https://doi.org/10.52326/jes.utm.2021.28(2).06
UDC 004.4:004.7

PETRI NETS MODELING FOR SDN TOPOLOGIES

Ali Ameen*, ORCID ID: 0000-0002-5451-8257,
Emilian Guţuleac, ORCID ID: 0000-0001-6839-514X

Technical University of Moldova, 168 Stefan cel Mare Av., MD-2004 Chisinau, Republic of Moldova
*Corresponding author: Ali Ameen, *alisalmanhussein@yahoo.com*

Received: 12.18.2020
Accepted: 04.05. 2021

Abstract. The article examines some new algorithms and focuses mainly on suggesting new working topologies for software-defined controllers in order to ensure SDN security and to prevent the occurrence of a potential central point of failure (SPOF) by overcoming the centralization problem. This is a positive feature of the SDN structure, but could also be a threat, caused by the use of several controllers in different working topologies. This article focuses on exactly one of the suggested topologies, which features and models based on the Petri Nets system. The usual topology of a single controller is compared to verify the advantages and privileges of the proposed serial topology over the existing one. The paper tries to obtain a formula from the modeling of the serial topology and its advantages over the usual topology and that formula will be used to measure the level of security or the defense capacity of the network defined by the software against cyber attacks; in particular, denial of service attacks / distributed denial of service attacks / DDoS.

Keywords: *blockchain, east-westbound API, GSPN, hydra, petri nets, RSA, virtual private network.*

1. Introduction

In previous articles [1 - 3] we discussed software-defined networks and what that title means and that it is not a new technology in precise but more like a new methodology of managing computer networks. So, basically the name meaning encompasses networks that are based on software or programmable networks and that is done basically by decoupling the control plane which is like the brain of the network management device from the forwarding plane which is like the muscle of the device and that will leave the switches as mindless dumb devices with a sole purpose of forwarding data packets as per the rules mapped and issued by the control plane which is in turn represented by a software-based or a hardware-based controller that will be connected to the data plane or network switches using a TLS-based protocol like OpenFlow [4] which is the most prominent used one in SDN currently. Now it is noticed that SDN has a great deal of advantages for the world of networking but, it also raises new security challenges as it solves and patches other security issues; some of the advantages it provides are:

- Dealing with growing technology needs: The amounts of the deployed internet connected devices like (Ipads, smartphones, and IoT devices etc) are really soaring up and with the new technologies like cloud technologies there is a growing need to develop computer networks to be able to keep up with these gigantic amounts of data that are created due to the previous reasons, and Software-defined networks could be a potential solution for that issue.

- Flexibility: Due to its programmability; the SDN can open the domain to develop new apps as per user, administrator or enterprise requirements also, that will free the consumer from vendor-based equipment hence, more freedom.
- Cost saving: Despite that it is a new way of managing networks but, it has a backwards-compatibility with legacy network devices and that means that is possible to work with SDN environment using classical network devices so, there is no need to change the entire equipment hence a big reduction on cost.
- Security: The centralization provided in this new paradigm represented by the control plane which could be usually one controller and it could be multiple controllers means that there will be one entity or brain capable of controlling, monitoring and managing the status of the whole network with the ability to enforce policies in the network from a central point with no need to configure every single switch in the network which will reduce that time needed and takes a big amount of the burden of configuring or changing policies in every single network node and that means reduction of human error; not to forget that this single point of management will be capable of filtering the data packets through restrict rules that will provide a granular control over the networks data.
- Enhancing data flow: The SDN controller is capable of identifying multiple paths for each flow; meaning that this permits the flow’s traffic to be distributed and divided among multiple network nodes. And that will give a better enhancement of the performance of the network.

Now as we addressed its advantages; we have to address some of the main issues or security challenges SDN raises to give a neutral overview and those issues are:

- **Centralization:** Despite that centralization of SDN architecture is one of the main positive features of SDN and an advantage in SDN over the classical architecture on one hand but on the other hand it represents a potential threat itself in the same time by creating a single point of failure.

