electronics

Article

Toward Network Worm Victims Identification Based on Cascading Motif Discovery

Hangyu Hu, Mingda Wang, Mingyu Ouyang and Guangmin Hu *

School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; huhangyuuestc@gmail.com (H.H.); behold.philly.wmd@gmail.com (M.W.); 201721010810@std.uestc.edu.cn (M.O.)

* Correspondence: hgm@uestc.edu.cn; Tel.: +86-28-6183-0205

Received: 24 December 2018; Accepted: 2 February 2019; Published: 5 February 2019

Abstract: Network worms spread widely over the global network within a short time, which are increasingly becoming one of the most potential threats to network security. However, the performance of traditional packet-oriented signature-based methods is questionable in the face of unknown worms, while anomaly-based approaches often exhibit high false positive rates. It is a common scenario that the life cycle of network worms consists of the same four stages, in which the target discovery phase and the transferring phase have specific interactive patterns. To this end, we propose Network Flow Connectivity Graph (NFCG) for identifying network worm victims. We model the flow-level interactions as graph and then identify sets of frequently occurring motifs related to network worms through Cascading Motif Discovery algorithm. In particular, a cascading motif is jointly extracted from graph target discovery phase and transferring phase. If a cascading motif exists in a connected behavior graph of one host, the host would be identified as a suspicious worm victim; the excess amount of suspicious network worm victims is used to reveal the outbreak of network worms. The simulated experiments show that our proposed method is effective and efficient in network worm victims' identification and helpful for improving network security.

Keywords: network worms; network flow connectivity graph; flow behavior analysis; motif discovery; network security

1. Introduction

Nowadays, network administrators increasingly rely on Intrusion Detection Systems (IDSs) to ensure security during network communication. However, there is an increasing trend that attackers are able to launch attacks by utilizing a large number of hosts instead of an individual host, which could have widespread impact on the entire network. For instance, attackers can scan large numbers of hosts concurrently in order to search for computer vulnerabilities (e.g., network scans); then, use a self-replicating program to propagate their malicious codes to target vulnerable hosts in a short time (e.g., network worms); and, finally, they can use those compromised hosts to flood a targeted system or host to disturb or disrupt its service (i.e., Distributed Denial-of-Service attack) [1–4].

In the present study, we focus more on network worm detection and evaluate whether the proposed approach can be extended to other types of attacks. There are two main purposes for attackers to launch network worms: (1) causing a traffic overload in the networks and congestion on backbone links, which disrupt affected hosts and lead to financial losses [5] and (2) recruiting compromised hosts for future attack use [6], as illustrated in Figure 1. For example, in 2013, the Symantec Internet security threat report [7] pointed out that target attacks increased by 42% in 2012, of which 31% were using worms to attack business; whereas, in 2016, Kaspersky laboratory [8] reports claimed that, between 2014 and 2015, the global economic machine was affected by worms, viruses, Trojan horses, and other malware attacks. The damage was approximately $1 billion. Even more seriously, although people can do much in the way of prevention, new types of the network worms are found from time to time, such as polymorphic worms, which can successfully avoid the capture of detection tools or antivirus software and infect millions of hosts in a few seconds. Network worms increasingly pose a great threat to network-based military equipment, economic institutions, and communication networks. Hence, fast and accurate detection and identification of network worms play an essential role in establishing a secure, stable, and reliable network environment that covers a wide range of research in the field of network security and management and has attracted significant attention from both the academia and industry all over the world.

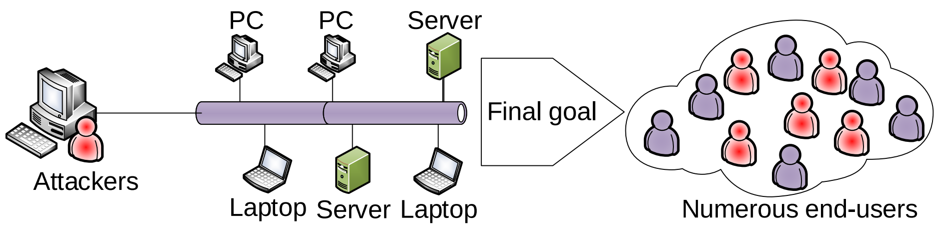

Figure 1. Worm propagation in the network.

The approaches to the worm detection problem can be divided into two categories. Most network-worm detection methods begin by inspecting each file that enters a system (i.e., antivirus software packages) and then looks for known signatures hiding in network traffic payloads; these are known as signature-based approaches [9–11]. The main drawback of these approaches is that their performance is questionable, particularly against unknown worms (a previously unknown version or unreleased worm) or polymorphic or metamorphic worms. However, anomaly-based approaches focus on the identification of observations that do not conform to an expected pattern from normal network traffic instead of looking for fixed regular expressions in payloads. Examples of these techniques can be found in the literature [12–17]. Unlike signature-based methods, anomaly-based methods often have high false positive rates since it is difficult to determine the boundary between normal and abnormal patterns. Therefore, network worm detection is still a challenging problem for network administrators.

However, many researchers neglect to focus on a common scenario: that the life cycle of network worms typically comprises the same stages. In addition, network worms have their own specified and similar connected patterns in each stage. After a network worm has been released, the life cycle of a network worm consists of four stages: (1) target discovery, in which the worm scans for vulnerable hosts and then exploits the vulnerability to prepare for the next stage; (2) transferring the malicious code to the target hosts; (3) activation and (4) infection [16,18]. During the target discovery and transferring stages, network worms can be detected by a series of tools, such as anti-virus software, IDSs, and firewalls [18]. However, network worm behaviors in the last two stages are limited since it is difficult to detect the network worm by IDSs in these activities [5]. A typical example is the existence of anomalous flows generated by scanning in identifying the vulnerable hosts as well as worm propagation [19,20].

In this paper, the approach proposed is mainly relying on detecting the inherent connected patterns of worms during the target discovery and transferring stages. Ideally, these patterns can be distinct from normal network traffic, just like a network "footprint" for worms. Thus, we take a moderate stand based on the network flows that record communication between a source and destination IP addresses to capture those suspicious "footprints" of network worms. In our previous work, we present the advantage of leveraging a novel graph model called Network Flow Connectivity Graphs (NFCG) to comprehensively study network-wide flow connectivity relationships. In an NFCG, $G = (V, E)$ is defined as a set of vertices V (representing network entities, IP addresses in our case) and edges E (representing interaction relationships between pairs of vertex sets). We could build a series of interaction graphs to capture a whole worm propagation process under an NFCG. An NFCG is consisting of multiple types of motifs, a small, induced subgraph structure that can be seen as the basic cell of a graph. All of the motifs can be used to describe specific and similar connected patterns of network worms.

Thereafter, we developed a Cascading Motif Discovery (CMD) algorithm to capture connected patterns that exhibit typical characteristics of network worms. The CMD algorithm consists of two phases: the scanning motif discovery (SMD) phase, which is used to find the presence of the scanning behaviors in the initial stage of a network worm, and the transferring motif detection (TMD) phase, which is used to detect the similar transferring behaviors within network worms. A host would be identified as a worm victim if it performs network scanning activities and similar transferring behaviors. Experimental results with a simulated dataset obtained from the Georgia Tech Network Simulator (GTNetS) suggest that our approach is effective and efficient in identifying worm victims and detecting the outbreak of network worms. The main contributions of the present study are summarized as follows:

The main contributions of the present study are summarized as follows:

- We reveal the characteristics flow connected patterns during a worm outbreak. The target discovery and transferring stages of the worm life cycle have a specific fan-out connection mode, and the cascading appearance and large repetition of these two connection modes are typical characteristics of the flow-connected behavior of network worms.
- We propose a technique for worm detection based on network-flow-connected behavior analysis. Additionally, we develop a novel solution for building the worm-life-cycle motifs. Unlike the traditional motif discovery methods in a complex network, we focus on the inherent behavior of network worms and reveal the specific motif classes that conform to the essential characteristics of a worm outbreak.
- Due to the scalability of worm connection behaviors and the complexity of subgraph mining, the CMD algorithm realizes the decomposition of worm attacks in different scales and solves the scale problem of subgraph matching.
- We conduct many experiments by collecting dataset through a mature simulation system. The experimental results demonstrate that our approach effectively detect regardless of known released worms or never-seen-before worms.

The remainder of this paper is organized as follows: in Section 2, we introduce the preliminaries and background of the Network Flow Connectivity Graphs (NFCG) and network motif. Section 3 presents the algorithm of using Cascading Motif Discovery for network worm victims' identification and gives an illustrative example for the proposed algorithm. In Section 4, we briefly introduce the GTNetS simulation platform for network worm propagation and present the evaluation results of the proposed approach. The discussion of our proposed method is presented in Section 5. Finally, Section 6 gives the conclusions of our work and presents possible future works.

