An Ontological Graph Identification Method for improving Localisation of IP Prefix Hijacking in Network Systems

Osama S. Alkadi, *Member, IEEE*, Nour Moustafa, *Member, IEEE*, Benjamin Turnbull, *Member, IEEE* and Kim-Kwang Raymond Choo, *Senior Member, IEEE*

IP prefix hijacking continues to be a pervasive cyber security threat to the core internet routing infrastructure. The data security of multiple cloud-based services is also susceptible to these threats, due to the high dependency on traditional routing protocols. Although a number of hijacking detection techniques have been recently proposed, no existing system has effectively addressed the problem of detecting malicious transit Autonomous System (AS) services in any detected hijacking occurrences. The ability to locate and isolate malicious services is critical for conducting a necessary mitigation strategy at an early stage, to minimise the impact of the attack, to restore cloud services quickly. In this paper, we propose an effective real-time processing method, so-called Ontological Graph Identification (OGI), for detecting IP prefix hijacking of nodes and suspicious transit nodes caused by the hijacked nodes through ASs. The proposed method is evaluated using the two public datasets of RIPE RIS and RouteView. Experimental results revealed improved performance for the detection of malicious transit nodes compared with peer techniques. It is, therefore, shown that the proposed method has utility in automating the process of investigating nodes with suspicious activities in real network systems.

***Index Terms*—IP prefix hijacking, anomaly detection, BGP hijacking, network systems**

I. Introduction

THE underlying routing infrastructure of the Internet contains thousands of independent autonomous systems (AS), all of which rely on the Border Gateway Protocol (BGP) to exchange routing information. While BGP is designed for routing network traffic and adequately maintaining network reachability information between peered autonomous systems, it lacks any form of trust mechanism, leaving it susceptible to misconfiguration or malicious prefix hijacking [1], [2]. This phenomenon occurs when an AS advertises to surrounding neighbours existing prefixes, i.e., groups of IP addresses, to falsify ownership and maliciously reroute someone else's traffic to other network destinations.

Routing attacks have the potential to cause severe threats to inter-domain routing infrastructures [3]. For example, malicious users exploited vulnerabilities of BGP and DNS protocols to intercept and reroute traffic of Amazon's Route 53 to third-party servers in Russia [4]. A hacker also rerouted traffic destined to networks of 19 Internet Service Providers (ISPs), involving data from the networks of Amazon and other service providers, such as DigitalOcean and OVH, to steal cryptocurrency of bitcoin users [5]. A hijacking attack occurred when an ISP in Pakistan forwarded the announcement of Routing Information Service (RIS) to hijack the YouTube website for several hours [6]. These examples highlight the potential scale of hijacking against global infrastructures.

In literature, to defend against IP prefix hijacking attacks, mechanisms are categorised into *reactive* and *proactive* [7]–[12]. Reactive mechanisms respond to hijacking attacks as they occur based on the two stages of detection and mitigation. Detection is often offered by third-party services to route information, such as BGP updates and traceroutes, to inform networks about malicious events including their prefixes. Then, the targeted networks mitigate the events using different strategies, for example, announcing particular prefixes or contacting other ASs to clean announcements [7]. Proactive mechanisms try to prevent hijacking events from occurring or minimise their effect [13]–[16]. Network systems depend on practical reactive mechanisms to defend against prefix hijacking since proactive methods are expensive in terms of costs and deployments [17]. Although there has been rapid adoption of resource public-key infrastructure (RPKI) framework to BGP by cryptographically signing IP prefixes using Route Origin Authorizations (ROAs) [18], [19], particularly by Regional Internet Registries (RIRs), ISPs and various cloud providers, it would still require re-implementation of existing systems, making it pragmatically a long-term strategy, rather than of immediate use.

The mechanisms previously outlined provide partial solutions to the issue of malicious transit ASs, but are not specifically designed as comprehensive detection systems against all forms of malicious behaviour. Malicious transit nodes are ASs that are mostly benign, but behave suspiciously in forwarding malware from one or more sources to end points as a transit service. Although upstream transit ASs can be used to disconnect a maliciously behaving AS, they may also be controlled by cyber criminals. Such malicious transit ASs are harder to detect as their malicious behaviour is sporadic, and they are therefore unlikely to be listed in IP blacklists [20]. There is a need for a robust technique that can locate hijackers' services and their nodes with transit malicious activity [7].

This paper proposes a new Ontological Graph Identification (OGI) method that can effectively identify malicious nodes through ASs and discover suspicious events from transit nodes caused by malicious nodes in real-time. This method would

O. S. Alkadi, N. Moustafa, and B. Turnbull are with the School of Engineering and Information Technology, University of New South Wales, Canberra, ACT 2600 Australia, K.-K. R. Choo is with the Department of Information Systems and Cyber Security and Department of Electrical and Computer Engineering, University of Texas at San Antonio, San Antonio, TX 78249-0631, USA. He also has a courtesy appointment at the School of Information Technology and Mathematical Sciences, University of South Australia, Adelaide, SA 5095 Australia.
E-mail: o.alkadi@student.unsw.edu.au; nour.moustafa@unsw.edu.au; benjamin.turnbull@unsw.edu.au; raymond.choo@fulbrightmail.org

allow countermeasures, such as automatic de-peering, to be proactively implemented in a timely manner. It also serves as an intelligent system for assessing and detecting early malicious activities. The method has a new capability of being operated by an AS itself without relying on third-party cloud infrastructure such as edge locations. While speed may be a limitation for very large network graphs in real world applications, the proposed OGI malicious transit nodes detection system could take advantage of existing blacklists for better node weight re-initialisation. A single iteration of the proposed OGI method requires $\sim$1 second, and thus the node scores recalculation in this case would make it feasible for supporting online hijacking detection systems in real-time.

The key contributions in this paper can be summarised as follows:

(a) We propose a *path verification approach* for identifying possible illegitimate paths (i.e. paths containing one or more malicious nodes) in the network prior to AS malicious node localisation. Malicious nodes would be penalised whenever they appear on an illegitimate path. All other AS nodes on illegitimate paths besides the malicious ones are labelled as suspicious, and corresponding penalty values would be passed on to the next stage of malicious transit node detection. Using this methodology, an exhaustive search on the whole network graph is avoided and the speed of malicious activity detection is significantly improved.

(b) We suggest an *Ontological graph-based ranking* model that classifies illegitimate paths. A PageRank algorithm is utilised to give higher weights to malicious nodes according to the previous stage. By that, the neighbours of malicious nodes receive higher weights depending on how close they are to how many malicious nodes.

(c) We also *reduced noise of malicious transit nodes* (i.e. legitimate nodes compromised by adversary malicious nodes) by normalising the estimated nodes scores according to the total average score of all AS with similar number of adjacent neighbours. Hence a node score which falls short compared to the adaptively set threshold would be discarded.

The paper is organised as follows. The background and state-of-the-art work on IP prefix hijacking are discussed in Section II. The proposed ontological graph identification method is described in Section III. Following that, Section IV describes the experimental results of our framework on the Reseaux IP Europeens (RIPE) dataset. Lastly, we draw our conclusions and give some perspectives of future work in Section V.

II. Background and Related Work

This section discusses most common prefix hijacking mechanisms that are considered a major threat for ASs in cloud services, including different scenarios related to blackholing, imposture and interception attacks. Where relevant, real world examples are also given. This section also summarises and critically analyses major related recent work in the field.

