

Time Influence on Security Protocol

Sabina Szymoniak^a

Department of Computer Science, Czestochowa University of Technology, Poland

Keywords: Formal Methods, Security Protocols, Timed Analysis, Verification.

Abstract: The paper discusses the problem of influence of time parameters on protocols security. It is a significant issue because some periods may affect us to be or not to be in the real and virtual world. Time can decide about the security of our private data, money and many others. It is necessary to check whether used protocols provide an appropriate security level of our data. Also, Intruder capabilities and knowledge may evolve with time. With wrongly selected time parameters, the Intruder may perform an attack on protocol and deceive honest users. The research has expanded the formal model and computational structure designed previously. Based on this, we implemented a tool. This tool can calculate the correct protocol execution and carry out simulations. Thanks to this checking the possibility of Intruder attack including various time parameters was possible. We presented experimental results on NSPK protocol and WooLamPi protocol examples.

1 INTRODUCTION

Time is one of the most significant parameters of the private and professional life of every person. Fractions of seconds often decide about our to be or not to be in this world. Internet communication is similar. Every day at any time Internet users sends much information electronically. They want to participate in secure communications. It means the information will be delivered at the right time and will remain safe and intact (Čibej Uroš et al., 2019).

Our communication is secured by security protocols. Unfortunately, every message is exposed to a wicked user. This user is called Intruder. The Intruder can steal and use transmitted data. Sometimes, time favours Intruder. The transmission time can give Intruder many opportunities to carry out an attack. We use timestamps to determine the date and time of the message generation. Despite the use of them, the attacker can successfully exploit message. For this reason, it is significant to verify security protocols to protect against unwanted situations.


Also, respectable monitoring and solid users identification is a very important step in ensuring an adequate level of resource security. Unfortunately, there are many types of attacks. Each of them can share the features of other attacks. One of the most significant attacks is a Man In The Middle Attack. This attack consists of executing a protocol not directly between

two honest users ($A \Leftrightarrow B$) but executing a protocol with Intruder standing in the middle of the communication ($A \Leftrightarrow I \Leftrightarrow B$).

Modelling and verification of security protocols (SP) is an important step in avoiding the Intruder attack. For the last fifty years, a scientist from different countries proposed many methods of SP modelling and verification. We can indicate inductive methods (Bella and Paulson, 1997), deductive methods (Burrows et al., 1990), model checking (Kurkowski, 2013) and others methods (Chadha et al., 2017; Basin et al., 2018; Siedlecka-Lamch et al., 2016; Nigam et al., 2016; Perháč et al., 2019) as examples. These methods started to appear and evolve with the first security protocols such as Needham Schroeder Public Key protocol (Needham and Schroeder, 1978). The analysis of the protocols became more exact after the introduction of the Intruder model by Dolev and Yao (Dolev and Yao, 1981).

Time modelling is an integral part of a protocol study. The most important times, which must be considered, are delays in the network. These values have a huge influence on the duration of the protocol session. In the network, different situations occur. Selecting the appropriate interval of delay in the network values we can obtain a critical influence on the protocol correctness and also on Intruder capabilities.

Corin proposed a very interesting approach in (Corin et al., 2007) and (Corin et al., 2004). In this approach, Corin modelled SP using timed automata

^a  <https://orcid.org/0000-0003-1148-5691>

taken from the Alur and Dill work (Alur and Dill, 1994). Corin noticed the SP sensitivity to the passage of time. They presented the individual users' behaviours by individual automata. Communication between the automata, which modelled users, was synchronised accordingly. However, it was necessary to prepare a precise and very detailed protocol specification. Such a specification should enable disambiguation of users behaviours. Also, the time values should be set on each state.

