Annales UMCS Informatica AI IX, 1 (2009) 59–75
DOI: 10.2478/v10065-009-0005-0

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

Optimization of TLS security protocol using the adaptable security model

Paweł Szałachowski[1], Bogdan Księżopolski[1], Zbigniew Kotulski[23]

[1] *Institute of Computer Science, Maria Curie Skłodowska University, pl. M. Curie-Skłodowskiej 1, 20–031 Lublin, Poland*
[2] *Institute of Fundamental Technological Research of PAS, Swietokrzyska 21, 00–049 Warsaw, Poland*
[3] *Institute of Telecommunications of WUT, Nowowiejska 15/19, 00–665 Warsaw, Poland*

Abstract

Security protocols used in today's communication are complex and it is very difficult to analyze and optimize them. Literature reports some results which optimize security protocols. In the case of devices with limited resources (mobile phones, PDA, sensors) the speed and efficiency of the process is crucial for their stable work. Security methods used during transporting the data between parties are crucial as for as efficiency is concerned. However, optimization cannot significantly reduce the security of the process. We must remember that in many fields (e.g. e-banking, e-court etc.) security level will always be the main factor. In this paper, we show how to optimize security protocols in terms of the security level. We present the visualization tool for the adaptable security model, which defines the protection level of the transmitted data. These elements help us analyze and optimize a cryptographic protocol. The presented optimization results are based on the TLS protocol. We describe this protocol by the adaptable model and we create different versions of the protocol. Finally, we discuss differences between them and their impact on the protection level.

1. Introduction

Nowadays, for secure realization of electronic processes we can use different security protocols. If the sites of the protocols are communicating by means of the high performance devices then the protocol can be used in the standard way defined by the authors. The problem is if the protocol is realized by the systems with limited resources (mobile phones, PDA, sensors). The capability of these devices is the barrier of secure communication with each other.

In the first stage, finding a compromise between security and efficiency often begins at hardware. R.Roman et al. in [**1**] present a survey for cryptographic primitives and implementations for extremely constrained hardware - sensor network nodes. In turn, B.Kayayurt and T.Tuglular in [**2**] use the TLS Protocol to create end-to-end security implementation for mobile devices. Unfortunately, sometimes the dedicated implementation, for a given hardware is not sufficient and we must improve protocol in a different layer. The authors of cryptographic protocols often allow us to select the features such as encryption algorithms or their modes of operation. These modifications affect performance. In many cases, modification of configuration is not enough and the only option in order to improve the properties is the modification of the construction protocol. A. Elgohary et al. in [**3**] introduce the enhancement for SSL/TLS, making it more efficient. The modification of the protocol configuration is well tested and indicates that this kind of the optimization is safe. The optimization which refers to modification of the protocol construction is very risky.

Another problem is the evaluation of the protocol optimization process. Security protocols are very complex and we need models with analysis tools to determine how our modifications influence the whole. Generally, security protocols are analyzed using formal models. These methods use high abstraction to represent the security protocol and are mainly used for protocol verification. Therefore, these models can detect only restricted class of attacks. The best known representative of this group of models is BAN Logic, introduced in [**7**]. A. Yasinsac and J. Childs in [**4**] analyze the TLS protocol by formal methods. Besides formal correctness analysis, estimation of the risk of the processes is very important. M.Gerber and R.Solms in [**13**], based on properties of assets, characterize the probability of incident occurrence. Another approach to estimate the risk of processes is presented by Z.Dwaikat and F.Parisi-Presicce in [**14**]. Their model specifies security services used in the protocol, and then determines the risk of an attack.

In this article, we use the model which is a representative of Quality of Protection (QoP) models [**4, 8, 9**]. QoP models allow calculating different versions of the protocol which protects the transmitted data on different security levels.