East-westbound API: In multi-controller topologies there are a channel that connects between every two controllers and that channel is an application programming interface API called east-westbound API [5] since that the connection between the higher planes and the lower planes like control and data planes is referenced by the directions north and south. There is not much concentration on them, and it could be vulnerable to some cyber-attacks like MITM [6], DoS or DDoS [7] types of attacks which have a destructive effect [8].

- **Security level assessment and defense ability measurement:** most works that try to do some modelling for computer networks and measure security their level based on some existing general laws of risk assessment or try to conduct some specific mathematical analysis for that particular instance of network. To the best of our knowledge there’s no fixed solid security level assessment methodology for software-defined networks due to many reasons like; their dynamic and ever-evolving properties, various topologies, different numbers of controllers used, etc.

The proposed solutions for the previous issues could include a suite of some algorithms and SDN controllers’ topologies organized together as a full framework that could be incorporated with the SDN environment to enhance it and patch up the aforementioned issues.

First it is needed to describe the algorithms briefly and after that the topologies will be discussed briefly as well, with a main concentration on the firstly proposed serial topology here in this article.

Those algorithms are optional to be incorporated with the three proposed topologies or with any other SDN controllers' topology; that's why in the later described petri nets modelling they will not be incorporated with the SDN controllers' topologies and also since we want to give a pure description of those topologies and derive a formula that will address the security level of the software-defined network that leverages that specific topology; if it was based on one of the 3 proposed topology in this research of course.

2. Proposed Algorithms

In this article we elaborated and proposed the main methods and algorithms to be integrated together in a whole framework to solve the problems noticed in the research:

- Hydra: we have designed a framework that contains the next algorithms with some techniques incorporated alongside those methods like counter measurement precautions to counter attack the Denial of Service/ Distributed Denial of Service (DoS/DDoS) attacks [9]; by installing botnets [10, 11] into network computers connected to the controller that has the Hydra software to make them as potential zombie guards to attack the attacker's IP.
- VPN: virtual private network (VPN) is well known for its ability to secure connection channels through its technique that is known as internet protocol security (IPsec). It is capable of creating secure communication virtual channels between connected nodes and since it is widely used in different networks then it is possible to include it into the proposed framework. A VPN is needed to specify a certainty that the confidentiality of sensitive data can be kept transmitted on the network a Local Area Network (LAN) or workable so that only authorized users are able to access sensitive data [12]. We have established the usage of VPN. Basically it is possible to connect two controllers using the secure channel of VPN even if they were in the same building and exchange the information between them securely.
- **RSA:** RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and it is different from the decryption key which is kept secret (private). In RSA, this asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem". The acronym RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1977. Clifford Cocks, an English mathematician working for the British intelligence agency Government Communications Headquarters (GCHQ), had developed an equivalent system in 1973, but this was not declassified until 1997 [13]. This algorithm is already used in most network's communications nowadays but we have included it here in this framework in a different way and that's by doubling the channel of cryptography; meaning that instead of using one public key and 2 private keys for every encryption-decryption procedure; this framework will use 2 public keys and 4 private keys and every node will have a channel for sending encrypted information and a channel for receiving information hence; two channels of encrypted communications.

- **Blockchain:** which is also incorporated with the aforementioned framework but in a different way. As it is known blockchain's best and biggest participation is in cryptocurrencies like bitcoin but, it is also used in some other fields and it is already used in the Marconi protocol [14], but here we have designed the usage of blockchain in a different way to secure the configuration updates between multiple controllers.

3. Suggested Topologies

We have proposed in this research and in this article different topologies to overcome the centralization issue which is already an advantage over the classical structure of networks; since it is giving the software-defined network's structure the ability to manage the whole network and facilitate policy enforcement but, at the same time it could be leveraged as a single point of failure (SPOF) in case of an attack on that point which is represented by the single controller. Those proposed topologies could leverage the proposed Hydra framework and whether they activate the framework or not; they can help to overcome the centralization issue. Those topologies could differ in the number of controllers used and the kind of interaction between them. In this article we'll focus on the serial topology which is the first proposed topology and despite that it could already been in use by some researches but, here the difference will be in the kind of interaction between the controllers themselves.

