Mitigating Malicious Packets Attack via Vulnerability-aware Heterogeneous Network Devices Assignment

Jianjian Ai[1], Hongchang Chen[1], Zehua Guo[2], Guozhen Cheng[1], Thar Baker[3]
[1]National Digital Switching System Engineering & Technological R&D Center,
[2]University of Minnesota Twin Cities, [3]Liverpool John Moores University

***Abstract*—Due to high homogeneity of current network devices, a network is compromised if one node in the network is compromised by exploiting its vulnerability (e.g., malicious packets attack). Many existing works adopt heterogeneity philosophy to improve network survivability. For example, "*diverse variants*" are assigned to nodes in the network. However, these works assume that diverse variants do not have common vulnerabilities, which deem an invalid assumption in real networks. Therefore, existing diverse variants deployment schemes could not achieve optimal performance. This paper considers that some variants have common vulnerabilities, and proposes a novel solution called Vulnerability-aware Heterogeneous Network Devices Assignment (VHNDA). Firstly, we introduce a new metric named Expected Infected Ratio (EIR) to measure the impact of malicious packets' attacks spread on the network. Secondly, we use EIR to model the vulnerability-aware diverse variants deployment problem as an integer-programming optimization problem with NP-hard complexity. Considering NP-hardness, we then design a heuristic algorithm named Simulated Annealing Vulnerability-aware Diverse Variants Deployment (SA-VDVD) to address the problem. Finally, we present a low complexity algorithm named Graph Segmentation-based Simulated Annealing Vulnerability-aware Diverse Variants Deployment (GSSA-VDVD) for large-scale networks named graph segmentation-based simulated annealing. The experimental results demonstrate that the proposed algorithms restrain effectively the spread of malicious packets attack with a reasonable computation cost when compared with baseline algorithms.**

***Index Terms*—network device, malicious packets attack, diversity, simulated annealing**

I. Introduction

IN traditional networks, network devices (e.g., switches and routers) exhibit high homogeneity due to some practical considerations (e.g., simplifying network operation and maintenance, unifying operator training, reducing complexity, and enhancing interoperability). However, such network device may suffer serious security threats. For instance, as soon as one node is compromised, the entire network will be subsequently compromised.

Malicious attackers exploit vulnerabilities and bugs behind network devices to launch attacks. These attacks result in significant information leakage from networks, normal operational interference, and even worse network destruction. Recent studies show that the current network devices are vulnerable to various attacks, such as a DNS spoofing attack [1], information exposure [2][3], Denial of Service (DoS) attack [4][5], buffer overflow attack [6][7], and stored cross-site scripting (XSS) attack [8]. The problem would be more serious as more new attacks are generating increasingly and evolving at a rapid pace. Malicious packets attack is a severe attack to crash and/or control nodes. In a network with homogenous devices, a malicious packets attack can send a single message to compromise the entire network.

To address this problem, some researchers are inspired by the survivability through heterogeneity philosophy [9] and have investigated the technique of diversity to prevent the propagation of malicious packets attack. Different network devices' vendors use different techniques and implementation methods (i.e., in terms of hardware, software and operating system) to provide services. In this paper, we define variants as the different implementations of the same function. We take the operating system as an example. For simplicity, we use "*operating system variant*" and "*variant*" interchangeably. If the number of variants is large enough, each node can have a unique variant. However, the number of variants is limited in the real condition. Efficiently assigning limited variants to nodes is critical to the defense efficiency, which is named diverse variants deployment problem. Existing works formulate the problem as a graph coloring problem with an objective to minimize the number of defective edges, whose two endpoints have the same variant, while maximizing the number of disconnected components. These approaches can achieve a satisfactory defense performance when diverse variants are completely independent. However, these works assume that diverse variants do not possess common vulnerabilities, and thus, malicious packets' attacks cannot propagate among different nodes. This assumption is not always valid. Heterogeneous variants may have common vulnerabilities since different network device vendors reuse the same code for different devices. For instance, using a third-party library may already contain vulnerabilities [10]. Attackers could compromise distinct nodes by exploiting common vulnerabilities among the nodes. In this case,

malicious packets attacks can continue propagating even though the adjacent nodes are heterogeneous, and the existing solutions cannot efficiently prevent such malicious packets attack.

In this paper, we devise a novel solution named Vulnerability-aware Heterogeneous Network Devices Assignment (VHNDA) that effectively deploys diverse variants to nodes in order to restrain the propagation of malicious packets attack in the network. By this solution, the network can effectively defend against the propagation of malicious packets attack because the proposed solution can use the correlation information between variants. To achieve the proposed solution, white box switches can be a feasible choice. Different from conventional switches, users can run any network operating system software in a white box switch on demand, which enables users to design and configure the network flexibly. In this way, the heterogeneity of network devices can be realized.

The main contributions of this paper are summarized as follow:

1. New metric for packets attack. We first analyze potential attack events with the assumption that diverse variants have common vulnerabilities. Then, we devise a metric named Expected Infected Ratio (EIR) that quantitatively measures the malicious packets attack propagation's impact on the network. Besides, we design a method using the connected graph to efficiently calculate EIR.
2. Problem formulation. Based on the above-mentioned metric, we propose the Vulnerability-aware Diverse Variants Deployment (VDVD) problem and formulate the problem as an integer-programming problem with NP-hard complexity.
3. Efficient algorithm design. We use simulated annealing to efficiently solve the problem and present a graph segmentation-based simulated annealing to achieve the tradeoff between computational complexity and network scalability. It is proved theoretically that the graph segmentation-based simulated annealing method can reduce the computational complexity.
4. Simulations verification. We conduct experiments to verify the validity of the proposed algorithms in terms of various aspects (e.g., coloring algorithms, compromised probability, inaccurate information and topology size).

The rest of this paper is structured as follows. The attack model and motivation of the work are introduced in Section II. In Section III, we discuss the overview of the proposed scheme; followed by the formulation of VDVD problem in Section IV. Section V discusses the proposed two solutions to the problem of different scales of network. Section VI shows the experimental evaluation and analyses the results gained from the experiments. Section VII presents the related work followed by Section VIII that concludes the paper and paves the way for the future work.

II. Attack Model and Motivation

A. Attack Model

First and foremost, we assume that it is not possible to make a network device completely immune to malicious attacks. Put it simply, an attacker can find the vulnerabilities behind a network device to launch a malicious attack. The attacker can destroy simultaneously the network devices with the same operating system. In this paper, we focus on a malicious packets attack. We define a malicious packet as a crafted message that exploits the buffer overflow vulnerability of nodes.

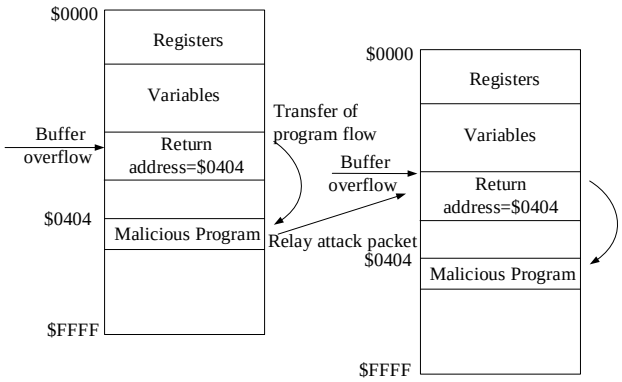


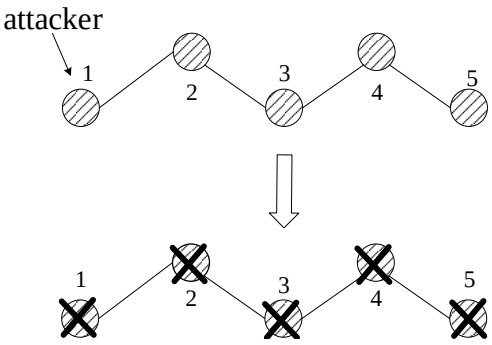
Fig. 1 The spreading process of malicious packets attack

Fig. 1 shows the spreading process of malicious packets attack. A buffer overflow vulnerability, which is the common trigger of attacks, may cause transferring program flow to a transmission component in the code space. Then, an exploited node may relay the attack packet it received before becoming irresponsive. This leads to the propagation of the malicious packet over the entire network which causes failure/corruption of all nodes in the network. Different from the assumption in the literature, we assume that there are common vulnerabilities between different variants. Thus, a malicious packets' attack can propagate if there are common vulnerabilities between the infected node and its adjacent nodes.