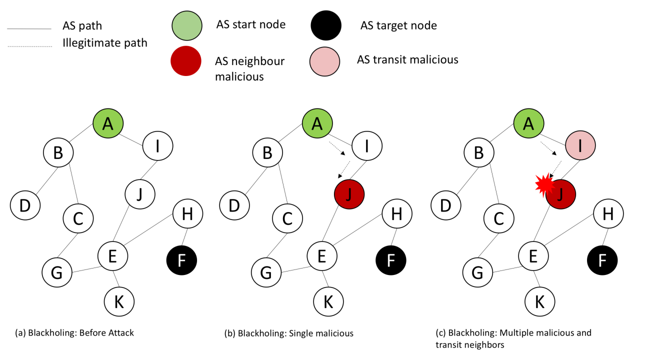

AS path
Illegitimate path
AS start node
AS target node
AS neighbour malicious
AS transit malicious
A
B
I
D
J
C
H
E
F
G
K
(a) Blackholing: Before Attack
(b) Blackholing: Single malicious
(c) Blackholing: Multiple malicious and transit neighbors

Fig. 1. Blackholing attack scenario showing AS graph structure (a) before the attack, and during the attack in case of single and multiple malicious neighbours in (b) and (c), respectively.

A. Malicious Attack Models

IP Prefix hijacking falsely advertises to neighbouring AS the ownership of IP prefixes or addresses. The malicious AS reroutes traffic destined for these IPs to itself. Hijackers use the control plane (i.e., crafted or misconfigured BGP announcements) to eventually affect the data plane. Thus, IP Prefix hijacking is typically launched on the control plane to affect the behavior of the data plane. The outcomes of a malicious AS successfully hijacking an IP Prefix that can be categorised into one of the following three scenarios [7], [10]; *blackholing*, *imposture*, and *interception*. Each of these is discussed separately.

Blackholing is a type of denial-of-service attack on a network system when an attacker discards incoming or outgoing network traffic, without informing the source that the data has not reached its destination [7]. Figure 1 illustrates a scenario in which a malicious host could make another network unavailable by falsely hijacking the route, and then using its position as a route for all traffic destined for a legitimate address and discarding it. From the victim's perspective, there is no incoming traffic at all. In Figure 1(a), a normal scenario is presented before the attack. Figure 1(b) represents an attack with a single malicious neighbour for node J, while Figure 1(c) illustrates an attack with multiple malicious neighbours when node I has accepted malicious routing path of node J while dropping the original packet of node A. In this occurrence, node I is still considered an legitimate node and is not aware of malicious activity which is occurring. It is not yet classified or blacklisted as malicious, but has a potential to become a malicious node. Thus, we call such nodes as 'transit malicious', as the malicious activities are passed across or through node I, and orchestrated by the malicious node J. In this case, the traffic is dropped. The traffic is not dropped on node I, so it can be exploited for further attacks to conceal the identity of the malicious node J.

An Imposture scenario is analogous to the man-in-the-middle attack, where an attacker deliberately intercepts traffic by altering packet information. However, this traffic will never reach its destined target, but will instead be acknowledged and replied by the same malicious host [10]. Figure 2 represents the imposture attack scenario which shows the graph structure

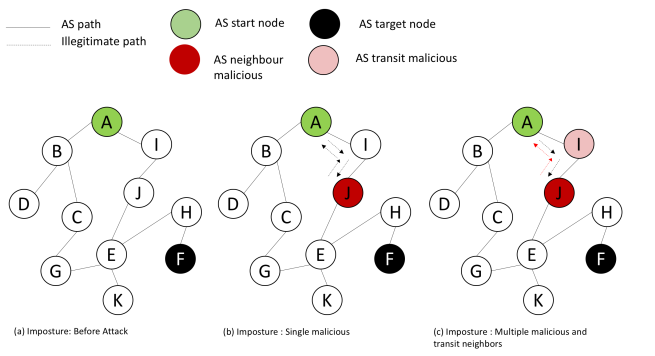

AS path
Illegitimate path
AS start node
AS neighbour malicious
AS target node
AS transit malicious
A
B
C
D
E
F
G
H
I
J
K
(a) Imposture: Before Attack
(b) Imposture : Single malicious
(c) Imposture : Multiple malicious and transit neighbors

Fig. 2. Imposture attack scenario showing AS graph structure (a) before the attack, and during the attack in case of single and multiple malicious neighbours in (b) and (c), respectively.

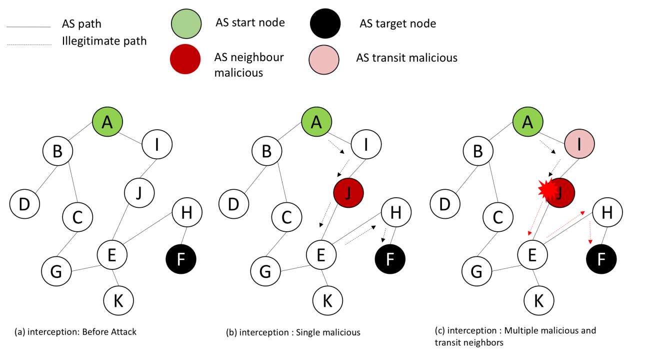

AS path
Illegitimate path
AS start node
AS neighbour malicious
AS target node
AS transit malicious
A
B
C
D
E
F
G
H
I
J
K
(a) interception: Before Attack
(b) interception : Single malicious
(c) interception : Multiple malicious and transit neighbors

Fig. 3. Interception attack scenario including a transit activity, showing AS graph structure (a) before the attack, and during the attack in case of single and multiple malicious neighbours in (b) and (c) respectively.

before the imposture attack and with single and multiple malicious neighbours. Figure 2(a) illustrates an initial state of a normal scenario before the start of the attack. Figure 2(b) represents a single malicious neighbour in node J, while Figure 2(c) is an example of an imposture attack scenario with many malicious neighbours, that is, when node I has accepted the malicious AS routing path of node J. Particularly, node J alters the original packet from node A and imitates the final destination (i.e., AS targeted node F).

Interception is the most sophisticated hijacking scenario, as illustrated in Figure 3, where an attacker eavesdrops and possibly alters the victims packet information before forwarding it to the target destination [10]. Figure 3 presents an interception attack scenario which shows the AS graph structure before and with single and multiple malicious neighbours. In Figure 3(a), a legitimate scenario is represented before the start of an attack. Figure 3(b) shows an interception attack with a single malicious neighbour in node J, where the attacker alters the original packet from node A, and then forwards the packet to the final destination F. Figure 3(c) represents an interception attack with multiple malicious neighbours in the nodes J and I, respectively, that forwards the original packet from node A after eavesdropping and altering the packet information.

TABLE I
COMPARING DIFFERENT IP PREFIX HIJACKING DETECTION TECHNIQUES

Method	Control plane	Data plane	Hybrid method	Identify attacker	Malicious transit
Lad at al. [9] Chaviaras et al. [24]	Yes	No	No	No	No
Zhang et al. [12] Zheng et al. [10] Tahara et al. [25] Mickens et al. [26]	No	Yes	No	No	No
Kruegel et al. [27]	Yes	Yes	Yes	No	No
Shi et al. [11] Hong et al. [28]	Yes	Yes	Yes	No	No
Qui et al. [29]	Yes	Yes	Yes	Yes	No
Our method	Yes	Yes	Yes	Yes	Yes

B. Related Work

Several research studies have proposed detection systems for identifying BGP hijacking. The systems depend on *control plane* and/or *data plane* techniques [7], [21], [22]. Control plane techniques monitor BGP updates and routing tables to identify malicious behaviours of BGP messages such as the origin of prefixes or unexpected path changes. Data plane techniques monitor the routes reachability from the victim to detect suspicious events, where they employ traceroutes to discover abnormal behaviours in the data routes. The aforementioned three scenarios of IP prefix hijacking could be identified and prevented using data-plane approaches because the approaches can effectively examine routes of malicious data.