Kurkowski formal model (Kurkowski, 2013) is base for the presented research. This model enables the timed and untimed SP examination. For the needs of our analysis, we extended this model to consider delays in the network. Only in (Jakubowska and Penczek, 2007b; Jakubowska and Penczek, 2007a) Jakubowska and Penczek considered delays, but they did not continue these research. To perform our considerations, we used Kurkowski and Jakubowska and Penczek models. We extended combined research methods (Kurkowski and also Jakubowska and Penczek methods). We include the ability to calculate the duration of individual operations performed during the protocol execution. Delay in the network values and encryption and decryption times could be calculated according to the selected probability distributions. Thanks to this, it is possible to simulate the protocols executions in the network in a correct way. The main contributions of this paper are a method and tool which allow to examined time-sensitivity of SP.

The organisation of the rest of this paper is as follows. In the next Section, we presented our parameters assumptions. Next, we introduced Needham Schroeder Public Key protocol (NSPK) and WooLamPi protocol (WLP) as examples. The third Section presented research assumptions and experimental results for exemplary protocols. The last Section included our conclusions and plans for the future.

2 ASSUMPTIONS AND PROTOCOLS

2.1 Research Methodology

This paper is an extension and continuation of previous research (Szymoniak et al., 2015) and (Szymoniak et al., 2017). Mentioned papers introduced the formal model and the computational structure. The formal model enables defining sets of time conditions, the protocol step and a protocol.

All definitions allow the full specification of a timed and untimed protocol. We include external and

internal actions performed during the protocol execution and time conditions. In time conditions, we took into account delays in the network. Time conditions are imposed on each step and also on the whole protocol execution. Thanks to this, we can consider time influence on protocol step and the protocol.

The computational structure makes it possible to consider the protocols executions in the physical network. Thanks to this, we can consider the protocol with Intruder. The Intruder is a user who may try to deceive other users. Intruder's behaviour may cause not the fulfilment of protocol goals. We consider four Intruder models: Dolev-Yao model, restricted Dolev-Yao model, lazy Intruder model and restricted lazy Intruder model (Dolev and Yao, 1981).

The main part of computational structure are definitions of timed protocol step, users knowledge and the ways of its acquiring and time dependencies. In case of time dependencies we use time of composing the message (T_c), step time (minimal T_k^{min} , current T_k , maximal T_k^{max}), session time (minimal T_s^{min} , current T_s , maximal T_s^{max}), lifetime (T_k^{out}).

Lifetime in k -th step indicates the maximal waiting for a response time. The correct formula for lifetime calculating is as follows:

$$T_k^{out} = \sum_{i=k}^n T_i^{max} \quad (1)$$

where:

- k - step number,
- i - step counter, for $i = k \dots n$,
- n - number of steps in the protocol,
- T_k^{out} - lifetime in the k -th step,
- T_i^{max} - maximal step time.

In (Szymoniak et al., 2017), we presented other formulas.

For the research, we implemented a tool in C++. This tool enabled testing including the assumptions adopted at each research stage. We presented the tool activities in detail in (Szymoniak, 2021). The tool enables three kinds of research during which we can examine all protocol executions:

- a) *Timed Analysis* - which allows examining the duration of the protocol steps and protocol vulnerability on attacks,
- b) *Delay Simulations* - which allows performing simulations of delays in the network and their influence on Intruder activity,
- c) *E&D Simulations* - which allows performing simulations of encryption and decryption times and their influence on Intruder activity.

The most interesting aspect of the tool is the ability to set one of four statuses for each executions:

- a) *correct* - for executions completed in the correct time interval with preserved time conditions,
- b) *!min* - for executions completed with preserved time conditions, but current session time T_s was lower than minimal session time T_s^{min} ,
- c) *!max* - for executions completed with preserved time conditions, but current session time T_s was greater than maximal session time T_s^{max} ,
- d) *error* - for executions where any time condition was not fulfilled (execution was ended with error).

By *preserved time conditions* we mean conditions defined in every protocol step. So if the T_k is lower than T_k^{out} for k -th step, we can mark the time conditions preserved in the k -th step were met. Also, the protocol must fulfil dependencies limiting its duration. When the time conditions for every step are met and session time is between minimal and maximal session time, the protocol was made correctly (situation a)). Also, we must consider some different situations. When the Intruder tries to execute the protocol, he may not have the right set of knowledge for ending the protocol. Then, the Intruder must establish a parallel session to acquire knowledge. Time of additional steps, executed between two steps from basic execution, must be added to basic step time and also to session time. If Intruder actions do not consist of some operations like encryption or decryption, the step time (and session time) can be lower than minimal values for this parameter calculated according to protocol structure (situation b)). On the other hand, Intruder actions in basic and additional steps may consist of all protocol operation, so the step time and session time can be greater than maximal values for this parameter calculated according to protocol structure (situation c)).