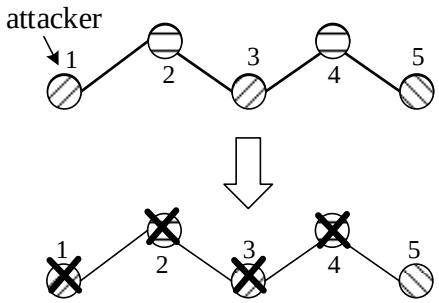
B. Motivation

We exemplify different variant deployments using a simple network that has 5 nodes and 3 software variants. In the below figures, a circle denotes a node, a circle with a cross indicates a node infected by a malicious packets' attack (infected node for short), and circles with different filling patterns indicate nodes with different variants. Left slash indicates variant A, horizontal line indicates variant B and right slash line indicates variant C. For example, in Fig. 2 in which the initial injected node is node 1, 2(a) has one variant A, and 2(b) has three variants: A, B and C.

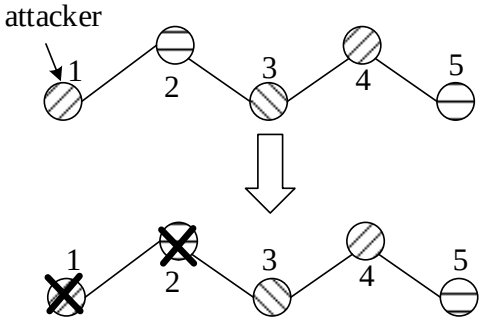
Fig. 2(a) shows the spread of malicious packets attack under homogeneity deployment, in which all nodes are assigned with the same variant A. Due to the homogeneity of nodes, the entire network would be infected if only one node is injected.



(a) Malicious packets attacks spread under homogeneity deployment



(b) Malicious packets attacks spread under existing heterogeneity deployment



(c) Malicious packets attacks spread under correlation-aware heterogeneity deployment

Fig. 2 Malicious packets attacks spread under different deployment

In the same vein, Fig. 2(b) shows diverse variants assigned to nodes using the existing heterogeneity deployment, and the variants of adjacent nodes are heterogeneous. In the figure, the variants are deployed in the following form: A is on nodes 1 and 3, B is on nodes 2 and 4, and variant C is deployed on node 5. We study specific attack cases where common vulnerabilities exist in different variants. For example, exploited vulnerability exists in variants A and B. Hence, malicious packets attack can propagate even though the adjacent nodes are heterogeneous. In Fig. 2(b), since the exploited vulnerability belongs to the common vulnerabilities between A and B, a malicious packets attack contaminates nodes (1–4). Hence, we perceive that deploying different variants on adjacent nodes cannot effectively restrain malicious packets attack spread in the network. In the same vein, Fig. 2(c) depicts the spread of malicious packets attacks under correlation-aware heterogeneity deployment. In Fig. 2(c), variant A is deployed on node 1 and 4, variant B is deployed on node 2 and 5, and variant C is deployed on node 3. Thus, malicious packets attack only contaminates nodes 1 and 2.

Therefore, the existing diverse variants deployment cannot effectively prevent the spread of malicious packets attacks. This paper proposes a novel diverse variant deployment to restrain the propagation of malicious packets attack in the case of a realistic situation.

III. Scheme Overview

Fig. 3 depicts the workflow of the proposed Vulnerability-aware Heterogeneous Network Devices Assignment (VHNDA) solution. It comprises of two modules: the EIR calculation, and VDVD problem solving.

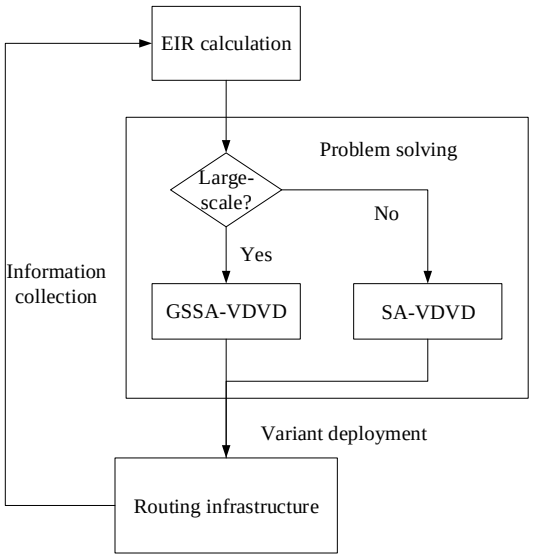


Fig. 3 The workflow of VHNDA

In the EIR calculation module, we analyze the potential attack events, and calculate a metric named EIR to measure the impact of malicious packets attack propagation based on the network information (e.g., topology information and flow state), which obtained periodically from network devices. Since the characteristic of malicious packets attack propagation is similar to the conception of the connected component, we propose a connected component-based algorithm to calculate the EIR. In the VDVD problem solving module, we propose two algorithms, SA-VDVD and GSSA-VDVD, for different scale networks. We, first, formulate the VDVD problem as an integer-programming problem with the objective to minimize the EIR while satisfying constraints. Due to the NP-hardness of the problem, we propose a SA-VDVD algorithm for medium-scale networks. Considering that the computational complexity will significantly rise with the increase of network size, we also present a low complexity algorithm named GSSA-VDVD for large-scale networks.

IV. The Diverse Variants Deployment Problem Formulation

This section starts with an introduction to the system description; then, analyzes the potential attack events and presents a metric named EIR to measure the impact of malicious packets attack spread.

A. System Description

TABLE I
NOTATIONS

Symbol	Description
V	Set of network devices. Represented by circles in figures
L	Set of edges in the network
S	Set of variants. Represented by filling patterns of circles in figures
n	Number of nodes in the network
m	Number of edges in the network
$\lvert\bullet\rvert$	Cardinal number for set
E_{s_i}	Set of all the potential events that the malicious packets attack can propagate in the case that the incipient injected node belongs to variant s_i
A_i	Set of vulnerabilities for variant s_i
d_i	Size of A_i
w_i^j	Binary. w_i^j is 1 if variant s_j is deployed onto node i and 0 otherwise
h_i	The weight for node i
$X(i)$	The variant deployment of node i . $X(i) \in S$
$P_e^{X(i)}$	The probability that event e occurs in the case that the initial injected node belongs to variant $X(i)$
$E_{X(i)}$	Set of all the potential events that the malicious packets attack can propagate corresponding to the case that the variant type of incipient injected node is $X(i)$
$\varphi_{D,e}(i)$	Number of infected nodes in the case that node i is initially infected for a given deployment D and a compromised variant event e
$r_{D,e}(i)$	The infected ratio for a given deployment D and a compromised variant event e
R_D	EIR
R_D^h	The weighed EIR

The network topology is modeled as a graph $G=(V,L)$, where V denotes the set of all nodes corresponding to network devices, and L is the set of edges representing links between network devices. Suppose that there are n nodes and m edges in the network, such that $n=|V|$ and $m=|L|$. Also, $S=\{s_1,s_2,\cdots,s_k\}$ is a set of k software variants. TABLE I lists the symbols used in the problem statement and formulations in section IV.

B. The Probability of Potential Attack Events

An attacker can exploit a number of vulnerabilities in each variant. Let $A_i=\{a_i^1,a_i^2,\cdots,a_i^{d_i}\}$ be the set of vulnerabilities for variant s_i with size of $d_i=|A_i|$. For simplicity, we assume that an attacker employs only one vulnerability to launch an attack.

