IMPACT OF AN ATTACK ON A NETWORK EXECUTING DISTRIBUTED COMPUTING

Martin KENYERES[1], Jozef KENYERES[2]

In this paper, we examine the effect of an attack on a network executing the distributed algorithm average consensus. We assume that only one attack is committed during the process of reaching the convergence. We examine how intensively an attack affects features of the average consensus algorithm. We define two parameters: the number of additional iterations and the change of the final value. Then we examine how they are changing when the following parameters are being changed: the number of iteration during which an attack is committed, an attacker's internal value, the initial values, the range of the initial values. At the end, we examine how the position of the attacked element affects the impact of an attack on the network. Firstly, we perform experiments in an example network containing 24 densely placed elements and whose topology is randomly generated. We decide to apply TDMA as a method to share a transmission medium. Then we execute another experiment in which we examine how intensive an attack is when the size of a network changes. This paper is motivated by the publications where a potential failure of a node such as a dead node, a misbehaving node etc. significantly affects the whole computation process. In contrast to the previous works, we assume the presence of an attacker who is aware of the weaknesses of distributed computing.

Keywords: parallel distributed computing, average consensus, wireless sensor networks

1. Introduction

In this publication, we focus our attention on examining negatives effects caused by unauthorized interference in the distributed systems. In the practical application, the scenario which we focused on may happen for example in the wireless sensor networks where because of the simplicity of used devices, network security mechanisms are very limited [5], [6], [7]. They contain limited battery sources, which causes adding additional bits not to be the best solution to protect them [8]. In [9], the authors showed how a failure within a network affected the whole computation process of the average consensus algorithm. Due to the simplicity of WSN devices, communication within these networks often misses any cryptographic security. Thus, they pose an easy target for potential cybernetic aggressors. The average consensus algorithm is a representative of the

[1] Brno University of Technology, Czech Republic, e-mail: kenyeres@phd.feec.vutbr.cz
[2] Zelisko GmbH, Austria, e-mail: Jozef.Kenyeres@knorr-bremse.com

consensus-based protocols and is chosen as the goal of our interest within this paper for its simplicity. Thus, the motivation of this work is to show how vulnerable the distributed systems whose functionality is based on mutual cooperation among the entities without implementation of cryptographic mechanisms are. In contrast to the failures in [9], an attacker is assumed to be aware of distributed computing weaknesses and exploits them.

We focused our attention on the algorithm called "average consensus" presented in [10]. Average consensus is a distributed iterative algorithm which is found simple and whose output represents the average value counted from the initial values of the elements forming a network. We were inspired by [9], where the authors presented how vulnerable the algorithm is and [11], where degradation of the final results is analyzed. The authors focused their attention on how a network failure may affect the network. We assumed that an attack is performed by an intelligent being who knows the weak features of distributed computing and utilize this knowledge for their advantage.

In the first section of this paper, we introduced the Average consensus algorithm, our assumptions and suggested the mathematical tool which we used in this paper. Then we described the committed attacks and explained how mathematical tool describing the distributed computing behavior was changed when we consider the presence of an attacker. In the experimental section, we simulated the functionality of a network and within the simulations, we committed an attack many times, always with different conditions. The obtained results were compared together in order to analyze the effect of an attack on a network executing distributed computing and derived the theoretical conclusions.

2. Average consensus

The functionality of distributed algorithms is referred to distributed systems which consist of separated subsystems whose correlative communication is significantly limited [12]. Despite it, they are able to fulfill particular tasks. In this paper, we assume that the network executes the distributed algorithm A network executing Average consensus is such a system which achieves the desired results in a distributed manner. The elements are able to achieve the consensus according to the values sent by their adjacent neighbors [13]. In this section, we present algorithm and the assumptions defining our experiment.

Discrete iterations instead of the continuous time

We assume that each element of a graph represents one device of a network. For example, a node forming a wireless sensor network could be such an element. It is required to emphasize the method of access used in a network to an access transport medium. We were inspired by [14], where authors presented

TDMA model of an access to a shared medium. Choosing TDMA [15] completely prevents potential collisions or minimizes their probability so significantly that it could be considered to be zero. In this section, we list at least the main features of this access method. TDMA is an access mechanism where every element in a network knows the time slot during which it is allowed to send a message. During the other slots, it can only listen to the network traffic and receive messages sent by other elements. More details about this method used in WSN could be found in [16]. It is important to mention that this scheme synchronizes the elements.

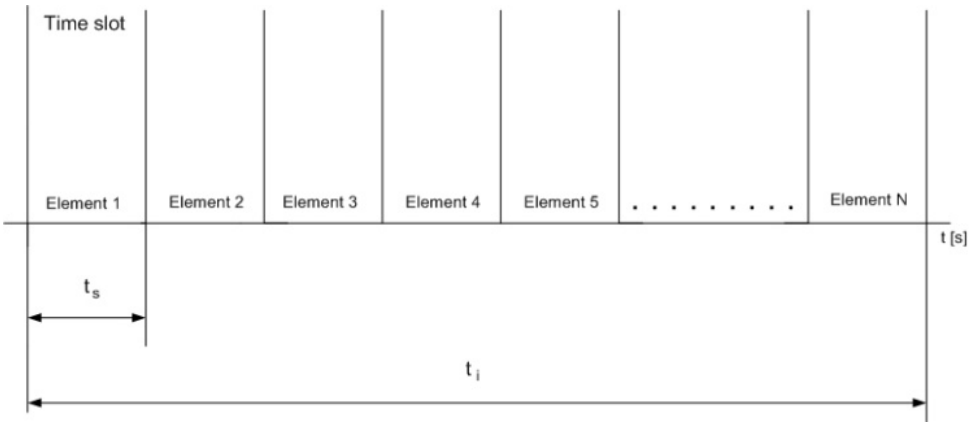

Fig 1.Dividing the time into time slots and assigning them to each element forming a network

As the Average consensus is an iterative algorithm, the TDMA scheme fits well.

$$t_i = N * t_s \tag{1}$$

In our simulation, we substituted the time interval t_i with a discreet iteration i. This was possible because we are not interested in what happens during the time interval t_i, but only in the final state of an element after it elapses. So, the state of every element after t_i is what we use in our calculations. This decision does not affect obtained results. We defined the slot distribution in (1). Here, t_s defines the slots' size and *N* the number of elements.