2. Preliminaries and Background

This section first introduces network flows, and then we provide the construction of Network Flow Connectivity Graph (NFCG) and present the visualization of NFCGs. Finally, we describe the basic concept and application of the network motif and explain how to apply the motif to worm detection.

2.1. Network Flows

Network flows that record the interaction process between source and destination hosts are the information carriers of network operation. In literature, there are several definitions of an IP flow can be found in [21–23]. Traditional network flows give the data (timestamp, IP addresses, port number, protocol, etc.) to describe the interaction procedure between a pair of IP addresses. Since each communication between source and destination hosts will generate a flow, the behavioral characteristics of network flows could be used to represent the communication patterns among network servers, end-users, switching-devices and various application services, the architecture of network flow collecting and processing process. In general, the architecture of network flow collecting and processing process contains the flow exporter module and flow collector module [24], as Figure 2 shows.

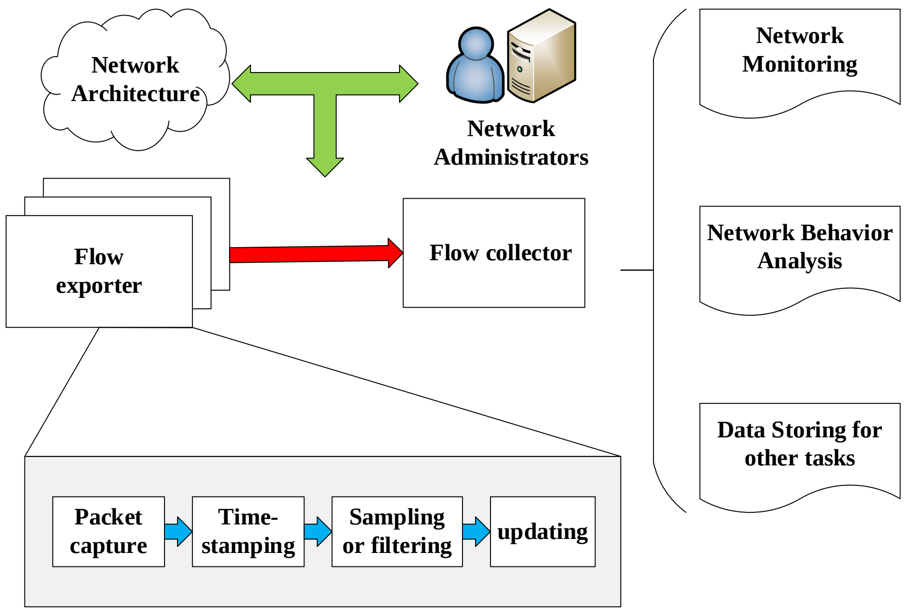

Figure 2. The architecture of network flow exporting and collecting procedure.

2.2. *The Network Flow Connectivity Graph*

The detailed processes of NFCG construction are given as in Algorithm 1:

Algorithm 1 Constructing Network Flow Connectivity Graph

```
Input: Network flow information sequence $N =< H, F >$, $H$: sets of hosts, $F$: sets of flows;
Output: Network Flow Connectivity Graph $G =< V, E, R, W >$;
function GENERATING AGGRE-FLOWS($AF$)
    for pairs of hosts $h_i$ and $h_j$ do
        (1) extract all communication flows $F_(ij)$ between $h_i$ and $h_j$
        (2) create aggre-Flows set $AF_{ij} = \{AF_1^{ij}, AF_2^{ij}, AF_3^{ij}, \cdots, AF_n^{ij}\}$, the set of aggregated flows originating from $h_i$ to $h_j$
    end for
end function

function CONSTRUCT NETWORK FLOW CONNECTIVITY GRAPH($G =< V, E, R, W >$)
    for All aggre-Flows between $h_i$ and $h_j$ do
        (1) set up the vector set $\alpha_v$ to represent the volume of flow connection, $\alpha_v = \sum_{k=1}^{n} Vol(AF_k^{ij})$
        (2) set up the vector set $\alpha_{pn}$to represent the number of open ports opened during the time interval, $\alpha_{pn} = \sum_{k=1}^{n} Nop(AF_k^{ij})$
    end for
    for each $h \in H$ do
        ranking nodes based on the value of flow properties, and divide nodes into different levels $R_v$ (with different colors); weighting the edge $e_{ij}$ between $h_i$ and $h_j$ by $W_{ij} \Leftarrow (h_i, h_j, \alpha_v, \alpha_{pn})$
    end for
end function

return Network Flow Connectivity Graph module $G =< V, E, R, W >$;
```

A. Network flow information sequence extraction

In this paper, we introduce the concept of aggre-flow, which is defined as meeting the following two conditions: (a) these flows are between the same pair of IP addresses regardless of their protocol and opened port number, and (b) the time interval between network flow should be relatively small, so that two flows would not have any associated relationships for a long time. The steps for generating network information sequence are as follows:

- Collect network traffic data and build network flow trace using the standard definition of the five-tuple flow.
- Trace initial network flow and consider every flow to be an independent network.
- Merge two flows into an aggre-flow when two flows are between the same pair of IP addresses regardless of their protocol and opened port number.
- Repeat step 3 until there are no more flows to aggregate; then, you have the new network flow information sequence.

B. The original structure of NFCG construction using the flow information

Based on the new network flow information sequence, the original graph $G =< V, E >$ is defined as a set of vertices, $v_i \in V$, which represent network entities, and edges, $e_{ij} \in E$, which represent connection between pairs of vertices. However, in modeling a computer network, the traditional method [25,26] has correlated each unique IP address with a node in the graph, whereas edges were used to represent the interactions from a source host to a destination host. This is depicted in Figure 3 as an interaction graph. However, the edges in the interaction graph only indicate whether there exists a connection between two entities, but they cannot provide more information about network flow behaviors among network hosts. With the development of network infrastructure, it is hard to perform accurate analysis for computer network activity merely based on whether two hosts have communication. Therefore, we intend to mine much more useful information from network flows, and combine flow information to the interaction graph to construct a novel graph model which could be more flexible and contain much more abundant information. The embedding of this information helps us to comprehensively understand network flow connectivity behaviors.

C. Graph optimization of NFCG

Intuitively, many researchers believe that there are hierarchical structures in many kinds of networks [27]. In general, network flow data is often dispersed from high ranking level hosts (e.g., the nodes have large volume traffic flowing over by) to the low ranking level ones, and aggregated to the high level hosts from the low level ones, respectively. In order to explore more detailed connected relations of different hosts, we propose a ranking mechanism based on the statistics of network flow properties to classify nodes into different levels. Moreover, edges can be weighted so that they contain as much network flow information as possible. In a general form, we use the feature vector $w_{i,j} \in W$ to represent the weight of an edge. According to different research purposes, various feature vectors are selected, such as the number of packets during the interaction, ports number opened, live time of the flow connection, and etc.

In Figure 3, we have 64 nodes the different colors mean the ranking of nodes, and each edge in the NFCG would have different weighted vectors. In general, network flow data is often dispersed from high ranking level hosts (e.g., the nodes have large volume traffic flowing over by) to the low ranking level ones, and aggregated to the high level hosts from the low level ones, respectively. For different ranking levels of hosts, they may have some similar connected patterns or distinct patterns. Therefore, in order to explore more detailed connected relations of network flows, we propose a ranking mechanism based on the statistics of network flow properties to classify nodes into different levels. Moreover, edges can be weighted so that they contain as much network flow information as possible. In a general form, we use the feature vector to represent the weight of an edge. According to different research purposes, various feature vectors are selected, such as the number of packets during the interaction, ports number opened, live time of the flow connection, etc. It is clearly that the interaction graph model shows whether there exist connections between two entities like the plain graph model while NFCG can be extended to rich representation of weighted graphs which show more network flow connectivity information by node ranking and edge weighting mechanism.