Let $E_{s_i}=\{e_1,e_2,\cdots,e_k\}$ be the set of potential events that malicious packets attacks can propagate corresponding to the case that the variant type of incipient injected node is s_i. The main idea of calculating the probability is to count up the vulnerabilities corresponding to the attack event. Note that, the probability for each potential case is related to the variant incipient injected nodes. We take the case that the incipient injected node belongs to variant s_1 as an example to present the calculation.

Next, we describe how to calculate the probability by the example of three variants. In this case, E_{s_i} can be expressed as $\{\{s_1\},\{s_1,s_2\},\{s_1,s_3\},\{s_1,s_2,s_3\}\}$. As shown in Fig. 4, $\{s_1\}$ means that only variant s_1 is compromised; $\{s_1,s_2\}$ means that only variant s_1 and s_2 are simultaneously compromised; $\{s_1,s_3\}$ means that only variant s_1 and s_3 are simultaneously compromised; and $\{s_1,s_2,s_3\}$ means variant s_1,s_2,s_3 are simultaneously compromised. Let $P_e^{s_1}$ be the probability of event e in the case that the incipient injected node belongs to variant s_1. Let $A-B$ be the difference set in which elements exist in A but not in B. Therefore, the formal expressions are as follow:

$$P_{\{s_1\}}^{s_1}=\frac{|A_1-A_2-A_3|}{|A_1|} \tag{1}$$

$$P_{\{s_1,s_2\}}^{s_1}=\frac{|A_1\cap A_2-A_1\cap A_2\cap A_3|}{|A_1|} \tag{2}$$

$$P_{\{s_1,s_3\}}^{s_1}=\frac{|A_1\cap A_3-A_1\cap A_2\cap A_3|}{|A_1|} \tag{3}$$

$$P_{\{s_1,s_2,s_3\}}^{s_1}=\frac{|A_1\cap A_2\cap A_3|}{|A_1|} \tag{4}$$

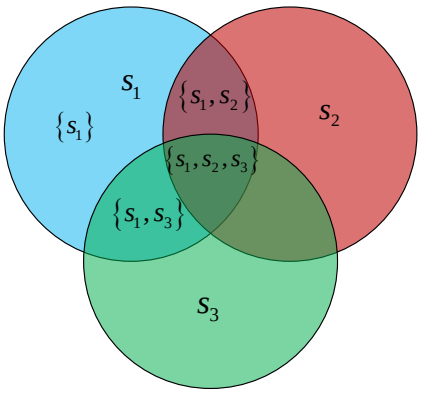


Fig. 4 Depiction of different potential events when the incipient injected node belongs to variant s_1

C. Measuring the Impact of Malicious Packets Attack Propagation

We devise a metric named EIR which is defined as the expected value for the proportion of infected nodes to measure the impact of malicious packets propagation. Let w_i^j be the

binary variable indicating whether variant s_j is deployed onto node i. Let $W_i = \{w_i^j, j \in [1,k]\}$ be the variant placement vector for node i. The variant placement can be indicated as $D = \{W_i, i \in V\}$. Specifically, for a given deployment D and a compromised variant event e, numerous nodes are infected after a period of time. Let $\varphi_{D,e}(i)$ be the number of infected nodes in the case that node i is initially infected. Then, the corresponding infected ratio can be expressed as follows,

$$r_{D,e}(i) = \frac{\varphi_{D,e}(i)}{n} \tag{5}$$

In fact, it is a persistent process that malicious packets propagate. In this paper, we do not consider the concrete propagation process, but we focus on the final decisive effect of the malicious packets spread on the network. Furthermore, we present a method to count the number of infected nodes.

A novel graph $G_f^{D,e}$ is generated according to the given deployment scheme D and the current compromised event e. For each node $i \in V$, if $X(i) \in e$, then node i will be inserted to $G_f^{D,e}$. For the inserted nodes, if there exist links between inserted nodes in previous graph G, then these links remain in $G_f^{D,e}$. In fact, not all the nodes in $G_f^{D,e}$ will be ultimately infected. If there are no paths between two nodes, at least one node will not be infected. Intuitively, the characteristic of malicious packets attack propagation is similar to the conception of the connected component which is defined as a subgraph where any two vertices are connected to each other by paths. Thus, we can adopt connected components to analyze the final infected nodes. Let $C_f^{D,e}(i)$ be the connected component of $G_f^{D,e}$ containing the initial infected node i. Then, the number of final infected nodes in the network can be defined as the number of nodes in $C_f^{D,e}(i)$. Thus Equation (5) can be rewritten as,

$$r_{D,e}(i) = \frac{\left|C_f^{D,e}(i)\right|}{n} \tag{6}$$

Algorithm 1 details the process of calculating $r_{D,e}(i)$

Algorithm 1 Calculating infected ratio $r_{D,e}(i)$

Input: a given deployment $D = \{W_i, i \in V\}$, the current compromised event e, the original graph G
Output: infected ratio $r_{D,e}(i)$

initialization $G_f^{D,e} = (V', L')$
for each node $i \in V$ **do**
  if $(X(i) \in e)$ **then**
    insert node i into V';
  end if
end for
for each node $i \in V'$ **do**
  for each node $j \in V'$ **do**
    if $(l(i,j) \in L)$ **then**
      insert link $l(i,j)$ into E';
    end if
  end for
end for
for each node $i \in V'$ **do**
  obtain by calculating the connected component of $G_f^{D,e}$ containing node i;
  $r_{D,e}(i) = \frac{1}{n}\left|C_f^{D,e}(i)\right|$;
end for

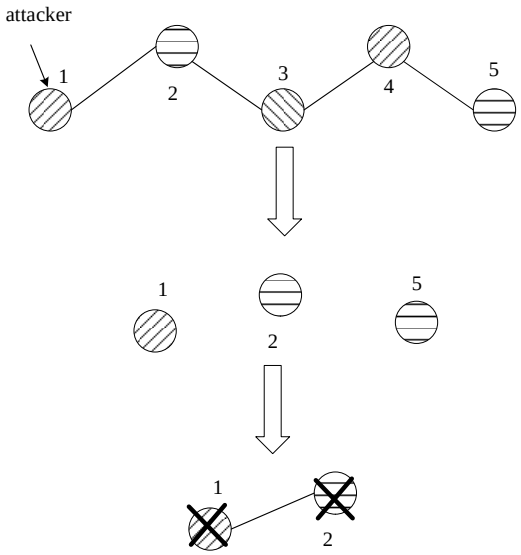


Fig. 5 The process of calculating infected ratio

We exemplify the process of calculating infected ratio using a sample network in Section II. Suppose that the initial injected node is node 1. As shown in Fig. 5, variant A is deployed on node 1 and 4, variant B is deployed on node 2 and 5, and variant C is deployed on node 3. We consider a specific attack case that the attacker exploits the vulnerability existing in variants A and B to launch an attack. It can be seen that node 1, 2 and 5 may be attacked. There is no single path between node 1 and 5, the same case between node 2 and 5. Thus, node 5 will not be infected.

Considering all the initial injected nodes and various compromised events, the EIR can be expressed as,

$$R_D = E\left[r_{D,e}(i)\right] = \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{e \in E_{X(i)}} P_e^{X(i)} r_{D,e}(i)\right) \tag{7}$$

where $P_e^{X(i)}$ represents the probability of event e in the case that the initial injected node belongs to variant $X(i)$, $E_{X(i)}$ indicates the set of all the potential events that malicious packets can propagate corresponding to the case that the variant type of incipient injected node is $X(i)$.

The EIR R_D assumes that all the nodes in the network are equally important. In fact, this assumption is not realistic; in

real cases, it is hard to satisfy this assumption as some nodes are more significant than the others. For instance, a node can be regarded as vital when it carries more traffic. Alternatively, a node with a higher degree, which means the node connects to more customers, can be considered significant. This paper measures the importance via assigning different weights to different nodes in the network. We use the degree of node to represent the weights.

