Fig. 8).

Figures. 9 and 10 show a comparison of transmission speeds of different data sizes (from 1 byte) with the alignment of the length of the entire sent packet to 1500 bytes for the ADTP and ESP protocols over TCP and UDP with "Magma" encryption algorithm.

According to the experiments, additional data transfer overheads are determined by the speed of the encryption algorithm used. Fragmentation and defragmentation processes can be omitted if the transmitted data is small. Transformation of secret information does not cause delays.

[1] Perfect forward secrecy (PFS) is a feature of specific key agreement protocols that gives assurances that session keys will not be compromised even if the private key of the server is compromised [16].

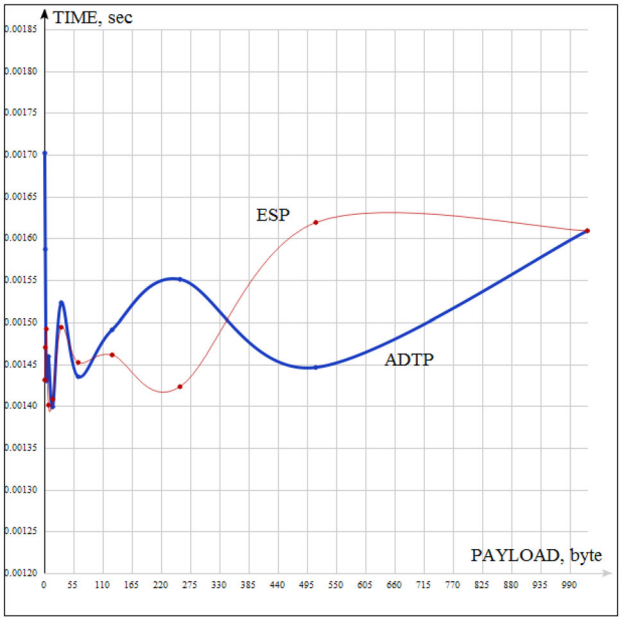

Fig. 9 ADTP and ESP protocols over TCP

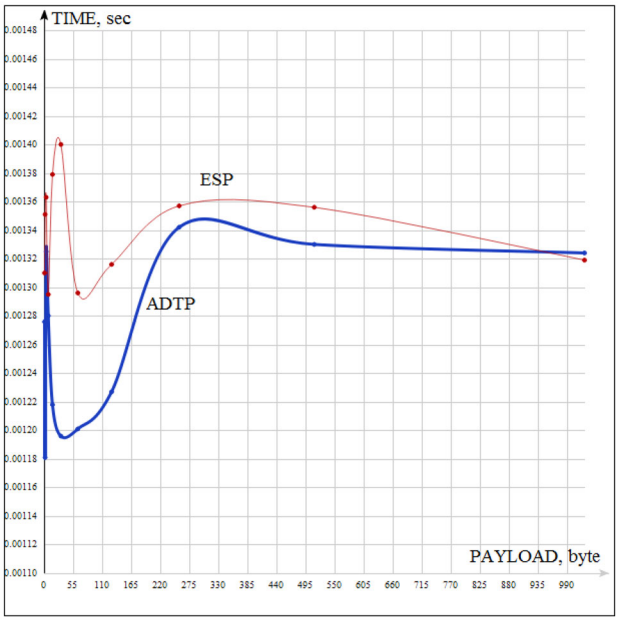

Fig. 10 ADTP and ESP protocols over UDP

6 Conclusion

The family of cryptographic mechanisms described in the article allows to implement a secure interaction of IoT-devices, as well as ordinary Internet devices at various levels of the network stack. The proposed key system permits to transfer data up to 1 pentabyte before restart of the KGP protocol. Used cryptographic algorithms allows to achieve the speed of data transfer the same as speed of IPSec connections.

References

1. R 1323565.1.028-2019: Information Technology. Cryptographic Data Security. Cryptographic Mechanisms for Secure Interaction of Control and Measuring Devices, Standardinform, Moscow (In Russian) (2018)
2. GOST R 34.11-2012: Information Technology. Cryptographic Data Security. Hash Function. Standardinform, Moscow (In Russian) (2012)
3. RFC 6986. GOST R 34.11-2012: Hash Function (2013)
4. GOST R 34.12-2015: Information Technology. Cryptographic Data Security. Block Ciphers, Standardinform, Moscow (In Russian) (2015)
5. GOST R 34.13-2015: Information Technology. Cryptographic Data Security. Modes of operation for block ciphers, Standardinform, Moscow (In Russian) (2015)
6. R 1323565.1.026-2019: Information Technology. Cryptographic Data Security. Block Encryption Modes of Operation that Implement Authenticated Encryption, Standardinform, Moscow (In Russian) (2019)
7. Russian Federal law from 06.04.2011: N 63 “About the digital signature” (In Russian)
8. GOST R 34.10-2012: Information Technology. Cryptographic Data Security. The Processes of Formation and Verification of Electronic Digital Signature, Standardinform, Moscow (In Russian) (2012)
9. R 1323565.1.004-2017: Information Technology. Cryptographic Data Security. Public Key Generation Schemes with Public Key Authentication, Standardinform, Moscow (In Russian) (2017)
10. R 50.1.113-2016: Information Technology. Cryptographic Data Security. Cryptographic Algorithms Associated with the Use of Electronic Digital Signature Algorithms and Hashing Functions, Standardinform, Moscow (In Russian) (2016)
11. R 1323565.1.017-2018: Information Technology. Cryptographic Data Security. Cryptographic Algorithms Associated with the Use of Block Encryption Algorithms, Standardinform, Moscow (In Russian) (2018)
12. RFC 8645: Re-keying Mechanisms for Symmetric Keys (2019)
13. Lebedev P.A., Nesterenko A.Y., Semenov A.M. Brief analysis of cryptographic mechanisms of protected interaction of control and measuring devices (2019). https://tc26.ru/standarts/kriptograficheskie-issledovaniya/-kratkiy-analiz-kriptograficheskikh-mekhanizmov-zashchishchennogo-vzaimodeystviya-kontrolnykh-i-izmeritelnykh-ustroystv.html. Retrived 02 Dec 2020
14. RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3 (2018)
15. Akhmetzyanova, L., Alekseev, E., Smyshlyaeva, E., Sokolov A.: Continuing to reflect on TLS 1.3 with external PSK (2019). https://eprint.iacr.org/2019/421.pdf. Retrived 02 Dec 2020
16. Manoilov, G., Radichkova B.: Elsevier’s Dictionary of Information Security (1 edn), Kindle Edition (2007)
17. R 1323565.1.012-2017: Information Technology. Cryptographic Protection of Information. Principles of Development and Modernization of Encryption (cryptographic) Devices of Information Protection, Standardinform, Moscow (In Russian) (2017)
18. ISO 7498-1: Information Technology. Open Systems Interconnection. Basic Reference Model. Part 1. The Basic Model

19. IEEE Standard for Local and metropolitan area networks. Media Access Control (MAC) Security. IEEE. (2018). https://doi.org/10.1109/IEEESTD.2018.8585421. ISBN 978-1-5044-5215-1. Retrieved 10 Dec 2019
20. Protocol Numbers. Internet Assigned Numbers Authority (IANA). http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml. Retrived 12 Dec 2019
21. Libakrypt: Software crypto module for user space. https://github.com/axelkenzo/libakrypt-0.x. Retrived 10 Dec 2019