The Internet of Things (IoT) refers to the millions of devices around the world that are connected to the Internet. Insecure IoT devices designed without proper security features are the targets of many Internet threats. The rapid integration of the Internet into the IoT infrastructure in various areas of human activity, including vulnerable critical infrastructure, makes the detection of malware in the Internet of Things increasingly important. Annual reports from IoT infrastructure cybersecurity companies and antivirus software vendors show an increase in malware attacks targeting IoT infrastructure. This demonstrates the failure of modern methods for detecting malware on the Internet of things. This is why there is an urgent need for new approaches to IoT malware detection and to protect IoT devices from IoT malware attacks. The subject of the research is the malware detection process on the Internet of Things. This study aims to develop a technique for malware detection based on the control flow graph analysis. Results. This paper presents a new approach for IoT malware detection based on control flow graph analysis. Control flow graphs were built for suspicious IoT applications. The control flow graph is represented as a directed graph, which contains information about the components of the suspicious program and the transitions between them. Based on the control flow graph, metrics can be extracted that describe the structure of the program. Considering that IoT applications are small due to the simplicity and limitations of the IoT operating system environment, malware detection based on control flow graph analysis seems to be possible in the IoT environment. To analyze the behavior of the IoT application for each control flow graph, the action graph is to be built. It shows an abstract graph and a description of the program. Based on the action graph for each IoT application, a sequence is formed. This allows for defining the program’s behavior. Thus, with the aim of IoT malware detection, two malware detection models based on control flow graph metrics and the action sequences are used. Since the approach allows you to analyze both the overall structure and behavior of each application, it allows you to achieve high malware detection accuracy. The proposed approach allows the detection of unknown IoT malware, which are the modified versions of known IoT malware. As the mean of conclusion-making concerning the malware presence, the set of machine learning classifiers was employed. The experimental results demonstrated the high accuracy of IoT malware detection. Conclusions. A new technique for IoT malware detection based on control flow graph analysis has been developed. It can detect IoT malware with high efficiency. Keywords: malware; IoT; IoT devices; IoT application; cybersecurity; cyberattack; control flow graph; detection of cyber threats.

Introduction

The infrastructure building of modern cities is impossible without the usage of the Internet of Things (IoT) devices. The rapid proliferation of IoT devices has led to virtual infrastructure controlling physical objects everywhere: from CCTV cameras and smart devices in private homes, on various purposes and highways to machine tools in factories, dams and power plants. It increases the probability of cyberattacks executed by the malware on those areas that previously did not pose risks to cybersecurity [1, 2]. Critical infrastructure is particularly vulnerable to IoT cyberattacks. Extensive exploitation of the Android platform in the IoT infrastructure leads to a rapid increase in cyberattacks on Android-based devices [2]. The main factors that contribute to the unprecedented increase in the number of IoT cyberattacks [3] are:

- exponential growth of IoT devices amount;
- dangerous deployment of IoT devices with the possibility of direct access to them via the Internet;
- lack of security updates for IoT devices, that makes them vulnerable to already known attacks;
- lack of transparency regarding the security status of IoT devices.

Vulnerable components in the IoT infrastructure can be [4]:

© Kira Bobrovnikova, Sergii Lysenko, Bohdan Savenko, Piotr Gaj, Oleg Savenko, 2022
IoT devices, that are usually the main means of initiating attacks. Malicious users can use, in particular, unsecured default settings, legacy components and unsecured update mechanisms;

- channels connecting IoT components. Protocols used in IoT systems can have security issues that can affect the entire infrastructure;

- applications and software.

IoT vulnerabilities in web applications and software are able to compromise systems, such as stealing user credentials or sending malicious updates. In this situation, any smart IoT device, such as a coffee maker, can be a source of cyberattacks and affect critical organization systems such as Intranet and database servers, through the ability to collect data and monitor IoT systems [1].