Now, we extend the EIR by considering the node importance. Therefore, the weighed EIR is rewritten by replacing the factor $1/n$ with weight factor h_i,

$$R_D^h = E\left[r_{D,e}(i)\right] = \sum_{i=1}^{n}\left(\sum_{e\in E_{X(i)}} P_e^{X(i)} h_i r_{D,e}(i)\right) \quad (8)$$

where h_i is the weight for node i. Obviously, Equation (7) is a special case of Equation(8). Thus R_D^h can be regarded as the universal metric.

D. Problem Formulation

The VDVD problem aims to find the optimal variant assignment for each node in the network in order to prevent malicious packets spread. Our objective is to devise a deployment or mapping $M: S \mapsto V$ while minimizing the EIR.

We formulate the VDVD problem as follows,

$$\min \sum_{i=1}^{n}\left(\sum_{e\in E_{X(i)}} P_e^{X(i)} h_i r_{D,e}(i)\right) \quad (9)$$

subject to

$$w_i^j = \{0,1\} \quad \forall i \in V,\ s_j \in S \quad (10)$$

$$\sum_{s_j \in S} w_i^j = 1 \quad \forall i \in V \quad (11)$$

$$X(i) = j, \quad if\ w_i^j = 1 \quad (12)$$

In the above formulations, Equation (10) indicates that the variant deployment for nodes must be one of the only two statuses including completely a software variant or completely not of that variant. It is not allowed that a part of variant is deployed onto one node. Equation (11) represents only one variant can be deployed on a certain node in the network. Equation (12) shows the relationship between $X(i)$ and w_i^j. In the VDVD problem, the possible values of variables are 0 or 1, thus it is a typical integer-programming problem.

V. Solution

In this section, we propose two algorithms for different scale networks: SA-VDVD and GSSA-VDVD. The following subsections discuss these algorithms and the complexity analysis of them.

A. SA-VDVD algorithm

Obviously, the VDVD problem is an integer-programming problem. It is difficult to solve such a problem due to its nature of NP-hard. The computational complexity of the brute force solution that exhausts all possibilities in the search space is $O(k^n)$, where n is the number of nodes in the network and k is the number of available variants. It is impractical to employ the brute force solution when n and k are huge.

In this work, we resort to simulated annealing (SA) [11] which is regarded as an effective probabilistic searching technique for approximating the global optimum of a given function to solve the VDVD problem. As stated in TABLE I, X is a solution of VDVD problem corresponding to the state in SA. We define Ω as the state space and $H(X)$ as the energy at state X. Let T be the temperature, which controls the rate of progress of SA, and the probability making the transition from the current state X to a candidate new state X' is specified by an acceptance probability function in the metropolis algorithm [12],

$$P_T(X, X') = \begin{cases} 1, & if\ H(X') < H(X) \\ \exp\left(-\dfrac{H(X') - H(X)}{T}\right), & otherwise \end{cases} \quad (13)$$

The concrete SA-VDVD algorithm is presented as follows.

Algorithm 2 SA-VDVD algorithm

Input: the original graph G, the set of available variants S

Output: deployment $D = \{W_i, i \in V\}$

1: Initialize the temperature T and number of iterations L for each T, randomly generate the initial state X;
2: **for** $k = 1 : L$ **do**
3: Generate a new state X' which is in the state space Ω via a random perturbation;
4: Calculate the increment of energy $\Delta = H(X') - H(X)$;
5: **if** $\Delta < 0$ **then**
6: Accept the state transition from X to X';
7: **else**
8: Accept the state transition with the probability of $\exp\left(-\dfrac{H(X') - H(X)}{T}\right)$;
9: **end if**
10: **end for**
11: **if** the cease criterion is attained **then**
12: stop the process and return the optimal solution;
13: **else**
14: Cut down the temperature T, and then go back to Line 2;
15: **end if**

The efficiency and efficacy of SA algorithm depend on the cooling scheme parameters including the value of the initial temperature, number of iterations for each temperature, and the initial state. Several methods have been proposed to set it. Out of those methods, we adopt ANDYMARK [13] – an analytical method, which can obtain high quality solution while with less effort to tune the parameters of the cooling scheme for the SA algorithm. Let S be the solution space of the problem solved and V_{S_i} be the neighborhood set of S_i. The maximum cost deteriorations can be written as $\Delta H_{V_{\max}} = Max\left\{H(S_j) - H(S_i)\right\},\ \forall S_j \in V_{S_i}, \forall S_i \in S$. The

initial temperature is $-\Delta H_{V_{\max}} \Big/ \ln\left(P_A\left(\Delta H_{V_{\max}}\right)\right)$. In general, $P_A\left(\Delta H_{V_{\max}}\right) \approx 1$. Here let $P_A\left(\Delta H_{V_{\max}}\right)$=0.9. Let $\left|V_{S_i}\right|$ be the neighborhood size. Let $P_R\left(S_j\right)$ be the rejection probability and generally the value is close to zero. The number of iterations for each temperature is $L = \ln\left(P_R\left(S_j\right)\right)\left|V_{S_i}\right|$.

Typical cooling schedule includes the linear cooling schedule [11] and the geometric cooling scheme [14]. The authors in [15] show no significant difference in performance between linear and geometric schemes. We embrace the geometric cooling schedule as shown in Equation (14),

$$T(\lambda+1) = \alpha T(\lambda), \quad \lambda = 1, 2, \cdots, \tag{14}$$

where α is a constant representing the cooling factor and the value is approximate to 1, λ indicates the number of cooling. In our scenario, we set $\alpha = 0.95$.

B. GSSA-VDVD algorithm

In this section, we first analyze the computational complexity of the proposed SA-VDVD algorithm. We define a parameter K which reflects the external loop and inner loop in the process of SA. According to Algorithm 1, the computational complexity for calculating the EIR, which comes from calculating connected component, can be approximated to $O(n^3)$. Thus, the total computational complexity can be expressed as $O(Kn^3)$. Generally, the parameter K is relatively constant in the process of SA. It is obvious that the computational complexity will significantly rise with the increase of network size n. When the network size n increases to a certain value, it is not practical to obtain the deployment scheme via the proposed SA-VDVD algorithm. Therefore, it is urgent to seek a scheme which can effectively decrease the computational complexity while maintaining the performance.

To address this problem, we devise a Graph Segmentation-based Simulated Annealing Vulnerability-aware Diverse Variants Deployment (GSSA-VDVD) algorithm drawing lessons from the graph theory. The idea of this scheme is to transform the network with large size into smaller networks and then color them respectively. To do so, there are 2 alternatives including graph clustering [16] and graph partitioning [17]. Graph clustering algorithms attempt to find peninsulas of connectivity, while graph partitioning algorithms try to split the network into balanced partitions.

The concrete GSSA-VDVD algorithm is shown as follows.

Algorithm 3 GSSA-VDVD algorithm

Input: the original graph G, the number of divided components l, the set of available variants V

Output: deployment $D = \{W_i, i \in V\}$

1: obtain subgraph G_i, $i = 1, 2, \cdots, l$ by cutting the original graph G into l partitions via graph partitioning or graph clustering algorithm;
2: generate the set of node pairs N in which nodes in a node pair are adjacent and belong to different subgraph;
3: **for** each G_i **do**
4: obtain deployment D_i corresponding to subgraph G_i by executing SA-VDVD algorithm;
5: **end for**
6: **for** each element in N **do**
7: **if** nodes belong to the same variant **then**
8: update the variant deployed on one node of the node pair;
9: **end if**
10: **end for**
11: obtain the final deployment D by the integration of different D_i, $i = 1, 2, \cdots, l$

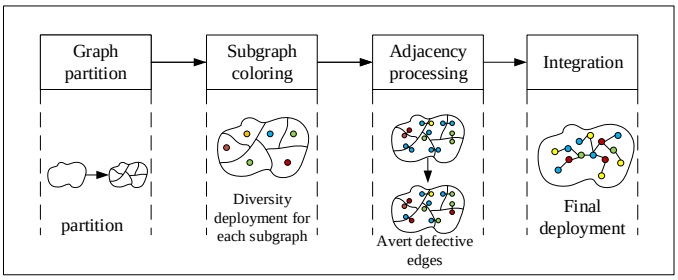


Fig. 6 The workflow of GSSA-VDVD algorithm

As shown in Fig. 6, the GSSA-VDVD algorithm consists of four phases: graph partition phase (line 1-2 in Algorithm 3), subgraph coloring phase (line 3-5 in Algorithm 3), adjacency processing phase (line 6-10 in Algorithm 3) and integration phase (line 11 in Algorithm 3).