Situations b), c) and d) indicate a network failure. The distinction between the four situations was necessary to determine what the Intruder capabilities were and also to know how the activity of an Intruder affects other users and the whole network. The occurrence of the situation d) ends tool work with an error. These assumptions mean that the action of a given execution is possible as long as the session time is between T_s^{min} and T_s^{max} and the time conditions are met.

In (Szymoniak et al., 2017) we used the constant delays value. Then, we decided to consider the random values of delays in the network and carry out simulations of the computer network including these values. Thanks to this, it will be possible to demonstrate the influence of this time parameter on the protocol correctness and Intruder capabilities. Also, we decided to consider the influence of encryption and

decryption times on tested properties. We carried out research using an abstract time unit ([tu]). It is an arbitrarily selected and considered period.

2.2 Security Protocol Example

The most characteristic security protocol is the Needham Schroeder Public Key protocol (NSPK), proposed in (Needham and Schroeder, 1978). The purpose of this protocol is mutual authentication of two users. The scheme of timed version¹ of this protocol in Alice-Bob notation is as follows:

$$\begin{aligned} \alpha_1 \quad A, B & : \{T_A, I_A\}_{K_B} \\ \alpha_2 \quad B, A & : \{T_A, T_B\}_{K_A} \\ \alpha_3 \quad A, B & : \{T_B\}_{K_B} \end{aligned}$$

There are two users A and B in this protocol. In the first step, user A generates his timestamp. Then, he creates a ciphertext which contains timestamp T_A and his identifier I_A . The message is encrypted with the public key of user B . In response, B sends to A his timestamp and newly generated timestamp T_B . This message is encrypted with the key K_A . In the last step, A confirms his identity by sending his timestamp encrypted with K_B to B .

Our tool generated eighteen executions of NSPK. These executions differed from each other in the parameters used by the Intruder. However, only fourteen of them are real executions, i.e. those that can execute on a computer network. Remaining executions are impossible to perform. One of the participants in these executions is not able to acquire relevant knowledge. Designation of a real protocol executions set was done using SAT-solver. These executions were not included in our analysis.

The next protocol is symmetric WooLampPi protocol (WLP). WLP aimed in one way users authentication with public keys and the trusted server. Its timed version scheme in Alice-Bob notation is as follows:

$$\begin{aligned} \alpha_1 \quad A, B & : I_A \\ \alpha_2 \quad B, A & : T_B \\ \alpha_3 \quad A, B & : \{T_B\}_{K_{AS}} \\ \alpha_4 \quad B, S & : \{I_A, \{T_B\}_{K_{AS}}\}_{K_{BS}} \\ \alpha_5 \quad S, B & : \{T_B\}_{K_{BS}} \end{aligned}$$

There are three users in this protocol: A , B and S (the trusted server). In the first step, user A sends to B his identifier. Then, B generates his timestamp and sends them to A . In third step user A creates a ciphertext which contains timestamp T_B and sends them to B . The message is encrypted by A key K_{AS} which is shared between user A and server. User B cannot

¹Timed version of the protocol was created by replacing the random numbers (nonces) by timestamps

decrypts this message, so he send it with its ID encrypted by key K_{BS} to the server. In response, the server sends back to B his timestamp encrypted by key K_{BS} and confirms A 's identity to B .

We assumed that Intruder impersonates only honest users, not the trusted server. This restriction caused that the implemented tool generated only fourteen executions. Next, SAT-solver marked six of them as impossible to execute.