At the same time, the motivation of criminals can be different: terrorist acts for political or other purposes, theft of information that is a trade secret, blackmail and extortion for financial gain, revenge or entertainment. The Fourth Industrial Revolution provides new possibilities for cyber-criminals that employ networks, web bots, botnets etc. [9–12]. The inability to apply security or monitoring solutions on IoT devices is a major problem to prevent malicious activity in the IoT infrastructure. Thus, the growing number of cyberattacks caused by malware on IoT infrastructure [13–16] is currently a very important issue that needs to be addressed.

The aim of this paper is to develop a technique for IoT malware detection based on control flow graph analysis.

The paper structure is the following.

Section 1 presents the state-of-art works section – a brief analysis of the very modern and the latest ideas and methods addressed to solve the problem of IoT malware detection with its advantages and disadvantages.

Sections 2 discusses the stages of the proposed technique for IoT malware detection based on control flowgraph analysis.

Section 3 describes the experiments. In addition, conclusions present further work concerning the IoT malware detection.

1. Related works

There are a large number of various works devoted to the problem of preventing, detecting cyberattacks on the IoT infrastructure.

In [17] several methods to mitigate the attacks on IoT infrastructure have been developed. One of the methods is based on opcodes analysis to classify IoT malware. These opcodes are retrieved from disassembled IoT malware programs. To evaluate the effectiveness of the proposed approach supervised machine learning classifiers such as Support Vector Machine, Decision Tree, Random Forest, Naïve Bayes and K-Nearest Neighbor and data set of 512 IoT malware samples and benign application have been used. Results of experiments showed that the Random Forest classifier performed better than other classifiers.

In [18] a wrapper technique-based feature selection algorithm to filter the features accurately and to select effective features for the machine learning algorithms was developed.

In [19] the prospects of using machine learning classification algorithms for securing IoT infrastructure against attacks were investigated.

In [20] the security issues inherent to federated learning paradigm and possibilities enabled by this paradigm concerning IoT malware detection. A framework for IoT malware detection based on federated learning was presented. To evaluate the proposed framework the dataset N-BaIoT which modeling network traffic of real IoT devices infected by malware, has been used. Two federated models - supervised and unsupervised (multi-layer perceptron and autoencoder) - were evaluated. The performance of both models has been compared with using two traditional approaches. The first approach allowed each participant train a model locally with only its own data. The second approach used to share data by the participants with a central entity which is responsible of training a global model. It was concluded that the use of more diverse data has a considerable positive impact on the performance of model. However, the federated models showed similar results as the centralized, and wherein they allowed preserving the participant’s privacy. At the same time, the aggregation averaging step of baseline model which was used in most of federated learning algorithms was highly vulnerable to different attacks.

The paper [21] focuses on capturing the attacks in IoT networks by using Cowrie honeypot.

In paper [22] deep learning model DBN-IDS (Deep Belief Network intrusion detection system) was proposed. For experiments conduction the dataset was utilized which contains the examples of attacks caused by malware.

In [23] a web attack detection system which is based on URLs analysis and distributed deep learning was developed. It employs several concurrent deep models which deploys in different servers and trains separately. The overall detection results are obtained from these concurrent deep models to enhance the stability of the system and the convenience in updating.

In paper [24] deep learning models were proposed and evaluated using datasets for attack detection. Also, aspects of using deep learning algorithm for IoT cyberattack detection were reviewed.
In [25] an intelligent ADE-based (the averaged dependence estimator) framework for IoT attack detection was presented.

In [26] Security Information and Event Management-based system which detects and blocks IoT malicious traffic by monitoring of the specific packet types including TCP SYN, ICMP and DNS packets was proposed.

The paper [27] proposes a decentralized security architecture that relies on the three core technologies in order to detect the IoT attacks: Software Defined Networking (SDN) to continuously monitor and analyze the traffic data; Blockchain which supports decentralized attack detection to overcome the single point of failure problem; Fog and mobile edge computing which contributes to lesser storage constraints, cheaper computation, and shortening the time taken to detect and mitigate attack.