One of the early attempts in providing a detection system of IP prefix origin changes of ownership is the Prefix Hijacking Alert System (PHAS) [9]. PHAS is designed to detect multi-origin AS path conflicts. PHAS runs exclusively on the control plane as a means of ensuring a low false alarm rate. PHAS utilises a time-window function to concurrently train and validate the model as a means of reducing the number of repeated alerts. However, PHAS is unable to detect complex IP hijacking scenarios such as the ones described in [21], [22]. In [23], the authors provided alternative heuristic techniques to multi-origin autonomous system conflicts by modifying routing policies, which produced promising results to reduce false negative alarms. The two methods cannot detect sophisticated malicious events due to their complex design for tracking prior information about normal or suspicious events.

The method outlined in [30] also seeks to solve the issues of working with multi- and sub-origin autonomous systems in real-time. The approach taken is similar to the method outlined by Tahara et al. [25], utilising active probing tests to differentiate legitimate ownership from false claims. Also, a tool for monitoring per-host availability trends in enterprise settings is proposed in [26]. The method operates by drawing live measurements to detect network anomalies, like IP hijacking in real-time. However, these approaches of continuous pinging of the entire Internet would introduce considerable load and network performance issues. In a similar approach, the Lightweight Distributed Measurement Scheme (LDMS) by Zheng et al. [10] proposed using data plane techniques. LDMS relies on pre-selected external third-party vantage points and monitoring services. However, in the previous scheme, announcing a route to a subnet within a publicly announced prefix and then using it for spam and phishing can go undetected - unless an individual sub-prefix is monitored.

There are also hybrid approaches to detecting prefix hijacking events, combining both the data and control planes in an effort to improve detection rates [11]. These two processes may be initiated simultaneously, by gathering the control route status and data reachability of the identified hijacked events. The authors in [27] utilised the data of Route Views project to distinguish between legitimate and potential malicious traffic. The method utilises topology information of the AS connectivity, and is capable of detecting malicious inter-domain routing update messages through passive monitoring of BGP traffic. However, the proposed method does not automatically account for changes of IP address ownership and removal of connections between ASs. Other hybrid methods, such as Chaviaras et al. [24], designed a framework for accelerating the detection of prefix hijacking incidents from real-time BGP data. Schutrup et al. [31] proposed an approach for detecting five different types of hijacks by less specific route announcement. A direct view was provided from operational BGP routers without intermediate collectors via Looking Glass servers, and real-time feeds are made available via BGP collectors with live data streaming (e.g. RIPE RIS [6] and BGPmon projects [30]). Also, the work in [29] located the prefix hijacker ASs based on distributed AS path measurements. Each attacker AS was located by actively monitoring paths to the victim prefix from selected monitors distributed on the Internet. Nevertheless, the problem of highlighting potential attackers (i.e. normal AS with suspicious behaviour) has not been dealt with. A summary comparing different recent proposed IP prefix hijacking detection techniques based on different assessment metrics is listed in Table I. As shown in Table I, our method can effectively detect IP prefix hijacking attacks, along with considering control and data planes. As our method uses graph models to statistically structure network nodes using an adaptive PageRank algorithm [16], it is hybrid and scalable, as it estimates a small number of optimised parameters that can be easily self-tuned in real-time processing.

III. Proposed Methodology

We propose a novel method, named Ontological Graph-based Identification (OGI) for identifying IP prefix hijacking attacks via recognising their illegitimate paths. The novelty of this method comes from detecting suspicious nodes of ASs and recognising their malicious events within any transit node in real-time processing. The OGI method has two new functions: 1) it does not rely on an external blacklist of hostile IP addresses; and 2) normal nodes scoring higher ranks are identified as malicious based on the accumulated weights of neighbours for every node.

A systematic architecture of the proposed OGI method is depicted in Figure 4. Assume that we have different Cloud Providers (CP) labelled $\{CP_t, CP_x, CP_z, CP_y, CP_v\}$, each with multiple ASs. The collected BGP update announcements from these and surrounding AS neighbours are the training and

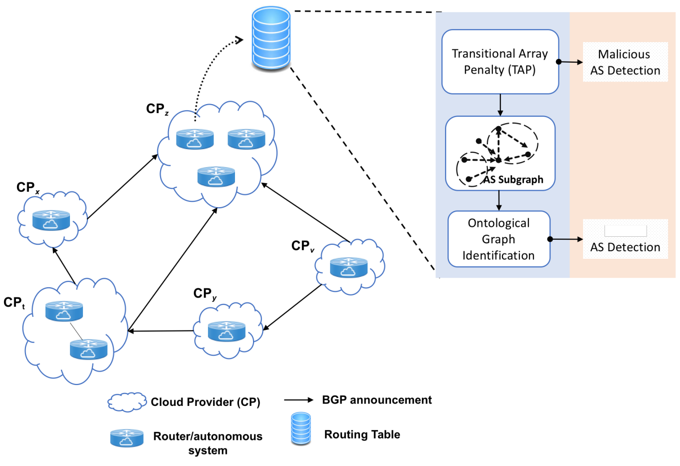


Fig. 4. System Architecture of identifying normal AS exhibiting malicious behaviour.

testing data that are used for validating the proposed method. The method depends on dividing a network into subgraphs for easily tracking suspicious events, whereby illegitimate paths containing malicious nodes are flagged. This is discussed further in Section III.A.

The reliability of the AS route is assessed through the use of a proposed *Transitional Array Penalty* (TAP) metric. The TAP metric penalises each AS in relation to the proportion of hosted malicious transit AS. If one of the nodes does not have a physical path to the destination, the TAP will assign a penalty value of 1 to the node. All penalty values are normalised to make the computational process much easier before the phase of the path validity verification outlined in Section III-B.

The output of the TAP is used to generate a graph using the PageRank algorithm [16]. The method can assign a score to each AS individually by computing its likelihood probability. Consequently, this helps in properly specifying prior weights for each AS before running the method. By assessing the weights of the highest-scoring node in ASs before and after running the PageRank algorithm, every node is scored using the estimated likelihood for computing the weights of transit nodes and their originally hijacked nodes through ASs, as explained in Section III-C2.

In order to evaluate the efficiency of the OGI method, the AS node weights are equally initiated to run the PageRank algorithm. The algorithm estimates the appearance frequency of nodes on illegitimate paths as malicious nodes when their weights are much higher than the weights of normal nodes. This process is detailed in Sections III-C.

A. Subgraph partitioning

As the scale of the network increASs, the number of AS hosts to be investigated in real-time also expands. This renders the process of investigating the integrity of the BGP update message infeasible within a reasonable convergence time. To counter this, we propose the use of a subgraph analysis approach. It is an effective approach which divides the network structure into multiple subgraphs prior to AS malicious node localisation. The use of subgraphs can assist in identifying illegitimate paths contaminated with malicious nodes.

Through the use of subgraph partitioning, an exhaustive search on the entire network graph is avoided, improving the search speed for malicious activity. A K-means algorithm [32], based on the Euclidean distance function, is used for clustering the network into multiple network regions using a predefined k value. Each region/cluster reflects a subgraph with roughly similar probability densities for their network nodes, and then each of these subgraphs can be dealt as a separate graph with the same statistical characteristics. The generated subgraphs can assist in identifying illegitimate paths contaminated with malicious nodes based on estimating their low likelihood probability. The subgraph partitioning phase helps to significantly improve the computational resources of the method by avoiding the exhaustive search on the entire network graph. To identify illegitimate paths including malicious nodes and their suspicious transit nodes, a path verification approach is proposed as illustrated below.