In the literature one can find only a couple of articles about QoP because this security topic is one of the latest approaches. S.Lindskog and E.Jonsson try to extend the security layer in a few Quality of Service (QoS) architectures [**9**]. Unfortunately, the described methods are limited to the confidentiality of the data. These methods are based on different configurations of the cryptographic modules. C.S. Ong et al. in [**8**] present QoP mechanisms, which define security levels depending on security parameters. These parameters are: a key length, the length and contents of an encrypted block of data. P. Schneck and K. Schwan [**16**] proposed the adaptable protocol concentrating on the authorization. By means of this protocol one can change the version of the authorization protocol which finally changes the parameters of the asymmetric and symmetric ciphers. Y. Sun and A. Kumar [**15**] create the QoP models based on the vulnerabilities analysis which are represented by the attack trees. The leaves of the trees are described by means of special metrics of security. These metrics are used for describing individual characteristics of the attack. Unfortunately, the majority of the QoP models can be realized only for the three main security services: confidentiality, integrity and authorization. In article [**5**] B. Księżopolski and Z. Kotulski introduce a mechanism for adaptable security which can be realized for all the security services. In Section 3 we briefly present the model, which B. Księżopolski and Z. Kotulski introduce in [**5**].

In addition to the models, we often need a program that makes it easier to work with the model. Such a program is AVISPA that is a project for automated validation of security protocols. This tool and work with it are described in [**11**]. The configuration of the model was prepared by means of the SPOT application [**12**] which is the visualization of the model.

We have organized this paper as follows: Section 2 presents the TLS Record and TLS Handshake protocols. Section 3 briefly describes the model and TLS Handshake Protocol in the model. Next, in Section 4 we specify the SPOT analysis tool. Section 5 presents the methods of optimization of the protocol and the obtained results. Finally, in Section 6, we comment results and notify conclusions.

2. TLS protocol

In this section, we shortly present the protocol, which we have chosen for the analysis. It will be Transport Layer Security (TLS) Handshake Protocol version 1.2 [**6**]. The TLS protocol is continuation of the SSL protocol and it is designed to provide privacy and data integrity between two communicating applications. The TLS and SSL protocols are widely used with many network protocols (like

HTTP, SMTP, POP, SSH) and many applications provide TLS/SSL support to ensure security. TLS is the application protocol independent and it consists of two layers: the TLS Handshake Protocol and the TLS Record Protocol. The stack of protocols is shown in Figure 1. The task of the TLS Record Protocol is to provide two main security properties for the network connections: confidentiality and reliability. The confidentiality of the transmitted data is realized by symmetric cryptography for data encryption and keyed MAC for message integrity (reliable). The TLS Record Protocol is also used for encapsulation. In the next section the TLS Handshake Protocol will be described in detail.

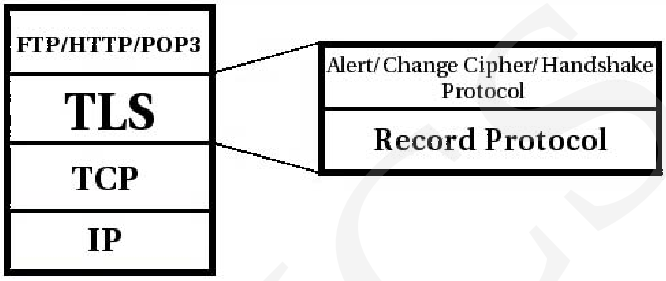

Fig. 1. TLS Protocol stack

2.1. TLS Handshake Protocol.

The goals of TLS Handshake Protocol are: authenticate the server and client between themselves, set cryptographic keys and an encryption algorithm by negotiation. These actions are performed before the transmission of data. At least one participant of communication must be authenticated and it is realized by asymmetric or public key cryptography. The negotiation process must be resistant to Man-in-the-middle attack so the shared secret is known only for the server and client. The negotiations process must be also reliable so any modification of the negotiation process is detected by the parties to the communication.

In the following we present a simple scenario in TLS Handshake which will be further optimized. Client wants to verify the server and next connect with the server by secure connection. The full message flow is presented in Figure 2.