In [28] a security scheme named learning-driven detection mitigation (LEDEM) that detects and mitigates attacks was proposed. LEDEM scheme is based on semi-supervised machine-learning and leverages the cloud and software-defined network (SDN) paradigm to mitigate the attacks.

Majority of machine learning approaches for malware detection require a significant prior knowledge about malware features. To solve the problem of limited availability of malware samples in [29] a framework for generating and detecting new IoT malware samples was proposed. For this purpose, the raw byte code at the edge layer of the IoT networks was utilized. Convolutional Neural Network was used for high-level features extraction. To generate new IoT malware samples technique of boundary-seeking Generative Adversarial Network was used. To capture the long-term and short-term features dependency an attention-based model, a combination of CNN and LSTM (Long Short-Term Memory) were applied. The attention mechanism allows improving the performance of model by decreasing or increasing attention to certain parts of the malware features.

In [30] two defense approaches against attack to a IoT malware detection system for mobile multimedia applications were proposed. They are CNN and 1-nearest neighbors (C4N) combination and Robust-NN. These approaches allow to modify training data set that has been poisoned by an attack. Thus, the trained machine learning model accuracy is improved, and if the malicious IoT program is executing on any IoT device, the model generates the alerts.

In [31] a dynamic analysis-based approach for new and well-known IoT malware detection was proposed. The proposed approach used the convolution neural network model for dynamic analysis IoT malware in cloud environment. It extracted behaviors related to process, system call, memory, virtual system and network. Extracted and analyzed behavior data are converted into the IoT malware behavior images. These images are classified with using the Convolution Neural Network.

In the paper [32] thorough survey of approaches based on static IoT malware detection was conducted. The definition, security threats and evolution of IoT malware were introduced. The latest IoT malware detection approaches were analyzed and compared. IoT malware detection approaches were divided into two groups: graph-based and non-graph-based methods. The non-graph based approaches can be useful when detecting known malware without obfuscation or customization. But these methods lose accuracy for unknown malware detection. The graph-based approaches are based on analysis of the control flow and therefore characterized by certain complexity. Wherein these approaches show advantages to accurately detect complicated or unknown IoT malware. Thereafter these approaches were tested and evaluated with using the same experimental configuration and IoT malware dataset, consisting of 11200 samples. The advantages and limitations of these approaches summarized that they can be used to improve the efficiency of IoT malware detection.

In [33] MQTTset – the legitimate dataset related to the MQTT protocol, which can be used to train machine learning models to implement the detection systems ability to protect IoT networks was described.

Despite the large number of different methods for detecting and mitigating cyberattacks caused by malware on the IoT infrastructure, the steady increase in their number confirms that this problem is not solved today.

2. Technique for IoT malware detection based on control flowgraph analysis

The proposed method is based on the analysis of control flow graphs built for IoT applications. It allows detecting unknown IoT threats caused by malicious software, which are the modified versions of known IoT threats. The control flow graph is represented as a directed graph, which contains information about the components of the program and the transitions between them, taking into account unconditional transitions, branches, cycles, function calls and exclusions. The control flow graph describes the control flows that may occur during the program execution.

Typically, methods based on CFG analysis deal with the NP-complete problem when are used for large codes analysis. However, IoT applications are small in size due to the simplicity and limitations of the IoT op-
erating systems environment. This makes it possible to effectively detect IoT threats based on CFG analysis.

The proposed method consists of the following stages:
- creation of training/testing data;
- system training;
- system testing and evaluation of detection accuracy;
- IoT malware detection.

Let’s consider the steps of the stages of the proposed method.

2.1. Stage of Training / Testing Data Creation

The steps of the stage of training/testing data creation are shown in Fig. 1.

In order to create training /test data, a set of malware and benign IoT applications are disassembled.

For each IoT application the control flow graph from source codes is constructed.