B. Path Verification Approach

A Transitional Array Penalty (TAP) metric is proposed for verifying paths. The TAP metric can be defined as a directed graph $G(V, E)$ derived from BGP AS-path P, where the set of nodes or vertices V are the AS, and E denotes the directed edges between two connected AS. By definition, this graph could contain directed cycles or loops. Let $P_v(i)$ be the accumulated penalty value of a node $i \in V$ occurring on a certain AS path of a route $R, (j, i) \in E$ which describes a directed edge from a node j to a node i. Once a BGP update message reaches node i from a neighbouring node j, a bottom-up approach is followed by i to verify the authenticity of the advertised message, where i starts first with the destination node prefix n of the route R and checks whether its predecessor node $n-1$ is directly connected (i.e. has a physical path) to n. The same scenario is used to analyse if node $n-1$ is a neighbour of node $n-2$, until reaching the starting node i.

If a disconnected path is discovered between any of the traced nodes, then a penalty value of 1 is assigned to this path. The weight is given to the node up the stream (i.e. node $n-1$ if we are checking node n), and the penalised node is considered malicious. The same approach applies to the rest of the possible paths from node i to n. Each path containing one or more malicious nodes is considered an invalid path M_p. For example, transitional matrices for all possible paths are shown in Figure 5(b), where D_r is the traced AS path of route R; H_n is the number of hop counts from node i to node k; D_p is the predecessor of the destination node k in R to origin; P_v is the penalty value for a node i.

The TAP metric is used for determining actual paths of nodes with their neighbours to discriminate between actual paths of nodes and disconnected nodes. The model assigns a higher score to the discounted nodes compared with the actual path. It can, therefore, better localise transit nodes, which are legitimate (normal) nodes manipulated by malicious nodes. In more detail, the TAP scores assist in estimating the invalid

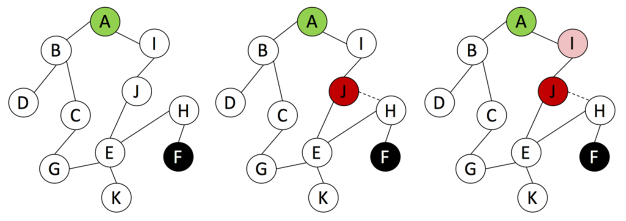
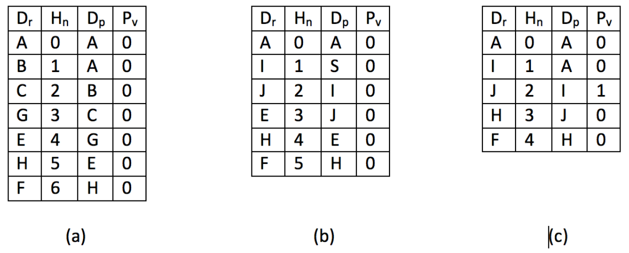


Fig. 5. Autonomous System (AS) graph showing starting node A in green and destination node F in black for (a) normal case and (b) malicious case showing malicious node J, and (c) malicious case showing J and malicious transit node I.



(a)

D_r	H_n	D_p	P_v
A	0	A	0
B	1	A	0
C	2	B	0
G	3	C	0
E	4	G	0
H	5	E	0
F	6	H	0

(b)

D_r	H_n	D_p	P_v
A	0	A	0
I	1	S	0
J	2	I	0
E	3	J	0
H	4	E	0
F	5	H	0

(c)

D_r	H_n	D_p	P_v
A	0	A	0
I	1	A	0
J	2	I	1
H	3	J	0
F	4	H	0

Fig. 6. Assessing the reliability of the AS route by the Transitional Array Penalty metric for AS graphs shown in Fig. 5, where A is the originating node and F is the target node.

paths in the network, where these paths are highly likely to have malicious transit nodes. The use of TAP in partitioning of the network into multiple subgraphs containing invalid paths improves detection speed.

Each time a malicious node appears in an invalid path, the corresponding P_v variable is incremented, as shown in Figure 6. The accuracy of the detected malicious nodes is benchmarked against the criteria in [33]. Hence, the AS paths containing any of the penalised nodes (e.g. node J) would be considered illegitimate and are quarantined for further investigation. This includes specifying the malicious nodes along with the invalid path, and furthermore transit nodes which are normal nodes with suspicious activity, as in node I in Figure 5(c). The steps of applying the TAP metric are described in Algorithm 1.

C. Ontological Graph Identification

1) PageRank algorithm

A PageRank algorithm is utilised to give higher weights to malicious nodes according to the initialised weights, as discussed in Section III.B. The PageRank algorithm [16] is a transformational technique developed by Google, and was an underpinning technology in the development of their search engine. A weighted ranking function is employed by the algorithm for estimating the number of hyperlinks pointing towards a connected nodes and their interrelation with other neighbour nodes. In order to fairly implement the PageRank algorithm, the weights for AS nodes are set to equal values using an uniform probability distribution.

Formally, it can be defined as a directed graph $G(V, E)$ derived from BGP AS-path , where the number of nodes or vertices V are the AS and E denotes the directed edges

Algorithm 1: Transitional Array Penalty

```
Input: Graph ($G$), BGP routing tables
1  Initialisation: read routing updated table;
2  if flag = true then /* flag raised for received updating message */
3      for node $i \in G$ do /* verify authenticity of advertised message */
4          while node $j \neq n$ /* where node $n$ is destination node; */ do
5              if $j$ is not connected to $j-1$ then
6                  $j-1$ is a malicious node;
7                  $R$ is an invalid path;
8                  $p \leftarrow p+1$ /* increase penalty */
9              end
10             $j = j-1$
11         end
12     end
13 end
14 return $p$;
Output: Nodes penalty vector values $p$
```

between two connected AS [16], [34]. Let $P_t(i)$ be the score of a node $i \in V$ for an iteration $t, (j,i) \in E$ which describes a directed edge from a node j to a node i for a total number of nodes n. O_j is denoted as the number of outgoing links in a node j. I_j is denoted as the number of incoming links in a node j. $W(i)$ is the initial weight of the node $i \in V$.

The weight $W(k)$ for a certain node k is estimated based on its penalty-ranking score, as outlined in Section III.B. This relies on known malicious activities from threat sharing platforms [33]). This is estimated using equation (1).

$$W(k) = \begin{cases} P_v(i), & \text{if node } i \text{ is malicious according to [33]} \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

The damping factor d in the PageRank algorithm simulates a random visit to a webpage (set as $d = 0.85$), i.e. the probability at any i to continue to j. It can be seen as a balancing factor between the previously estimated score and the weight of a node, i.e. the penalty-ranking of the AS. Its computation efficiency [13], [35] is optimised as follows,

$$P_t(i) = (1-d)\sum_{k=1}^{n} W(k) + d \sum_{(j,i)\in E} \frac{P_{t-1}(j)}{O_j}, \tag{2}$$

and the iteration stops when $|P_t - P_{t-1}| < 10^{-12}$.

2) Malicious transit nodes identification

Evaluating the quality and quantity of links for which a node is a destination to determine a relative score and importance would be useful in determining nodes significance in a network topology. The algorithm ranks the significance of AS nodes, such that an AS is referred to by other nodes, if other nodes are connected to it. A frequently-referred AS node can be considered popular, as would be an AS referred to by a few popular ASes. An AS linked by only one popular node should be considered as popular too, but less so. The mechanisms used allow this.

We further assess the relevance of this scenario in localising malicious transit AS nodes, as such AS are well connected to malicious nodes. Based on the assumption that a malicious node does not have a physical link with its predecessor, while a transit node always has a direct link with its neighbours. By dividing the AS nodes into multiple subgraphs and giving different weights for each individual node, the number of AS on an illegitimate path is computed and the nodes weighted rank is estimated recursively until all scores become stable. For each stage, the observed score of a node is evenly distributed throughout its outgoing edges. One modification made to the original PageRank algorithm approach when applied to this work is that the weights of the nodes are initialised dynamically based on the TAP score. Finally, the new score for each node is given by summing the scores of all incoming edges. This approach is termed as neighbourhood-ranking since the localised malicious transit nodes, unlike the malicious ones, tend to have a good connectivity with other neighbouring nodes within a subgraph.