No collision cause by the attacker's effort

We also assume that an attacker utilizes the time slot assigned to an element which is attacked. The possibility of a collision between the attacked element's message and an attacker's is ignored and so, its probability equals to zero.

As it has already been mentioned, the average consensus algorithm is a distributed algorithm whose output is determined as the average of the initial values at elements. The average is counted by distributed manner in such a way that internal value of every element iteratively converges to the average value. Every element is assigned an initial value which is unknown by other elements forming a network. An element is aware of other elements' values only if it is sent it. We proposed a set of mathematical tools to describe our experiments. We defined a two-dimensional field containing both initial values of every element and the updated in every iteration. It is defined as follows:

$$x(a,i) \text{ for } a = 1,2 \ldots . N \text{ , } i = 1,2 \ldots i_1 \tag{2}$$

Here the variable a expresses the identification number of the elements (*ID*) and the value of i represents an iteration. The value of i equaling to 1 represents the initial state. N is the amount of the elements forming a network and the parameter i_1 is the last iteration. *ID* is unique among the elements and varies from 1 to *N*.

For example $x(2,2) = 10$ means that the element number 2 in the iteration number 2 has its value equaled to 10. The initial values of the elements are deterministically set. We decided to use following formula to set those values:

$$x(a,i) = k * a \text{ for } a = 1,2 \ldots . i \text{ , } i = 1 \tag{3}$$

where k is a constant which we are going to vary for some experiments in order to show the algorithm's behavior when an attack infected the network.

In this section, we explain the algorithm's functionality and how reaches the consensus. As the first, we would like to explain our experimental scenario. Every element is aware of its ID only. Thus, it is able to determine its initial value x as well as to find out the time intervals during which it is allowed to send a message. Information of neither a network's topology nor the amount of elements in a network is available to an element nor is it able to acquire them (it **is**, but that process would be very ineffective and long-lasting and so, we leave this option out). It can only either listen to a traffic nearby it or receive messages sent by elements in its range. A received message is processed and the current value is calculated as follows:

$$x(a,i) = x(a,i-1) + * \sum_{b=1}^{N} \{[x(b,i-1) - x(a,i-1)]$$

$$* A(a, b, i-1)\} \text{ for } i = 2,3\ldots..i_1 \quad (4)$$

Here A is a three-dimensional field which serves as the adjacent matrix (see [17], [18]) and so, determines neighborhood relations. The size of it is determined by the number of elements forming a graph. This field is defined as follows:

$$\begin{cases} A(a,b,i) = 1 & A(b,a,i) = 1 \quad (5) \\ A(a,b,i) = 0 & A(b,a,i) = 0 \quad (6) \end{cases}$$

For the formulas (5) and (6), these conditions have to be fulfilled:

(5) if a and b are adjacent during i iteration and if $a \neq b$
(6) f a and b are not adjacent during i iteration and if $a \neq b$
(6) if $a = b$ for i $= 1,2 \ldots . i_1$

In (4), is a parameter to determine the speed of the algorithm's convergence and it varies according to [14] as follows:

$$0 < \leq \frac{1}{\max\{|N_i|\}} \quad (7)$$

The label $\max\{|N_i|\}$ represents the maximal number of neighbors in a network.

As mentioned, the average consensus is considered to have reached the consensus when the following statement is valid:

$$x(a, i_l) = \bar{x} = \frac{1}{N}\sum_{b=1}^{N} x(b, i) \text{ for } i = 1,\ a = 1,2 \ldots . N \quad (8)$$

The parameter i equals to 1 because the final state is counted from the initial values.

The formula (8) is unreachable because the behavior of the algorithm is defined according to:

$$\lim_{i\to\infty} x(a, i) = \frac{1}{N}\sum_{b=1}^{N} x(b, j) \text{ for } j = 1 \quad (9)$$

Therefore, we had to define some parameter to determine the state of the convergence. We defined б as follows:

$$б = \left|\frac{\sum_{a=1}^{N} x_a(i)}{N}\right| * \alpha \text{ for } i = 1 \quad (10)$$

We labeled the parameter б as the condition of convergence and determines when the algorithm is considered to have reached the consensus and

the process of achieving it is found finished and therefore, it is not more requested that the algorithm continue. Its value, as it can be seen in the formula (10), is determined by the parameter α, the number of the elements as well as the initial values of them. α is labeled the accuracy of the convergence. The smaller value it is of, the less the final value at the elements differs from the real average and the more iterations are required for the algorithm to achieve the consensus.

Then we can define the event of convergence as follows:

$$max\big(x(a,i)\big) - \min\big(x(a,i)\big) < б \tag{11}$$

If this condition holds i_1 equals to i. It is a value of which no element is aware at the beginning of a simulation process and whose value is affected by several factors. For instance, a network's topology, the number of the elements, element density etc. The parameter i_1 also determines how many iterations are necessary for a network to converge.