In the graph partition phase, the original graph is divided into multiple subgraphs by executing the graph cutting algorithm. Furthermore, the information of adjacency nodes between different subgraphs is preserved in a specified set. Then, SA-VDVD algorithm is executed to deploy diverse variants for each subgraph in the subgraph coloring phase. Once the first two phases are complete, the initial deployment scheme for the entire graph is said to be achieved. Up until this point, it seems that all the work has been done. In fact, the initial scheme is not the desired one, because there may exist defective links between subgraphs. To eliminate these links, the adjacency processing phase is performed in which we can recolor one node of these defective links. Ultimately, the respective deployment D_i for each subgraph is integrated as the final deployment scheme.

C. Complexity analysis of GSSA-VDVD

The main questions that one may ask about the effectiveness and performance of GSSA-VDVD are, why can the proposed scheme decrease the computational complexity? And which one is better? To answer these questions, we will first introduce several theorems as follows.

Theorem 1: With a given number of divided components l, the computational complexity can attain the minimum if and

only if the size for each component is equal.

Proof: Assume that the network is divided into l components, let x_i be the size of the *ith* component, then the total computational complexity can be expressed as $O(K\sum_{i=1}^{l} x_i^3)$, and our target is to minimize it under constraint condition. Naturally, we can establish the following formula,

$$\begin{aligned} &\min \sum_{i=1}^{l} x_i^3 \\ &\textit{subject to } \sum_{i=1}^{l} x_i = n \end{aligned} \tag{15}$$

where n denotes the number of nodes in the network. We can transform the problem to the following via Lagrange Multiplier Approach. First Lagrange function $F(x_1, x_2, \cdots x_l, \lambda)$ is defined as follows,

$$F(x_1, x_2, \cdots x_l, \lambda) = \sum_{i=1}^{l} x_i^3 + \lambda\left(\sum_{i=1}^{l} x_i - n\right) \tag{16}$$

where λ is the Lagrange multiplier. Computing partial derivative towards equation(16), then we can obtain the following equations,

$$\begin{aligned} &\frac{\partial F(x_1, x_2, \cdots x_l, \lambda)}{\partial x_1} = 3x_1^2 + \lambda = 0 \\ &\frac{\partial F(x_1, x_2, \cdots x_l, \lambda)}{\partial x_2} = 3x_2^2 + \lambda = 0 \\ &\quad\vdots \\ &\frac{\partial F(x_1, x_2, \cdots x_l, \lambda)}{\partial x_l} = 3x_l^2 + \lambda = 0 \\ &\frac{\partial F(x_1, x_2, \cdots x_l, \lambda)}{\partial \lambda} = \sum_{i=1}^{l} x_i - n = 0 \end{aligned} \tag{17}$$

Then we can get the final result by solving(17),

$$x_1 = x_2 = \cdots = x_l = n/l \tag{18}$$

$$f_{\min}(x_1, x_2, \cdots x_l) = \left(n/l\right)^3 l = n^3/l^2 \tag{19}$$

From the results, we can conclude that $f(x_1, x_2, \cdots x_l) = \sum_{i=1}^{l} x_i^3$ can get the minimum value n^3/l^2 if and only if $x_1 = x_2 = \cdots = x_l = n/l$.

The results indicate that graph partitioning algorithms can attain lower computational complexity than graph clustering algorithms on the same network topology. This is because that clustering algorithms may produce unbalanced partitions resulting in high computation cost.

***Theorem 2*:** The computational complexity of network divided into l components will decrease by a factor of $1/l^2$ compared with the original network.

Proof: In the derivation process of the above-mentioned *Theorem 1*, we can easily get the relationship between the number of components l and the computational complexity. Note that the computational complexity here indicates the minimal computing cost for a given number of components l .

For the original network, the computational complexity can be expressed as $O(Kn^3)$.

For the divided network, the computational complexity can be expressed as $O_{\min}\left(K\sum_{i=1}^{l} x_i^3\right) = O\left(Kn^3/l^2\right)$.

Thus, we can get the relationship of computational complexity of these two schemes,

$$O_{\min}\left(K\sum_{i=1}^{l} x_i^3\right) = O\left(Kn^3/l^2\right) = O(Kn^3)/l^2 \tag{20}$$

It is clear that the computational complexity of network divided into l components will decrease by a factor of $1/l^2$ compared with the original network. Therefore, we can conclude that the more the number of divided components l , the lower the computational complexity. Nevertheless, we should not select the tremendous l for its performance degradation. As for the suitable l , it is difficult to derive from rigorous theory. Alternatively, we can determine the value of l through experiments and no more details will be shown here.

VI. EVALUATION

In this section, the proposed SA-VDVD algorithm is evaluated in terms of various aspects, namely coloring algorithms, compromised probability, inaccurate information and topology size. We organize this section as follows. First, the experimental setup used in our experiment is introduced in subsection A. Then, the comparison algorithms are presented in subsection B. Finally, in subsection C, we discuss the simulation results.

A. Experiment Setup

In our evaluation, we use Cenic topology [18] and Interoute[1]. Note that the topology is a network device-level topology, in which each node and edge correspond to a network device and link between network devices, respectively. The topology information is shown in TABLE II.

TABLE II
NETWORK TOPOLOGY USED IN OUR EVALUATION

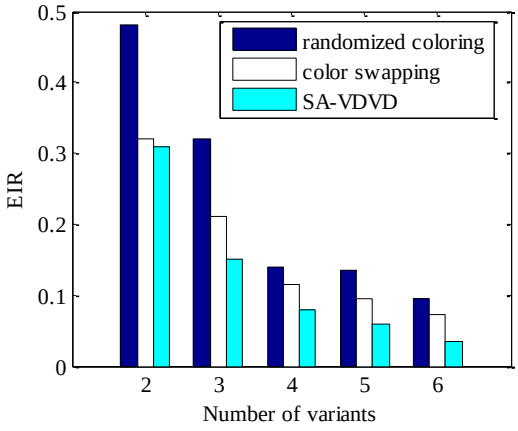
Topology	Nodes	Edges
Cenic	51	91
Interoute	110	158

B. Comparison Algorithms

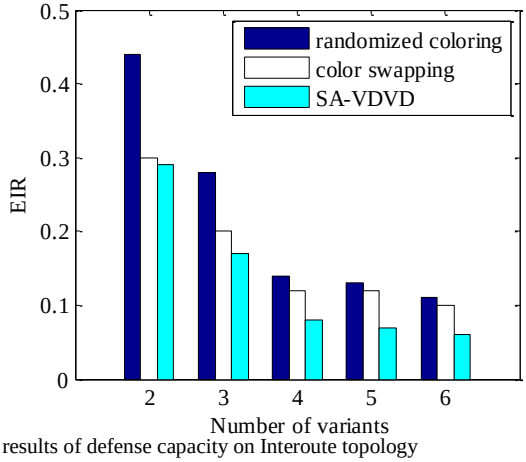
The comparison algorithms are introduced below:

(1) randomized coloring: randomly assigns a variant from the set of variants to each node.
(2) color swapping [19]: each node performs a local search amongst its immediate neighbors to determine if switching to a new variant would decrease the number of locally defective edges.
(3) SA-VDVD: it assigns a variant to each node based on SA.