An example of the malicious transit node identification method is presented in Figure 7. This figure illustrates how an appropriate initialisation of the algorithm can influence the node weights from the first iteration. In the original PageRank approach, each node starts with a unit score as no prior knowledge is assumed. The initial weight score is distributed through the outgoing links to the neighbouring nodes within each iteration step. Some nodes, as C, can have a relatively higher score than other nodes which are less connected. However, without a proper weight initialisation, malware transit nodes, e.g. I and F, can have a roughly similar score – although less connected – and hence go undetected. Therefore, the output of the process could be important in propagating higher values from malicious AS to transit nodes.

After the malicious nodes are localised and given high scores, the penalty ranking is transmitted to all connected neighbouring nodes. Other nodes which play a central connectivity role in the network topology are expected to receive high scores as well, c.f. node J in Figure 7(b). However, upstream nodes, as J, in the network topology can be excluded from being malicious transit if they are also well connected to other nodes. Once the malicious node is disconnected from the incoming links, node Js score drops to 2.16, which is well below the score of I. Identifying and disconnecting malicious nodes is not straightforward and the process requires time.

Therefore, identifying malicious transit nodes early can also assist in taking countermeasures against malicious activities originating from normal nodes. A way to differentiate between a malicious and normal activity performed by a specific AS node is to recall the connectivity property mentioned earlier. The score of the nodes can be normalised by the number of connected neighbours. The higher connectivity is, the lower node score, and vice versa. Applying this concept to Fig. 7-b, the normal node J would score 0.72 for its four neighbours, less than the score of 1.3 for the malicious transit node I with two neighbours. Algorithm 2, abbreviated as Neighbour-Rank summarises the steps of the AS OGI technique.

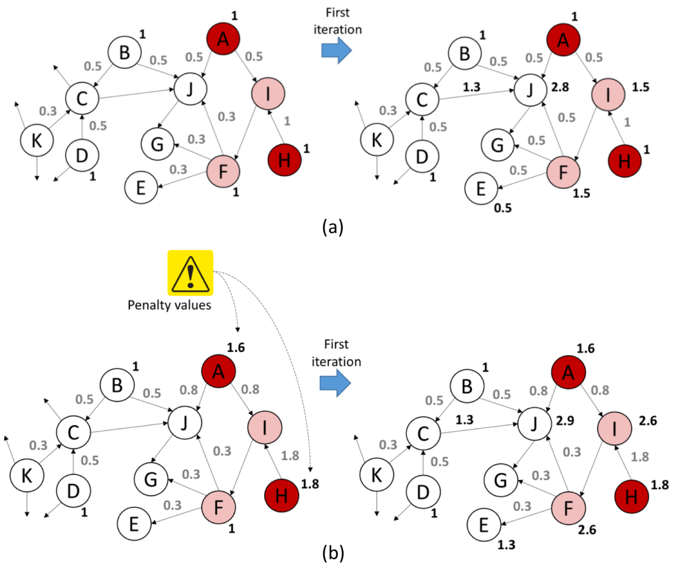


Fig. 7. Illustration of AS Ontological Graph Identification based on neighbourhood ranking before and after first iteration for (a) initialised with unit weights and (b) with weighted malicious AS using path verification method (malicious AS A and H had a higher initialised weighted). The red, pink and white colours indicate malicious, transit and normal nodes, respectively.

Algorithm 2: AS Neighbour-Ranking based on Ontological Graph Identification

```
Input: $G(k.O)$, $G$: graph, $k$: nodes, $O$: neighbouring
       links (outgoing links) from node $j$ to node $i$
1  while $t > 0$ do /* t is the number of
     iteration per subgraphs $g$ */
2      foreach $g \in G$ do
3          forall nodes $k$ in subgraph $g$ do
4              Initialisation based on neighbourhood ranking;
5              $P[k] \leftarrow \frac{P_{t-1}(j)}{O}$ /* where $P_{t-1}(j)$ is
               score at previous node $j$ */
               sorted neighbourhood;
6              $P_t \leftarrow (d \times P + (1-d)W)$ /* where $d$
               and $W$ denote damping factor
               and weights from the AS path
               verification step (section
               3.2) */
7          end
8          Normalise each $k$ in $P_t$ to get $P_t'$;
9          $P_t'(k) = P(k) - P_{tn}$ /* where $P_{tn}$ is the
           average of all nodes $k$ with equal
           number of neighbours $n$ */
10         Estimate Neighbour-Rank vector;
11         $N_r = P_t' \geq max\{W(k)\}$ /* Highest rank
           $P_t'$ are considered to be malicious
           transit AS */
12     end
13     $t \leftarrow t - 1$
14 end
15 return $N_r$;
Output: Estimated Neighbour-Rank vector $N_r$
```

The output of Algorithm 2 is further refined for filtering out normal ASes which might be engaged with minor transit services to malicious ASes [36], relying on the number of neighbours N for a certain AS. To normalise the algorithm score, it can be assumed that most AS are normal or not transit. Hence, the average score of all AS with similar number of adjacent neighbours (N) is subtracted from the score of the current AS. This operation is performed recursively for all AS in each subgraph.

$$P_t^{'}(i) = P_t(i) - \frac{\sum_{j \in V, N(j)=N(i)} P_t(i)}{card(\{j \in V, N(j) = N(i)\})} \tag{3}$$

Finally, the AS achieving the condition $P_t^{'} \geq max\{W(k)\} \forall k \in V$, would be selected as potential transit node(s), i.e. having a tendency of becoming malicious.

The OGI algorithm can detect interception attacks that have a type-0 footprint on BGP [24]. Different attacks of control-plane incidents can actually trigger the OGI algorithm re-computation, and thus the malicious transit node characterisation. Such examples include: Distributed Denial of Service (DDoS), man-in-the-middle, and path manipulation attacks. This is due to the fact that the model can define malicious transit nodes based on estimating TAP score and their likelihood probabilities according to AS Neighbour-Ranking.

3) *Transit nodes classification*

We have set the ground truth of the transit nodes from the RIPE RIS dataset [6] according to the following criteria; transit nodes are located down-stream according to the malicious nodes, data collection is within a few hop counts from a malicious node (e.g. up to 2 hops were empirically set), and data collection is well connected (i.e. has at least 3 incoming neighbour links). The different parameters are empirically specified as in Section 4.

Evaluating the transit detection results against the specified ground truth is more reasonable. Formally, given an observation O with a local bi-hop topology $G_0^{(2)}$, we want to determine $\forall i \in T_0$ whether $I_j = I_0$, where $T_0 \in V$ is the set of test nodes. $T_0 = N_0^{(2)}$indicates that within every second hop, the algorithm is evaluated; referring to a connectivity with a far-neighbour transit node with a malicious AS node. Similarly, only first level nodes are dealt with in the algorithm for $T_0 = N_0^{(1)}$, which fits the process of an immediate neighbour transit node connectivity. Both choices of T_0 will be evaluated in the Section 4. This decision problem is represented as a binary-classification problem, i.e. assign a label C_i(the ground-truth) to each node:

$$C_i = \begin{cases} 1 & I_i = I_0 \\ 0 & Otherwise \end{cases} \tag{4}$$

The algorithm should take 1 and 2-hop topology as input and output predicted labels $\hat{C}_i \in 1, 0$.

The classification results represented as a confusion matrix are shown in Table II. Using the notation in Table II, we have multiple standard ways to evaluate the quality of predicted labels $\hat{C}_i$, such as accuracy (Ac), true positive rate (TPR) and false positive rate (FPR):

$$Ac = (x_1 + x_4)/(x_1 + x_2 + x_3 + 4) \tag{5}$$

TABLE II
CONFUSION MATRIX

No. of Samples	$\hat{C} = 1$	$\hat{C} = 0$
C = 1	x_1	x_2
C = 0	x_3	x_4

$$TPR = x_1/(x_1 + x_2) \tag{6}$$

$$FPR = x_3/(x_3 + x_4) \tag{7}$$

In Section 4, we will further analyse the shortcomings of the above evaluation methods and propose to use Receiver Operating Characteristics (ROC) and Area Under the ROC Curve (AUC) to evaluate key steps of the algorithm in depth.