3. Infiltration into a network

In this section, we are going to describe the features of an attack in a network. We assume that the attacker's effort is always successful. We also assume that neighborly relations of an attacker will be same as an attacked element's and an attacker will become a part of a network. In contrast to the other elements, an attacker cannot receive any message. Therefore, we have edited the adjacent field A in such a way that we have increased its size by the amount of the attacker (in our case 1). Let i_λ be the iteration during which the attacker commits their attack and λ be the label of an attacked element. Then the following statements are valid:

$$A(N+1,b,i_\lambda) = A(\lambda,b,i_\lambda)$$

$$A(a,N+1,i_\lambda) = A(b,\lambda,i_\lambda) \tag{12}$$

$$for\ a,b = 1,2 \ldots N\ if\ a,b \neq \lambda$$

$$A(N+1,b,i_\lambda) = 1\ \ A(a,N+1,i_\lambda) = 1\ \ if\ a,b = \lambda \tag{13}$$

$$A(N+1,N+1,i_\lambda) = 0 \tag{14}$$

and

$$A(N+1,b,i)\ \ A(a,N+1,i) = 0$$

$$for\ a,b = 1,2\ \ldots N\ if\ i \neq i_\lambda \tag{15}$$

We also have to expand the field containing *x* by 1 and put the attacker's value into the new position for $i = i_\lambda$. The attacker is a part of a network and affects counted values with their value; therefore, the formula to count the current value has to be changed as well:

$$x(a,i) = x(a,i-1) + * \sum_{b=1}^{N+1}\{[x(b,i-1) - x(a,i-1)] * A(a,b,i-1)\} \quad (16)$$

4. Experiments

In order to perform intended experiments, we have to generate a network with an appropriate topology on which we will execute our experiments. We decided to generate a random topology consisting of 24 elements. We used the generator described in [19] to create a network and utilized theoretical knowledge from [20]. We assumed the network whose topology would be interconnected because performing a distributed algorithm on such one whose elements are not connected to one another has no sense. An interconnected network is such a structure in which every element can provide information about itself, which is realized by sequent resending that information by other elements forming the same network. This information is involved in an upgraded state of adjacent elements. If a network is not interconnected, separated parts would converge to different values and the average counted from the whole topology would never be achieved. The generator [19] is for generating a network according to the parameters: the communication range of an element, the size of the area covered by elements and the amount of elements, which are set by experimenters. Our goal was setting the mentioned parameters with such values that a generated network could be classified as a network with a dense topology. We chose a small area with a high amount of elements of a relatively long communication range, which resulted in generating a network with dense topology. The results are depicted in Fig. 2.

In our first experiments, we examine $\Delta(k, x_\lambda)$. We define Δ as a difference in the number of the iterations required to reach the consensus without the attack and with it. We examine the effect of an attack on Δ in relation to the parameter k. We suppose that the iteration number, when an attack was executed is constant. The parameter x_λ defines the value of an attack as follows:

$$x(N+1, i_\lambda) = x_\lambda * \sum_{a=1}^{N} x(a,i) \text{ for } i = 1 \quad (17)$$

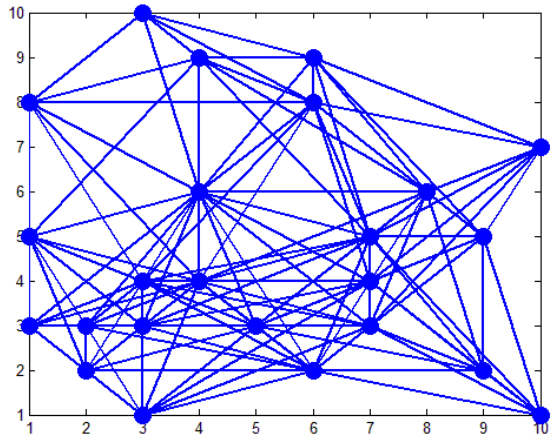

Fig. 2. A random topology consisting of 24 nodes

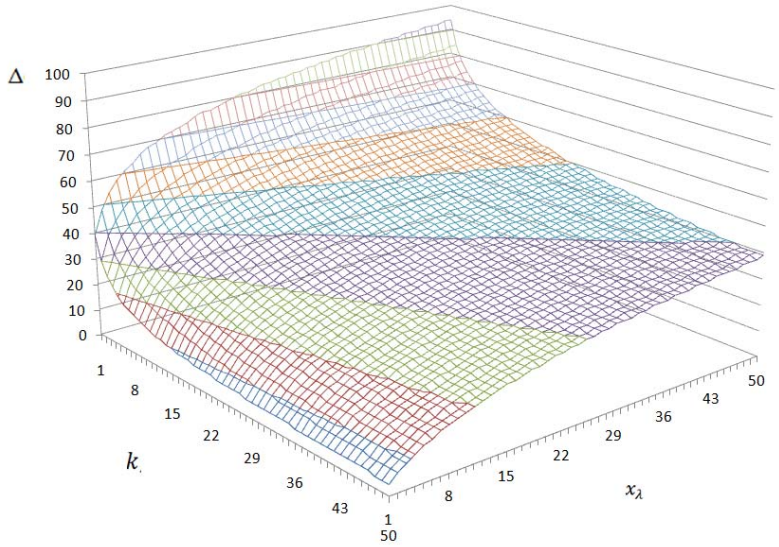

Fig. 3. A network illustrating the effect of increasing attack's value on the number of iterations

As we see the results (Fig. 3), regardless k, the number of iterations was increasing as the parameter x_λ was of a higher value. The speed of the function's growth was being decreased when x_λ reached higher values. For the high values, the change of the function was very small. These relations evoked the logarithm function. It means that increasing value of the attack caused more damage. When

the k parameter grew, the function was declining. For small k, the function was rapidly decreasing as k was growing. For a bigger k, the growth was consequently decreased. The behavior of the function evoked a negative logarithm function. It means the less values at elements differ from one another, the less resistant the network was.

In the second simulation, we examine $\Delta(k, i_\lambda)$ and assume that the internal value of the attacker was constant for every instance.

Regardless k, we can see in the picture Fig. 4, that Δ is increasing when i_λ's value is being increased. Then we see a point after which the function's values are of zero value. The reason is that the attack was committed after the iteration process had been finished and so, the attack's activity did not affect the network. The contribution of this simulation is the later the attack is performed, the more damages it causes. But if it is committed too late (after the network has converged), it does not affect a network at all. This function evokes a linear function. When the parameter k is being changed, we can notice similar behavior as in the previous simulation. In this case, we can also see k does not affect the number of the iterations if no attack occurs.