- **Serial Topology.**

We designed the serial topology to contain 3 controllers, where there's a main controller and 2 backup ones just in case of an attack or a disruption that may stop the first or main controller. the main controller controls the whole network and its nodes and sends an update every 10 seconds that informs the backup controllers about any change in the topology or network configuration every 10 seconds and it also acts like a beacon that alerts the controllers if there's a latency in the update message and it took more than 10 seconds then, it will be taken that the main controller has been infected by a DDoS attack or any type of threat hence, comes the role for the second controller which was a backup to become the next main controller and the same thing now goes between the new main controller and the third controller which now becomes the new 1st backup controller until the previous main one will be maintained and restored then everything goes back to its previous state.

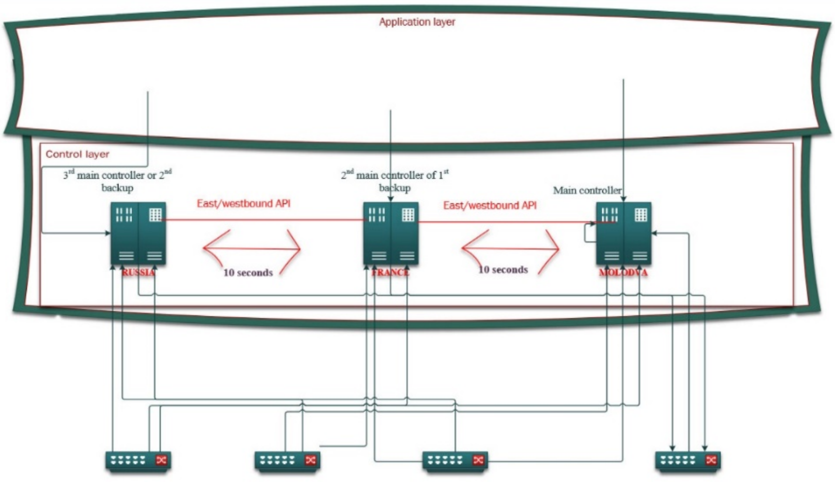


Figure 1. Serial Topology.

And of course if the new main controller which was previously the second one got attacked as well then, the same procedure will be applied and it will be replaced with the third controller. Of course the assignment of controllers could be by using a priority number to choose the main, the first backup (second) and the (second backup) third controller. And as mentioned above in the Hydra section, a botnet program will be added to all controllers so that they can install it in the computers connected to them for a counterattack measurement to disrupt the attacker's machine and prevent it from continuing its attack by creating a DoS/DDoS attack on it or by sending a simple virus to it that will stop it or force it to restart then the next procedure will be isolating or blocking the IP of that attacker's machine.

- **Ordinary Topology.**

Here we have a basic topology of software-defined networks, where we have one controller that controls the whole network, it controls the switches and they control the rest of the network of course; here the controller will be serving the computers by serving the switches that transfer the requests of the computers.

But, here if we have to many computers requesting to be served or a DoS/DDoS attack on the controller and that attack was somehow able to disrupt the server/controller and we don't have a backup controller that works with our main controller hence, that will stop the controller with no substitute and it might jeopardize the whole network by stopping it or hacking into switches by controlling the controller itself or by giving the network commands to let unauthorized entities or devices to be connected to the network and that will mean the end with no ability to recover. The figure 2 below shows an example of the ordinary one-controller topology.