The WLP executions structure is different than the structure of NSPK protocol executions. The protocol initiator occurs in the first three protocol steps only. He does not generate any confidential information. The recipient (user with whom initiator desires to communicate) in protocol occurs in all protocol steps, and in fourth and fifth steps communicates with the server. Recipient generates its timestamp. However, during the generation of executions, we adopted assumption limiting Intruder capabilities as a recipient. Intruder as a recipient can appear in two situations: as himself (with his cryptographic objects) and impersonating an honest user and using user data (cryptographic objects).

3 EXPERIMENTAL RESULTS

We researched three stages. In the first stage, we evaluated the security of the protocol for each real execution during timed analysis. Values of encryption and decryption times were constant. They were equal to 4 [tu]. We assumed the interval of delay in the network value (1 - 3 [tu]) and constant current delay in the network (1 [tu]). The second and third stages involved simulation tests for all real protocol executions. In the second stage, we assumed constant values of encryption and decryption times (2 [tu]) and the interval of delay in the network (1 - 10 [tu]). We randomised current delay values with the following probability distributions: uniform, normal, Cauchy's, exponential.

We selected probability distributions to model different loads of the computer network. Normal and uniform distributions modelled real network work. This network sometimes has problems with the load, but they do not affect the activity of users. Cauchy's distribution modelled the heavily loaded network. In this network, there are many users, but few resources. We try to model a fast network by an exponential distribution. In that network, there are no problems. In the third stage of the study, we consider constant delay values (2, 4, 6, 8, 10 [tu]). We randomised encryption and decryption times according to a uniform probability distribution. Also, we assumed that no one can guess timestamp. We tested every real exe-

cution in a thousand test series during the simulation. We will present our experimental results in more detail on NSPK and WLP protocol example. Obtained results explain the behaviour of protocol and its sensitivity to time flow.

3.1 Times Analysis for NSPK Protocol

For the first stage of studies, we calculated following lifetimes for steps: $T_1^{out} = 35$ [tu], $T_2^{out} = 23$ [tu], $T_3^{out} = 11$ [tu]. Also, we calculated minimal and maximal session times $T_s^{min} = 29$ [tu], $T_s^{max} = 35$ [tu].

Let's analyze one of NSPK execution that maps Lowe's attack². In this execution, Intruder uses user A 's timestamp and his ID and public key. Unfortunately, Intruder does not have the appropriate knowledge to execute step α_1 . Therefore, he must establish additional execution (β) to acquire knowledge of T_A .

In Table 1 we show timed analysis of attacking execution. This analysis includes times of basic (column BS) and additional (column AS) steps and comments regard to steps executions. Each step time consists of four values: encryption time, time of confidential information generation, delay in the network and decryption time. In basic steps also appeared five value (on the first position). This value is an additional step time which was executed between earlier and current basic step.

Table 1: Timed analysis of first attacking execution.

BS	AS	Time [tu]	Comment
α_1		$4 + 1 + 1 + 4 = 10$	ok
	β_1	$4 + 0 + 1 + 4 = 9$	ok
	β_2	$4 + 1 + 1 + 4 = 10$	ok
α_2		$19 + 4 + 0 + 1 + 4 = 28$	$!T_2^{out}$
α_3		$4 + 0 + 1 + 4 = 9$	$!T_s^{max}$
	β_3	$18 + 4 + 0 + 1 + 4 = 27$	$!T_3^{out} \&\&!T_s^{max}$

Step α_1 lasted 10 [tu] (encryption time - 4 [tu], time of confidential information generation - 1 [tu], delay in the network - 1 [tu], and decryption time - 4 [tu]). Time conditions for the first step were met. Before step α_2 had to execute additional steps (β_1 and β_2). These steps lasted 19 [tu]. Additional steps time made that the step α_2 should not be executed because T_2^{out} will not be exceeded. Step α_2 lasted for 28 [tu]. The lifetime for the second protocol step was equal to 23 [tu]. In this situation, the honest user should cancel

α_1	B, I	:	$\{I_B, T_B\}_{K_I}$,
β_1	I, A	:	$\{I_I, T_I\}_{K_A}$,
β_2	A, I	:	$\{T_I, T_A\}_{K_I}$,
α_2	I, B	:	$\{T_B, T_A\}_{K_B}$,
α_3	B, I	:	$\{T_A\}_{K_I}$,
β_3	I, A	:	$\{T_A\}_{K_A}$.