Let’s denote the control flow graph for IoT application $\upsilon$ as:

$$ G_{\text{CFG}}(F,R) = \langle F, R \rangle, $$

where $F=\{f_i\}_{i=1}^O$ – the nodes (vertices) of the CFG, which present functions of the IoT application $\upsilon$;

$R=\{r_i\}_{i=1}^S$ – the edges of the CFG, which present the call relationship between $f_i$ in $\upsilon$;

$O$ – is the order of the CFG;

$S$ – is the size of the CFG.

At the next step the obtained control flows graphs are to be analyzed in order to evaluate the set of metrics (Table 1), that characterize the structure of the CFG and thus describe the structure of the IoT application code.

In order to process the control flow graph $G_{\text{CFG}}(F,R)$ it can be presented as the adjacency matrix $A_{\text{CFG}} = \{a_{ij}\}_{i,j=1}^{O,O}$. Then let us denote the CFG metric extraction and CFG metrics feature vectors $\tilde{M}$ formation function $\phi_1$ as: $\phi_1(A_{\text{CFG}}) \rightarrow \tilde{M}$, where $\tilde{M}$ – is feature vector formed from the metrics obtained for each IoT application $\upsilon$, $\tilde{M}=\{O,S,m_1..m_B\}$.

The set of feature vectors $\tilde{M}$ are further marked as one of the IoT malware families class or a benign IoT application and are added to the database of training / testing data.

Also, in order to analyze the behavior of the IoT application for each of the control flow graph $G_{\text{CFG}}(F,R)$ the actions graph $G_B(\mathbf{B}, \mathbf{E})$ is to be built. It is an abstract graph description of the program. Such description consists of IoT application’s actions (for example, API calls). Let denote the actions graph $G_B(\mathbf{B}, \mathbf{E})=\langle \mathbf{B}, \mathbf{E} \rangle$, where $\mathbf{B}=\{b_i\}_{i=1}^L$ – the nodes of the $G_B(\mathbf{B}, \mathbf{E})$, each of which presented the action made by IoT application $\upsilon$; $L$ – the number of nodes;

$\mathbf{E}=\{e_i\}_{i=1}^P$ – the edges of the $G_B(\mathbf{B}, \mathbf{E})$, which presented transitions between $b_i$; $P$ – the number of edges.

On the basis of the obtained actions graph $G_B(\mathbf{B}, \mathbf{E})$ for each IoT application $\upsilon$ the set of sequences of actions $\mathbf{B}_{\text{Act}}$ is formed.

---

**Figure 1. Stage of training / testing data creation**
The set of CFG metrics that characterize the structure of the IoT application

<table>
<thead>
<tr>
<th>Metric</th>
<th>The metrics value</th>
</tr>
</thead>
<tbody>
<tr>
<td>O</td>
<td>the order of graph (the number of nodes)</td>
</tr>
<tr>
<td>S</td>
<td>the size of graph (the number of edges)</td>
</tr>
<tr>
<td>m_1</td>
<td>the radius – the minimum distances among all the maximum distances (the edges number in a shortest path) between a node to all other nodes</td>
</tr>
<tr>
<td>m_2</td>
<td>the diameter – the maximal distance (the edges number in a shortest path) between the pair of nodes</td>
</tr>
<tr>
<td>m_3</td>
<td>The graph density – the ratio of the number of graph edges with respect to the maximum possible number of edges</td>
</tr>
<tr>
<td>m_4</td>
<td>the shortest path in the graph</td>
</tr>
<tr>
<td>m_5</td>
<td>the centrality degree is defined as the number of links incident upon a node</td>
</tr>
<tr>
<td>m_6</td>
<td>the normalized closeness centrality – the reciprocal of the sum of the shortest path’s length between the graph node and all other nodes</td>
</tr>
<tr>
<td>m_7</td>
<td>the normalized betweenness centrality – the number of shortest paths that pass through the node</td>
</tr>
<tr>
<td>m_8</td>
<td>the number of components – number of subgraphs in which any two nodes are connected to each other by paths and is not connected to other nodes in the rest of the graph</td>
</tr>
</tbody>
</table>

Let us denote the formation function \( \varphi_2 \) for actions sequences \( \text{B}_{\text{Act}} \) as: \( \varphi_2 = (G_{B}(B,E)) \rightarrow \text{B}_{\text{Act}} \), where \( \text{B}_{\text{Act}} = \{B_i\}^3_{i=1} \).