[1] More information about the Interoute topology can be found at www.topology-zoo.org

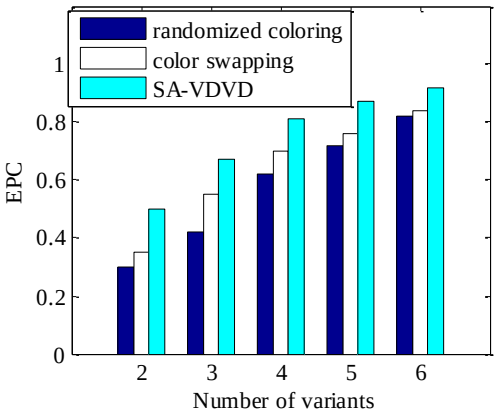


(a) The results of defense capacity on Cenic topology

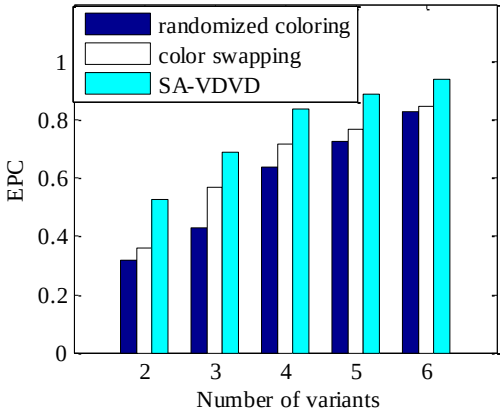


(b) The results of defense capacity on Interoute topology

Fig. 7 EIR under the different number of variants



(a) The results of network robustness on Cenic topology



(b) The results of network robustness on Interoute topology

Fig. 8 EPC under the different number of variants

(4) GSSA-VDVD: it divides the given network into numerous smaller subnets and then assigns a variant to each node based on SA.

The experimental results from these algorithms are stochastic. To eliminate this effect, the Monte Carlo method is adopted to reflect the final results. For a given coloring algorithm, each value in our evaluation is obtained from the following equation,

$$R = \frac{1}{K}\sum_{i=1}^{K} R_{D_i} \tag{21}$$

where D_i represents the *ith* deployment for a certain coloring algorithm. K is the number of Monte Carlo simulations. In the experiment, K is 200.

C. Simulation results

1) The impact of coloring algorithms

In this section, we compare the defense capacity against malicious packets attacks achieved by various algorithms on the same network topology. Fig. 7 shows the results achieved by the different algorithms measured using the EIR on Cenic and Interoute, respectively. With the increase number of variants, the RIR will decrease for each algorithm. Besides, the proposed SA-VDVD algorithm clearly outperforms other coloring algorithms by 35% in terms of the EIR, as the proposed SA-VDVD algorithm considers the spread across the distinct nodes. As a good result, it can restrain the propagation of malicious packets attacks to the greatest extent. Note that when the number of variants is 2, the defense performance of the proposed algorithm is similar to that of color swapping but superior to randomized coloring. This is explained by the fact that the proposed algorithm and color swapping both aim to make the adjacent nodes heterogeneous under this scenario.

There is no doubt that the infected nodes can impact on the network robustness. The network robustness means that the property of the network keeping connected in the case of a failure. In such a case, Pair Connectivity (PC) is adopted as the metric to measure the network robustness. PC is defined as the following equation,

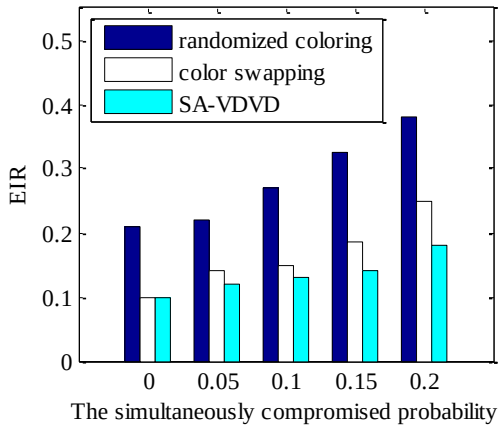
$$U = \frac{\sum_{i=1}^{num} \frac{1}{2}\left|comp_i\right|\left(\left|comp_i\right|-1\right)}{C_n^2} \tag{22}$$

where C_n^2 is the number of all node pairs, $comp_i$ is the *ith* component and *num* is the number of components in the network. Considering all the possible initial injected nodes, we define a metric named Expected Pair Connectivity (EPC) in the following equation,

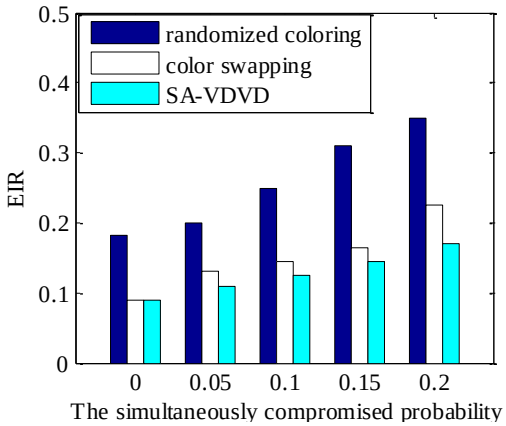
$$EPC = \frac{1}{n}\sum_{k=1}^{n} U_k \tag{23}$$

where U_k indicates the PC in the case that node k is initially injected. It is worth mentioning that we assume the probability for each node initially injected is equal for the sake of simplicity. We also assume that there are 6 variants in total. Fig.

8 shows the EPC using different algorithms against the number of



(a) The results on Cenic topology



(b) The results on Interoute topology

Fig. 9 EIR under the different simultaneously compromised probabilities for different coloring algorithms

variants. As per the figure, the EPC increases as the number of variants rises for each algorithm. Obviously, the proposed SA-VDVD algorithm outperforms other algorithms for a certain number of variants. Moreover, with the increase of the number of variants, the EPC for all algorithms will be gradually closed.

2) The impact of compromised probability

In this section, we investigate the effect from the probability of compromised events on the defense capability. Next, we take an example to illustrate the effect. The related parameter is configured as follows. Assume that there are 3 variants in total which can be used in our coloring algorithms. For the clarity purpose, assume that the compromised events comprised only of two types. One is the nodes with the same variants are compromised, and the other is the two variants in the variants set that can be compromised simultaneously. We study the probability that two variants are simultaneously compromised varying from 0 to 0.2.

Fig. 9 shows the defense capability using different coloring algorithms against the probability that two variants are simultaneously compromised on Cenic and Interoute respectively. As can be observed, for each algorithm, the EIR increases as the probability that two variants are compromised rises. Obviously, with the increase of the probability that two variants are compromised, the spread ability of the malicious

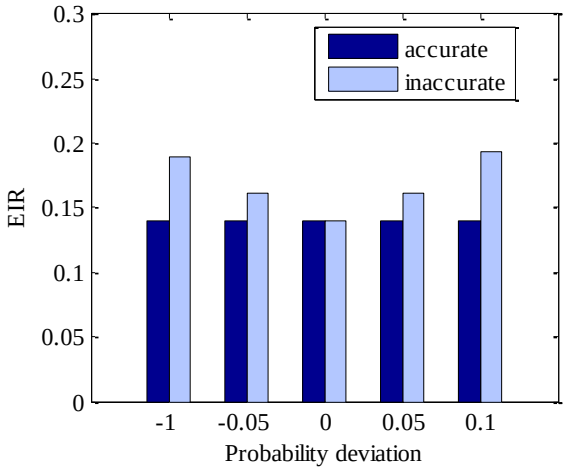


Fig. 10 EIR obtained by different information under the different probability deviations

packets attack will improve across the distinct nodes. As a result, more nodes will be infected. Moreover, the proposed SA-VDVD algorithm outperforms the other coloring algorithms and randomized coloring performs the worst in the case of any probability. When the probability that two variants are compromised is relatively low, the goodness of the proposed SA-VDVD algorithm is not fully clear. In this case, spread ability of the malicious packets attack across the distinct nodes is not powerful. Thus, defense effectiveness is similar among the algorithms except for randomized coloring. As the probability that two variants are compromised increases, the advantage of the proposed SA-VDVD algorithm turns to be clear. In such a case, spread ability across distinct nodes will increase quickly. Hence, the SA-VDVD algorithm considering common vulnerabilities can restrain the propagation of malicious packets attacks more effectively than other coloring algorithms without considering common vulnerabilities.