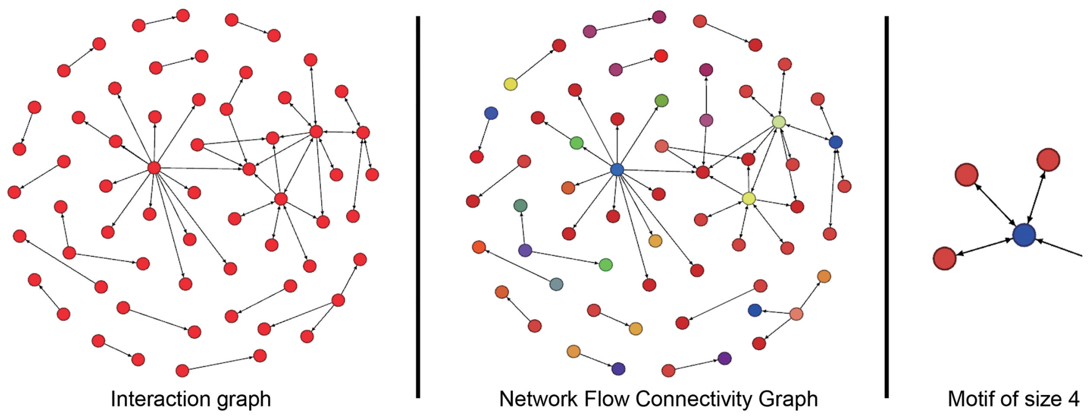

Figure 3. Visualization example of interaction graph, NFCG and motif. The interaction graph consists of 64 nodes, with each edge representing the interaction relationships between different nodes. The NFCG graph is an augmented interaction graph that can be encoded with supplementary flow information, with each node denoted by a different color/style edge, while the motif is a reoccurring subgraph of the NFCG graph.

2.3. The Concept of a Network Motif

The network motif is a basic concept in the field of complex networks and is an effective method of studying the local structural characteristics and evolution laws of complex systems [28]. The motif is a small, connected subgraph and is composed of a few nodes, which can be considered to be the "molecular" topology describing the connected relationships in a network. As the size of a motif is typically between an individual node and a network community, a motif can be defined as a pattern of interconnections that occurs in a graph. Additionally, we can even consider some special network architectures as an extension of several types of "network motifs." For example, research conducted by Milo [29] has found some meaningful network motif structures in complex networks, and it supports the notion that a small number of motifs occur repeatedly across the entire network, which might have been widely employed to reveal and understand the dynamic characteristics of a network architecture. Milo [30,31] has also conducted many experiments in many specific network environments, such as biological neural networks, food chain networks, and the Internet, and concluded that the general appearance of these motifs indicates that they are likely to have specific functions in different networks. This is the key initial point of our proposed approach: the occurrence of some events in networks brings about corresponding changes in the network structures, which result in some specific network motifs (let these be defined as highly significant motifs). At this point, we believe that the frequent appearance of highly significant motifs can help a lot in detecting network events. However, since many applications or events use the client–server architecture, they may produce the same motif strucutres in the interaction graph model, which is hard to separate network events. To address this problem, we propose the augmented interaction graph model called NFCG, in which graphs can be encoded with supplementary flow information so that nodes and edges can be assigned different colors and weights. Encoding with flow behavior information helps us to understand the details of network flow connections better. Thus, this allows for two motifs to share similar structures and behaviors under the effect of network events.

3. Proposed Approach

The proposed approach for detecting networks is introduced in this section. In the present study, the behavior of the target discovery and propagation stages in the worm life cycle can be expressed as different fan-out connected patterns, which can be represented by corresponding motifs. Since the characteristics of worm victims are [32]:

1. Worm victims would generate a lot of similar flows (from one victim to target a lot of target hosts)
2. Each flow generated by worm victims contains only a few packets, but these flow behaviors are very similar since they are caused by the same source reason.
3. Worm victims often use particular same port numbers to spread a known worm, except for polymorphic worms.

Therefore, several cascading motifs can be used to describe the sequential cascading relationships generated by the whole worm life cycle. Our motif-based method is to find cascading motifs in the NFCGs that can be seen as the problem of motif discovery and subgraph matching. The traditional motif discovery method, such as FANMOD [33], is used to enumerate size-N motifs in a graph, so that it can be used to analyze and summarize the structure and changes of complex networks. However, our method does not refer to a particular size of motif but a class of motifs with specific fan-out connected patterns. To perform such motif discovery in our method, subgraph matching often needs to be completed with exact patterns, and motifs reflected in worm propagation are similar to fuzzy patterns. Here, we develop a CMD algorithm to capture connected patterns that exhibit typical characteristics of network worms. The CMD algorithm consists of two phases: the scanning motif discovery (SMD) phase, which is used to find the presence of scanning behaviors in the initial stage of a network worm, and the transferring motif detection (TMD) phase, which is used to detect similar transferring behaviors within network worms. The complete flow chart of the CMD algorithm is presented in Figure 4.

3.1. The Scanning Motif Discovery Phase

The SMD phase aims to detect scanning activities among network flows. As known from the worm life cycle and discussed before, a majority of network worms are typically caused by scan activity (i.e., source host sending packets to a number of destination hosts and/or ports) [34], except for some special worms, such as the email-spammer. Thus, we need to first distinguish scanning flows from non-scanning flows. There are three cases in the scanning flows: (a) port scan, in which the flows have a host scanning several ports on a single destination host; (b) IP scan, in which the flow from a host scanning a particular port on many destination hosts; and (c) hybrid scan, which is basically a combination of both port scan and IP scan. To this end, for an NFCG graph, we could extract the scanning motif structures by measuring the flow property information embedded in the NFCG weight vector.

Regardless of the type of scanning activities, we group all subgraphs of NFCG into scanning motifs and non-scan motifs by using a combination of the following flow features, including the number of destination hosts targeted, the number of packets sent, and the set of destination ports used. The motifs and network flows would have similar communication features if they caused by the same reason. The output of the SMD phase is to identify those suspicious motifs in which the source hosts perform scanning behaviors. However, there would be a large number of suspicious motifs extracted when those features were calculated for motif discovery. In order to increase accuracy and reduce the false positive rate of the proposed approach, we set the threshold for SMD to filter out those motifs which have the exact scanning behaviors. These motifs and network flow data are used for the next TMD phase.

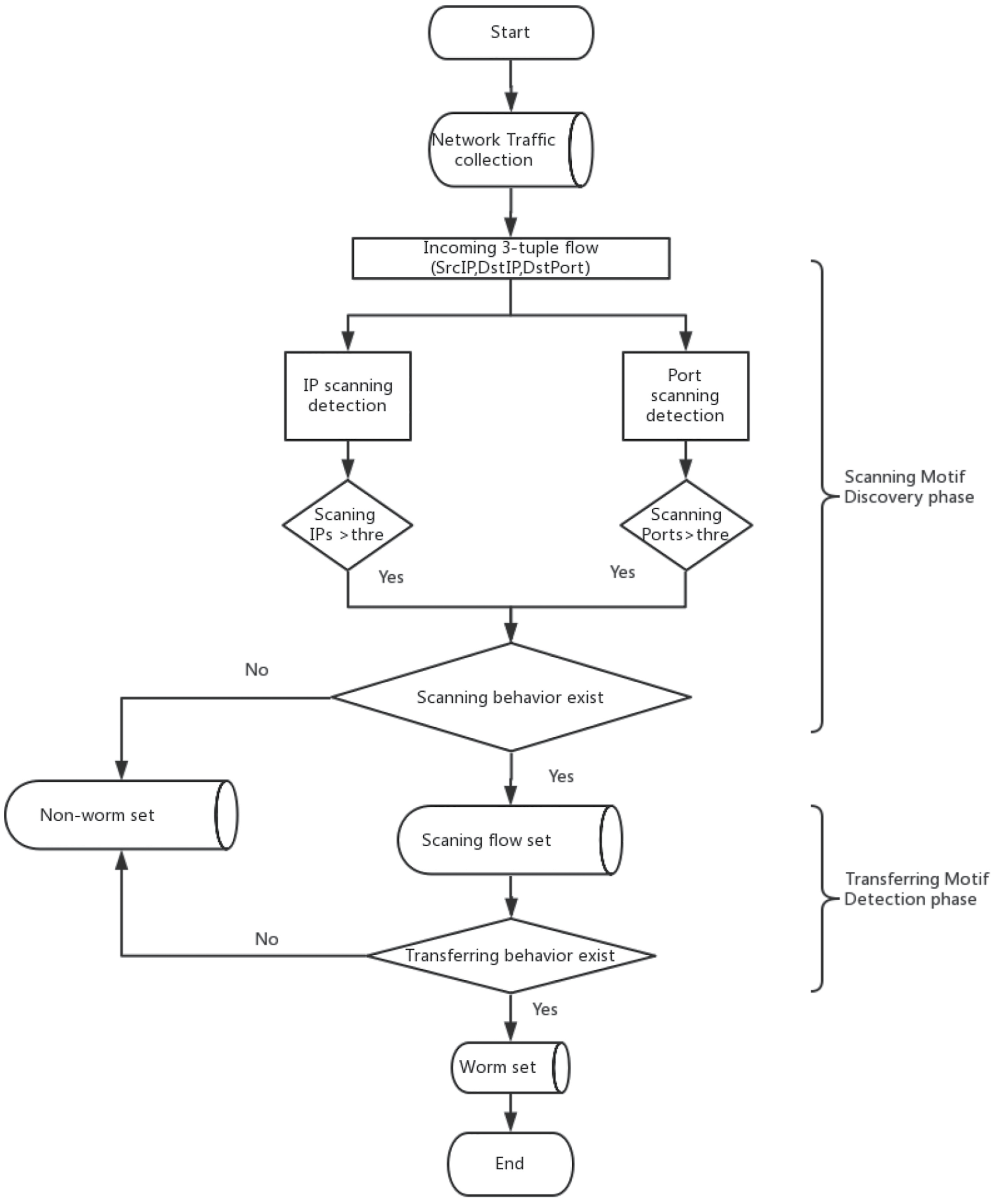

Figure 4. Flow chart of our approach operation.