Let us denote the actions sequences as following:

- \( B_1 = \{b_i\}^N_{i=1} \) – a sequences of unique actions, that determines what unique actions are made by the IoT application, where \( N \) is the number of unique actions;
- \( B_2 = \{b_i\}^K_{i=1} \) – a sequences of actions, which determines the order of actions in time, where \( K \) is the total number of all actions;
- \( B_3 = \{b_n\}^N_{i=1} \) – a sequences of actions, which determines how many times each action is uses, where \( n \) is the number of times the action was executed.

At the next step, the Multiple Sequence Alignment algorithm (MSA) is applied to the resulting actions sequences, then they are also marked and added to the database of training / testing data.

Further, the resulting database \( D \) is divided into two parts: training data \( T \) and test data \( E \) samples: \( D = T \cup E \).

Thus, each of training and test data samples consists of two subsets: \( T = T_M \cup T_B \) and \( E = E_M \cup E_B \), where \( T_M \) and \( E_M \) are CFG metric-based data, which are built from the feature vectors \( \tilde{M} \); \( T_B \) and \( E_B \) are data, which are built from actions sequences \( \text{B}_{\text{Act}} \).

### 2.2. Stages of System Testing and Evaluation of Detection Accuracy

The steps of the stage of system testing and evaluation of detection accuracy are shown in Fig. 2.

At this stage the system is trained. For this purpose, we use the training data \( T \).

After that we use the test data \( E \) for two malware detection models based on CFG metric and on sequences of actions.

Mentioned models make it possible to obtain its accuracy score that indicates that the IoT application belongs to a particular malware class or is a benign application.

These accuracy score obtained by each of models is further applied as weight and will be used to evaluate the overall result via the soft voting method.

### 2.3. IoT Malware Detection Stage

The steps of the IoT malware detection stage are shown in Fig. 3. In order to detect IoT malware for each suspicious IoT application the control flow graph is built. On the basis of each constructed graph the feature vectors \( \tilde{M} \) of CFG metrics that characterize the structure of the IoT application is formed. Also, on the basis of each constructed graph the sequences of actions set \( \text{B}_{\text{Act}} \) are formed which describe behavior of IoT application.

After that, a decision on the presence or absence of IoT malware using both attack detection models is made.

The classifier in each of two malware detection models provides the prediction that a suspicious application belongs to a malicious or benign class. The prediction of each malware detection models is weighted according to the previously obtained detection accuracy and are summed up.

Let’s define an IoT malware detection function \( \varphi_3 \) as:

\[
\varphi_3: \{\psi((\tilde{M},\psi(\varphi_1))\times\varphi_2,\psi(\varphi_2))\} \rightarrow D, \quad (2)
\]
3. Experiments

In order to evaluate the efficiency of the proposed approach the experiments were held. They were based on the usage of the BotGRABBER – cyberattacks’ detection tool – was used [34–38].

In order to conduct the experiments, 5762 samples of IoT malicious applications such as Mirai, Ares, IPStorm, ADB.miner, Pbot, Matryosh, Trinity, Guerrilla, Hiddad, Hajime, Gafgyt and other from [39–41] and 1824 samples of IoT benign applications collected from Google Play Store [42] and [43] were used.

With purpose of the IoT applications disassembling as well as the CFG’s and the actions graphs construction, a framework for reverse-engineering and binaries analyzing Radare2 [44] was used.