1. A Client sends the ClientHello message. This message contains the following attributes: TLS Protocol version, session id, a list of available cipher suite, compression method and random values. The server responds with a ServerHello, which establishes the version of TLS Protocol, session id (when

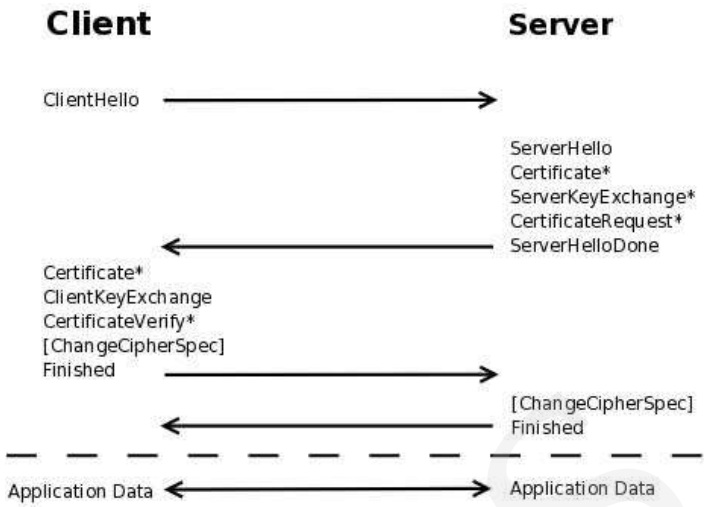

Fig. 2. Message flow for a full handshake in TLS Protocol 1.2

session is resumed), cipher suite and compression method. It also sends random values within ServerHello.

2. Next the Server sends the Certificate message which includes its certificate. Sending this message is not necessary, it depends on the selected cipher suite. Next message which may be sent is ServerKeyExchange. The server sends it when the server certificate is only for signing, or the server has no certificate. Next the server may send the CertificateRequest message. This message requests a certificate from the client, it may be sent if the server is authenticated and it also depends on the selected cipher suite. The server sends ServerHelloDone, signalling that this phase is done.
3. Now it is time for messages from a client. The client must send its certificate in the Certificate message only if the server requested it by the CertificateRequest message. Next the client sends the ClientKeyExchange message. Depending on the selected cipher this message may have different contents. To prove that a client has access to the private key in the certificate, the client sends a digitally-signed CertificateVerify message. Now a ChangeCipherSpec message is sent. The client sends it to signal the server that he started to use the encryption. Finally, the encrypted Finished message is sent by the client.
4. Similarly, in response, the server sends ChangeCipherSpec and the encrypted Finished message.
5. The handshake is complete. Now, the server and client can exchange encrypted Application Data.

3. The model

The realization of an electronic process strongly depends on the proper level of security.

The security level of an electronic process depends on several factors. This level can be modified by the choice of security elements applied in a protection system. In the model of the adaptable security [4], one can suggest an analytical expression to calculate the security level; its numerical value is a function of three primary parameters:

1. L - the protection level;
2. P - the probability of an incident occurrence;
3. ω - the impact of a successful attack.

In the following subsections we describe these elements. Every protocol is divided into subprotocols and, within these subprotocols, into steps. The main parameters listed in this section are computed for each service in each step. The calculation is prepared by means of the formula introduced in Section 3.4. The TLS Handshake is subprotocol of the TLS and we split it only into steps. We presented this division in the previous subsection.

3.1. The protection level (L).

Security services are realized by security mechanisms and every service can be realized in different ways. The security mechanism has attribute L. It is the protection level, defined in percent and describes the contribution of the protection of a particular service to the global protection level. In Table 1 the security services and the security mechanisms, with appropriate L values, for the TLS Handshake protocol are presented.

3.2. The probability of an incident occurrence (P).

The details about the used security mechanisms are represented by the trees. In Fig 3 we can see the component tree for the integrity service. Selection of leaves refers to the selection of the security mechanism particular configuration which will be used in the protocol. Every leaf is described by the following parameters:

– LZ – assets gained during successful attach on a given security element (100% = compromising the whole protocol);
– LK – knowledge needed for an attack (100% = expert);
– LP – costs needed for an attack (100% = the highest cost);
– C – communication steps as an additional possibility of attack, $C \in [0/0.1]$ (0.1 = the highest threat);