Since the goal of the worm is to spread their malicious codes over the entire network in a short time, the worm would never stop progressing after successfully infecting a host. The infected hosts first turn to network victims, and then they will go and scan their neighboring hosts for finding the next victims. It is clear that, if we count the number of IP addresses scanned by the victim, and if the amount exceeds a pre-defined threshold, then we can draw a conclusion that a worm has been detected. Therefore, the TMD phase is used to detect whether a motif that contains the "scanner" host exhibits strong correlated behaviors in the next step. In particular, a cascading motif that is created by a scanning motif and a transferring motif depicts the anomalous connection pattern step by step in the target-finding stage and the transferring stage of worm propagation. It is claimed that, if a cascading motif exists in the connection behavior graph of one host, the host would be identified as a worm victim.

The starting point of the TMD phase comes after the SMD approach detects the suspicious motifs performing scanning behaviors. The primary objective of the correlation approach is to check whether suspicious motifs that contain scanner hosts have a strong correlated behavior toward other destination hosts. The flow chart of the TMD phase is also illustrated in the corresponding part of Figure 4. Figure 5 illustrates the motif discovery procedure of our proposed approach. Through motif discovery, we can achieve a few motifs that exhibit worm-like behaviors. In addition, these suspicious motifs become more and more obvious with the interaction of the network.

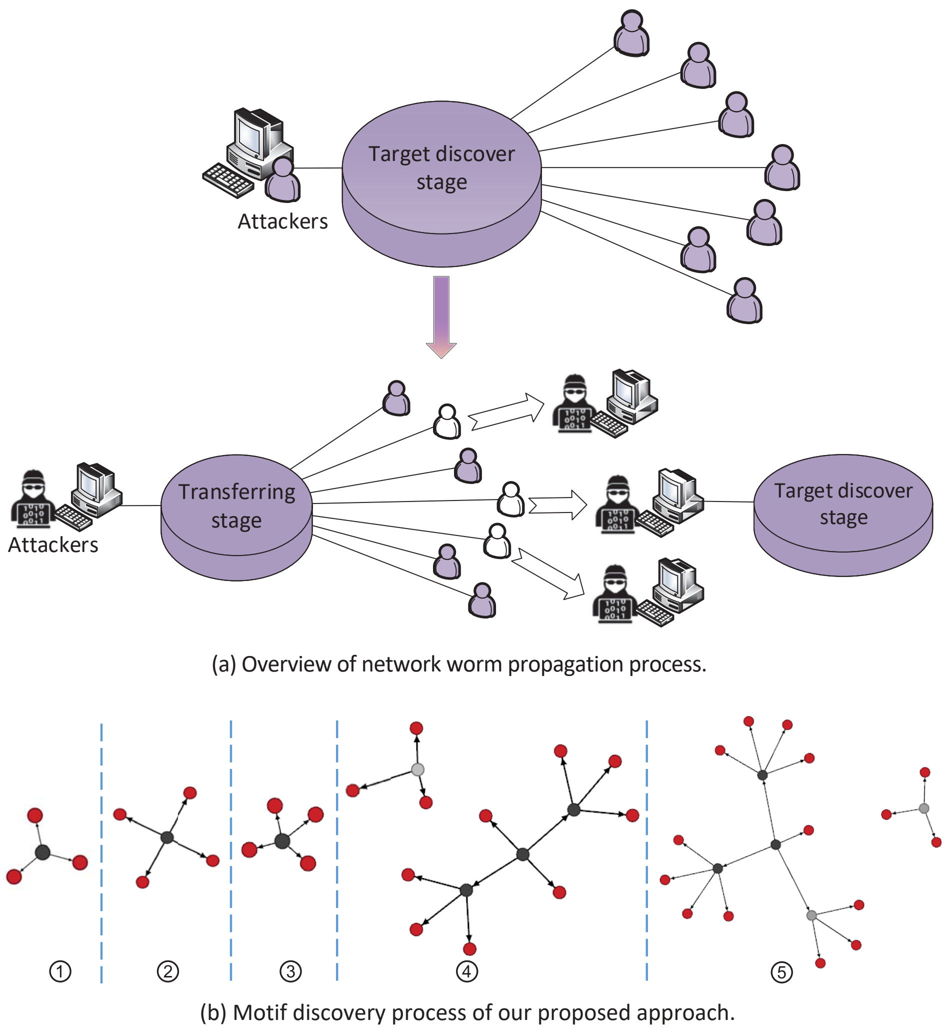
Target discover stage
Attackers
Transferring stage
Attackers
Target discover stage
(a) Overview of network worm propagation process.
①
②
③
④
⑤
(b) Motif discovery process of our proposed approach.

Figure 5. Motif discovery procedure of our proposed approach: (**a**) overview of the network worm propagation process and the (**b**) motif discovery process of the proposed approach.

3.2. The Transferring Motif Detection Phase

Informed by Figures 4 and 5, we need first to determine a motif that contains "scanner" hosts. If a scanner host has been detected, then this suspicious host must be checked to establish whether it has strong correlated behaviors toward other hosts. The counter will increase if the suspicious host does perform this behavior. Afterwards, if the counter exceeds a pre-defined threshold, the alert is triggered and the IP address of the worm victim is saved in the worm set; otherwise, it is saved in the non-worm set. The pre-defined threshold in the TMD phase is based on some empirical knowledge and analysis of the network worm flows.

3.3. An Illustrative Example

We provide an illustrative example to show how to identify network worm victims by our approach. As Figure 6 shows, supposing we have the input network flow data as follows, the typical procedure are given as follows:

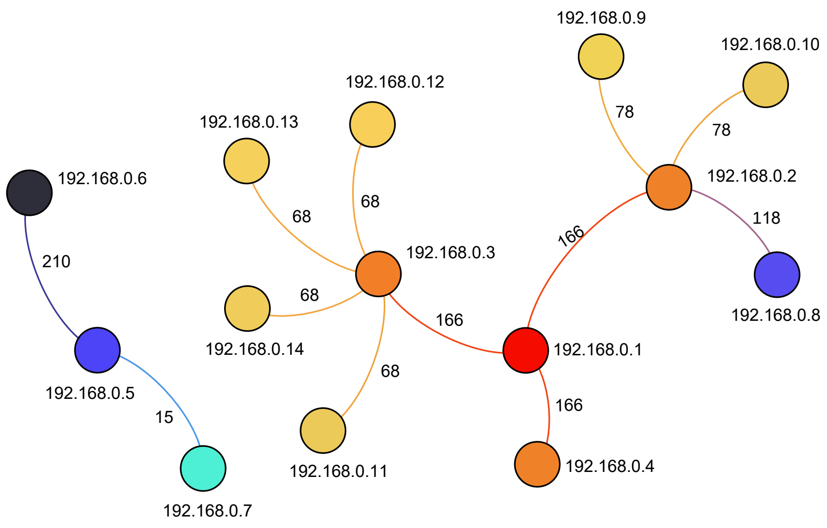

Figure 6. An illustrative example of our proposed approach.