For CFG metrics to obtained a library for studying graphs and networks NetworkX [45] was employed. For each sample of the IoT application from the obtained CFG metrics the feature vectors \( \bar{M} \) were formed. In addition, the actions series \( B_{\text{Act}} \) from the action graphs were extracted.

The obtained feature vectors and actions series were labelled as malicious or benign IoT class, respec-
tively. Further they were split into three parts in the ratio of 2:3:5 to create:

1. The training data.
2. The test data E that is used to obtain the accuracy score of each of two malware detection models based on CFG metric and on sequences of actions are used.
3. Data for reliability assessment of proposed method, respectively.

For the experiments conducting the following machine learning algorithms for each detection model were employed: C-means, Support Vector Machine, Naïve Bayes, K-Nearest Neighbors, Random Forest and Rotation Forest [46-49].

The prediction of each detection models is weighted according to its detection accuracy and are summed up.

In order to evaluate the efficiency of the proposed approach the following quantity measures were used:

\[
SN = \frac{TP}{TP+FN},
\]

\[
SP = \frac{TN}{TN+FP},
\]

\[
Q = \frac{(TP+TN)}{(TP+TN+FP+FN)},
\]

where TP – True Positives;
 TN – True Negatives;
 FP – False Positives;
 FN – False Negatives;
 SN – Sensitivity;
 SP – Specificity;
 Q – Overall Accuracy.

The obtained results are presented in the Table 2, and they showed that detection efficiency is up to 98 %, while the false positives is about 3-5 %.

**Conclusion**

The research presents the new technique for IoT malware detection based on control flow graph analysis. It is based on the analysis of control flow graphs built for IoT applications.

Proposed approach allows detecting unknown IoT threats caused by malicious software, which are the modified versions of known IoT threats.

The main core of the approach is the usage of the control flow graph that is represented as a directed graph and contains the information about the components of the program. Developed approach is based on CFG analysis.

As the mean of conclusion making concerning the malware presence a set of machine learning classifiers were employed.

Experimental results demonstrated that the proposed technique’s detection efficiency is up to 98 %, while the false positives is about 3-5 %.

Further work will be devoted to techniques for IoT malware detection that may be further improved by choosing a more refined set of malicious samples, as well as the involving different machine learning algorithms.

**Contribution of authors:** developed the conceptual provisions of the technique for IoT malware detection based on control flow graph analysis – **K. Bobrovnikova**; performed the review and analysis of references devoted to the problem of preventing, detecting cyberattacks on the IoT infrastructure – **S. Lyseenko**; held the experimental research and provided main results – **B. Savenko** and **P. Gaj**; made the introduction section of the paper – **O. Savenko**. All authors have read and agreed to the published version of the manuscript.

### Table 2

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Number of evaluation samples</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Malicious</td>
<td>Benign</td>
</tr>
<tr>
<td></td>
<td>TP</td>
<td>FN</td>
</tr>
<tr>
<td>c-means</td>
<td>7586</td>
<td>44</td>
</tr>
<tr>
<td>SVM</td>
<td>7586</td>
<td>98</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>7586</td>
<td>19</td>
</tr>
<tr>
<td>K-Nearest Neighbors</td>
<td>7586</td>
<td>78</td>
</tr>
<tr>
<td>Random Forest</td>
<td>7586</td>
<td>91</td>
</tr>
<tr>
<td>Rotation Forest</td>
<td>7586</td>
<td>85</td>
</tr>
</tbody>
</table>
References (GOST 7.1:2006)


4. OWASP Internet of Things. Available at: https://owasp.org (accessed 11.01.2022).


40. IoT POT. Available at: https://sec.ynu.codes/iot/ (accessed 11.01.2022).


43. OpenWrt. Available at: https://openwrt.org/ (accessed 11.01.2022).


45. NetworkX. Available at: https://networkx.org/ (accessed 11.01.2022).


МЕТОД ВИЯВЛЕННЯ ЗЛОВМІСНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТУ РЕЧЕЙ НА ОСНОВІ АНАЛІЗУ ГРАФОВ ПОТОКІВ КЕРУВАННЯ