3) The impact of inaccurate information

So far, we have assumed that the probabilities for all the compromised events are priori known for us. In real scenarios, these probabilities could be obtained via two ways, one is expert opinion and the other is real world statistics. However, both techniques cannot be completely accurate. In this section, we study the case that the deployment is based on inaccurate information.

We use the following equation to depict the inaccurate information,

$$P'(e_i) = P(e_i) + \Delta_i \tag{24}$$

where $P'(e_i)$ indicates the available probability, $P(e_i)$ indicates the realistic probability and Δ_i indicates the deviation between the available probability and the realistic probability for a compromised event e_i. In this section, the evaluated probability refers to the probability that two variants are simultaneously compromised. The probability deviation is set ranging from -0.1 to 0.1 with the common difference 0.05.

Fig. 10 shows the EIR obtained by different information using the proposed SA-VDVD algorithm against the probability deviation that two variants are compromised. As is

seen in the figure, the deviation value between accurate and inaccurate increases as the absolute value of probability

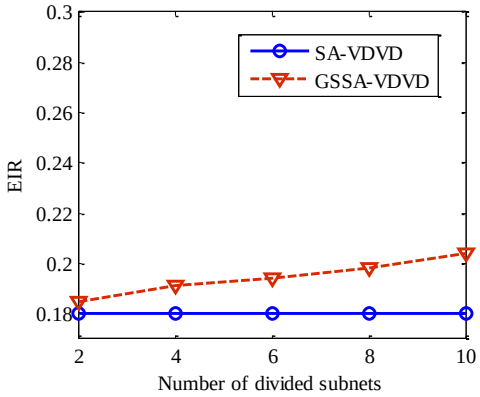


Fig. 11 EIR under the different number of divided subnets for SA-VDVD and GSSA-VDVD

deviation increases. The explanation is that the obtained deployment with inaccurate information can be not optimal.

4) The impact of the topology size

As stated in Section V, the computation time becomes excessive when the size of the network is large. In this section, we explore this particular situation in the aspect of defense performance and computational complexity. Assume that there are 3 available variants. Simulations are conducted on Interoute topology where we first compare the defense performance of different deployment schemes on the same network topology. Fig. 11 shows the EIR versus the divided subnets for SA-VDVD and GSSA-VDVD. As can be seen in Fig. 11, the deployment based on SA-VDVD algorithm clearly outperforms the deployment based on GSSA-VDVD algorithm. Furthermore, as the number of divided components increases, the EIR will increase simultaneously. When the number of divided subnets is relatively small, the performance degradation based on GSSA-VDVD algorithm is not obvious.

Afterward, we investigate the relationship of computational complexity for different algorithms. To reflect the compared results clearly, normalization method is adopted. More specifically, the computational complexity of deployment based on SA-VDVD algorithm is selected as the benchmark. For the other scheme, the computational complexity is calculated as the ratio of the actual computational complexity of a certain scheme to that of the benchmark. Fig. 12 shows the computational complexity versus the number of divided subnets for various algorithms. We can observe that the deployment based on GSSA-VDVD algorithm has lower computational complexity than schemes based on SA-VDVD algorithm. Moreover, the computational complexity for the deployment scheme based on GSSA-VDVD algorithm will decrease rapidly with the increase of the number of divided subnets.

VII. Related Work

A. Cyber security

Keeping in view the exponentially increasing cybercrime and fraudulent online activities by fraudster, [20] delves into the various cyber fraud and probable solutions. B. B. Gupta identifies in [21] the emergent research and techniques being utilized in the field of cryptology and cyber threat prevention.

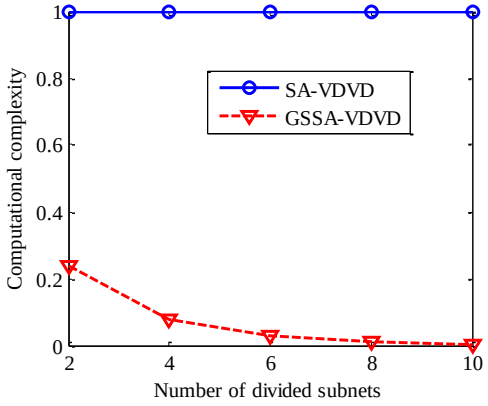


Fig. 12 Computational complexity under the different divided subnets for SA-VDVD and GSSA-VDVD

L. Wang et al. [22] propose a novel compressive sensing scheme that supports medical image sampling, compressing, encryption and confidentially homomorphic aggregation simultaneously. Internet of Things (IoT) provides new types of services in order to improve everyday life [23]. [24] describes challenges that threaten IoT diffusion and presents open research questions. [25] surveys the security challenges of the integration of IoT and cloud computing. [26] presents a comprehensive survey of secured web application by identifying numerous serious threats faced by several-related organizations. [27] presents a cloud-based framework that thwarts the DOM-based XSS vulnerabilities caused due to the injection of advanced HTML5 attack vectors in the HTML5 web applications.

B. Diversity

Diversity has long been recognized as a promising solution to improve the resilience of a software system against various vulnerabilities. In fact, diversity is no longer a new thing, and it has been extensively applied in a variety of fields such as biology and organic systems [19]. Here, we only investigate its application in computer and network context. At the beginning, design diversity has been investigated in fault tolerance for a long time. A typical case is N-Variant programming which builds $N \geq 2$ functionally equivalent programs and detects the faulty version by comparing the output results [28]. Then the N-Variant system extends N-version programming to detect intrusions [29]. Apparently, it is an important premise to generate diversity. So far, substantial literature adopting randomization techniques has been put forward to auto-generate diversity [30][31][32]. Recently, from the point of view of application fields, diversity has been extensively used in some new scenarios such as cloud computing security [33], Moving Target Defense (MTD) [34] and network routing [35][36][37].

The aforementioned studies are basically directed against a point defense which focuses on the security of network node. To achieve the global network defense, inspired by the survivability through heterogeneity philosophy, Zhang et al. [9] first propose a novel survivability paradigm named heterogeneous networking to improve the survivability of a network. In terms of the relationship between diversity and the

robustness of a network, Juan Caballero et al. [10] employ a graph theoretic approach to explore the benefits of diversity for the robustness of a network, where robustness is the property of a network staying connected under a software failure. Though diversity has been extensively used, most of the existing efforts rely on intuitive and imprecise notions of diversity. At a higher abstraction level, as a global property of the entire network, diversity and its effect on security have been overlooked. Zhang et al. [38] model network diversity as a security metric by designing and evaluating a series of diversity metrics and provide guidelines for the instantiation of the proposed metrics.

C. Graph Coloring

As a well-known problem in graph theory, graph coloring problem focuses on the assignment of colors to nodes of the graph subject to certain constraints [39]. In general, it guarantees that two adjacent nodes possess distinct colors. So far, graph coloring problem has been extensively used in various fields such as curriculum schedule, traffic management and network, etc. Here we mainly focus on its application in the area of networking. O'Donnell et al. [19] transform the problem of limiting the spread of malware via diversity on a network topology into the graph coloring problem and propose a series of distributed coloring algorithms. Similarly, graph coloring problem is adopted in order to study how to maximize the robustness of a network via diversity [10].

D. Malicious Packets Attack

The authors in [40] propose a range of attack approaches to illustrate that a mal-packet, which only carries specially crafted data, can exploit memory-related vulnerabilities and utilize existing application codes in a sensor to propagate itself without disrupting sensor's functionality. In [41], the authors illustrate the feasibility of launching sensor worms through trial experiments on Mica2 motes and investigates the technique of software diversity to combat sensor worms.

