UDC 004

STOCHASTIC VIOLATOR MODEL

A. B. Kaczynski [1], O. V. Kireienko[1], O. V. Kozlenko[1]

[1] *National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Educational and Research Institute of Physics and Technology*

Abstract

This paper introduces a new type of violator model that is based on Markov chains. It can be used as a scenario model AS IS or as a mathematical model with quantitative estimates if additional information is presented. Our aim with this paper was to develop a model that will allow to restore missing data, using existing knowledge about violator. The results show that presented scenario for general cases cover the majority of attacks and can be applied to real-life scenarios too. Summing up the results, it can be concluded that additional improvement of the model should be focused on data gathering to ensure that existing data will be enough to recover the rest.

Keywords: violator model, Markov processes, attack scenarios

Introduction

As of today, informal approaches to development of violator's models are dominant. These approaches depict causes and motives of malicious actor actions, his knowledge, capabilities, priorities for achieving set goals like tactics, location and character of such actions, ways for attacking system[1] [2]. With a few tweaks, according to [3], we will consider a malicious actor to be a person or a group of people who ensure realisation of cyber threats by the means of malicious or non-malicious action. This article describes a model that is based on continuous-time Markov process that has a fundamental role in violator's model development. This model is based on "birth-death" processes . It is notable for describing malicious behavior as a random process with a discreet amount of states in the system S [4]. Among advantages of this approach should be mentioned an ability to use information from special services, analytic groups, data about existing information access methods, its processing and storing, and about previously registered cyber incidents to develop a violator's model. Furthermore, it takes into account real operational and technical capabilities of a malicious actor's influence on system/subsystem that provides cybersecurity or an object in question. Such approach allows us to examine a formalized description of such actor as a violator of access control rules, who can only move from any given state only to its neighboring state. With a few tweaks such model can be applied to external or internal malicious actors. Adequate violator's model guarantees development of an effective system that provides cybersecurity, which in turn allows development of appropriate mechanisms of cyber threats forecasting and averting. This article describes 4 scenarios of malicious actor's actions. They determine classification types for such actions and take into account methods and approaches at each step with the aid of "birth-death" process. Notably, each action corresponds to a specific state. The following states are common for all scenarios:

- $S0$: normal operation mode for the targeted system;
- $S1$: normal operation mode for the targeted system (the difference between $S1$ and $S0$ is determined by the presence of inbound connection arcs;
- $S2$: preparation for attack;
- $S3$: infiltration into a system and selection of attack method;
- $S(3+i+kn)$: these states correspond to malicious actions within the system.

Parameter n is the amount of branches, parameter k is a sequence number of a malicious action within attack, parameter i is a sequence number of a branch. If $n = 1$, there is no branching, sequence for k starts from zero and sequence for i starts from one. These states correspond to malicious actions that a actor can perform within the system after infiltration. They can represent uploading of the malicious software (viruses, ransomware, botnet-client), privilege escalation (exploiting known vulnerabilities, shellcode), disabling security mechanisms, halting the applied tasks that are running within the system, interaction with authorized users, covering malicious actions. $S(r+j+lm)$: refers to the states of system recovery. Parameter r is the amount of malicious actions, parameter j is a sequence number of a branch, parameter l is a sequence number of an action, and parameter m is the amount of branches. In this case, sequence for j starts from one, sequence for l starts from zero. These states represent ways to remove vulnerabilities and consequences from attack, system testing before restoring normal mode of operation. Transition from one state to another refers to the lifecycle of a cyberattack. In case of $S(3+i+kn)$ states it means completing the next malicious action or completing the same action for a different segment of the system. Movement in the reversed direction (from right to left) represents cancellation of performed ma-

licious actions for whatever reason(discovering a more threatening attack, inability to cover malicious actions, human mistakes, technical issues on either malicious actor's or information system's side), which forces to restart an attack. Let's examine each of these scenarios in more detail.

1. Scenario 1: user-related attacks

User-related attacks exploit an authorized user to voluntarily shut down security mechanisms, configuration change of the aforementioned mechanisms and even the configuration of an entire system while such user is under influence of malicious actor. For that specific reasons such attacks are particularly agile and contain a possibility to switch branches:

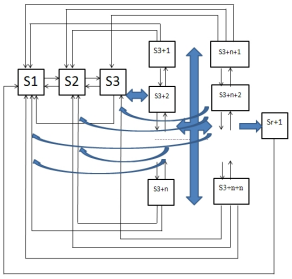

Fig. 1. general case for user-related attacks

We will use an infamous attack Soloriagate [5, 6], to examine the peculiarities of this scenario's implementation. Picture below contains a system's states graph with a branching cycling process:

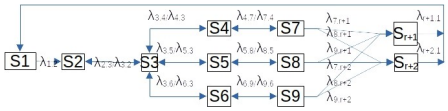

Fig. 2. Graph of system's states of first scenario for Solorigate attack

In case of Solorigate attack there are a few specific states and relations:

- $S4$: the use of stolen passwords;
- $S5$: forgery of SAML tokens;
- $S6$: attack that uses GoldMax, GoldFinder and Sibot software;
- $S7 - S9$: data compromise, persistance, covering malicious actions by an attacker;
- $\lambda_{1,2}$: flow intensity from $S1$ to $S2$
- $\lambda_{2,3}/\lambda_{3,2}$: values of flow intensity from $S2$ to $S3$/ from $S3$ to $S2$
- $\lambda_{k,i}/\lambda_{i,k}$: values of flow intensity from Sk to Si, from Si to $Sk(i = 4,...,9;k=4,...,6)$
- $\lambda_{i,r+1}$: values flow intensity from Si to $Sr + 1(i = 7,...,9)$
- $\lambda_{i,r+2}$: values flow intensity from Si to $Sr + 2$ $(i = 7,...,9)$
- $\lambda_{r+j,1}$: values flow intensity from $Sr + j$ to $S1$ $(j = 1,2)$

The corresponding matrix of a Markov process for this scenario is presented as:

$$\begin{matrix}
0 & \lambda_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda_{2,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \lambda_{3,2} & 0 & \lambda_{3,4} & \lambda_{3,5} & \lambda_{3,6} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda_{4,3} & 0 & 0 & 0 & \lambda_{4,7} & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda_{5,3} & 0 & 0 & 0 & 0 & \lambda_{5,8} & 0 & 0 & 0 \\
0 & 0 & \lambda_{6,3} & 0 & 0 & 0 & 0 & 0 & \lambda_{6,9} & 0 & 0 \\
0 & 0 & 0 & \lambda_{7,4} & 0 & 0 & 0 & 0 & 0 & \lambda_{7,r+1} & \lambda_{7,r+2} \\
0 & 0 & 0 & 0 & \lambda_{8,5} & 0 & 0 & 0 & 0 & \lambda_{8,r+1} & \lambda_{8,r+2} \\
0 & 0 & 0 & 0 & 0 & \lambda_{9,6} & 0 & 0 & 0 & \lambda_{9,r+1} & \lambda_{9,r+2} \\
\lambda_{r+1,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_{r+2,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{matrix}$$

System of Kolmogorov's differential equations for the first scenario is given below:

$$\begin{aligned}
\lambda_{1,2}P_1 + \lambda_{1,2}P_3 &= P_2\lambda_{2,3} \\
\lambda_{2,3}P_2 + \lambda_{4,7}P_4 + \lambda_{5,8}P_5 + \lambda_{6,9}P_6 &= P_3(\lambda_{3,2} + \lambda_{3,4} + \lambda_{3,5} + \lambda_{3,6}) \\
\lambda_{3,4}P_3 + \lambda_{7,8}P_7 &= P_4(\lambda_{4,3} + \lambda_{4,7}) \\
\lambda_{3,5}P_3 + \lambda_{8,9}P_8 &= P_5(\lambda_{5,3} + \lambda_{5,8}) \\
\lambda_{3,6}P_3 + \lambda_{9,6}P_9 &= P_6(\lambda_{6,3} + \lambda_{6,9}) \\
\lambda_{4,7}P_4 &= P_7(\lambda_{7,r+1} + \lambda_{7,r+2} + \lambda_{7,4}) \\
\lambda_{5,8}P_5 &= P_8(\lambda_{8,r+1} + \lambda_{8,r+2} + \lambda_{8,5}) \\
\lambda_{6,9}P_6 &= P_9(\lambda_{9,r+1} + \lambda_{9,r+2} + \lambda_{9,6}) \\
\lambda_{7,r+1}P_7 + \lambda_{8,r+1}P_8 + \lambda_{9,r+1}P_9 &= P_{r+1}\lambda_{r+1,1} \\
\lambda_{7,r+2}P_7 + \lambda_{8,r+2}P_8 + \lambda_{9,r+2}P_9 &= P_{r+2}\lambda_{r+2,1} \\
\lambda_{r+1,1}P_{r+1} + \lambda_{r+2,1}P_{r+2} &= P_1\lambda_{1,2}
\end{aligned}$$

2. Scenario 2: operating system attack

Attacks that are targeting an OS (operating system) contribute greatly to the overall amount of hacking attacks. In the current scenario, malicious actor is preparing for attack without any consideration of capabilities of authorized users in targeted information system. Exploiting OS vulnerabilities, malicious actor can perform a plethora of attacks that don't rely on victim's preparations. Furthermore, most actions will be automated/scripted while the discovery of the incident won't allow to reduce the losses from the successful attack below a certain threshold.