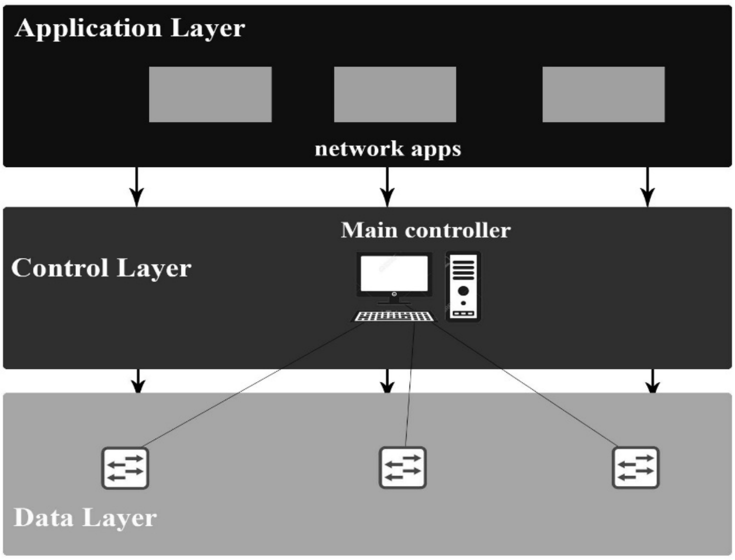


Figure 2. Ordinary Topology.

4. Petri Nets Modelling

Petri nets field was invented by Carl Adam Petri for the purpose of describing chemical processes. A Petri net, which is called as a place/transition (PT) net as well. It is described as one of many available mathematical modeling techniques used for the purpose of modelling distributed systems. Also, it could be described as a discrete event dynamic system. The petri net is a directed bipartite graph, meaning that it contains mainly of two types of nodes which are places (i.e. conditions, represented by circles) and transitions (i.e. events that may occur, represented by bars). The directed arcs or arrows describe the direction of the procedure meaning which places are pre or post conditions for which transitions. Petri nets technique offers a graphical notation for stepwise procedures or processes that could include iteration, concurrent execution and/or choice. This technique has an exact mathematical definition [15]. We have used petri nets for modelling of the serial topology, to have a better understanding of the topology's capabilities and to derive a formula from its behavior that could be leveraged to measure the security level of a software-defined network especially if it was based on that topology.

- **Serial topology.**

As mentioned earlier; this topology has 3 controllers working as 1 main controller and 2 backup controllers of course regardless of whether they were software-based or hardware-based controllers. We have designed this topology using PIPE software that is based on Petri Nets system.

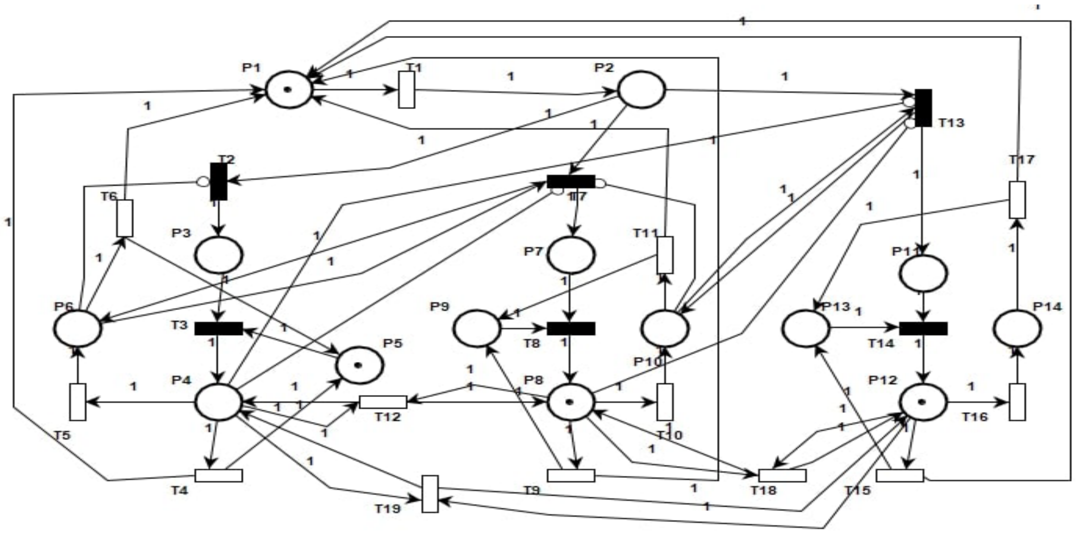


Figure 3. Serial topology modeling using Petri Nets.