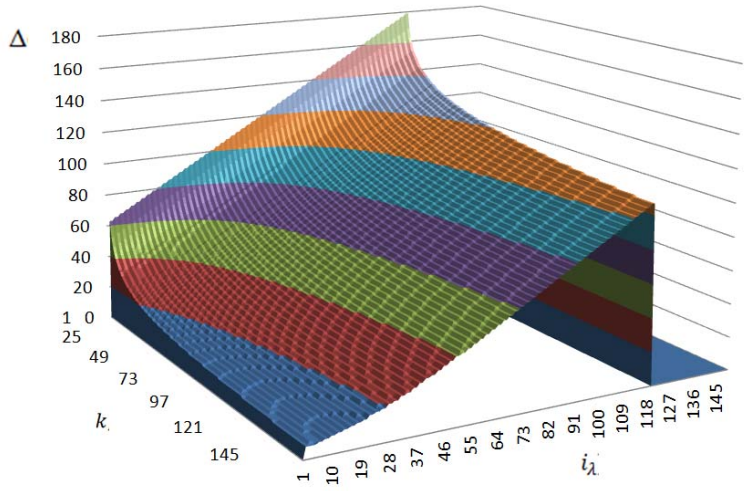

Fig. 4. A graph illustrating the effect of the increasing number of iteration when the attack occurred

In this experiment (Fig. 5), we examined how an attack affects the value of the convergence value to which the network structure converged (Δ_{av}). We concentrated our effort on examining $\Delta_{av}(i_\lambda, x_\lambda)$. The value of the parameter k was 1. The parameter Δ_{av} represents the difference of the value to which the network converges when no attack happens and when an attack does. We changed

both i_λ and x_λ and found out that the higher the attack's value was, the bigger Δ_{av} was. The function grows linearly. The parameter i_λ has no effect on Δ_{av}. The function is constant.

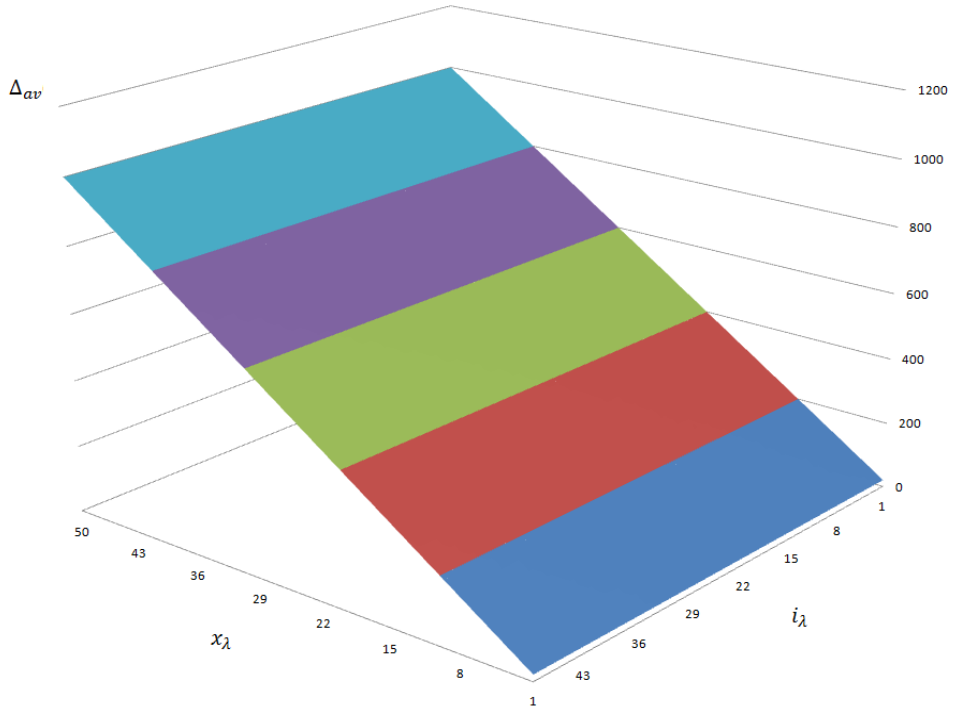

Fig. 5. A graph illustrating the increase of the convergence value caused by the attack

Then we performed the fourth (Fig. 6) where we examined the influence λ on the algorithm's behavior. We statically set these parameters on the values, they equalled to k = 10, $i_\lambda = 20$ $x_\lambda = 10$. We examined how the number of the iterations would be changed when the number of an attacked element differed. As we can see in the Fig. 6, the number of iterations was not affected significantly, even though the topology was random.