IV. Results and Discussion

A. Dataset and experimental environment

We used the Routing Information Services (RIS) of RIPE dataset [6] to extract BGP updates messages related to the destination prefix. The information contains raw data packets such as origin AS, adjacent AS and prefix length. A total of 4256k distinct ASes were observed, for which the TAP function was computed. We also collected data from another resource for means of robustness assessment. BGP update data from routing table snapshots were also extracted from the University of Oregon RouteViews Server [37] that peers with 57 BGP routers in 46 different ASes to validate the proposed method. The RouteViews routing tables and updates were collected for the same duration as the RIPE RIS dataset.

The ground truth of malicious nodes were chosen according to the Computer Incident Response Centre Luxembourg (CIRCL) BGP Ranking software [33]. The software assists in validating security events by scoring blacklists of malicious source IP addresses[1]. The BGP Ranking software has been reported to show an average false positive rate of 2.5% per month when tested over a 3 year period [38], and was recently used in related studies for detecting malicious network blocks that have been sub-allocated [39], and in identifying malicious ASs using the routing behavior of ASs exclusively [38].

Although other sources of blacklists exist, such as CleanMX, SpamHaus (Edrop and ROKSO) and BL-A, the BGP Ranking used in this study computes the node scores of ASs based on labelled data captured from a variety of public IP addresses and domain names, which are based on external blacklists. This racking mechanism draws upon sources such as dshield, blocklist.de, spamhaus, bambenek, shadowserver and arbor atlas. This facilitates BGP Ranking to be used for a more comprehensive determination of AS node scores. Additionally, BGP ranking can be used to set a malicious AS threshold by calculating the average score of the top ASs, where any AS with a score that is equal to or higher than the threshold is considered a malicious AS.

[1]http://bgpranking.circl.lu/

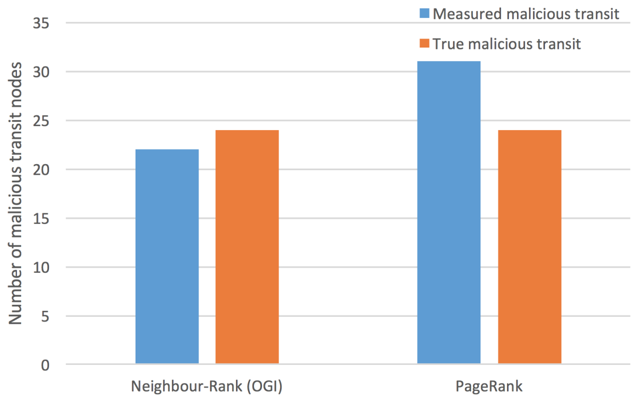


Fig. 8. Comparison between the original PageRank algorithm and the proposed Neighbour-Rank algorithm of the AS Ontological Graph Identification method for localising malicious transit nodes.

B. Performance assessment

In order to fairly assess our OGI method, the original PageRank algorithm is compared with the proposed Neighbour-Rank algorithm using the RIPE RIS dataset. The results reveal that the PageRank algorithm has more false positives compared to the OGI method based on Neighbour-Rank algorithm; namely, the OGI method has fewer false negatives and less false positives, as shown in Figure 8. The malicious transit nodes I_m are the maximum scoring normal nodes (i_k) in a certain subgraph g which their associated scores are greater than the highest-scoring malicious node in the corresponding g, as previously outlined. This gives an indication that the proposed algorithm operates as a feature reduction technique that highlights the most prominent nodes which exhibited highest scores for further investigation. This is crucial when the node numbers are large, as there are likely to be large numbers of detected malicious transit nodes.

Figure 8 is a representation of an average number of 2917 of nodes having a neighbourhood (i.e. number of hops) of $k = 2$ in the K-means algorithm. Thus, we would expect the malicious transit nodes to be much higher when $k > 3$, as depicted in Figure 9. Hence, a reduced representation of the possible nodes exhibiting transit malicious activities assists in reducing the complexity and investigation time.

Also, as shown in Figure 10, the number of iterations grows higher for higher damping factor (d) values. However, the d value is co-related to the size of the subgraph, as low d values tend to damp the algorithm flow, and the iterations will quickly converge, and vice versa. Thus, this might not hold for large AS subgraphs, where little damping (i.e. high d value) would be required for covering as much nodes as possible before iterations slowly converging. In this paper, a d value of 0.85 was applied as it is widely used in the literature [34]. It should be noted that a single iteration of the Neighbour-Rank algorithm running algorithm running on an Intel Core i7-6700 CPU with 3.4GHz base clocking speed, 8 GB of RAM and L3 cache size of 8192 MB running Microsoft Windows 10, 64 bit requires 1.145 seconds.

Additionally, experiments were performed using the proposed method via n number of hop-counts (H), for the purpose

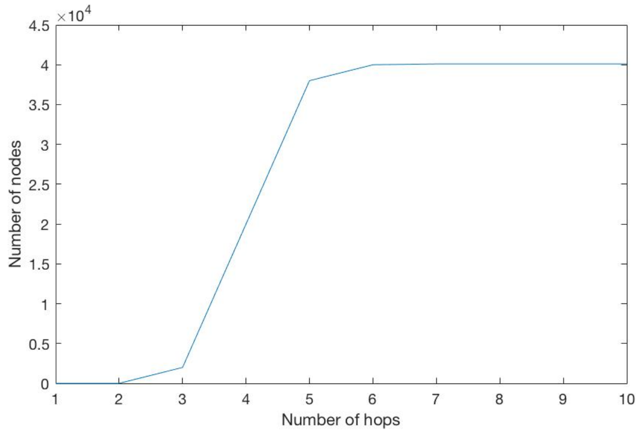


Fig. 9. Average number of nodes in a sub-graph with different neighbours based on number of hops (k).

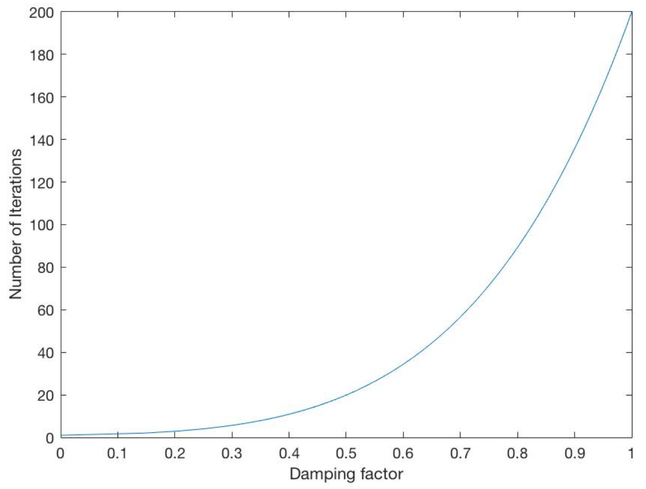


Fig. 10. Effect of different damping factor (d) values on the number of Neighbour-Rank algorithm iterations.