the communication should. If for some reason communication would continue, at α_3 a similar situation would occur. This step will last for 9 [tu] (encryption time - 4 [tu], time of confidential information generation - 0 [tu], delay in the network - 1 [tu], and decryption time - 1 [tu]). Please note that, when we add previous steps times to session time, T_s will be equal to 47 [tu], while T_s^{max} is equal to 35 [tu] for adopted assumptions. Also, T_3^{out} for additional step β_3 will not be exceeded. This step must include α_2 and α_3 steps times, so β_3 step time will be equal to 27 [tu] and will be greater than T_3^{out} . Session time for this step would be greater than T_s . It is impossible to conduct an attack on this protocol for such time parameters.

Let's analyse another attacking execution which shows the *Man In The Middle Attack* on NSPK protocol. In this case, Intruder must establish additional execution in order to perform an attack³. The attacker uses the entire ciphertext, so it does not try to encrypt and decrypt it. We do not add these times to the total step time, so these steps are shorter than they should.

Table 2: Timed analysis of second attacking execution.

α	β	Time [tu]	Comment
α_1		$4 + 1 + 1 + 0 = 6$	ok
	β_1	$0 + 0 + 1 + 4 = 5$	ok
	β_2	$4 + 1 + 1 + 0 = 6$	ok
α_2		$11 + 0 + 0 + 1 + 4 = 16$	ok
α_3		$4 + 0 + 1 + 0 = 5$	ok
	β_3	$5 + 0 + 0 + 1 + 4 = 10$	ok

In Table 2 we show timed analysis of this execution. The initiator of this execution is user A, so he generates his timestamp and encrypts the message. We add these times values to the step time. This step receiver is an Intruder who impersonates the user B. In this execution Intruder uses user B's key and timestamp, so he cannot decrypt messages which are addressed to user B. Decryption time would not be added to the step time. According to above analysis, step α_1 lasted 6 [tu] (encryption time - 4 [tu], time of confidential information generation - 1 [tu], delay in the network - 1 [tu], and decryption time - 0 [tu]). Time conditions for the first step were met. Before step α_2 had to be executed, additional steps (β_1 and β_2) must be performed. In step α_1 an Intruder possessed entire ciphertext which was sent in step β_1 without encryption by him. Encryption time would

α_1	$A, I(B)$:	$\{I_A, T_A\}_{K_B}$,
β_1	$I(A), B$:	$\{I_A, T_A\}_{K_B}$,
β_2	$B, I(A)$:	$\{T_A, T_B\}_{K_A}$,
α_2	$I(B), A$:	$\{T_A, T_B\}_{K_A}$,
α_3	$A, I(B)$:	$\{T_B\}_{K_B}$,
β_3	$I(A), B$:	$\{T_B\}_{K_B}$.

not be added to the step time. Please note that for such executions when the Intruder is message sender in the current step, encryption time would not be added to step time, and when the Intruder is message receiver in the current step, decryption time would not be added to step time. In step β_2 an Intruder was not decrypted again. These steps lasted 11 [tu]. Additional steps time made that step α_2 should be done. Step α_2 lasted for 16 [tu], while the lifetime for the second protocol step was equal to 23 [tu]. Time conditions for the second step were met. Steps α_3 and β_3 were executed without problems at 5 [tu] and 10 [tu]. All time conditions for each protocol step were met, so a Man In The Middle Attack was possible.

3.2 Delays Simulations for NSPK Protocol

In the second stage, we conducted simulations of the network using randomly generated current delay in the network values. For the previously adopted assumptions, we calculated new lifetime values for each step: $T_1^{out} = 44$ [tu], $T_2^{out} = 29$ [tu], $T_3^{out} = 14$ [tu]. Also, we calculated minimal and maximal session times: $T_s^{min} = 29$ [tu], $T_s^{max} = 35$ [tu]. Simulations allowed to verify users behaviour and Intruder activity in computer networks with a different load. We present simulations experimental results on uniform and normal probability distributions examples.