К. Ю. Бобровникова, С. М. Лисенко, Б. О. Савенко, П. Гай, О. С. Савенко

Під Інтернетом речей розуміють мільйони пристроїв у всьому світі, підключені до Інтернету. Незахищені пристрої Інтернету речей, розроблені без належних функцій безпеки, є метою багатьох Інтернет-загроз. Швидка інтеграція Інтернету в інфраструктуру Інтернету речей у різних сферах людської діяльності, включаючи визнання критично важливі інфраструктури, робить виявлення зловмісних програм в Інтернеті речей все більш важливим. Щорічні звіти компаній, що займаються кібербезпекою інфраструктур INTERNET речей, та виробників антивірусного програмного забезпечення показують зростання кількості атак зловмісного програмного забезпечення, спрямованих на інфраструктуру Інтернету речей. Це свідчить про неспроможність сучасних методів виявлення зловмисних програм в Інтернеті речей. Ось чому існує нагальна потреба в розробленні нових підходів до виявлення зловмисних програм в Інтернеті речей і захисту пристроїв Інтернету речей від атак зловмисного програмного забезпечення Інтернету речей. Підхід дозволяє виявляти невідомі зловмисні програми, а також побудувати граф дій. Це абстрактний опис програми у вигляді графа. На основі отриманого графа дій для кожного додатка Інтернету речей формується набір послідовностей дій. Це дозволяє визначити поведінку програми Інтернету речей, яка є можливою у середовищі Інтернету речей. Коли відбувається невідома дія програми Інтернету речей, вона може викликати відповідну деякі послідовності дій.

Висновки.

На основі проведеного дослідження є можливим використання графів потоків інтернету речей в якості інструментів для аналізу структури дії програми Інтернету речей на основі зловмисних програм. Сучасні методи виявлення зловмисних програм в Інтернеті речей мають невеликі розміри через простоту обчислів, яка сприяє розробці нових підходів до виявлення зловмисних програм в Інтернеті речей.

Ключові слова: зловмисні програмне забезпечення; Інтернет речей; пристрої Інтернету речей; додаток для аналізу графів потоків інтернету речей; виявлення кіберзагроз.