1. Input network flow data:

$$Flow\ data = \begin{bmatrix} SrcIP & DstIP & DstPort & Pro. & \#Packets \\ 192.168.0.1 & 192.168.0.2 & 135 & TCP & 166 \\ 192.168.0.1 & 192.168.0.3 & 135 & TCP & 166 \\ 192.168.0.1 & 192.168.0.4 & 135 & TCP & 166 \\ 192.168.0.5 & 192.168.0.6 & 80 & TCP & 210 \\ 192.168.0.5 & 192.168.0.7 & 53 & UDP & 15 \\ 192.168.0.2 & 192.168.0.8 & 23 & TCP & 118 \\ 192.168.0.2 & 192.168.0.9 & 135 & TCP & 78 \\ 192.168.0.2 & 192.168.0.10 & 135 & TCP & 78 \\ 192.168.0.3 & 192.168.0.11 & 135 & TCP & 68 \\ 192.168.0.3 & 192.168.0.12 & 135 & TCP & 68 \\ 192.168.0.3 & 192.168.0.13 & 135 & TCP & 68 \\ 192.168.0.3 & 192.168.0.14 & 135 & TCP & 68 \end{bmatrix}.$$

2. Then, we check out whether hosts exist that are performing scanning behaviors (hosts that are detected by the scanning motif discovery phase):

$$Scanning\ hosts = \begin{bmatrix} IP\ addresses \\ 192.168.0.1 \\ 192.168.0.2 \\ 192.168.0.3 \end{bmatrix}.$$

3. Set the threshold for the Transferring Motif Detection phase; in this case, we set the threshold as 3. It is means when applying the Transferring Motif Detection phase, if the amount of suspicious worm victims exceed the threshold 3, we can get a conclusion that the network worm has been detected. In this illustrate example case, the result is given as follows:

$$Result = \begin{bmatrix} Worm\ victims \\ 192.168.0.1 \\ 192.168.0.3 \end{bmatrix}.$$

IP addresses 192.168.0.1 and 192.168.0.3 are firstly detected as scanner hosts as they scan several hosts on the same port by the scanning motif discovery phase. Additionally, 192.168.0.3 generate communication flows to four hosts on destination port 135, which have exceeded the preset threshold; then, 192.168.0.3 is identified as a worm victim seed by the Transferring Motif Detection phase.

4. Experimental Evaluation

In this section, we explain our experimental validation of the proposed method using a real network trace from a WIDE trace [35] and a simulated traffic dataset from the Georgia Tech Network Simulator (GTNetS) [36,37]. The WIDE trace is taken at a US–Japan trans-Pacific backbone line (a 150 Mbps Ethernet link) on which all IP addresses are anonymous but forty bytes of application layer payload are kept for each packet. GTNetS, however, is a full-featured network simulation environment that allows researchers to study the behavior of moderate to large-scale computer networks under a variety of conditions. The effectiveness of our approach is demonstrated in three simulation experiments, namely TCP worm, UDP worm, and polyworm scenarios. Figure 7 shows an NFCG illustration example with 500 flows of the MAWI Working Group WIDE Project Traffic traces by using GraphViz tools [38].

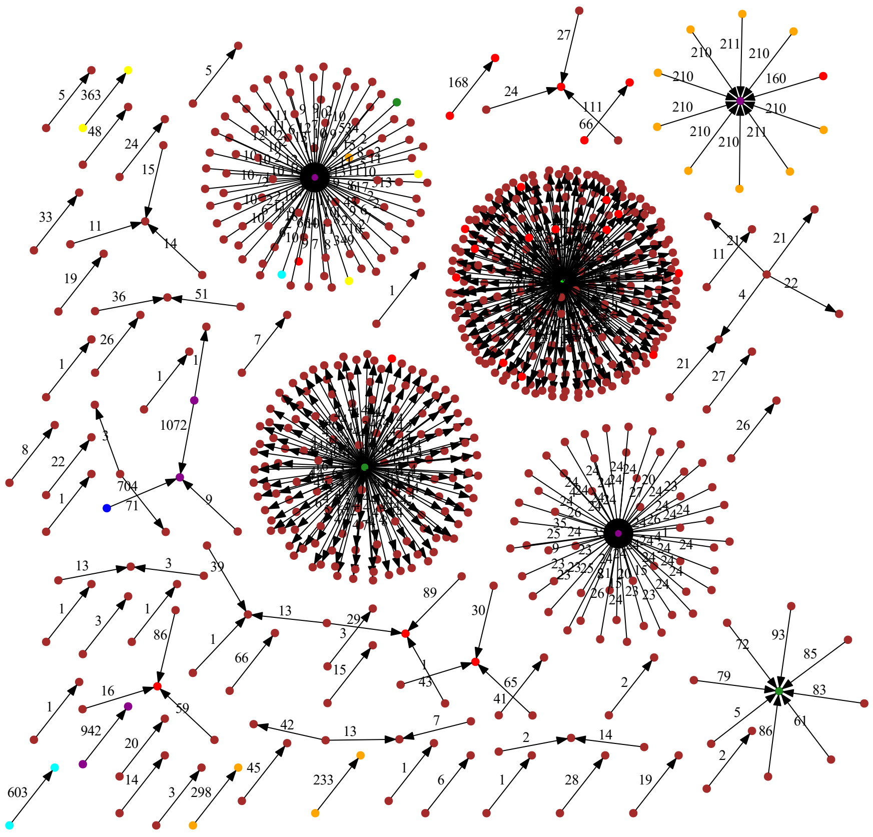

Figure 7. An illustration example of NFCG visualization, using the GraphViz tool, 500 flows of MAWI Working Group WIDE Project Traffic traces on 15 October 2018. The traffic of MAWI Working Group WIDE Project Traffic traces will be used for simulating network normal traffic. Combined with the simulated traffic dataset from the Georgia Tech Network Simulator (GTNetS), the ultimate mixed network flow dataset are validated for identifying hosts in NFCG which act as network worm victims (the cascading motif exists in the connected behavior graph of a host).

4.1. Simulation Environment

The GTNetS environment allows the creation of simulation network topologies (consisting of nodes and their associated communication links) and end-user applications describing the flow of data over the simulated topology. The GTNetS has been widely used for network worm detection because it can successfully simulate network worm traffic [15,16]. This simulator starts with an infected host (also known as network worm initiator) that randomly scans other vulnerable hosts for finding targets. Then, the designed topologies in the GTNetS are created based on the following input parameters:

- Time duration of the simulation experiment: the total time needed to simulate a target network worm, within which the network worm must complete its life cycle (target finding, transferring, activation, and infection);
- The basic topology for a network scenario test;
- Worm infection characteristics: (1) transport layer protocol, (2) the number of victim nodes in the initial stage, and (3) the scan rate, etc.

4.2. *Network Worm Scenario*

We have conducted several simulated experiments for different network worm scenarios: TCP, UDP, and polymorphic network worms. The TCP worm scenario tests our approach in identifying network victims under TCP worms; the UDP worm scenario tests the presence of UDP network worms; and the polymorphic worm scenario tests the polyworms that change their appearance while preserving the semantics each time they spread from one victim to another. The experimental scenarios are created by the GTNetS based on previously described input parameters. Table 1 presents the details of our simulated network worms.

Table 1. Input parameters of experiment configuration for the simulated network worm scenarios.

Experiment Configuration			
Network worm type	Blaster	Qaz, Opasof	polyworm
Target service	epmap	netbios-ns	netbios-ns
Target port	135	137	TCP for 135, UDP for 137
Transmission Protocol	TCP	UDP	TCP, UDP
Target OS	Windows	Windows	KALI system
Success infecting rate	40–60%	60–80%	50–70%
Scan rate	15 host/s	15 host/s	15 host/s
Simulated experiment time	5 s	5 s	5 s
Total row packet number	#7150	#10363	#6762
Total aggregated flow numbers	#5152	#7140	#5343

4.2.1. TCP Worm Scenario

The worm network initially contains one infected host as an attacker and then launches scanning behaviors on TCP port 135 for target finding. Once a vulnerable victim has been found, the malicious code from the attacker node is then transferred to the target victim. In this simulation, we set the scan rate as 15 host/s and simulation time as 5 s. The dataset consists of 7150 row packets (including normal flows and worm flows), which are extracted from the simulator log file. Then, we aggregated the dataset as 5152 flows. At the first stage, the attack starter had scanned eight hosts for vulnerabilities with four hosts that are successfully infected; the victim topology for TCP worm scenario is shown in Figure 8. In the next step, those four worm-infected hosts were going to search other hosts, respectively, on TCP port 135 with a success rate of infecting at 40–60%. Then, they kept repeating this step in order to spread the TCP worm to the entire network architecture. The green nodes in Figure 8 are actual network worm victims while 10 red label IP addresses are identified as worm victims by the CMD algorithm and two black label IP addresses 147.32.84.229 and 173.255.251.17 are missed by our algorithm.