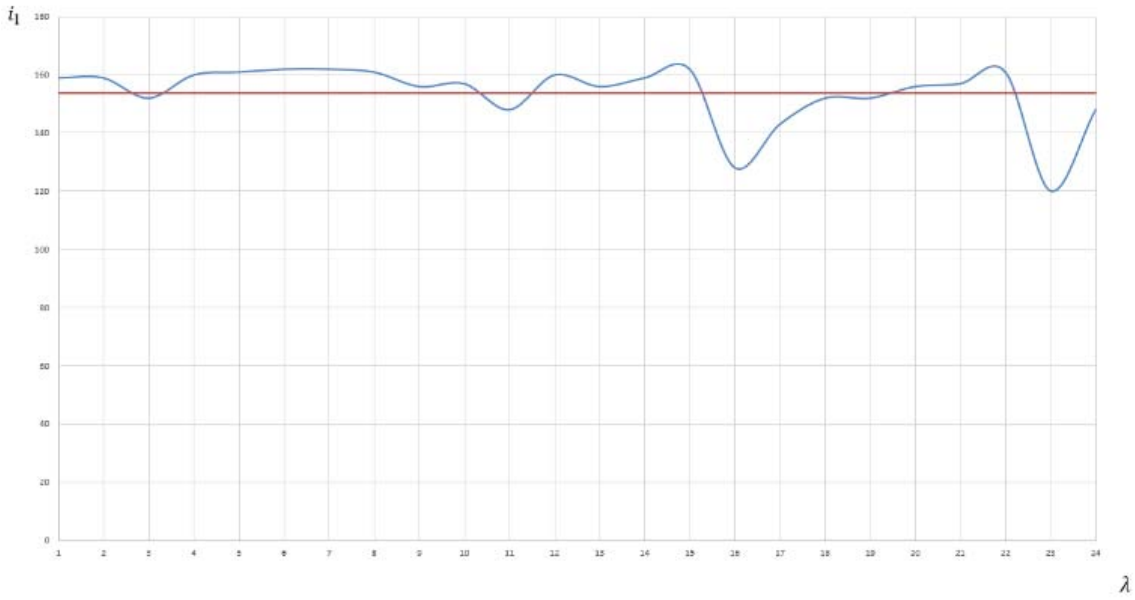

Fig. 6. A graph showing the effect of how change of an attacked element influences the amount of iterations

In order to obtain general results, the experiments were repeated for the network with the following sizes: 20, 30, 40, 50, 60, 70, 80, 90 and 100 nodes. We generated 10 networks for each size and therefore, executed 900 experiments. In Tab.1, the results for all the network with the size of 20 nodes are depicted. In the Tab.2, Tab.3 and Tab.4, the results for each size are shown. In order to ensure the transparency of the paper, we have depicted only the network whose convergence rate was most significantly affected by the attack. In Fig. 7 and Fig.8, the results for the $x_\lambda = 35$ are shown.