Description of the Serial topology scheme:

1- The main controller will be working normally as the only main brain unit or entity that manages the whole network behavior, interacting with switches and managing the requests of hosts through them.

2- The main controller will be sending updates of network configuration every 10 seconds to both backup controllers so that they be up to date and aware of overall network behavior, status and structure and be able to take control of the network in case of an attack or any kind of disruption of the main controller.

3- As mentioned before in case of an attack that will disrupt the main controller, the control and management of the whole network will be handed over to the next backup controller with the next closest priority number, in this case controller No. 2.

4- A bot will be sent to infect the attacking source and then the attacker's IP will be blocked in one direction like what happens with the access control list so that we can still control it but it cannot reach our network, and the infected controller will be blocked and isolated as well.

5- The new main controller which was previously the backup one now manages the network and sends information updates of network configuration to the remaining backup controller till the maintenance of the previously main controller finishes. The table 1 shows the description of places.

Table 1

Description of Places

Place	Description
P1	Packets processing by main server or controller/ the input place that sends data tokens
P2	Selection of servers
P3/P7/P11	Allocation of servers
P4/P8/P12	Server 1, 2, 3 Active processing
P5/P9/P13	Server 1, 2, 3 Free controllers
P6/P10/P14	Attack

While the table 2 shows the transitions descriptions.

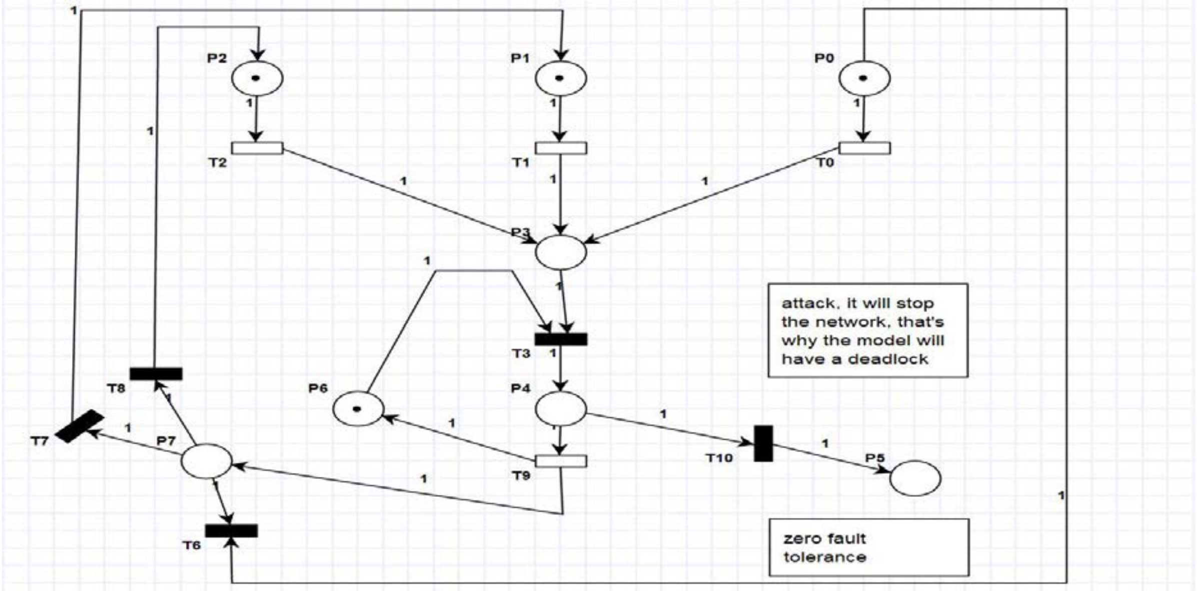
Table 2

Description of Transitions

Transition	Description
T1	Generated task i.e. packets processing
T2/T7/T13	Selection of servers 1, 2, 3
T3/T8/T14	Allocation of servers
T4/T9/T15	Task processing
T5/T10/T16	Exiting the stage
T6/T11/T17	Restoring the controller
T12/T18/T19	Updating the information and going back to initial stage of controllers