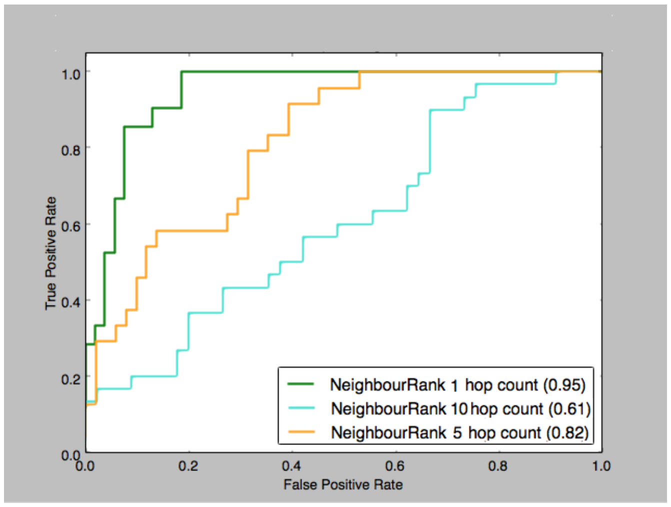


Fig. 11. Receiver operating characteristic curve (ROC) for proposed Ontological Graph Identification technique with varying number of hop-counts using the Neighbour-Rank algorithm.

of investigating the effect of the subgraph size on the algorithms performance (see Figure 11). As the number of links per AS vary from one subgraph to another, the classification performance was assessed using different incoming links while fixing the number of hop-counts. For completeness, results were benchmarked with the original PageRank algorithm. Table III shows the confusion matrix for the Neighbour-Rank algorithm with different number of hop-counts while keeping the number of incoming links fixed. Moreover, the performance while varying the number of incoming links for $n = 1$ and 2 is shown in Table IV and V, respectively. Results of the other dataset, which uses RouteViews routing tables and updates of 3 months, also exhibited similar performance (see Table VI), suggesting that improved performance via our OGI method is robust and independent of a particular use of a certain dataset.

The proposed method does not assume a uniform or random distribution of the AS as with the original PageRank. Instead, the original PageRank converges to the same values regardless of initial probabilities. Therefore, giving the malicious nodes a higher weighted value via the TAP metric, better reveals suspicious activity of normal nodes performing malicious transit activities, as outlined in section III.C. This assists in starting with realistic weights by giving a better separation between normal nodes performing minor malicious transit activities and the ones with major suspicious activities. Moreover, the algorithm improves the scalability by working on independent subsets of the AS data, i.e. subgraphs, facilitating for faster convergence of iterations.

The proposed method can identify IP hijacking and their malicious transit nodes better than the PageRank algorithm and other related methods. As it defines the illegitimate paths with the malicious nodes before adding the estimated TAP metric to the initialised weights of the Neighbour-Rank algorithm. The proposed OGI method could be deployed in a real-time and online hijack detection system. The method is designed to estimate the likelihood probabilities of nodes and their neighbours that depend on graph models of prior (i.e., root nodes) and likelihood (i.e., their edges), which can be estimated in real-time. It can be trained and tested in an offline mode if the server does not have high computational resources, and then it will be deployed in real-time to detect IP hijacking through the examination of nodes and their contiguous ones. If the method is trained and validated using a server with GPU can be built in real-time and deployed in production networks.

In comparison with other relevant recent works, our method was also compared with four control plane and data plane methods; PHAS [9], a Route Purging and Promotion (RPP) scheme [12], an Argus Agile System (AAS) [11] and the LOCK method [29]. Comparison was made in terms of accuracy and False Alarm Rate (FAR), with respect to the same settings of the proposed technique. The PHAS and AAS methods achieved about 80% accuracy and 18.82% FAR, the RPP accomplished a 57% accuracy and roundly 32% FAR, and the LOCK method obtained almost 88.4% accuracy

TABLE III
MALICIOUS TRANSIT AS CLASSIFICATION CONFUSION MATRICES FOR ORIGINAL PAGERANK ALGORITHM (PR) AND CORRESPONDING NEIGHBOUR-RANK ALGORITHM (NR) WITH NUMBER OF HOP-COUNTS ($n = 1$ AND 2) AND FIXED INCOMING LINKS ($k = 3$)

		Classification					
		NR($n = 1$)		NR($n = 2$)		PR	
True class	No. of Samples	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$
	$C = 1$	0.8251	0.1749	0.9512	0.0488	0.8200	0.1800
	$C = 0$	0.1489	0.8511	0.0253	0.9747	0.1588	0.8412

TABLE IV
MALICIOUS TRANSIT AS CLASSIFICATION CONFUSION MATRICES FOR ORIGINAL PAGERANK ALGORITHM (PR) AND CORRESPONDING NEIGHBOUR-RANK ALGORITHM (NR) WITH AS NUMBER OF INCOMING LINKS (INL) HAVING ($k = 3$, 4 AND 5) WHILE HAVING FIXED NUMBER OF HOP-COUNT ($n = 2$)

		Classification							
		INL($k = 3$)		INL($k = 4$)		INL($k = 5$)		PR	
True class	No. of Samples	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$
	$C = 1$	0.9512	0.0488	0.9622	0.0378	0.9501	0.0499	0.8200	0.1800
	$C = 0$	0.0253	0.9747	0.0202	0.9798	0.0391	0.9609	0.1588	0.8412

TABLE V
MALICIOUS TRANSIT AS CLASSIFICATION CONFUSION MATRICES FOR ORIGINAL PAGERANK ALGORITHM (PR) AND CORRESPONDING NEIGHBOUR-RANK ALGORITHM (NR) WITH AS NUMBER OF INCOMING LINKS (INL) HAVING ($k = 3$, 4 AND 5) WHILE HAVING FIXED NUMBER OF HOP-COUNT ($n = 1$)

		Classification							
		INL($k = 3$)		INL($k = 4$)		INL($k = 5$)		PR	
True class	No. of Samples	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$
	$C = 1$	0.9303	0.0697	0.9415	0.0585	0.9270	0.0730	0.8200	0.1800
	$C = 0$	0.0485	0.9515	0.0422	0.9578	0.0598	0.9402	0.1588	0.8412

TABLE VI
MALICIOUS TRANSIT AS CLASSIFICATION CONFUSION MATRICES ON ROUTEVIEWS DATASET FOR ORIGINAL PAGERANK ALGORITHM (PR) AND CORRESPONDING NEIGHBOUR-RANK ALGORITHM (NR) WITH NUMBER OF HOP-COUNTS ($n = 1$ AND 2) AND FIXED INCOMING LINKS ($k = 3$)

		Classification							
		INL($k = 3$)		INL($k = 4$)		INL($k = 5$)		PR	
True class	No. of Samples	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$	$\hat{C} = 1$	$\hat{C} = 0$
	$C = 1$	0.9498	0.0502	0.9610	0.0390	0.9493	0.0507	0.8200	0.1800
	$C = 0$	0.0267	0.9733	0.0213	0.9787	0.0399	0.9601	0.1588	0.8412

and 10.82% FAR. Finally, our method achieved an average of 94% accuracy and 4.2% FAR when $k = 3$. From the results, it is observed that the proposed method outperforms the other methods due to its potential design that can efficiently recognise malicious IP nodes.

V. CONCLUSION

Hijacking attacks can contribute to major network disruption, and in some cases the entire network inside the victims domain, may became unreachable. Hence, sophisticated hijacking attacks can be launched against the victims network such as the previously discussed interception or man in the middle attacks. Therefore, a novel ontological graph approach based on initialising the PageRank algorithm with different weights according to AS nodes malicious connectivity and activity has been proposed. Firstly, the illegitimate paths containing malicious and transit nodes are defined by the TAP metric. Secondly the extracted nodes are divided into several subgraphs with a certain connectivity neighbourhood (i.e. number of hops), facilitating for faster iteration convergence. Thirdly, the Neighbour-Rank algorithm is initialised with weights estimated according to the AS path verification derived from the TAP metric. Finally, the transit nodes which exhibit malicious activity are located and refined from each subgraph, and hence isolated before becoming malicious.