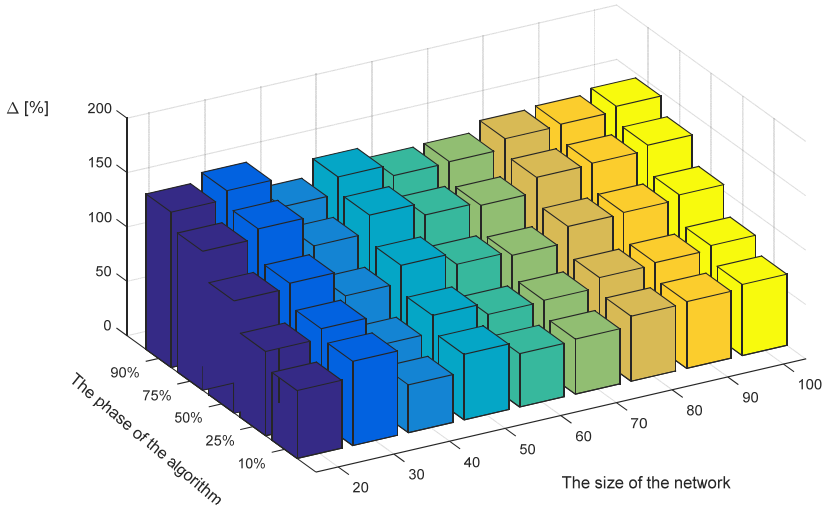

Fig. 7. The percentage growth of the iterations

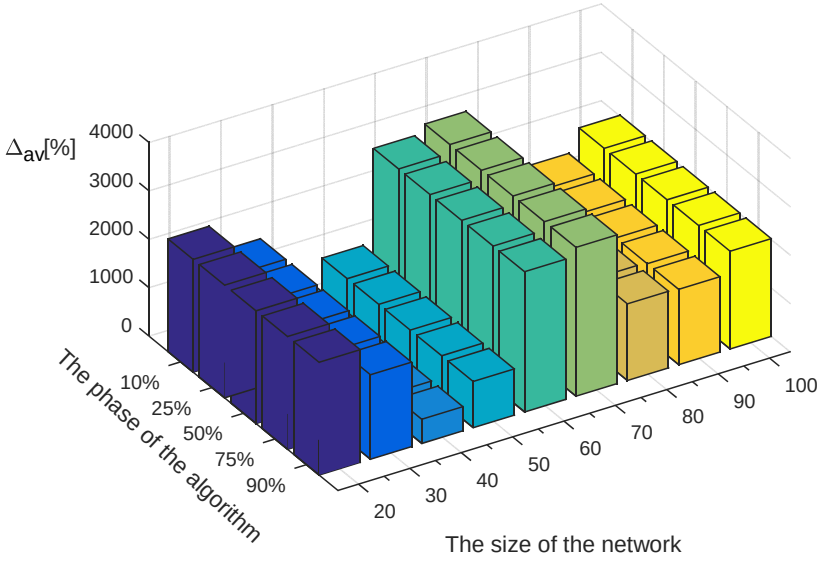

Fig. 8. The percentage growth of the final values

Table 1

The results for the networks with the size of 20 nodes

		10%		25%		50%		75%		90%	
		Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]
#1	5	41.35	239.87	56.39	239.87	82.7	239.87	107.51	239.87	122.55	239.87
	20	56.39	1329.48	71.42	1329.48	96.99	1329.48	121.80	1329.48	136.84	1329.48
	35	62.40	2329.08	77.44	2329.08	103.05	2329.08	127.81	2329.08	142.85	2329.08
	50	66.16	3328.68	81.20	3328.68	106.76	3328.68	131.57	3328.68	146.61	3328.68
#2	5	36.02	165.1	50.93	165.1	165.10	165.1	101.24	165.1	116.14	165.1
	20	49.06	664.9	63.97	664.9	664.90	664.9	114.28	664.9	129.19	664.9
	35	54.03	1164.7	68.94	1164.7	1164.70	1164.7	119.87	1164.7	134.78	1164.7
	50	57.76	1664.5	72.67	1664.5	1664.50	1664.5	122.98	1664.5	137.88	1664.5
#3	5	30.08	412.34	45.42	412.34	412.34	412.34	95.28	412.34	110.32	412.34
	20	43.95	1661.84	58.99	1661.84	1661.84	1661.84	108.84	1661.84	123.89	1661.84
	35	49.26	2911.34	64.30	2911.34	2911.34	2911.34	114.15	2911.34	129.20	2911.34
	50	52.80	4160.84	67.84	4160.84	4160.84	4160.84	117.69	4160.84	132.74	4160.84
#4	5	32.87	423.96	47.68	423.96	423.96	423.96	97.68	423.96	112.73	423.96
	20	45.37	1079.16	60.41	1079.16	1709.16	1079.16	110.41	1079.16	125.46	1079.16
	35	50.69	2994.36	65.74	2994.36	2994.36	2994.36	115.74	2994.36	130.78	2994.36
	50	53.93	4279.56	68.98	4279.56	4279.56	4279.56	118.98	4279.56	134.02	4279.56
#5	5	32.15	288.69	47.45	288.69	288.69	288.69	97.45	288.69	112.54	288.69
	20	46.07	1163.34	61.17	1163.34	1163.34	1163.34	111.17	1163.34	136.27	1163.34
	35	51.56	2037.99	66.86	2037.99	2037.99	2037.99	116.66	2037.99	131.76	2037.99
	50	55.09	2912.64	70.39	2912.64	2912.64	2912.64	120.19	2912.64	135.29	2912.64
#6	5	33.33	182.56	48.22	182.56	182.56	182.56	98.58	182.56	113.47	182.56
	20	46.09	733.89	60.99	733.89	733.90	733.89	111.34	733.89	126.24	733.89
	35	51.06	1284.3	66.66	1284.3	1284.30	1284.3	117.02	1284.3	131.91	1284.3
	50	54.60	1823.71	69.50	1823.71	1823.71	1823.71	119.85	1823.71	134.75	1823.71
#7	5	32.28	308.94	47.53	308.94	308.94	308.94	97.35	308.94	112.12	308.94
	20	45.21	1241.93	60.25	1241.93	1241.93	1241.93	110.19	1241.93	125.25	1241.93
	35	49.98	2173.39	65.28	2173.39	2173.39	2173.39	115.25	2173.39	130.25	2173.39
	50	53.45	3086.21	67.97	3086.21	3086.21	3086.21	118.59	3086.21	133.95	3086.21
#8	5	33.67	125.56	48.53	125.56	125.56	125.56	98.89	125.56	113.85	125.56
	20	46.23	504.75	61.24	504.75	504.75	504.75	111.59	504.75	126.48	504.75
	35	51.45	883.31	66.87	883.31	883.31	883.31	117.45	883.31	132.19	883.31
	50	54.92	1254.30	69.89	1254.30	1254.30	1254.30	120.05	1254.30	135.05	1254.30
#9	5	29.85	279.19	45.12	279.19	279.19	279.19	95.05	279.19	110.12	279.19
	20	43.57	1122.34	58.75	1122.34	1122.34	1122.34	108.45	1122.34	123.55	1122.34
	35	49.06	1964.10	64.05	1964.10	1964.10	1964.10	114.09	1964.10	129.05	1964.10
	50	52.65	2789.02	67.67	2789.02	2789.02	2789.02	117.38	2789.02	132.45	2789.02
#10	5	38.25	311.87	52.05	311.87	311.87	311.87	102.89	311.87	118.35	311.87
	20	51.15	1253.71	65.12	1253.71	1253.71	1253.71	116.21	1253.71	131.25	1253.71
	35	55.98	2194.00	70.25	2194.00	2194.00	2194.00	121.53	2194.00	136.58	2194.00
	50	59.05	3115.48	73.98	3115.48	3115.48	3115.48	124.35	3115.48	139.45	3115.48