3.2.1 Uniform Probability Distribution

We observed the following results during the simulation of real NSPK executions (using the uniform probability distribution). To present the results, we numbered the possible executions (column *No* in Table 3).

Table 3: Summary of NSPK simulations with uniform distribution which ended in correct interval in [tu].

No	$Min(T_s)$	$Avg(T_s)$	$Max(T_s)$	$Avg(D)$
1	17.1	30.73	43.4	5.58
2	18.1	30.69	43.3	5.57
3	α_2	29.6	41.5	5.53
4	39.2	42.4	43.9	3.27
5	17.6	29.89	41.6	5.63
6	36.3	42.56	44	3.15
7	28.5	39.57	44	4.39
8	17.3	29.57	42.6	5.52
9	26.3	41.7	44	2.92
10	17.1	29.47	42.4	5.5
11	17.7	40.75	44	2.75
12	27.7	39.75	44	4.45

Table 3 summarised the session times completed

within the correct time (in [tu]). We showed the smallest, average, and highest session times (in [tu]) and the average value of delay in the network. During the simulation we obtained values greater by 0.1 [tu] from T_s^{min} and equal to T_s^{max} .

We observed that only one execution ended below the correct interval. Also, we observed that for sessions ended with *!max* status the average session time ranged between 48.65 [tu] and 57.85 [tu]. The average delay in the network in these sessions was between 5.09 [tu] and 6.15 [tu]. The results showed how the delay value effect on session time, the session correctness and Intruder activities. Also, they showed the effect of additional steps executed by Intruder while the time conditions remain maintained.

3.2.2 Normal Probability Distribution

For some situations, in our tool, we implemented specific activities. In some cases, we drew out of range delays in the network values. It was most visible during simulations with a normal probability distribution. In this phase of simulations only honest execution have test series with *correct* and *!max* statuses. Rest of executions ended with an error. None of the test series ended with status *!min*.

Table 4: Summary of NSPK simulations with normal distribution which was ended in correct interval in [tu].

No	$Min(T_s)$	$Avg(T_s)$	$Max(T_s)$	$Avg(D)$
1	19.78	35.71	43.98	7.37
2	19.85	36.62	43.9	7.65

Table 4 summarised session times completed in the correct time, including the smallest, average, and highest session times and the average value of delay in the network.

These results showed delays in the network values influence on the protocol step and whole protocol. Even using high delay value time constraints can remain met. Wrongly chosen time parameters with fulfilled time conditions may affect the correctness of the honest users communication.

3.3 Encryption and Decryption Simulations for NSPK Protocol

During the last stage of the study, we carried out the next simulations of the protocols executions according to assumptions mentioned earlier. We generated encryption and decryption times according to a uniform probability distribution (from 1 to 10 [tu]). Random values of encryption and decryption times have resulted in a different correct interval for each test se-

ries. We observed that constant delays in the network and random encryption values cause a significant reduction in the chance of completing a protocol session before or after the correct protocol session time.

3.4 Timed Analysis for WLP Protocol

For WLP we assumed that Intruder impersonates only for honest users, not for the trusted server. Please note that in the fourth protocol step user *B* sends a message to server encrypting it with his symmetric key. In the case of impersonating Intruder, this is the user key whose Intruder impersonates. The honest user can encrypt the message and continue the protocol. However, Intruder, who does not have an appropriate cryptographic key, cannot acquire knowledge about the cryptogram to continue the protocol.

The research involved an analysis of times influence on attack possibility. However, due to this protocol structure, it turned out that with the current delay in the network value set to lower limit of delays range, all real executions ended in correct time or before the minimal session time. A similar situation occurred in the case of checking encryption time influence on executions correctness. Regardless of encryption and decryption times value, all executions ended in the correct session time.

3.5 Delays Simulations for WLP Protocol

For WLP simulation we made following assumptions $T_e = T_d = 2$ [tu], $T_g = 2$ [tu], delays range - 1 - 10 [tu]. We randomised current delays in the network values according to earlier selected probability distributions.