Table 1. Security services and security elements realized in TLS Handshake

	Security mechanisms					
Security services		1	2	3	4	5
	Integrity of data (I)	HMAC codes $L^{I1} = 60\%$	Advanced keys management $L^{I2} = 10\%$	Increase keys length $L^{I3} = 20\%$	Audit $L^{I4} = 10\%$	
	Confidentiality of data (C)	Encryption $L^{C1} = 60\%$	Advanced keys management $L^{C2} = 10\%$	Increase keys length $L^{C3} = 30\%$		
	Authorization of parties of protocol (Au)	Digital signatures $L^{Au1} = 50\%$	Advanced keys management $L^{Au2} = 10\%$	Advanced certificates management $L^{Au3} = 10\%$	Increase keys length $L^{Au4} = 25\%$	Audit $L^{Au5} = 5\%$

– M – practical implementation. The difficulty in implementing increases the probability of incorrect configuration. Error reports are an additional source of information, etc. $M \in [0/0.1]$ (0.1 = the highest threat).

Within service we define the additional security parameters:

– PP – global assets possible to gain in a given process $PP \in [0/0.1]$ (0.1 = the highest threat);
– I – kind of institution realizing the information process. Some of the institutions are of high threat. $I \in [0/0.1]$ (0.1 = the highest threat);
– H – potential risk for an attacker in the case of identification. The legal system and punishment of the countries where the process is realized. $H \in [0/0.1]$ (0.1 = a country with the lowest legal restrictions);

When we determine (by selection, of leaves from tree) the elements which we want to use for realization of a given security service then we can compute P. For every selected leaf we compute a probability of an incident occurrence according to the formulas:

$$P_P = (1 - (LK(1 - \omega_{LK}) + LP(1 - \omega_{LP})))(LZ + (1 - LZ)(C + M))$$
$$P^{\delta} = P_P + [\delta(1 - P_P)] \; \delta = (PP + I + H)$$
$$P = \max(P^{\delta})$$

where:

ω_{LK} – the weight defining potential attackers' lack of preparation in the domain of knowledge;

ω_{LP} – the weight defining potential attackers' lack of preparation in the domain

of costs;
$\omega_{LK} + \omega_{LP} = 1$
P_P – the probability of a threat occurrence without considering the additional δ parameter
P^{δ} – the probability after taking into account the additional parameter δ
P – the probability of an incident occurrence for this service, within a given step.

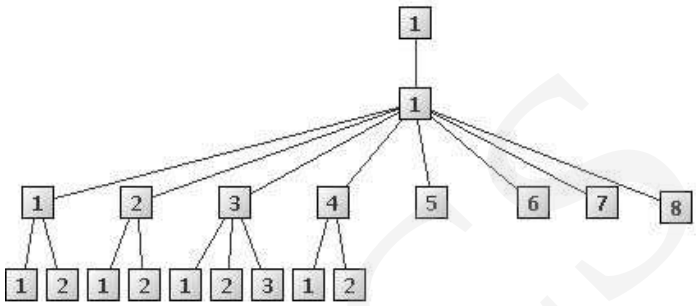

Fig. 3. The components tree for security service: integrity.

1 Integrity
1.1 Integrity of data
1.1.1 HMAC codes
1.1.1.1 Cryptographic modules (min. level 2) (LZ=80%, LK=70%, LP=80%, C=0.05, M=0.01)
1.1.1.2 Cryptographic modules (min. level 3) (LZ=80%, LK=80%, LP=90%, C=0.05, M=0.02)
1.1.2 Ports and interfaces of cryptographic modules
1.1.2.1 Cryptographic modules (min. level 2) (LZ=70%, LK=50%, LP=80%)
1.1.2.2 Cryptographic modules (min. level 3) (LZ=70%, LK=70%, LP=80%)
1.1.3 Specification of cryptographic modules
1.1.3.1 Cryptographic modules (min. level 2) (LZ=70%, LK=50%, LP=80%)
1.1.3.2 Cryptographic modules (min. level 3) (LZ=70%, LK=70%, LP=80%)
1.1.3.3 Increase digest lengths (LZ=10%, LK=60%, LP=40%)
1.1.4 Encryption mode supports integrity
1.1.4.1 Cryptographic modules (min. level 2) (LZ=80%, LK=70%, LP=80%)
1.1.4.2 Cryptographic modules (min. level 3) (LZ=80%, LK=80%, LP=90%, M=0.01)
1.1.5 Keys distribution (LZ=80%, LK=50%, LP=80%, C=0.02)
1.1.6 Key usage (LZ=80%, LK=80%, LP=50%)
1.1.7 Compression method supports integrity (LZ=30%, LK=80%, LP=50%, C=0.01)
1.1.8 Audit (LZ=10%, LK=60%, LP=40%, C=0.01, M=0.03)

The leaves are described using the terms from [**10**].