To summarize, our work is inspired by existing work of [9, 10, 18, 40], but, also, different from them. We investigate an optimal scheme to restrain the spread of malicious packets attack via diversity deployment under a more realistic condition, and theoretically formulate it as integer-programming.

VIII. Conclusions and Future Works

The paper investigated the existing common vulnerabilities among different variants which coincides well with the actual situation. We, first, devised a quantitative metric reflecting the effect on the network brought by the spread of malicious packets attacks. Then, we modeled the VDVD problem as an integer-programming problem and proposed a SA-VDVD algorithm to solve it. In addition, we proposed a GSSA-VDVD algorithm to address the high computational complexity as the size of network increases. Finally, we performed a series of experiments to verify the validity of the proposed algorithms. The proposed SA-VDVD algorithm outperformed other coloring algorithms by 35% in defense capacity. When variants had a low correlation, the superiority of SA-VDVD algorithm was not significant. GSSA-VDVD algorithm can effectively reduce the computational complexity.

In future, we intend to study the case where attackers can explore more than one vulnerability.

Acknowledgment

This work is supported by the National Key Research and Development Plan [No.2016YFB0800101], the Foundation for Innovative Research Groups of the National Natural Science Foundation of China [No.61521003] and the National Natural Science Foundation of China [No.61602509].

References

[1] Vulnerability Note VU#403568(Dec 2015) http://www.kb.cert.org/vuls/id/403568
[2] Vulnerability Note VU#391604(Nov 2015) http://www.kb.cert.org/vuls/id/391604
[3] Vulnerability Note VU#185100(Jan 2013) http://www.kb.cert.org/vuls/id/185100
[4] Vulnerability Note VU#893726(Apr 2014) http://www.kb.cert.org/vuls/id/893726
[5] Vulnerability Note VU#583638(Jan 2005) http://www.kb.cert.org/vuls/id/583638
[6] Vulnerability Note VU#305448(Mar 2017) http://www.kb.cert.org/vuls/id/305448
[7] Vulnerability Note VU#677427(Nov 2016) http://www.kb.cert.org/vuls/id/677427
[8] Vulnerability Note VU#917700(Apr 2014) http://www.kb.cert.org/vuls/id/917700
[9] Zhang, Yongguang, et al. "Heterogeneous networking: a new survivability paradigm." Workshop on New Security Paradigms (2001):33-39.
[10] Caballero, Juan, et al. "Would Diversity Really Increase the Robustness of the Routing Infrastructure against Software Defects?" Network and Distributed System Security Symposium, NDSS 2008, San Diego, California, USA, February DBLP, 2008.
[11] S. Kirkpatrick, C.D. Gelett, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 621–630.
[12] N. Metropolis, A.W. Rosenbluth, M.N Rosenbluth, A.H. Teller, E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (1953) 1087–1092.
[13] Frausto-Solís, et al. "ANDYMARK: An Analytical Method to Establish Dynamically the Length of the Markov Chain in Simulated Annealing for the Satisfiability Problem." LNCS 4247, pp. 269–276, 2006.
[14] Van Laarhoven, P.J.M. and Aarts, E.H.L. (1987) Simulated Annealing: Theory and Applications. Reidel, Dordrecht
[15] Strenski, P.N. and Kirkpatrick. "Analysis of Finite Length Annealing Schedules." Algorithmica, 6, pp.346-366, 1991.
[16] Tabatabaei, Seyed Salim, M. Coates, and M. Rabbat. GANC: Greedy agglomerative normalized cut for graph clustering. Pattern Recognition. 2012, 45(2), pp. 831-843.
[17] Galinier P, Boujbel Z, Coutinho Fernandes M. An efficient memetic algorithm for the graph partitioning problem. Annals of Operations Research, 2011, 191(1), pp. 1-22.
[18] CENIC, available at http://www.cenic.org [last accessed February 2019]
[19] O'Donnell, Adam J., and H. Sethu. "On achieving software diversity for improved network security using distributed coloring algorithms." ACM Conference on Computer and Communications Security, CCS 2004, Washington, Dc, Usa, October DBLP, 2004:121-131.
[20] B. B. Gupta. Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives. CRC Press, Taylor & Francis, 666, 2018.
[21] B. B. Gupta. Handbook of research on modern cryptographic solutions for computer and cyber security. IGI Global, 2016.
[22] Licheng, Wang, et al. Compressive Sensing of Medical Images with Confidentially Homomorphic Aggregations. IEEE Internet of Things Journal (2018):1-1.

[23] A.P. Plageras, K.E. Psannis, C. Stergiou, H. Wang, B.B. Gupta. Efficient IoT-based sensor BIG Data collection–processing and analysis in smart buildings. Future Generation Computer Systems 82 (2018): 349-357.

[24] Tewari, Aakanksha, and B. B. Gupta. Security, privacy and trust of different layers in Internet-of-Things (IoTs) framework. Future Generation Computer Systems, Elsevier, 2018.

[25] Stergiou, C, Psannis, KE, Kim, BG, B. B. Gupta. Secure integration of IoT and cloud computing. Future Generation Computer Systems, 2018, 78(3), pp. 964-975.

[26] Gupta, Shashank, and B. B. Gupta. Detection, avoidance, and attack pattern mechanisms in modern web application vulnerabilities: present and future challenges. International Journal of Cloud Applications and Computing (IJCAC), 2017, 7(3), pp. 1-43.

[27] Enhancing the browser-side context-aware sanitization of suspicious HTML5 code for halting the DOM-based XSS vulnerabilities in cloud." International Journal of Cloud Applications and Computing (IJCAC), 2017, 7(1), pp. 1-31.

[28] Iñaki Goirizelaia, Maider Huarte, Juanjo Unzilla, Ted Selker: An Optical Scan E-Voting System based on N-Version Programming. IEEE Security & Privacy 6(3): 47-53 (2008).

[29] B. Cox et al., "N-variant systems: A secretless framework for security through diversity," in Proc. 15th Conf. USENIX Secur. Symp., 2006, Art. ID 9

[30] S. Bhatkar, D. C. DuVarney, and R. Sekar, "Address obfuscation: An efficient approach to combat a broad range of memory error exploits," in Proc. 12th USENIX Secur. Symp., Washington, DC, USA, 2003, pp. 105–120.

[31] S. Bhatkar and R. Sekar, "Data space randomization," in Proc. 5th Int. Conf. Detection Intrusions Malware, Vulnerability Assessment (DIMVA), 2008, pp. 1–22.

[32] PaX Address Space Layout Randomization. [Online]. Available: http://pax.grsecurity.net/, accessed Jun. 2015.

[33] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," IEEE Internet Comput., vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.

[34] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats, 1st ed. New York, NY, USA: Springer-Verlag, 2011.

[35] Chen, Y. Y., Chang, E. J., Hsin, H. K., Chen, K. C., & Wu, A. Y. (2017). Path-diversity-aware fault-tolerant routing algorithm for network-on-chip systems. IEEE Transactions on Parallel & Distributed Systems, 28(3), 838-849.

[36] Chen, W., Lea, C. T., He, S., & Zhe, X. Y. (2017). Opportunistic routing and scheduling for wireless networks. IEEE Transactions on Wireless Communications, 16(1), 320-331.

[37] Tapolcai, J., et al. "Scalable and Efficient Multipath Routing: Complexity and Algorithms." (2016).

[38] Zhang, M., Wang, L., Jajodia, S., Singhal, A., & Albanese, M. (2016). Network diversity: a security metric for evaluating the resilience of networks against zero-day attacks. IEEE Transactions on Information Forensics & Security, 11(5), 1071-1086.

[39] T. R. Jensen. Graph Coloring Problems. Wiley, 1995.

[40] Gu, Q., & Noorani, R. (2008). Towards self-propagate mal-packets in sensor networks. Acm Conference on Wireless Network Security (pp.172-182).

[41] Yi Yang, Sencun Zhu, & Guohong Cao. (2016). Improving sensor network immunity under worm attacks: a software diversity approach. Ad Hoc Networks, 47(1), 26-40.