- **Ordinary Topology.**

As shown in the figure 4; this modelling represents the usual ordinary topology with a single controller and shows its weakness points.



attack, it will stop the network, that's why the model will have a deadlock

zero fault tolerance

Figure 4. Ordinary topology modeling using Petri Nets.

Description:

1. This topology is just representing the usual, simple, basic structure of Software-defined network using one controller.
2. It's just modelled for the sake of comparison to show how much effective our framework is with its proposed topologies.
3. This model shows how a one controller is really vulnerable and ineffective since there's a single point of failure (SPOF) which we want to overcome.
4. We have here one controller that processes switches' requests normally until an attack occurs.
5. In the case of an attack the above design shows that an attack can disrupt the controller and everything that relies on it since there's only one main controller. so, everything falls apart after infecting the controller and the whole network will be compromised. Which means that this topology has zero fault tolerance. The table 3 below shows the places of the diagram.

Table 3

Description of Places

Place	Description
P0/P1/P2	Selection of switches
P3	Main controller/ server
P4	Active processing
P6	Processing next request/ getting back to initial state
P7	Sending and receiving requests
P5	Attack on server/controller

The table 4 describes the transitions of the petri nets diagram.

Table 4

Description of Transitions

Transition	Description
T0/T1/T2	Sending requests to controller
T3	Active processing
T9	Initial state/ replying to switches
T6/T7/T8	Selection of switches
T10	attack

5. Defense Factor Formula: Derived from the Petri Nets modelling

- **GSPN Module.**

We have created a simple comparison between the 2 topologies in terms of Average Number of Tokens on places that represent the SDN controllers, where the tokens represent how many tasks or configuration updates the controllers have to implement every 10 seconds and the less busy controllers they are, the better it is; since it shows that the network controllers are less DoS/DDoS attacks prone and more capable of handling these attacks and dealing with them. In this comparison we left the weight ω of immediate transitions intact and gave the rate r of timed transitions a value of 0.1 since that we need those configurations of the network to be broadcasted every 10 seconds and that means that every 10 seconds the model state will change. The relationship between the time and rate/weight can be shown as below:

$$\tau = \frac{1}{r} \tag{1}$$

where τ represents the time, r represents the rate of timed transitions. That's why, if we want the time to be 10 seconds then, we have to change the rate value to 0.1.

We gave the models a fixed value of firings for the transitions in each model which is 20 firings and the result was as shown in the table 5.

Table 5

Average Number of Tokens in Places Representing SDN Controllers Using GSPN Module

Algorithms / **Places/k$_i$**	**Serial Topology/ Z_{Ki}**	**Ordinary Topology/ Z_{Ki}**
P3	0.16337	1.99975≈2
P7	0.06867	
P11	0.13233	

As inferred from the table 5 it is possible to notice that using this topology; the average number of tokens which is the average number of requests dealt with by the controller per unit of time is really reduced and small as compared to the average number of token/requests dealt with by the controller in the ordinary topology and that could prove the efficiency of the topology as compared to the efficiency of the single-controller SDN topology.

- **Elaboration of the Defense Factor formula.**

According to previous table 5 we can notice that the serial topology will be better since its model is empty of tokens most of time during which the firings took place and that means that this network topology was less occupied with tasks or its controllers were more available in unit of time hence, more capable of resisting DoS/DDoS attacks.