МЕТОД ВИЯВЛЕННЯ ВРЕДОНОСНОГО ПРОГРАМНОГО ОБЕСПЕЧЕННЯ ІНТЕРНЕТУ ВЕЩЕЙ НА ОСНОВЕ АНАЛІЗУ ГРАФОВ ПОТОКІВ УПРАВЛІННЯ

К. Ю. Бобровникова, С. Н. Лысенко, Б. О. Савенко, П. Гай, О. С. Савенко

Под Інтернетом вещей понимают миллионы устройств во всем мире, подключенные к Интернету. Не защищенные устройства Интернета вещей, разработанные без надлежащих функций безопасности, это метка большого Интернет-угроз. Быстрая интеграция Интернета в инфраструктуру Интернета вещей в различных сферах человеческой деятельности, включая важнейшую критически важную инфраструктуру, делает обнаружение вредоносных программ в Интернете вещей все более важным. Ежегодные отчеты компаний, занимающихся кибербезопасностью инфраструктуры Интернета вещей, и производителей антивирусного программного обеспечения показывают рост количества атак вредоносного программного обеспечения, направленных на инфраструктуру Интернета вещей. Это свидетельствует о несостоятельности современных методов обнаружения вредоносных программ в Интернете вещей. Вот почему существует острая потребность в разработке новых подходов к выявлению вредоносных программ в Интернете вещей и защите устройств Интернета вещей от атак вредоносного программного обеспечения Интернета вещей. Метод исследования является процессом обнаружения вредоносных программ в Интернете вещей. Цель работы является разработка метода обнаружения вредоносного программного обеспечения в Интернете вещей на основе анализа графа потока управления. Результаты. В статье представлен новый подход к выявлению вредоносного
программного обеспечения в Интернете вещей на основе анализа графа потока управления. Графы потока управления строятся для подозрительных приложений Интернета вещей. Граф потока управления представляет в виде ориентированного графа, представляющего компоненты подозрительной программы и переходы между ними. На основе графа потока управления можно извлечь метрики, описывающие структуру программы. Учитывая, что программы Интернета вещей имеют небольшие размеры из-за простоты и ограничения среды операционных систем Интернета вещей, обнаружение вредоносного программного обеспечения на основе анализа графа потока управления возможны в среде Интернета вещей. Чтобы проанализировать поведение программы Интернета вещей, для каждого графа потока управления нужно построить граф действий. Это абстрактное описание программы в виде графа. На основе полученного графа действий для каждого приложения Интернета вещей формируется набор последовательностей действий. Это позволяет определить поведение программы. Таким образом, для выявления вредоносного программного обеспечения Интернета вещей используются две модели обнаружения вредоносных программ: основная и на метриках, изъятых из графа потока управления, и на последовательностях действий. Благодаря тому, что подход позволяет анализировать как общую структуру программы, так и поведение каждого приложения, он позволяет добиться высокой точности обнаружения вредоносных программ. Предлагаемый подход позволяет выявлять неизвестные вредоносные программы Интернета вещей, которые являются модифицированными версиями известных вредоносных программ Интернета вещей. В качестве средства вывода о наличии вредоносного программного обеспечения использовался набор классификаторов на основе машинного обучения. Результаты экспериментов продемонстрировали высокую точность обнаружения вредоносных программ Интернета вещей. Выводы. Разработан новый метод обнаружения вредоносных программ Интернета вещей на основе анализа графа потока управления. Метод способен с высокой эффективностью выявлять вредоносное программное обеспечение в Интернете вещей.

Ключевые слова: вредоносное программное обеспечение; Интернет вещей; устройства Интернета вещей; приложение Интернета вещей; кибербезопасность; кибератака; граф потока управления; обнаружение киберугрозы.

Бобровникова Кира Юліївна — к.т.наук., доц. каф. комп`ютерної інженерії та інформаційних систем, Хмельницький національний університет, Хмельницький, Україна.

Лисенко Сергій Миколайович — д-р техн. наук, проф., доц. каф. комп’ютерної інженерії та інформаційних систем, Хмельницький національний університет, Хмельницький, Україна.

Савенко Богдан Олегович — асп. каф. комп’ютерної інженерії та інформаційних систем, Хмельницький національний університет, Хмельницький, Україна.

Гай Пьotr — д-р техн. наук, інститут комп’ютерних наук, Сілезький технологічний університет, Глівіце, Польща.

Савенко Олег Станіславович — д-р техн. наук, проф., декан факультету інформаційних технологій, Хмельницький національний університет, Хмельницький, Україна.

Kira Bobrovnikova — PhD, Associate Professor of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: bobrovnikova.kira@gmail.com, ORCID: 0000-0002-1046-893X.

Sergii Lysenko — Doctor of Science, Full Professor, Professor of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: sirogyk@ukr.net, ORCID: 0000-0001-7243-8747.

Bohdan Savenko — PhD student of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: savenko_bohdan@ukr.net, ORCID: 0000-0001-5647-9979.

Piotr Gaj — Doctor of Science, Institute of Computer Science, Silesian University of Technology, Gliwice, Poland, e-mail: piotr.gaj@polsl.pl, ORCID: 0000-0002-2291-7341.

Oleg Savenko — Doctor of Science, Full Professor, the Dean of the Information Technologies Faculty of National University, Khmelnytskyi, Ukraine, e-mail: savenko_oleg_st@ukr.net, ORCID: 0000-0002-4104-745X.