3.3. The impact of a successful attack (ω).

The impact of a successful attack is the second parameter (besides P) associated with risk. We calculate it, as previously, for each service in each step. We use for calculation direct and indirect parameters, presented below.
The direct parameters:
LZ – assets gained during a successful attack on given security elements (100% is the compromise of the whole protocol);
F – financial losses during a successful attack on given security elements (100% is the total financial loss).

The indirect parameters:
α – necessary financial costs for repairing the damages gained during a successful attack (100% is the maximal cost);
β – losses of the value of the company shares or the company reputation (100% is the maximal market loss).

Impact of an attack is calculated by the formula:

$$\omega = \frac{LZ}{3}(F + \beta + \alpha).$$

3.4. Security level (F_S).

The global security level expresses the security of the whole cryptographic protocol. We calculate this factor according to the formula:

$$F_S = \frac{1}{a}\sum_{i=1}^{a}\frac{1}{b_i}\sum_{j=1}^{b_i}\frac{1}{c_{ij}}\sum_{x=1}^{c_{ij}}(L_{ij}^{x})^{Z}[(1-\omega_{ij}^{x})(1-P_{ij,ALL}^{x})]$$

where:
F_S is the security level realized by a given version of cryptographic protocol, $F_S \in (0,1)$
i is the number of subprotocols in a given protocol;
j is the number of steps in a given subprotocol;
x is the number of specific security services;
ω_{ij}^{x} is the weight describing an average cost of losses after a successful attack on a given service, $\omega \in (0,1)$;
L_{ij}^{x} is the value of a protection level for a given service, $L \in (0,1)$;
P_{ij}^{x} is the probability of an attack on a given service, $P \in (0,1)$;
Z is the scalability parameter for security elements, $Z \in (0,10)$.

4. Security Protocol Optimization Tool

Security Protocol Optimization Tool (SPOT) is the element of the architecture presented in [**12**]. For purposes of this article we present only key elements and aims of the SPOT. The main goal of the SPOT is to implement the Model briefly presented in Section 3 and completely in [**4**]. By this tool we can create versions of a given protocol, compare these versions and visualize the results. It is also designed to be portable and user–friendly so the interface to the application is realized by means of the graphic panels. The important feature of the SPOT is the optimization module, which is capable of generating all states of a given protocol and shows the optimal states according to the user's preferences. This module can print this information and their impact on the security level (P and LZ parameter). This way, the user can choose the security elements that give him desirable results.

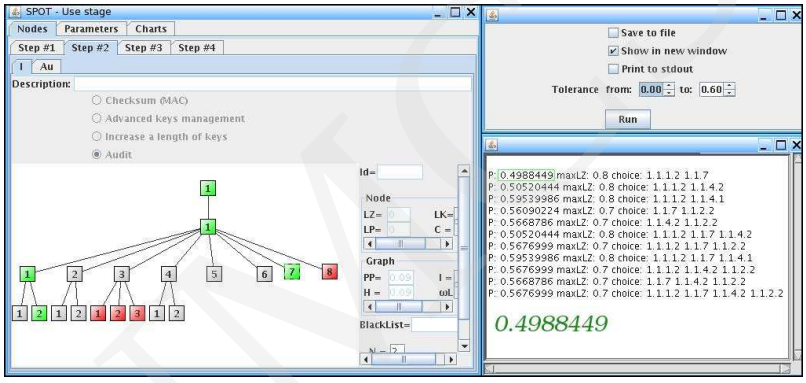

Fig. 4. (SPOT) Optimization of the security service: integrity.

5. Optimization of the protocol

In this section we characterize the scenario of the TLS protocol communication in detail. We define the parameters required for the calculation of the security level (F_S) for the particular version of the TLS protocol. Finally, we optimize the TLS protocol by selections of the security elements. For each service, after selection of appropriate elements, F_S with indirect parameters are presented and that configuration is represented a distinct version. The first version of the protocol is represented as V_F. In this version we change configuration to optimize Integrity and we obtain as results V_I. By selecting more secure mechanisms as regards confidentiality we achieve the version $V_{I,C}$, which is derived from V_I. Optimization of authorization, in $V_{I,C}$ gives us the last version $V_{I,C,Au}$.