4.2.2. UDP Worm Scenario

The same methodology used in the aforementioned phase was applied to the UDP worm scenario. Figure 9 shows the victim topology of the UDP worm scenario with a worm propagated on UDP port 137 (In this simulation, we set the scan rate as 15 host/s and simulation time as 5 s), and all the purple nodes with red label IP addresses in Figure 9 are network worm victims identified by the CMD algorithm. At the beginning, the attacker of NFCG in this simulation experiment had scanned six hosts for vulnerabilities with three hosts BEING successfully infected by worm propagation. Then, these worm-infected victims were proceeding to seek other vulnerable hosts on UDP port 137 with a 60–80% infection rate. After that, we could have obtained the simulation dataset of a UDP worm scenario. The dataset consists of 10,363 row packets and finally can be aggregated to 7140 flows.

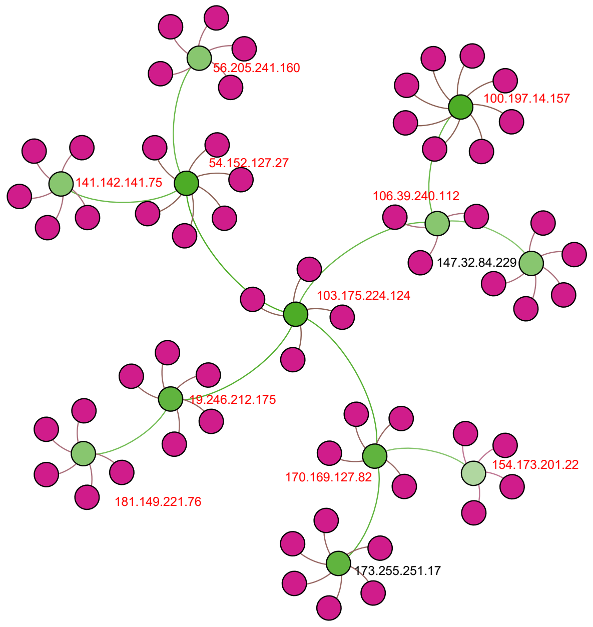

Figure 8. Complete cascading motif captured from the TCP worm test.

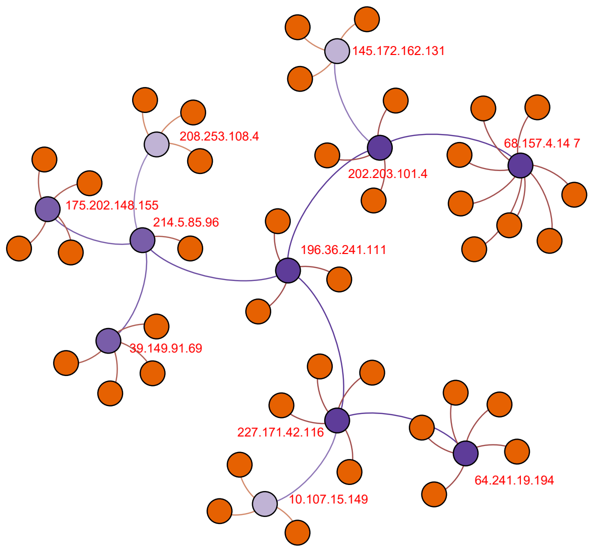

Figure 9. Complete cascading motif captured from the UDP worm test.

4.2.3. Polyworm Scenario

In a polyworm simulation scenario, we synthesize the polyworm by combining a TCP worm with a UDP worm together. Figure 10 shows the victim topology of the polyworm scenario. The attacker node begins scanning five hosts on TCP port 135 with two hosts successfully infected (success rate of infecting is 50–70%). Then, those worm-infected hosts scanned other hosts on UDP port 137 in the next infection stage. The dataset consists of 6762 row packets (including normal flows and worm flows), and can be aggregated into 5343 network flows. Finally, seven out of eight actual network worm victims have been identified by our CMD algorithm.

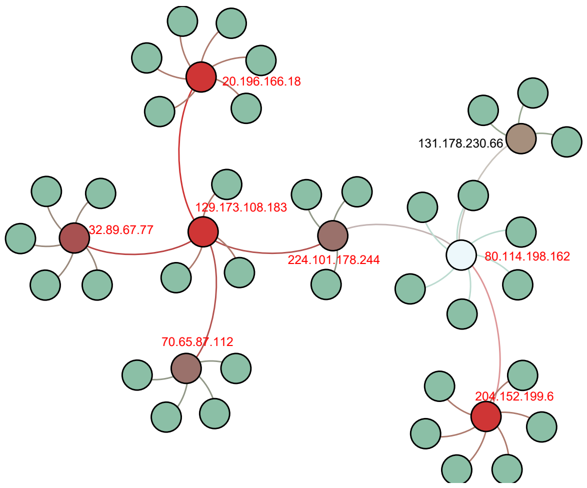

Figure 10. Complete cascading motif captured from the polyworm test.

4.3. Evaluation Results of Different Network Worm Scenarios

The simulation scenarios we conducted are used to test the accuracy of the proposed approach in identifying network worm victims. In our simulation experiments, we set the threshold based on some empirical knowledge for the TMD phase be equal to 3, which means that a host that receives network packets and then resends a packet to three or more destination hosts is identified as a suspicious host that demonstrates a strong correlation behavior. Tables 2–4 present the worm victim detected in the suspicious motif for each flow group under different network-worm simulation experiments. Table 2 indicates that our method can be used for network victim identification. For example, in the first flow group, only two hosts (103.175.224.124 and 54.152.127.27) have been identified since they have both scanning and transferring behaviors. However, the number of infected hosts quickly increases to 10 in the fourth flow group that exceeds the predefined threshold. Furthermore, Tables 3 and 4 also indicate that our method is efficient in identifying worm victims in the UDP worm and polyworm scenario separately.

Table 2. Infected IP addresses in the dataset for each flow group under the TCP worm simulation experiment.

Flow Group	Infected IP Addresses			
1∼1000	103.175.224.124	54.152.127.27		
1001∼2000	103.175.224.124	54.152.127.27	106.39.240.112	
2001∼3000	103.175.224.124	54.152.127.27	106.39.240.112	19.246.212.175
	170.169.127.82	141.142.141.75		
3001∼4000	103.175.224.124	54.152.127.27	106.39.240.112	19.246.212.175
	170.169.127.82	141.142.141.75	56.205.241.160	181.149.221.76
	100.197.14.157	154.173.201.22		
4001∼5152	103.175.224.124	54.152.127.27	106.39.240.112	19.246.212.175
	170.169.127.82	141.142.141.75	56.205.241.160	181.149.221.76
	100.197.14.157	154.173.201.22		

Table 3. Infected IP addresses in the dataset for each flow group under the UDP worm simulation experiment.

Flow Group	Infected IP Addresses			
1∼1400	196.36.241.111	202.203.101.4		
1401∼2800	196.36.241.111	202.203.101.4	68.157.4.147	227.171.42.116
2801∼4200	196.36.241.111 214.5.85.96	202.203.101.4	68.157.4.147	227.171.42.116
4201∼5600	196.36.241.111 214.5.85.96 175.202.148.155	202.203.101.4 145.172.162.131 39.149.91.69	68.157.4.147 10.107.15.149	227.171.42.116 64.241.19.194
5601∼7140	196.36.241.111 214.5.85.96 175.202.148.155	202.203.101.4 145.172.162.131 39.149.91.69	68.157.4.147 10.107.15.149 208.253.108.4	227.171.42.116 64.241.19.194

Table 4. Infected IP addresses in the dataset for each flow group under a polyworm simulation experiment.

Flow Group	Infected IP addresses			
1∼800	None			
801∼1600	224.101.178.244	129.173.108.183		
1601∼2400	224.101.178.244	129.173.108.183	32.89.67.77	
2401∼3200	224.101.178.244 204.152.199.6	129.173.108.183	32.89.67.77	80.114.198.162
3201∼4218	224.101.178.244 204.152.199.6	129.173.108.183 70.65.87.112	32.89.67.77 20.196.166.18	80.114.198.162

4.4. Comparison of Our Proposed Method with a Destination-Source Correlation Algorithm

In this section, we compare our proposed method with the Destination-source Correlation Algorithm method in [16] under three different worm scenarios. The Destination-source Correlation Algorithm is used to detect the network worms' victims in the case that a host received network traffic on port A and then starts to transfer packets to another vulnerable target host by port A. If the number of sending packets exceeds the predefined threshold, the host becomes suspicious.

Tables 5–7 illustrate data that confirm the high accuracy of the proposed approach and DCA method. The accuracy is calculated by the ratio between the True Positive (TP) to the sum of the True Positive (TP), False Positive (FP), and False Negative (FN) values:

$$Accuracy = \left(\frac{|TP|}{|FP| + |TP| + |FN|} \right) * 100\%,$$

where TP value reflects the capability of our approach to successfully and correctly identify the worm victims); FP value is defined to represent the case where hosts are incorrectly detected as worm victims; and FN value represents the number of worm victims that our approach cannot detect.