The future work of this study will examine different scenarios using different damping factor values on the reported results to adaptively select the optimal values in real network systems. The size of the subgraphs implemented in the Neighbour-Rank algorithm can assist in improving the detection performance for very large data. Thus, there is a need to balance the trade-off between increasing the connectivity of the subgraphs (i.e. having larger number of hops) and reaching to AS score stability in feasible processing time. However, this would come at the expense of increasing computational processing time, which would call for a parallel or distributed solution.

ACKNOWLEDGMENT

We would like to thank the reviewers for the carefully reading and constructive comments which substantially has led to improving the quality of the paper. This work was supported by the Australian Postgraduate Award (APA), and also in part by the National Science Foundation CREST under Grant HRD-1736209.

REFERENCES

[1] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," IBM, Tech. Rep., 2005.

[2] K. Butler, T. R. Farley, P. McDaniel, and J. Rexford, "A survey of BGP security issues and solutions," *Proceedings of the IEEE*, vol. 98, no. 1, pp. 100–122, 2010.

[3] H. Ballani, P. Francis, and X. Zhang, "A study of prefix hijacking and interception in the internet," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 265–276, 2007.

[4] S. Nichols, "AWS DNS network hijack turns MyEtherWallet into ThievesEtherWallet," http://www.theregister.co.uk/2018/04/24/myetherwallet_dns_hijack/, 2018, [Access Date: 20 February 2019].

[5] W. Greenberg, L. H. AndyNewman, E. Dreyfuss, B. Barrett, D. Gold, and I. Lapowsky, "Hacker redirects traffic from 19 internet providers to steal bitcoins," http://www.wired.com/2014/08/isp-bitcoin-theft/, 2014, [Access Date: 20 February 2019].

[6] RIPE, NCC, "Youtube hijacking a RIPE NCC RIS case study," urlhttp://www.ripe.net/news/study-youtube-hijacking.html9, 2008, [Access Date: 20 February 2019].

[7] P. Sermpezis, V. Kotronis, P. Gigis, X. Dimitropoulos, D. Cicalese, A. King, and A. Dainotti, "ARTEMIS: Neutralizing BGP hijacking within a minute," *IEEE/ACM Transactions on Networking (TON)*, vol. 26, no. 6, pp. 2471–2486, 2018.

[8] X. Hu and Z. M. Mao, "Accurate real-time identification of ip prefix hijacking," in *2007 IEEE Symposium on Security and Privacy (SP'07)*. IEEE, 2007, Conference Proceedings, pp. 3–17.

[9] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang, "Phas: A prefix hijack alert system," in *USENIX Security symposium*, vol. 1, 2006, Conference Proceedings, p. 3.

[10] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis, "A light-weight distributed scheme for detecting ip prefix hijacks in real-time," in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 277–288.

[11] X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu, "Detecting prefix hijackings in the internet with argus," in *Proceedings of the 2012 Internet Measurement Conference*. ACM, 2012, pp. 15–28.

[12] Z. Zhang, Y. Zhang, Y. C. Hu, and Z. M. Mao, "Practical defenses against BGP prefix hijacking," in *Proceedings of the 2007 ACM CoNEXT conference*. ACM, 2007, p. 3.

[13] K. Butler, P. McDaniel, and W. Aiello, "Optimizing BGP security by exploiting path stability," in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 298–310.

[14] Y.-C. Hu, A. Perrig, and M. Sirbu, "SPV: Secure path vector routing for securing BGP," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 179–192.

[15] J. Karlin, S. Forrest, and J. Rexford, "Pretty good BGP: Protecting BGP by cautiously selecting routes," Technical report, University of New Mexico, Tech. Rep., 2005.

[16] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[17] G. Chang, M. Arianezhad, and L. Trajković, "Using resource public key infrastructure for secure border gateway protocol," in *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE, 2016, pp. 1–6.

[18] NIST, "Global prefix/origin validation using rpki," urlhttps://rpki-monitor.antd.nist.gov, 2019, [Access Date: 20 February 2019].

[19] R. Bush and R. Austein, "The resource public key infrastructure (rpki) to router protocol," Internet Engineering Task Force (IETF), Tech. Rep., 2013.

[20] P.-A. Vervier, O. Thonnard, and M. Dacier, "Mind your blocks: On the stealthiness of malicious BGP hijacks." in *NDSS*, 2015.

[21] C. McArthur and M. Guirguis, "Stealthy ip prefix hijacking: don't bite off more than you can chew," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE, 2009, pp. 1–6.

[22] A. Pilosov and T. Kapela, "Stealing the internet: An internet-scale man in the middle attack," *NANOG-44, Los Angeles, October*, pp. 12–15, 2008.

[23] J. Qiu, L. Gao, S. Ranjan, and A. Nucci, "Detecting bogus BGP route information: Going beyond prefix hijacking," in *2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007*. IEEE, 2007, pp. 381–390.

[24] G. Chaviaras, P. Gigis, P. Sermpezis, and X. Dimitropoulos, "Artemis: Real-time detection and automatic mitigation for BGP prefix hijacking," in *Proceedings of the 2016 ACM SIGCOMM Conference*. ACM, 2016, pp. 625–626.

[25] M. Tahara, N. Tateishi, T. Oimatsu, and S. Majima, "A method to detect prefix hijacking by using ping tests," in *Asia-Pacific Network Operations and Management Symposium*. Springer, 2008, pp. 390–398.

[26] J. W. Mickens, J. R. Douceur, W. J. Bolosky, and B. D. Noble, "Strobelight: Lightweight availability mapping and anomaly detection," in *Proceedings of the 2009 Conference on USENIX Annual Technical Conference*, ser. USENIX'09. USENIX Association, 2009, pp. 5–5.

[27] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Topology-based detection of anomalous BGP messages," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 17–35.

[28] S.-C. Hong, H.-T. Ju, and J. W. Hong, "Ip prefix hijacking detection using idle scan," in *Asia-Pacific Network Operations and Management Symposium*. Springer, 2009, pp. 395–404.

[29] T. Qiu, L. Ji, D. Pei, J. Wang, J. J. Xu, and H. Ballani, "Locating prefix hijackers using lock." in *USENIX Security Symposium*, 2009, pp. 135–150.

[30] B. N. S. Inc., "Routing anomalies," http://www.bgpstream.com, 2016, [Access Date: 20 February 2019].

[31] J. Schutrup and B. ter Borch, "BGP hijack alert system," *University of Amsterdam, the Netherlands*, 2016.

[32] T. He and K. C. Chan, "MISAGA: An algorithm for mining interesting subgraphs in attributed graphs," *IEEE transactions on cybernetics*, vol. 48, no. 5, pp. 1369–1382, 2018.

[33] A. Dulaunoy, "BGP ranking project," http://www.terena.org/activities/tf-csirt/meeting32/dulaunoy-bgpranking.pdf, 2011, [Access Date: 27 May 2019].

[34] J. François, S. Wang, T. Engel *et al.*, "Bottrack: tracking botnets using netflow and pagerank," in *International Conference on Research in Networking*. Springer, 2011, pp. 1–14.

[35] T. Haveliwala, "Efficient computation of pagerank," Stanford, Tech. Rep., 1999.

[36] C. Wagner, J. François, R. State, A. Dulaunoy, T. Engel, and G. Massen, "Asmatra: Ranking ASs providing transit service to malware hosters," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, 2013, pp. 260–268.

[37] D. Meyer, "University of oregon route views archive project," 2001.

[38] M. Konte, R. Perdisci, and N. Feamster, "Aswatch: An as reputation system to expose bulletproof hosting ASes," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 625–638, 2015.

[39] S. Alrwais, X. Liao, X. Mi, P. Wang, X. Wang, F. Qian, R. Beyah, and D. McCoy, "Under the shadow of sunshine: Understanding and detecting bulletproof hosting on legitimate service provider networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 805–823.