Now after leveraging the PIPE software and the GSPN module it has; to determine those results mentioned previously, we have elaborated this equation based on the acquired results and based on the relationship between the readings gained from the different simulations, meaning the equation to assess the security level of networks especially the software-defined networks against cyberattacks especially those that could leverage the busy or flooded servers as a weakness point like DoS/DDoS attacks; we named this proposed equation as the network defense factor against cyberattacks law. Before we use this law or equation, we need to describe the basis of this law itself and how and why it was formulated. First we have to emphasize that this law is formulated for different kinds of attacks but it is especially used for DoS/DDoS attacks risk assessment due to its main concept or feature that it depends on; which is how many operations are conducted by the controller/server, means how many requests that the controller is dealing with at a specific unit of time and as we know and mentioned before that denial of service and distributed denial of service attacks DoS/DDoS are depending mainly on flooding the target with huge amounts of request packets to disrupt it or stop it completely so, the less the targeted device is occupied the better it is; because it will be more capable of dealing with that big amount of requests hence, it will have a better security defense level and longer time to respond to use its defense mechanisms like intrusion detection and intrusion prevention systems IDS/IPS, firewalls, etc. The law shows that the less requests a controller has the better security level and higher defense abilities it has and vice versa so, it's an opposite relationship between the number of requests being handled at a specific unit of time and the Defense level assessment. And as we know since this research focuses on assuring the security of the computer network generally and the security of the software-defined network in precise especially the control plane in its structure. And we have the control plane represented by the controller as our main element of interest to secure and also the main component of the SDN that we need to determine its security level, then it is of a great deal of importance to include that element mainly in the formula to figure out its security level hence, finding out the security level of the network itself.

In other words, let defense factor be DF then:

$$DF=f\ \{K, Z\} \tag{2}$$

Where K is the number of controllers in the network, and Z is the number of requests being served in each controller at the current unit of time. If we use the aforementioned relationship with the Petri Net models we have; then we can get a relationship between all of them which is our aforementioned equation that can be applied using the terms of Petri Nets as well. In terms of petri nets the requests will be represented by how many tokens are there in specific places which in turn represent specific nodes in the software-defined network and those specific nodes of interest are the SDN controllers. In the equation places representing the SDN controllers are denoted as K, where K ∈ P which is the whole group of places in the petri nets PN model, which in turn is a tuple of 5 objects, Pn= (P, T, I, O, M_0), where P is the finite set of places, T is a finite set of transitions, I is the input function, then we have O for output function and M_0 is the initial marking. So, here is the equation to measure the defense factor for a software-defined network that is based on one of the 3 specific topologies we proposed previously:

$$DF= \sum_{i=1}^{i=n} k_i \cdot \frac{1}{\sum_{z=0}^{z=\infty} Z_{k_i}} \tag{3}$$

Where K_i as mentioned previously is the number of the places that represent the controllers in a specific model, K_i= (K_1,K_2, ...K_n) and Z_{Ki} is the value of tokens in those places K_i, Z_{ki} = (0,......., ∞).

$$\text{Let } A = \frac{1}{\sum_{z=0}^{z=\infty} z_{k_i}} \tag{4}$$

$$A=\begin{bmatrix} \infty & ,(Z)=0 \\ <1 & ,(Z)>1 \\ 1 & ,(Z)=1 \\ 0 & ,(Z)=\infty \end{bmatrix} \tag{5}$$

So if we apply the values obtained from that simulation module's table on this proposed formula we have; then the Defense Factor for the ordinary single-controller topology of SDN will be:

DFO = Ko_i.1/Zo_i =1x (1/T_{P3}) → where T_{P3} is the value of the average number of tokens in the place No.3 (P3), then.

DFO=1x1/1.99975 = 0.50

While the Defense Factor for the proposed serial topology of SDN will be:

DFS= $Ks_i.1/Zs_i$= 3x $(1/T_{P3}+T_{P7}+T_{P11})$ → where T_{P3}, $T_{P7,}$ T_{P11} are the values of the average number of tokens in those places respectively, then.