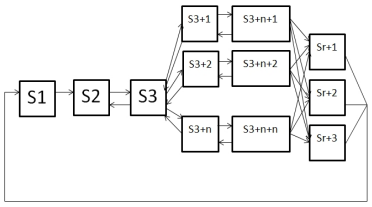

Fig. 3. general case for attack on OS

Attacks that use $CVE - 2017 - 5754$ (Meltdown)/$CVE - 2017 - 5753$(Spectre)/$CVE - 2017 - 5715$(Spectre $V2$) [7] vulnerabilities were selected as an example for the aforementioned scenario. This example differs from the general case by the amount of branches. Another difference is branching type for the recovery stage which occurs after the system reaches S_{r+3} state, when the recovery method is selected (S_{r+1} state):

Special states and relations for this example are mentioned below:

- $S4$ -– exploiting the meltdown vulnerability;
- $S5$ — exploiting the spectre vulnerability;
- $S6$ — exploiting the spectre2 vulnerability;
- $Sr + 1$ – installation of patches;
- $Sr + 2$ – register altering;
- $Sr + 3$ – antivirus;

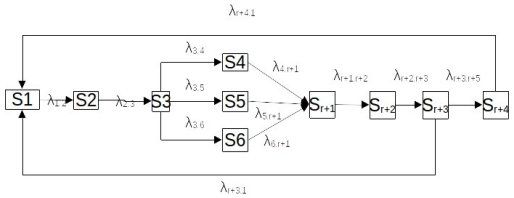

Fig. 4. Graph of system states for Meltdown/Spectre/Spectre2 attack

- $Sr+4$ — installation of an additional patch for Windows 7 64-bit or Windows Server 2008 R2 64-bit;
- $\lambda_{i,j}$: flow intensity from Si to Sj , $(i,j{=}1,...,r+4)$

The matrix of the Markov process for this scenario is presented below:

$$\begin{array}{c} \\ S1 \\ S2 \\ S3 \\ S4 \\ S5 \\ S6 \\ Sr+1 \\ Sr+2 \\ Sr+3 \\ Sr+4 \end{array} \begin{array}{c} \begin{array}{cccccccccc} S1 & S2 & S3 & S4 & S5 & S6 & Sr+1 & Sr+2 & Sr+3 & Sr+4 \end{array} \\ \begin{pmatrix} 0 & \lambda_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_{2,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_{3,4} & \lambda_{3,5} & \lambda_{3,6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{4,r+1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{5,r+1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{6,r+1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{r+1,r+2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{r+2,r+3} & 0 \\ \lambda_{r+3,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{r+3,r+4} \\ \lambda_{r+4,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Therefore, this system of equations is our violator's model:

$$\begin{gathered} \lambda_{r+4,1} P_{r+4} + \lambda_{r+3,1} P_{r+3} = \lambda_{1,2} P_1 \\ \lambda_{1,2} P_1 = \lambda_{2,3} P_2 \\ \lambda_{2,3} P_2 = P_3 (\lambda_{3,4} + \lambda_{3,5} + \lambda_{3,6}) \\ \lambda_{3,4} P_3 = \lambda_{4,r+1} P_4 \\ \lambda_{3,5} P_3 = \lambda_{5,r+1} P_5 \\ \lambda_{3,6} P_3 = \lambda_{6,r+1} P_6 \\ \lambda_{4,r+1} P_4 + \lambda_{5,r+1} P_5 + \lambda_{6,r+1} P_6 = \lambda_{r+1,r+2} P_{r+1} \\ \lambda_{r+1,r+2} P_{r+1} = \lambda_{r+2,r+3} P_{r+2} \\ \lambda_{r+2,r+3} P_{r+2} = P_{r+3} (\lambda_{r+3,1} + \lambda_{r+3,r+4}) \\ \lambda_{r+3,r+4} P_{r+3} = \lambda_{r+4,1} P_{r+4} \end{gathered}$$

3. Scenario 3: web-based attack

Unlike previous two scenarios this one characterizes interaction between malicious actor and a web resource, while an attack itself may or may not rely on user interaction. The lack of $S2$ or S_2 state makes this scenario different from the rest. Another difference is in recovery, represented as a sequence of nodes and same actions are applied to each node: Just like before,