Next, we calculated lifetimes values: $T_1^{out} = 72$ [tu], $T_2^{out} = 62$ [tu], $T_3^{out} = 42$ [tu], $T_4^{out} = 28$ [tu], $T_5^{out} = 14$ [tu].

We set a minimal session time to 27 [tu] and maximal session time to 72 [tu]. We did not analyse executions marked as impossible to carry out. We present simulations experimental results on exponential and Cauchy's probability distribution example.

3.5.1 Exponential Probability Distribution

During the simulation of WLP executions using the exponential probability distribution, we observed that the all test series ended in the correct session time.

Table 5 summarised session times completed in the correct time (in [tu]). We showed the smallest, average, and highest session times (in [tu]) and the average value of delay in the network. At all test series, the average value of delay in the network was equal to

Table 5: Summary of WLP simulations with exponential distribution which ended in correct interval in [tu].

No	$Min(T_s)$	$Avg(T_s)$	$Max(T_s)$	$Avg(D)$
1	27.07	27.5	28.35	1.1
2	27.09	27.51	28.91	1.1
3	27.04	27.52	28.37	1.1
4	27.06	27.52	28.46	1.1
5	27.09	27.5	28.54	1.1
6	29.15	29.71	30.91	1.1
7	29.21	29.72	30.88	1.1
8	29.12	29.72	30.63	1.1

1.1 [tu]. None of the session times exceeded 31 [tu]. Obtained results showed that using relatively low delay, the Intruder session time may end in the correct session time. These results showed how often honest users can be compromised by an Intruder.

3.5.2 Cauchy's Probability Distribution

The randomisation of out-of-range values, mentioned earlier, was particularly evident in the case of studies using Cauchy's probability distribution. Due to the nature of this probability distribution, a significant number of out-of-range values have been observed. Also, a large number of test series ended with error have been observed. None of the test series ended with status !min. We observed that for three executions almost all test series were ended with an error.

Table 6: Summary of WLP simulations with Cauchy's probability distribution which ended in correct interval in [tu].

No	$Min(T_s)$	$Avg(T_s)$	$Max(T_s)$	$Avg(D)$
1	50.35	63.41	71.77	8.32
2	48.87	64.49	71.74	8.56
3	45.75	63.89	71.20	8.46
4	46.59	63.45	71.93	8.34
5	48.75	64.28	71.99	8.54

Table 6 summarised session times completed in the correct time (in [tu]). We showed the smallest, average, and highest session times (in [tu]) and the average value of delay in the network. These results showed delays in the network values influence on the protocol step and whole protocol. Even the delay value is high, time constraints can remain met. Badly selected time parameters can also affect the correctness of honest users communication, while the time conditions are met.

3.6 Encryption and Decryption Simulations for WLP Protocol

During the last stage of WLP studies, simulations of these protocols executions were carried out. Constant values of delay in the network (2, 4, 6, 8 and 10 [tu]) and random values of encryption and decryption times were taken into account. For each delay in the network value, a thousand test series were performed. Encryption and decryption times were generated according to a uniform probability distribution (from 1 to 10 [tu]). Random values of encryption and decryption times have resulted in a different correct interval for each test series.

During analysis of encryption and decryption simulations for this protocol, it can be observed that for the stage with $D = 4$ [tu] all test series ended in correct session time. Also, only four test series ended below T_s^{min} for stage with $D = 2$ [tu]. For the rest stages, none of the test series ended below T_s^{min} . Except for stage with $D = 10$ [tu], for all stages, the number of !max sessions was bigger than the number of wrong sessions. This results showed how significant influence on completing a protocol session before or after correct session time have encryption and decryption times.

4 CONCLUSION

Time plays an important role in the private and professional life of every person. Its special aspects were discovered through Internet communication. Information transmitted by electronic connections should reach the recipient promptly. Data should also remain safe and intact. Internet communication is secured by special security protocols. Unfortunately, the protocols are exposed to the wicked person called Intruder. For this reason, it is important to model and verify the security protocols to avoid unwanted situations.