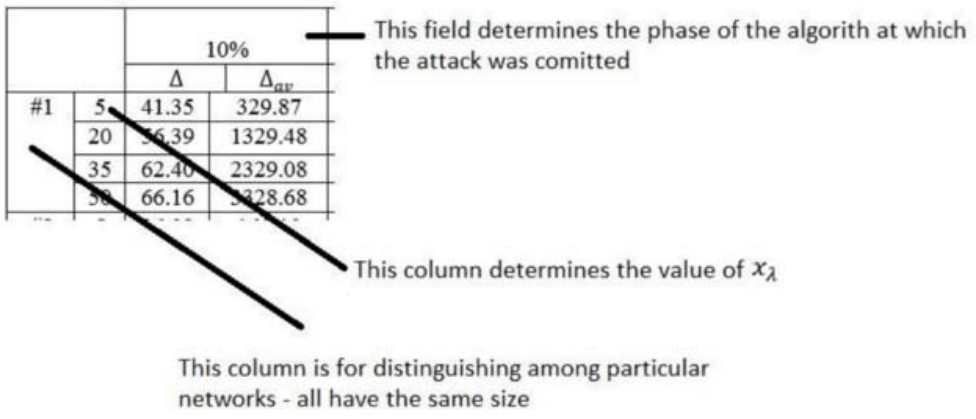

Fig. 9. The legend to the table 1

Table 2

The results for the slowest network with the size of 20,30 and 40 nodes

		20		30		40	
		Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]
10%	5	41.35	239.87	56.39	248.44	27.56	71.13
	20	56.39	1329.48	71.42	997.85	39.49	285.32
	35	62.40	2329.08	77.44	1745.58	44.03	499.52
	50	66.16	3328.68	81.20	2493.54	47.16	713.72
25%	5	56.39	329.87	66.4	248.44	42.90	71.13
	20	71.42	1329.48	80.59	997.85	54.55	285.32
	35	77.44	2329.08	86.58	1745.58	59.09	499.52
	50	81.20	3328.68	90.30	2493.54	62.22	713.72
50%	5	82.70	329.87	88.05	248.44	67.9	71.13
	20	96.99	1329.48	102.23	997.85	79.55	285.32
	35	103.05	2329.08	107.46	1745.58	84.09	499.52
	50	106.76	3328.68	111.19	2493.54	87.22	713.72
75%	5	107.51	329.87	116.41	248.44	92.90	71.13
	20	121.80	1329.48	130.59	997.85	104.55	285.32
	35	127.81	2329.08	136.56	1745.58	109.09	499.52
	50	131.57	3328.68	140.23	2493.55	112.22	713.72
90%	5	122.55	329.87	130.6	248.44	107.95	71.13
	20	136.84	1329.48	144.78	997.85	119.6	285.32
	35	142.85	2329.08	150.75	1745.58	124.15	499.52
	50	146.64	3328.68	154.48	2493.55	127.27	713.72

Table 3

The results for the slowest network with the size of 50,60 and 70 node

		50		60		70	
		Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]
10%	5	41.48	135.99	32.45	412.64	33.97	438.08
	20	54.55	545.45	44.37	1654.64	46.15	1756.14
	35	60.23	954.99	49.01	2896.64	50.64	3074.19
	50	63.64	1364.56	52.32	4138.65	53.85	4392.24
25%	5	55.68	135.99	47.68	412.64	48.72	438.08
	20	69.32	545.45	59.6	1654.64	60.90	1756.14
	35	75.12	954.99	64.24	2896.64	65.38	3074.19
	50	78.41	1364.56	67.55	4138.65	62.22	4392.24
50%	5	80.68	135.99	72.85	412.64	68.59	438.08
	20	94.32	545.45	84.77	1654.64	73.72	1756.14
	35	100.25	954.99	89.4	2896.64	85.90	3074.19
	50	103.41	1364.56	92.72	4138.65	90.38	4392.24
75%	5	105.68	135.99	97.35	412.64	93.59	438.08
	20	119.32	545.45	109.27	1654.64	98.72	1756.14
	35	125.56	954.99	113.91	2896.64	110.9	3074.19
	50	128.41	1364.56	117.22	4138.65	115.38	4392.24
90%	5	120.45	135.99	112.58	412.64	113.46	438.08
	20	134.09	545.45	124.5	1654.64	125.64	1756.14
	35	139.77	954.99	129.14	2896.64	130.13	3074.19
	50	143.19	1364.56	132.45	4138.65	133.3	4392.24

Table 4

The results for the slowest network with the size of 80,90 and 100 node

		80		90		100	
		Δ [%]	Δav[%]	Δ [%]	Δav[%]	Δ [%]	Δav[%]
10%	5	42.34	225.54	44.09	221.90	48.51	288.01
	20	54.74	903.84	56.69	889.10	60.40	1153.81
	35	59.85	1582.14	61.42	1556.3	65.35	2019.61
	50	62.77	2260.44	64.57	2223.51	69.31	2885.41
25%	5	56.96	225.54	59.06	221.90	63.37	288.01
	20	69.34	903.84	71.65	889.10	75.25	1153.81
	35	74.45	1582.14	76.38	1556.30	80.20	2019.61
	50	77.37	2260.44	79.53	2223.51	84.16	2885.41
50%	5	82.48	225.54	84.25	221.90	89.11	288.01
	20	94.89	903.84	96.85	889.10	100.99	1153.81
	35	100.54	1582.14	101.57	1556.3	105.94	2019.61
	50	102.92	2260.44	104.71	2223.51	109.94	2885.41
75%	5	107.3	225.54	108.66	221.90	113.86	288.01
	20	119.71	903.84	121.26	889.10	125.74	1153.81
	35	124.82	1582.14	125.98	1556.30	130.69	2019.61
	50	127.74	2260.44	129.13	2223.51	134.65	2885.41
90%	5	121.90	225.54	123.62	221.90	128.71	288.01
	20	134.31	903.84	136.22	889.10	140.59	1153.81
	35	139.42	1582.14	140.94	1556.30	145.54	2019.61
	50	142.34	2260.44	144.09	2223.51	149.52	2885.41

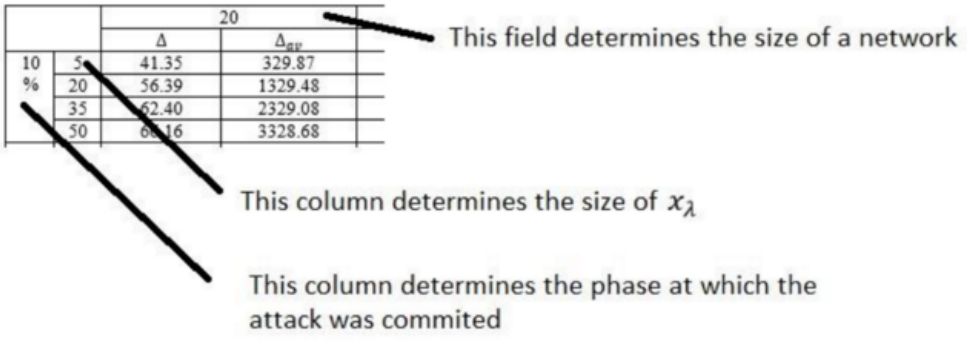

Fig. 10. The legend to the Table 2, 3 and 4

As we can see from the results shown in Tab.1, the growth of i_λ causes a linear growth of Δ . Also the growth of x_λ results in a growth of Δ, but the rate of the growth is being decreased as x_λ is reaching higher values. The parameter Δ_{av} grows as x_λ is being increased. The parameter i_λ has the negligible impact on this parameter. From Tab.2, Tab. 3 and Tab. 4, we can see the similar behavior. The size and the character of the network affect the values of Δ and Δ_{av}, but the behavior of these parameters is same for all the networks. Thus, the experiments prove that the impact of the attack is not directly related to the character and the size of the network. Thus, according to the executed experiments, the character of the attack impact does not change regardless the size of the network. Also, the variance of the initial values and the position of the attacked node result in the similar behavior, but in order to maintain the paper transparency, we have not depicted these results.