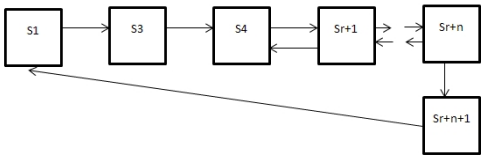

Fig. 5. General case for web(-environment) attack

we will use cyberattacks that exploit $CVE-2021-26855$, $CVE-2021-26857$, $CVE-2021-26858$, $CVE-2021-27065$ [8] vulnerabilities to demonstrate it in action. The graph of system states for these cyberattacks is presented as:

Special states of this scenario are listed below:

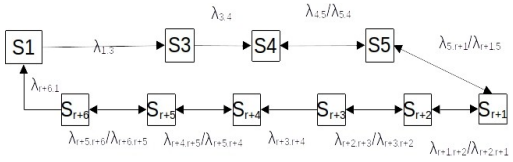

Fig. 6. Graph of system states for attack on Exchange Server

- $S4$ -– uploading a webshell, that is used to collect sensitive information and additional malware upload, C2 interaction or as a part of botnet;
- $S5$ – direct malicious actions performed by a actor;
- $Sr+1$ – Exchange Server latest patch installation on all nodes;
- $Sr+2$, $Sr+3$ – 443 port isolation and vulnerabilities sanitization;
- $Sr+4$ -– server reboot/restart - RAM flushing;
- $Sr+5$ -– antivirus scan for all nodes;
- $Sr+6$ — OS and application security;
- $\lambda_{i,j}$: flow intensity from Si to Sj, $(i,j{=}1,...,r+6)$.

Markov process matrix and Kolmogorov's system of this scenario is given below:

$$P_{ES} = \begin{array}{c} \\ S1 \\ S3 \\ S4 \\ S5 \\ Sr+1 \\ Sr+2 \\ Sr+3 \\ Sr+4 \\ Sr+5 \\ Sr+6 \end{array} \begin{array}{c} \begin{array}{cccccccccc} S1 & S3 & S4 & S5 & Sr+1 & Sr+2 & Sr+3 & Sr+4 & Sr+5 & Sr+6 \end{array} \\ \begin{pmatrix} 0 & \lambda_{1,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_{3,4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_{4,5} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_{5,4} & 0 & \lambda_{5,r+1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda_{r+1,r+2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_{r+2,r+1} & 0 & \lambda_{r+2,r+3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \lambda_{r+3,r+2} & 0 & \lambda_{r+3,r+4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{r+4,r+5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_{r+5,r+4} & 0 & \lambda_{r+5,r+6} \\ \lambda_{r+6,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{gathered} \lambda_{1,3} P_1 = \lambda_{3,4} P_3 \\ \lambda_{3,4} P_3 + \lambda_{5,4} P_5 = \lambda_{4,5} P_4 \\ \lambda_{4,5} P_4 = P_5 (\lambda_{5,4} + \lambda_{5,r+1}) \\ \lambda_{5,r+1} P_5 + \lambda_{r+2,r+1} P_{r+2} = P_{r+1} \lambda_{r+1,r+2} \\ \lambda_{r+1,r+2} P_{r+1} + \lambda_{r+3,r+2} P_{r+3} = P_{r+2} (\lambda_{r+2,r+3} + \lambda_{r+2,r+1}) \\ \lambda_{r+2,r+3} P_{r+2} = P_{r+3} (\lambda_{r+3,r+4} + \lambda_{r+3,r+2}) \\ \lambda_{r+3,r+4} P_{r+3} + \lambda_{r+5,r+4} P_{r+5} = P_{r+4} \lambda_{r+4,r+5} \\ \lambda_{r+4,r+5} P_{r+4} + \lambda_{r+6,r+5} P_{r+6} = P_{r+5} (\lambda_{r+5,r+6} + \lambda_{r+5,r+4}) \\ \lambda_{r+5,r+6} P_{r+5} = P_{r+6} (\lambda_{r+6,r+5} + \lambda_{r+6,1}) \\ \lambda_{r+6,1} P_{r+6} = P_1 \lambda_{1,3} \end{gathered}$$

4. Scenario 4: attack on data in transit

In order to intercept data, malicious actor needs to get access to data transmission channel. To achieve that he has to intercept traffic that is passing through a specific system node or to use special tapping hardware. General scheme is given below We will use an attack

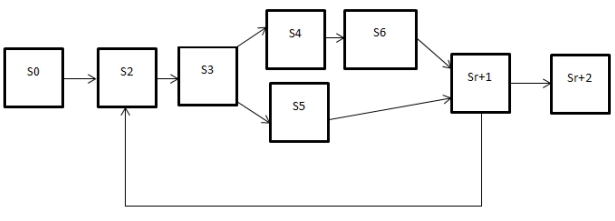

Fig. 7. general case for attack on data in transit

that exploits Heartbleed[9] vulnerability as an example for this scenario:

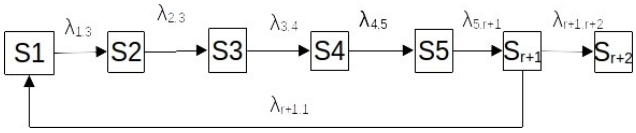

Fig. 8. Graph of system states for Hearthbleed attack

Specific states for this scenario are listed below:

- $S4$ -– valuable information recording on a certain storage device
- $S5$ -– active listening/eavesdropping
- $Sr+1$, $Sr+2$ -– threat removal before reboot
- $\lambda_{i,j}$:flow intensity from Si to Sj , (i,j=1,...,$r+2$)

The matrix of Markov process for this scenario is presented below:

$$\begin{array}{c} \\ S0 \\ S2 \\ S3 \\ S5 \\ Sr+1 \\ Sr+2 \end{array} \begin{array}{c} \begin{array}{cccccc} S0 & S2 & S3 & S5 & Sr+1 & Sr+2 \end{array} \\ \begin{pmatrix} 0 & \lambda_{0,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_{2,3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_{3,5} & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_{5,r+1} & 0 \\ 0 & \lambda_{r+1,2} & 0 & 0 & 0 & \lambda_{r+1,r+2} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

And this is the system of equations that describe violator's model for this scenario:

$$\begin{cases} \lambda_{r+1,1}P_{r+1} = P_1\lambda_{1,2} \\ \lambda_{1,2}P_1 = P_2\lambda_{2,3} \\ \lambda_{2,3}P_2 = P_3\lambda_{3,4} \\ \lambda_{3,4}P_3 = P_4\lambda_{4,5} \\ \lambda_{4,5}P_4 = P_5\lambda_{5,r+1} \\ \lambda_{5,r+1}P_5 = P_{r+1}(\lambda_{r+1,1} + \lambda_{r+1,r+2}) \end{cases}$$

Conclusions

Adequately developed violator's model is the basis appropriate threat design prognostication and prevention mechanisms by the security subsystem. To achieve his goals, malicious actor has to apply effort and resources,thus mathematical modeling of that actions is a complicated task with multiple approaches to its solution. Informal methods, such as Markov processes, are dominant in such approaches. Model of continuous-time Markov process was examined in this article. The main advantage of this method is the fact that malicious actions are viewed as a random process with a finite amount of system states. Such approach allowed us to examine 4 scenarios of attacks on information and communication system. For each scenario different types of malicious actor's behavior were considered. And they were demonstrated with specific examples of infamous cyberincidents that occurred within the past 5 years. To refine this approach further, the next step would require an improvement of a model and calculating gains/losses with incomplete information.

References

1. U. Dyohenes, E. Ozkaiia. Cybersecurity: strategy of attacks(offence) and defence. — M. : ДМК.Пресс, 2020. — С. 326 с.
2. S. Enson. 4. Respond/Reaction/Answering computer incidents. Applied course/module. — M. : ДМК.Пресс, 2021. — С. 436 с.
3. J. Maknamara. Secrets of computer/digital espionage. Tactics and counterexamples. — M. : БИНОМ. Лаборатория знаний, 2004. — С. 536 с.
4. S. Venttsel E., A. Ovcharov L. Random process theory and its engineering applications. — M. : Наука, 1991. — С. 384 с.
5. Remediating Networks Affected by the SolarWinds and Active Directory/M365 Compromise . — `https://us-cert.cisa.gov/remediating-apt-compromised-networks`. — 2021. — [Online; accessed 14-May-2021].
6. Detecting Post-Compromise Threat Activity in Microsoft Cloud Environments . — `https://us-cert.cisa.gov/ncas/alerts/aa21-008a`. — 2021. — [Online; accessed 14-May-2021].
7. Meltdown and Spectre Side-Channel Vulnerability Guidance. — `https://us-cert.cisa.gov/ncas/alerts/TA18-004A`. — 2021. — [Online; accessed 01-May-2021].
8. Mitigate Microsoft Exchange Server Vulnerabilities. — `https://us-cert.cisa.gov/ncas/alerts/aa21-062a`. — 2021. — [Online; accessed 14-April-2021].
9. Heartbleed. — `https://us-cert.cisa.gov/security-publications/Heartbleed-OpenSSL-Vulnerability`. — 2021. — [Online; accessed 23-April-2014].