DFS=3x1/0.36437 = 8.23

And as it is noticed here, the serial topology has a higher defensibility than the ordinary one. The table 6 below presents a comparison between the different topologies in their Defense Factor results.

Table 6

Comparison between Different SDN Topologies based on Their Defense Factor DF

Algorithm	Serial Topology	Ordinary Topology
Defense factor DF	8.23	0.50

Conclusions

- This article focuses on assuring the security of software-defined networks in order to make them a safer environment hence, securing computer networks in general by facilitating their transition to the SDN paradigm.
- In this article we have given a brief explanation of the algorithms proposed by us and that will be explained in later articles. They're incorporated together to form the Hydra framework.
- We have designed this framework to be used optionally by the other part of the solution which is the topologies proposed to overcome the centralization point in the SDN structure which is an advantage itself since it facilitates the management of the network but, it raises some new security challenges in the same time as well; like the single point of failure (SPOF).
- We have proposed three topologies for the SDN controllers in the research, and here we concentrate on one of them which is the serial topology.
- This article gave a modelling for the serial topology using Petri Nets system to derive a formula that can be used to assess the security risk level of the software-defined network that leverages one of the proposed topologies.

References

1. Ali Ameen. Software-defined networks - a general survey and analysis. In: Journal of Engineering Science, no. 3 (2018), pages 61-73.
2. Ali Ameen. The using of sdn technologies for security insurance of computer networks. In: Proceedings of Technical-Scientific Conference of TUM, March 2019, Volume 1, Pages 417-420.
3. Ali Ameen. Leveraging blockchain technology to assure security of SDN. In: Journal of Engineering Science, Proceedings of International conference on Electronics Communications and computing, October 2019, pages 128-139.
4. Azodolmolky S., Coker O. Software Defined Networking with OpenFlow 2nd edition: book. UK, 2017.
5. Ayesha I., SDN controllers' security issues: MS thesis. University of Jyväskylä - Finland, 2017.
6. Malik A., Ahsan A., Shahadat M., Tsou J., Man-in-the-middle-attack: Understanding in simple words. International Journal of Data and Network Science, 2019, China University of Technology Volume 3, pp. 77-92.
7. Prasad K., Reddy A., Rao K., DoS and DDoS Attacks: Defense, Detection and Trace Back Mechanisms – A Survey. Global journal of computer science and technology: E Network, Web and Security, Issue 7, 2014, JNTUH University, India. Volume 14, 18-pages.
8. Wong F., Tan C.X. A survey of trends in massive DDoS attacks and cloud-based mitigations. Int. J. Netw. Secur. Appl. (IJNSA), (2014), 6(3), pages 57–71.

9. Zakaria Bawany N., A. Shamsi J., Salah K. DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions, King Fahd University of Petroleum & Minerals, ISSUE 2017, Journal of Cryptology, 17-pages.
10. Osagie M., Enagbonma O., Inyang A. The Historical Perspective of Botnet Tools. Current Journal of Applied Science and Technology, Issue 6, Feb, 2019, Department of Physical Sciences, Faculty of Science, Benson Idahosa University. Volume 32, 8-pages.
11. CCTV-based botnet used for DDoS attacks. [online]. [accessed: 04.07.2017]. available: https://www.ddosattacks.net/a-massive-botnet-of-cctv-cameras-involved-inferocious-ddos-attacks .
12. Iqbal M., Riadi I., Analysis of Security Virtual Private Network (VPN) Using OpenVPN. International Journal of Cyber-Security and Digital Forensics, 2019, Ahmad Dahlan University, 8 (1), pp. 58-65.
13. RSA. [online]. 2019, [accessed: 20.07.2019] available: https://en.wikipedia.org/wiki/RSA_(cryptosystem).
14. How blockchain will manage networks. [online]. 2019, [accessed: 26.08.2019] available: https://www.networkworld.com/article/3356496/how-blockchain-will-manage-networks.html.
15. Petri Net. [online]. 2019, [accessed: 11.10.2019] available: https://en.wikipedia.org/wiki/Petri_net .