5. Conclusion

The aim of this paper is to present how vulnerable the networks performing a cooperative task can be. As the example, we chose WSN executing simple distributed algorithm - Average consensus. We simulated the injection attack and executed the same experiment on the networks with various size and topologies. We examined the impact of the changes of the parameters: the number of iteration during which an attack is committed, an attacker's internal value, the initial values, the range of the initial values and the position of the attacked node on the number of additional iterations and the change of the final value. Especially, we focused our effort on how rapidly just one attack can increase the time necessary for the algorithm to converge. This was achieved by comparison of the result gained with and without the attack on the same networks. We analyzed the algorithm behavior in several experimental scenarios, in which we modified

the occurrence and attacker's inner state together with the dispersion of the initial states. From the achieved results, we can see the $\Delta(x_\lambda)$ evokes a logarithm function and $\Delta(i_\lambda)$ evokes a linear one as long as $i_\lambda < i_1$. Changing k parameter evokes a negative logarithm function. We can see the less spread the values are, the less resistant against the injection the network is. Then we showed how obtained results were affected by both injected value and the iteration during which an attack was committed. We show that the iteration does not affect the final results and an attacker's internal state affects the network linearly. Our last goal was to show how the position of an attacked element affects the number of iterations. We show that it has only a small effect. The same experiments were repeated for the network with various size and the behavior of the attack impact was same for each scenario. For every experiment, we set the accuracy parameter α to 0.0001, which secured a sufficient accuracy. This paper motivates us to focus on techniques to minimize impacts of such attack in our future work. In the case of WSN, the advanced security mechanisms such as encryption are out of scope due to the energy and computation power limitations. Because of this, the security issues need to be considered by a sophisticated algorithm design. The major contribution of this paper is to emphasize, how critical this topic is.

Acknowledgment

Research described in this paper was financed by the National Sustainability Program under grant LO1401. For the research, infrastructure of the SIX Center was used.

REFERENCES

[1] *Bishop, Matt*. "What is computer security?." Security & Privacy, IEEE 1.1 (2003): 67-69.

[2] *Ingols, Kyle*, "Modeling modern network attacks and countermeasures using attack graphs." Computer Security Applications Conference, 2009. ACSAC'09. Annual. IEEE, 2009.

[3] *Gordon, Lawrence A., and Martin P. Loeb.* "The economics of information security investment." *ACM Transactions on Information and System Security (TISSEC)* 5.4 (2002): 438-457.

[4] *Gordon, Lawrence A., and Martin P. Loeb.* "The economics of information security investment." *ACM Transactions on Information and System Security (TISSEC)* 5.4 (2002): 438-457.

[5] *Carman, David W., Peter S. Kruus, and Brian J. Matt.* "Constraints and approaches for distributed sensor network security (final)." DARPA ProjectReport,(Cryptographic Technologies Group, TrustedInformation System, NAI Labs) 1 (2000): 1.

[6] *Dohler, Mischa*, "Kumars, Zipfs and Other Laws: How to Structure an Optimum Large-Scale Wireless (Sensor) Network?." European Wireless 2007 (2007).

[7] *Perrig, Adrian, John Stankovic, and David Wagner*, "Security in wireless sensor networks." Communications of the ACM 47.6 (2004): 53-57.

[8] *Ramakrishnan, Maharajan, and P. Vanaja Ranjan*. "Adaptive power control with overhearing avoidance for wireless sensor networks." *Communication Systems and Networks (COMSNETS), 2010 Second International Conference on*. IEEE, 2010.

[9] *Kenyeres, Jozef, Martin Kenyeres, and Markus Rupp.* "Experimental node failure analysis in WSNs." Systems, Signals and Image Processing (IWSSIP), 2011 18th International Conference on. IEEE, 2011.

[10] *Censi, Andrea, and Richard M. Murray.* "Real-valued average consensus over noisy quantized channels." American Control Conference, 2009. ACC'09.. IEEE, 2009.

[11] *Fagnani, Fabio, and Sandro Zampieri.* "Average consensus with packet drop communication." *SIAM Journal on Control and Optimization* 48.1 (2009): 102-133.

[12] *Lynch, Nancy A.* Distributed algorithms. Morgan Kaufmann, 1996.

[13] *Patterson, Stacy, Bassam Bamieh, and Amr El Abbadi.* "Distributed average consensus with stochastic communication failures." *Decision and Control, 2007 46th IEEE Conference on*. IEEE, 2007.

[14] *Kenyeres, Jozef,* "WSN implementation of the average consensus algorithm." Wireless Conference 2011-Sustainable Wireless Technologies (European Wireless), 11th European. VDE, 2011.

[15] *Ramakrishnan, Maharajan, and P. Vanaja Ranjan.* "Adaptive power control with overhearing avoidance for wireless sensor networks." *Communication Systems and Networks (COMSNETS), 2010 Second International Conference on*. IEEE, 2010.

[16] *Patterson, Stacy, Bassam Bamieh, and Amr El Abbadi*. "Distributed average consensus with stochastic communication failures." *Decision and Control, 2007 46th IEEE Conference on*. IEEE, 2007.

[17] *Bapat, Ravindra B*. Graphs and matrices. Springer, 2010.

[18] *Skiena S.*, "Adjacency Matrices." §3.1.1 Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica. Reading. MA: Addison-Wesley, 1990, s. 81-85.

[19] *Kenyeres, Jozef*, "Connectivity-Based Self-Localization in WSNs." Radioengineering 22.3 (2013).

[20] *Chen, Xiangqian*, "Sensor network security: a survey", Communications Surveys & Tutorials, IEEE 11.2 (2009): 52-73.