5.1. Version of the TLS Handshake protocol.

In our example we assume that parties of protocol transfer critical data within a reputable organization. We also assume that Client and Server are high–performance systems (a lot of CPU and RAM). In these conditions our priority, during protocol optimization, is security. In the first step of optimization, we determine which security service is required for single steps of a given protocol. In the article we chose the security services which are presented in Table 2. "YES" in the table means that the appropriate service is realized in a given step, and "NO" means that the service is not realized. So, in Step 1 we ensure only integrity of data. There is integrity and authorization in Step 2. All services are in Step 3 and Step 4: integrity, confidentiality, and authorization.

Table 2. Selected security services

	Steps of the subprotocol				
Security services		**Step 1**	**Step 2**	**Step 3**	**Step 4**
	I	YES	YES	YES	YES
	C	NO	NO	YES	YES
	Au	NO	YES	YES	YES

Table 3. Selected security mechanisms for the security services

	Steps of the subprotocol				
Security services		**Step 1**	**Step 2**	**Step 3**	**Step 4**
	I	1,4	1,2	1,2,3,4	1,2,3,4
	C	NO	NO	1,2,3	1,2,3
	Au	NO	3	1,2,3,4,5	1,2,3,4,5

Next we assign security mechanisms which realize the security service chosen before. In our case, security mechanisms possible to select are presented in Table 1. We present selection of mechanisms which realize appropriate security service in Table 3. Numbers in this table identify the mechanisms according to Table 1. In the next step, the value of other model parameters must be defined. According to our communication scenario (critical data and reputable organization) for the parameters $F, \alpha, \beta, PP, I, H$ we set a high value, it is presented in Table 4. Also, we set Z equal 1. The first choice of the security elements (from security trees) is standard in each step and service. To realize security services and mechanisms we choose in the base configuration: HMAC-MD5 for Integrity, RC4 with 128 bytes key for confidentiality and RSA with the standard verification for authorization.

Table 4. Parameters for the basic version of TLS protocol

	LZ	***F***	α	β
Step 1				
I	0.7	0.88	0.83	0.92
Step 2				
I	0.8	0.96	0.89	0.87
Au	0.7	0.97	0.87	0.95
Step 3				
I	0.7	0.82	0.79	0.92
C	0.8	0.98	0.86	0.92
Au	0.7	0.98	0.97	0.92
Step 4				
I	0.7	0.97	0.94	0.98
C	0.8	0.98	0.99	0.98
Au	0.7	0.99	0.98	0.97

5.2. First results and the analysis (V_F).

When we define all model parameters we can compute global security level F_S. This version of the protocol we define as V_F. Table 5 presents the global security level F_S with the main model parameters P, ω, L^Z. The results are presented for all protocol steps and required security services in these steps.

We have obtained $F_S = 0.0694744$ and probabilities range from 0.672065 to 0.8106986. These values for P parameter are high (maximum level = 1). In the article we use Optimization Module from SPOT, introduced in Section 4, for reducing the value of P parameter. In the next sections, the optimization of probability of incident occurrence (P) will be described. This process can be realized for all steps of the protocol and for specific security service in them.

5.3. Optimizing the results: integrity (V_I).

In each step of TLS protocol we use Optimization Module for integrity and the results are shown in Fig. 4. We have observed that using cryptographic modules on level 3 is more secure (according to [**10**]). In integrity trees, these modules represent the HMAC codes, encryption (with the appropriate mode) and compression methods. Applying to the Optimization Module we decided to increase length of key and use advanced keys management for Step 3 and Step 4. We have done it by using SHA–256 instead of MD5 for HMAC. SHA– is considered more secure and its digest is longer. By these changes we have achieved expected effects. That configuration is represented in our paper as

V_I. The results after integrity optimization are presented in Table 6. After this optimization $F_S = 0.08213736$. This is due to a large reduction in the parameter P in every step. The first stage of optimization refers to Integrity, in the next steps other security services will be optimized.