As illustrated in Tables 5–7, regardless of known released worms or never-seen-before worms, in most cases, our proposed approach can achieve higher average detection accuracy than the DCA method, especially in the polyworm simulated scenario as Table 7 reported. This result is caused by the use of SMD and TMD phases of our algorithm. The scanning motif discovery phase contributes much to the detection of network scanning behaviors. Furthermore, the TMD phase verifies worm victims in the cascading motif. However, the low accuracy of the DCA method in a polyworm simulated scenario is because it can only detect the host which exhibits destination-source correlation behaviors. Thus, we can draw a conclusion that our approach is effective and efficient in identifying worm victims and detecting network worms.

Table 5. Evaluated comparison of the Cascading Motif Discovery-based method and the Destination-source Correlation Algorithm for each flow group under a TCP worm simulated experiment.

Flow Group	**1**		**2**		**3**		**4**		**5**	
No. of Infected Hosts	**3**		**5**		**8**		**10**		**10**	
Approaches	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA
No. of identified victims	2	3	6	4	7	9	11	11	10	13
False Positive	0	1	1	0	0	2	1	3	0	4
False Negative	0	0	0	1	1	1	0	2	2	3
True positive	2	2	5	4	7	7	10	8	10	9
Accuracy	67%	50%	83%	80%	88%	70%	91%	62%	83%	56%

Table 6. Evaluated comparison of the Cascading Motif Discovery-based method and Destination-source Correlation Algorithm for each flow group under the UDP worm simulated experiment.

Flow Group	**1**		**2**		**3**		**4**		**5**	
No. of Infected Hosts	**4**		**5**		**5**		**10**		**11**	
Approaches	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA
No. of identified victims	2	3	4	5	5	5	10	7	11	11
False Positive	0	1	0	1	0	1	0	0	0	2
False Negative	2	2	1	1	0	1	0	3	0	2
True Positive	2	2	4	4	5	4	10	7	11	9
Accuracy	50%	40%	80%	67%	100%	67%	100%	70%	100%	69%

Table 7. Evaluated comparison of the Cascading Motif Discovery-based method and Destination-source Correlation Algorithm for each flow group under the polyworm simulated experiment.

Flow Group	**1**		**2**		**3**		**4**		**5**	
No. of Infected Hosts	**1**		**3**		**4**		**5**		**8**	
Approaches	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA	CMD	DCA
No. of identified victims	0	2	2	2	5	3	5	3	7	4
False Positive	0	1	0	1	1	1	0	0	0	1
False Negative	1	0	1	2	0	2	0	2	1	4
True Positive	0	1	2	1	4	2	5	3	7	3
Accuracy	0%	50%	67%	25%	80%	40%	100%	60%	88%	38%

5. Discussion

In this paper, our proposed algorithm is designed for worm detection through detecting the outbreak of connected patterns of network worms. In other words, the connected motifs of worms are used to successfully identify worm victims. Many worm detection methods often need the help of a detailed packet payload, such as the Honeypot logs. However, we propose a flow-based worm detection method without analyzing the payload, which mainly focuses on the connected patterns flowing over the entire network, and then digging out those suspicious flow patterns matched with the lifecycle of network worms. However, there are still several problems existing in our algorithm which require improvement regarding applicability.

The first and practical problem for our proposed method is that of the time interval used for flow data collection and graph construction procedure. It is infeasible to visualize all node pairs and edges due to the large scale nature of network size nowadays. As the time interval we selected increases, numerous nodes need to be visualized and processed; thus, selecting a large time interval has an influence on the fact that the graph is poorly visualized and it is hard to do real-time analysis. At the same time, however, if choosing a small time interval, NFCGs are becoming sparse so that they cannot provide abundant information for flow connectivity behaviors, which leads to false negatives increasing. On the other hand, some low rank level nodes which generate irrelative or insignificant edges can also affect results of the analysis. In this case, these edges are often treated as happening occasionally and easily make the analysis fall into confusion. To address this problem, we can adopt filtering rules to remove the insignificant and irrelative nodes and edges generated by network redundant information based on the quantity of flow attributes (e.g., 1000 flows visualization) or time-window (e.g., 30 s time window for large scale network, 3 min time window for small scale network) and select the edges whose weights are larger than a certain suitable value in order to accurately capture the principal connected relationships among network flows (e.g., opened port number larger than 100).

Another issue in our algorithm is that the threshold set up in our CMD algorithm would produce false positives and false negatives. If the threshold is set too large, this would result in more false negatives; while if the threshold is smaller, the false positive value is increasing. Actually, the threshold we used should have an adaptivity-element that is different under different scales of network size. For example, in the wide-area network (e.g., the Backbone Communication Network), the scale of worm attacks in this kind of network is relatively larger than others because of the size of the network and large-volume traffic. In this case, we can decrease the false positive rate of worm victim identification by setting an appropriate threshold value, such as more than 100 hosts in the Scanning Motif Detection Phase. In the Local Area Networks, such as company or campus networks, the threshold would be lower if there are 10 or 20 more hosts being found that have been scanned by the same target host or group hosts.

Moreover, for other worm behaviors, such as email-slammers (worms in Email), they do not have the scanning behaviors to hosts through network flow interacting (e.g., they find vulnerable hosts through an attachment to a mail [39]), which results in the fact the SMD of our algorithm fails in detection. Hoever, for most scanning worms, our algorithm is effective in identifying network victims no matter if they are known-released or zero-day worms. Although the connected patterns of worms may be similar in graphical visualization with other behaviors, the cascading motifs discovered by our CMD algorithm are completely differential from normal flow behaviors and other anomalous flow behaviors. The current algorithm is designed for communication networks because we mainly consider flow behavior analysis in these networks. If the worm behaviors in other types of networks do not have scanning behavior, the performance of our method would not be good. It is our next step to classify more worm patterns or other anomalous behaviors that produce large-scale similar flow behaviors and also apply our method under different network scenarios.

6. Conclusions and Future Work

The ability of an approach to detect and identify anomalous events that are active on a computer network is critical for network management and security. In this paper, we propose a behavioral detection approach towards the network worm victims identification. Instead of matching the fixed signature in network traffic payloads, our approach focused on discovering specific connected patterns that depicted the inherent behaviors of network worms.

We developed NFCGs, which contain plenty of network-flow behavior characteristics, in order to model the social behaviors among network hosts; and then presented a CMD algorithm to capture scanning and transferring behaviors of network worms. We used the SMD phase to find motifs that perform scanning activities and the TMD phase to detect whether the strongly correlated behaviors exist in the next step of the scanning motif. With the use of the GTNetS simulation platform, we have conducted a variety of experiments for network worms. Experimental results suggest that our approach is effective and efficient in identifying network worm victims. Additionally, we addressed three types of worm propagation situations in our experiment. The results demonstrate that our approach has high detection accuracy, regardless of whether a worm is a known worm or a new polymorphic worm.

In the future, aiming at increasing the robustness and effectiveness of the proposed approach, we would start from sensitive analysis to improve the accuracy of our approach, and performance characteristics will be reported in comparison with some classical worm detection methods. Moreover, our approach can be extended to utilize motif structures to discover anomalous characteristics for more network events and help in the evolution analysis of worm outbreaks.

Author Contributions: H.H. and G.H. conceived and designed the graph model and experiments. H.H. and M.W. analyzed and processed the data; H.H. performed the experiments; M.W. and M.O. helped in the experiment; and H.H. wrote the paper.

Funding: This research is supported by the National Natural Science Foundation of China (No. 61471101, No. 61571094).