The paper presented the methodology of testing timed security protocols including delays in the network. The research was based on a formal model and computational structure. Thanks to this, it was possible to calculate the time of proper protocol execution including different time parameters. The implemented tool enabled automatic verification of timed security protocols.

The study showed the influence of time parameters on the protocol. Intruder actions depended on the values of the time parameters. The higher value of the upper limit of delay in the network may increase Intruder opportunities to gain knowledge and allow the Intruder to carry out the attack. A similar situation

occurred with encryption and decryption times. More time-consuming encryption with constant delays in the network has made Intruder chances of deceiving honest users higher.

REFERENCES

- Alur, R. and Dill, D. L. (1994). A theory of timed automata. *Theoretical Computer Science*, 126:183–235.
- Basin, D. A., Cremers, C., and Meadows, C. A. (2018). Model checking security protocols. In *Handbook of Model Checking.*, pages 727–762. Springer.
- Bella, G. and Paulson, L. C. (1997). Using Isabelle to prove properties of the Kerberos authentication system. In *DIMACS Workshop on Design and Formal Verification of Security Protocols*.
- Burrows, M., Abadi, M., and Needham, R. (1990). A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36.
- Chadha, R., Sistla, A. P., and Viswanathan, M. (2017). Verification of randomized security protocols. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12.
- Corin, R., Etalle, S., Hartel, P. H., and Mader, A. (2004). Timed model checking of security protocols. In *Proceedings of the 2004 ACM Workshop on Formal Methods in Security Engineering, FMSE '04*. ACM.
- Corin, R., Etalle, S., Hartel, P. H., and Mader, A. (2007). Timed analysis of security protocols. *J. Comput. Secur.*, 15(6).
- Dolev, D. and Yao, A. C. (1981). On the security of public key protocols. Technical report, Stanford, CA, USA.
- Čibej Uroš, Luka, F., and Jurij, M. (2019). A symmetry-breaking node equivalence for pruning the search space in backtracking algorithms. *Symmetry*, 11(10).
- Jakubowska, G. and Penczek, W. (2007a). Is your security protocol on time? In *Proceedings of the 2007 International Conference on Fundamentals of Software Engineering, FSEN'07*, pages 65–80, Berlin, Heidelberg. Springer-Verlag.
- Jakubowska, G. and Penczek, W. (2007b). Modelling and checking timed authentication of security protocols. *Fundam. Inform.*, 79(3-4):363–378.
- Kurkowski, M. (2013). *Formalne metody weryfikacji własności protokołów zabezpieczających w sieciach komputerowych*. Informatyka - Akademicka Oficyna Wydawnicza EXIT. Akademicka Oficyna Wydawnicza Exit.
- Needham, R. M. and Schroeder, M. D. (1978). Using encryption for authentication in large networks of computers. *Commun. ACM*, 21(12):993–999.
- Nigam, V., Talcott, C. L., and Urquiza, A. A. (2016). Towards the automated verification of cyber-physical security protocols: Bounding the number of timed intruders. *CoRR*, abs/1605.08563.
- Perháč, J., Bilanová, Z., Steingartner, W., and Piatko, J. (2019). Benchmark of software developed in different component models. In *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 245–250.
- Siedlecka-Lamch, O., Kurkowski, M., and Piatkowski, J. (2016). Probabilistic model checking of security protocols without perfect cryptography assumption. In Gaj, P., Kwiecień, A., and Stera, P., editors, *Computer Networks*, pages 107–117, Cham. Springer International Publishing.
- Szymoniak, S. (2021). Security protocols analysis including various time parameters. *Mathematical Biosciences and Engineering*, 18(2):1136–1153.
- Szymoniak, S., Kurkowski, M., and Piatkowski, J. (2015). Timed models of security protocols including delays in the network. *Journal of Applied Mathematics and Computational Mechanics*, 14(3):127–139.
- Szymoniak, S., Siedlecka-Lamch, O., and Kurkowski, M. (2017). *Timed Analysis of Security Protocols*, pages 53–63. Springer International Publishing, Cham.