Table 5. First results for our version (V_F)

	$\boldsymbol{P}$	ω	L^Z
Step 1			
I	0.6858307	0.6136666	0.7
Step 2			
I	0.7187909	0.7253333	0.7
Au	0.7941681	0.650999	0.1
Step 3			
I	0.6858307	0.5903333	1.0
C	0.7287932	0.7360000	1.0
Au	0.8106986	0.6696667	1.0
Step 4			
I	0.672065	0.6743333	1.0
C	0.7287932	0.7866667	1.0
Au	0.8106986	0.6860000	1.0
F_S	**0.0694744**		

Table 6. Results after I optimization (V_I)

	$\boldsymbol{P}$	ω	L^Z
Step 1			
I	**0.477998**	0.7013333	0.7
Step 2			
I	**0.4988449**	0.7253333	0.7
Au	0.7941681	0.650999	0.1
Step 3			
I	**0.5900936**	0.5903333	1.0
C	0.7287932	0.7360000	1.0
Au	0.8106986	0.6696667	1.0
Step 4			
I	**0.5676999**	0.6743333	1.0
C	0.7287932	0.7866667	1.0
Au	0.8106986	0.6860000	1.0
F_S	**0.08213736**		

5.4. Optimizing the results: confidentiality ($V_{I,C}$).

The confidentiality of the data is required in Step 3 and Step 4 (Table 2). In Confidentiality we use cryptographic modules mainly for encoding/decoding data, additionally in Step 3 Client must securely generate keys. Originally selected security elements were: cryptographic modules for every operation on level 2, standard length of the key, standard keys management. Using Optimization Module for confidentiality we found the most secure selection for this version of the protocol. We set cryptographic modules to level 3, standard length of key has been increased and we have applied advanced keys management. These settings have been mainly caused by choice of AES256–CBC algorithm instead of RC4. In Step 3 we also assume using of the secure PRNG function for Key Generation. Each of these components is described in [**10**] and as mentioned before, this version designated as $V_{I,C}$. After applying these changes we have calculated new results. They are shown in Table 7. There was a significant increase the F_S. Before changes F_S was equal to 0.08213736, now $F_S = 0.09844751$. This is due to the change in the parameter P for Confidentiality. In Step 3, parameter P has changed from 0.7287932 to 0.5676999 but in Step 4 it has changed from 0.7287932 to 0.4988449.

5.5. Optimizing the results: authorization ($V_{I,C,Au}$).

Authorization of the parties of protocol is required in Steps 2, 3 and 4. The trees of the authorization are very complex so the use of the Optimization Module is necessary. We can notice that in Table 7 the probabilities of an incident occurrence are high only for the authorization service. For optimization we have switched cryptographic modules to level 3, we have applied longer keys and advanced keys management. These changes have improved P parameter only in Step 3. By using Optimization Module we have noticed that high P is caused by the security elements associated with a certificate authority (CA). Optimization Module has generated several selections, where P of each selection was < 0.5 for Step 2. For Step 3 and Step 4 we have obtained a few selections, where P was < 0.75. According to these selections we have changed security elements. A standard parties verification was replaced by detailed verification. We assumed that it is possible to verify party of the communication at any time and that the revocation of the certificate may take up to 24 hours. This configuration of the protocol is defined as the version $V_{I,C,Au}$.

5.6. Final results.

The TLS protocol guarantees the three main security services: integrity, confidentiality and authorization. The optimization process is completed when these required security services are optimized. The final results are presented in Table 8. Comparing the final results with the base results which are presented in Table 5 one can notice that probability of incident occurrence P for all steps decreased significantly. Consequently, the security level increased from 0.0694744 to 0.10750219.