Acknowledgments: At the point of finishing this paper, we would like to firstly thank Warren Jin from Software and Computational Systems of CSIRO, Canberra, Australia, who gave us useful advice for improving our paper. We would also like to thank LetPub (Waltham, Massachusetts 02451, United States) (www.letpub.com) for providing linguistic assistance during the preparation of this manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CMD	Cascading Motif Discovery
NFCG	Network Flow Connectivity Graph
IDSs	Intrusion Detection Systems
SMD	Scanning Motif Discovery
TMD	Transferring Motif Detection
MAWI	Measurement and Analysis on the WIDE Internet
WIDE	Widely Integrated Distributed Environment
OS	Operating System
TCP	Transfer Control Protocol
UDP	User Datagram Protocol
DCA	Destination-Source Correlation Algorithm
CSIRO	Commonwealth Scientific and Industrial Research Organisation

References

1. Zhou, C.V.; Leckie, C.; Karunasekera, S. A survey of coordinated attacks and collaborative intrusion detection. *Comput. Secur.* **2010**, *29*, 124–140. [CrossRef]
2. Choi, H.; Lee, H.; Kim, H. Fast detection and visualization of network attacks on parallel coordinates. *Comput. Secur.* **2009**, *28*, 276–288. [CrossRef]
3. Chen, W.; Liu, Y.; Guan, Y. Cardinality change-based early detection of large-scale cyber-attacks. In Proceedings of the 2013 IEEE INFOCOM, Turin, Italy, 14–19 April 2013; pp. 1788–1796.
4. Dainotti, A; Pescapé, A; Ventre, G. Worm traffic analysis and characterization. In Proceedings of the IEEE International Conference on ICC'07, Glasgow, Scotland, 24–28 June 2007; pp. 1435–1442.
5. Li, P.; Salour, M.; Su, X. A survey of internet worm detection and containment. *IEEE Commun. Surv. Tutor.* **2008**, *10*, 20–35. [CrossRef]
6. Sharma, V. An analytical survey of recent worm attacks. *IJCSNS* **2011**, *11*, 99–103.
7. Corporation, S. Internet Security Threat Report. 2013. Available online: www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v18_2012_21291018.en-us.pdf (accessed on 15 April 2013).
8. Denis Makrushin, V.D. Fooling the 'Smart City'. *Kaspersky Lab Report*, 15 September 2016; pp. 1–22.
9. Tang, Y.; Chen, S. Defending against internet worms: A signature-based approach. In Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Miami, FL, USA, 13–17 March 2005; Volume 2, pp. 1384–1394.

10. Moftah, R.A.; Maatuk, A.M.; Plasmann, P.; Aljawarneh, S. An Overview about the Polymorphic Worms Signatures. In Proceedings of the International Conference on Engineering & MIS, Istanbul, Turkey, 24–26 September 2015; p. 29.
11. Wang, K.; Stolfo, S.J. Anomalous payload-based network intrusion detection. In *International Workshop on Recent Advances in Intrusion Detection*; Springer: Berlin, Germany, 2004; pp. 203–222.
12. Schechter, S.E.; Jung, J.; Berger, A.W. Fast detection of scanning worm infections. In *International Workshop on Recent Advances in Intrusion Detection*; Springer: Berlin, Germany, 2004; pp. 59–81.
13. Sekar, V.; Xie, Y.; Reiter, M.K.; Zhang, H. A multi-resolution approach forworm detection and containment. In Proceedings of the International Conference on IEEE Dependable Systems and Networks (DSN 2006), Philadelphia, PA, USA, 25–28 June 2006; pp. 189–198.
14. Gu, G.; Sharif, M.; Qin, X.; Dagon, D.; Lee, W.; Riley, G. Worm detection, early warning and response based on local victim information. In Proceedings of the 20th Annual IEEE Computer Security Applications Conference, Tucson, AZ, USA, 6–10 December 2004; pp. 136–145.
15. Anbar, M.; Manasrah, A.; Manickam, S. Statistical cross-relation approach for detecting TCP and UDP random and sequential network scanning (SCANS). *Int. J. Comput. Math.* **2012**, *89*, 1952–1969. [CrossRef]
16. Anbar, M.; Abdullah, R.; Munther, A.; Al-Betar, M.A.; Saad, R.M. NADTW: New approach for detecting TCP worm. *Neural Comput. Appl.* **2017**, *28*, 525–538. [CrossRef]
17. Nissim, N.; Moskovitch, R.; Rokach, L.; Elovici, Y. Detecting unknown computer worm activity via support vector machines and active learning. *Pattern Anal. Appl.* **2012**, *15*, 459–475. [CrossRef]
18. Xu, W.; Zhang, F.; Zhu, S. Toward worm detection in online social networks. In Proceedings of the 26th Annual Computer Security Applications Conference, Austin, TX, USA, 6–10 December 2010; pp. 11–20.
19. Jiang, D.; Yao, C.; Xu, Z.; Qin, W. Multi-scale anomaly detection for high-speed network traffic. *Trans. Emerg. Telecommun. Technol.* **2015**, *26*, 308–317. [CrossRef]
20. Ye, Y.; Li, T.; Adjeroh, D.; Iyengar, S.S. A survey on malware detection using data mining techniques. *ACM Comput. Surv. (CSUR)* **2017**, *50*, 41. [CrossRef]
21. Cisco. *Cisco IOS NetFlow Configuration Guide, Release 12.4*; Technical Report; Cisco: San Jose, CA, USA, 2008.
22. Claise, B. *Cisco Systems Netflow Services Export Version 9*; Technical Report; Cisco: San Jose, CA, USA, 2004.
23. Fioreze, T.; Wolbers, M.O.; van de Meent, R.; Pras, A. Finding elephant flows for optical networks. In Proceedings of the 2007 10th IFIP/IEEE International Symposium on Integrated Network Management, Munich, Germany, 21–25 May 2007; pp. 627–640.
24. Sperotto, A.; Schaffrath, G.; Sadre, R.; Morariu, C.; Pras, A.; Stiller, B. An Overview of IP Flow-based Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2010**, *12*, 343–356. [CrossRef]
25. Iliofotou, M.; Pappu, P.; Faloutsos, M.; Mitzenmacher, M.; Singh, S.; Varghese, G. Network monitoring using traffic dispersion graphs (tdgs). In Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, San Diego, CA, USA, 23–26 October 2007; pp. 315–320.
26. Jin, Y.; Sharafuddin, E.; Zhang, Z.L. Unveiling core network-wide communication patterns through application traffic activity graph decomposition. *ACM SIGMETRICS Perform. Evaluat. Rev.* **2009**, *37*, 49–60.
27. Jalili, M.; Perc, M. Information cascades in complex networks. *J. Complex Netw.* **2017**, *5*, 665–693. [CrossRef]
28. Milo, R.; Shen-Orr, S.; Itzkovitz, S.; Kashtan, N.; Chklovskii, D.; Alon, U. Network motifs: Simple building blocks of complex networks. *Science* **2002**, *298*, 824–827. [CrossRef] [PubMed]
29. Shen-Orr, S.S.; Milo, R.; Mangan, S.; Alon, U. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.* **2002**, *31*, 64–68. [CrossRef] [PubMed]
30. Yeger-Lotem, E.; Sattath, S.; Kashtan, N.; Itzkovitz, S.; Milo, R.; Pinter, R.Y.; Alon, U.; Margalit, H. Network motifs in integrated cellular networks of transcription–regulation and protein–protein interaction. *Proc. Natl. Acad. Sci. USA* **2004**, *101*, 5934–5939. [CrossRef] [PubMed]
31. Kashtan, N.; Itzkovitz, S.; Milo, R.; Alon, U. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics* **2004**, *20*, 1746–1758. [CrossRef] [PubMed]
32. Ellis, D. Worm anatomy and model. In Proceedings of the 2003 ACM Workshop on Rapid Malcode, Washington, DC, USA, 27–30 October 2003; pp. 42–50.
33. Wernicke, S.; Rasche, F. FANMOD: A tool for fast network motif detection. *Bioinformatics* **2006**, *22*, 1152–1153. [CrossRef] [PubMed]
34. Pang, S.; Komosny, D.; Zhu, L.; Zhang, R.; Sarrafzadeh, A.; Ban, T.; Inoue, D. Malicious Events Grouping via Behavior Based Darknet Traffic Flow Analysis. *Wirel. Pers. Commun.* **2017**, *96*, 5335–5353. [CrossRef]

35. MAWI Working Group Traffic Archive, the WIDE Project. 2018. Available online: http://mawi.wide.ad.jp/mawi/ (accessed on 18 October 2018).
36. Riley, G.F. Simulation of large scale networks II: Large-scale network simulations with GTNetS. In Proceedings of the 35th conference on Winter simulation: Driving innovation, Winter Simulation Conference, New Orleans, LA, USA, 7–10 December 2003; pp. 676–684.
37. Riley, G.; Sharif, M.L.; Lee, W. Simulating internet worms. In Proceedings of the IEEE Computer Society's 12th Annual International Symposium on IEEE, Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS 2004), Volendam, The Netherlands, 8 October 2004; pp. 268–274.
38. Ellson, J.; Gansner, E.; Koutsofios, L.; North, S.C.; Woodhull, G. Graphviz—Open source graph drawing tools. In *International Symposium on Graph Drawing*; Springer: Berlin, Germany, 2001; pp. 483–484.
39. Syed, F. Understanding Worms, Their Behaviour and Containing Them. Project Report. 2009. Available online: https://www.cse.wustl.edu/~jain/cse571-09/ftp/worms/index.html (accessed on 15 April 2009).