Table 7. Results after C optimization ($V_{I,C}$)

	$\boldsymbol{P}$	ω	L^Z
Step 1			
I	0.477998	0.7013333	0.7
Step 2			
I	0.4988449	0.7253333	0.7
Au	0.7941681	0.650999	0.1
Step 3			
I	0.5900936	0.5903333	1.0
C	**0.5676999**	0.6440000	1.0
Au	0.8106986	0.6696667	1.0
Step 4			
I	0.5676999	0.6743333	1.0
C	**0.4988449**	0.7866667	1.0
Au	0.8106986	0.6860000	1.0
F_S	**0.09844751**		

Table 8. Final results ($V_{I,C,Au}$)

	$\boldsymbol{P}$	ω	L^Z
Step 1			
I	0.477998	0.7013333	0.7
Step 2			
I	0.4988449	0.7253333	0.7
Au	**0.4418600**	0.2790000	0.1
Step 3			
I	0.5900936	0.5903333	1.0
C	0.5676999	0.6440000	1.0
Au	**0.7079023**	0.6696667	1.0
Step 4			
I	0.5676999	0.6743333	1.0
C	0.4988449	0.7866667	1.0
Au	**0.7079023**	0.6860000	1.0
F_S	**0.10750219**		

6. Conclusions

The presented methodology of protocol optimization has many advantages. We can work with any security protocol or just communication process. Furthermore, using the SPOT we can easily create versions of a given protocol and we can change them depending on the assumed scenario. We have presented methodology on the very popular protocol and we assume the scenario often realized. The protection of the

TLS protocol was guaranteed only by the options provided by its developers. This gives us the confidence that the protocol can not be accidentally broken. The cipher suite used before optimization was TLS–RSA–WITH–RC4–128–MD5, now it is TLS–DH–RSA–WITH–AES–256–CBC–SHA256. The exact description of these suites is in [**6**]. Applied changes are unfavorable for the protocol efficiency but during our optimization we assume that the security not efficiency of the protocol will be optimized. Finally, the global security level increases by 0.03802779 and it is *"55%"* of the base value of F_S.

References

[1] Roman R., Alcaraz C., Lopez J., *A Survey of Cryptographic Primitives and Implementations for Hardware-Constrained Sensor Network nodes*, Mobile Networks and Applications 12 (2007) 231.

[2] Kayayurt B., Tuglular T., *End-to-end security implementation for mobile devices using TLS protocol*,Journal in Computer Virology 2 (2006) 87.

[3] Elgohary A., Sobh T.S., Zaki M., *Design of an enhancement for SSL/TSL protocols*, Computers & Security, Elsevier (2006)

[4] Yasinsac A., Childs J., *Formal analysis of modern security protocols*, Information Sciences, Elsevier (2004)

[5] Księżopolski B., Kotulski Z., *Adaptable security mechanism for dynamic environments*, Computers & Security 3 (2007) 246.

[6] Dierks T., Rescorla E., *The Transport Layer Security (TLS) Protocol Version 1.2*, RFC 5246, IETF (2008).

[7] Burrows M., Abadi M., Needham R., *The logic of authentication*, ACM Transactions on Computer Systems (1990).

[8] Ong C.S., Nahrstedt K., Yuan W., *Quality of protection for mobile multimedia applications*, In Proceedings of the 2003 IEEE International Conference on Multimedia & Expo (2003).

[9] Lindskog S., Jonsson E., *Adding Security to Quality of Service Architectures*, Proceedings of the SSGRR Conference, (2002).

[10] National Institute of Standards and Technology, *Security Requirements for Cryptographic Modules*, FIPS PUB 140-2 (2002).

[11] Vigano L., *Automated Security Protocol Analysis With the AVISPA Tool*, Electronic Notes in Theoretical Computer Science 155 (2006) 61.

[12] Szalachowski P., Księżopolski B., Kotulski Z., *SPOT: Optimization tool for the adaptable security mechanism for dynamic environments*: submitted to the publication (2009).

[13] Gerber M., Solms R., *Management of risk in the information age*, Computers & Security 14 (2005) 16.

[14] Dwaikat Z., Parisi-Presicce F., *Risky trust: risk-based analysis of software systems*, Proceedings of the 2005 workshop on Software engineering for secure systems—building trustworthy applications (2005).

[15] Sun Y., Kumar A. *Quality od Protection(QoP): A quantitative methodology to grade security services*, 28th Conference on Distributed Computing Systems Workshop (2008) 394.

[16] Schneck P. Schwan K., *Authenticast: An Adaptive Protocol for High-Performance*. Secure Network Applications, Technical Report GIT-CC-97-22 (1997).