The Creation of Network Intrusion Fingerprints by Graph Homomorphism

CHUCK EASTTOM, MO, ADDA,
School of Computing
University of Portsmouth
Portsmouth, Hampshire, UK

Abstract: - Attack attribution in cyber-attacks tends to be a qualitative exercise with a substantial room for error. Graph theory is already a proven tool for modeling any connected system. Utilizing graph theory can provide a quantitative, mathematically rigorous methodology for attack attribution. By identifying homomorphic subgraphs as points of comparison, one can create a fingerprint of an attack. That would allow one to match that fingerprint to new attacks and determine if the same threat actor conducted the attack. This current study provides a mathematical method to create network intrusion fingerprints by applying graph theory homomorphisms. This provides a rigorous method for attack attribution. A case study is used to test this methodology and determine its efficacy in identifying attacks perpetrated by the same threat actor and/or using the same threat vector.

Key-Words: - Graph theory, Fingerprinting, Attack Attribution

Received: June 12, 2020. Revised: July 31, 2020. Re-revised: August 4, 2020. Accepted: August 5, 2020. Published: August 6, 2020

1 Introduction

Network intrusions come in many different varieties. There are Denial of Service (DoS) attacks, malware infections, remote access, and others. One type of intrusion is the remote attacker who attempts to obtain confidential data. Unlike Denial of Service or botnet attacks, the goal is the exfiltration of data, rather than the damage or the disruption of the target system [1]. Spyware is delivered to the target machine and the data is ex-filtrated. Spyware can be used by anyone but is certainly a tool utilized in cyber espionage [2] [3].

One thing that all cyber-attacks have in common is the difficulty of attribution. Attribution in cyber-attacks is always problematic [4] [5]. It is difficult to determine who actually the perpetrator was. Unlike physical crimes, one cannot use fingerprints, fibre evidence, footprints, security camera footage, or similar sources of evidence. The problem is exacerbated in nation state scenarios [6] [7]. Nation state attacks tend to be sophisticated and the perpetrators adept at covering their identity.

What is needed is a mechanism for developing a fingerprint for spyware attacks. A methodology for analysing a particular spyware attack and comparing it to other attacks. The level of match between the attacks can point to a common attacker being responsible.

Graph theory provides a tool that is appropriate for developing a fingerprint of spyware activity. It has been utilized frequently to model network behaviour [8] [9], economics [10], chemical engineering [11] and a host of other phenomena. Graph theory has already been utilized to model digital attacks in by matching indicators of compromise on a system wide basis [12]. Algebraic graph theory has also been utilized to model network intrusions and better understand the attack [13][14].

Cyber attack attribution is not as clear as physical forensics. In the physical forensics arena one can rely on fingerprints, DNA evidence, hair and fiber evidence, and a wide variety of other physical markers. Identifying Indicators of Compromise in cyber attacks attempts to provide some of this level of rigor but falls short. What is needed in the domain of cyber investigations is something that is the equivalent in rigor with a fingerprint. This would move cyber attack attribution from a subjective, qualitative process to an objective, quantitative science.

2 Problem Formulation

There is a need for improved cyber-attack attribution. Attribution of attacks is often done in an informal manner that is purely qualitative [15]. This is even more challenging in cyber warfare scenarios due to threat actors actively obfuscating the origin of the attack [16]. What is needed is a rigorous

mathematical methodology to analyze and model attacks, then to utilize that model as a comparison for attribution. The ability to mathematically assign a probability that the same threat actor as a known attack executed a given attack that is currently being analyzed, would enhance threat attribution.

In order to understand the current problem, it is useful to examine current attack modeling and attribution tools. One prominent current model is the MITRE Corporation ATT&CK model. This model's name is an acronym for Adversarial Tactics, Techniques and Common Knowledge (ATT&CK). The concept of this particular modeling technique was created by the MITRE Corporation. The ATT&CK model documents the specific tactics as well as the particular techniques that an adversary uses in an attack [17]. This model is widely used and can be quite effective in gaining a general understanding of an attack. However, the ATT&CK method has some limitations. ATT&CK is formally designed to understand the attack vectors, actors, and methodologies. The model is not designed to fingerprint a given attack for comparison to other attacks. The ATT&CK methodology helps the analyst to understand how a specific attack was conducted but is only moderately useful for attack attribution. It does not adequately integrate the attack path, or the target network into its model. While useful, this methodology actually highlights a gap in the literature, that the current study proposes to fill.

Another popular approach to analyzing cyber-attacks is the Common Vulnerability Scoring System(CVSS). CVSS is a qualitative mechanism to score information-security vulnerabilities. The CVSS' numerical score reflects the severity of exploits using descriptions such as low, medium, high, and critical [18]. The scoring is designed to aid in vulnerability management process. There are three groups of metrics: base, temporal, and environmental. The base group describes the basic characteristics of the vulnerability that are not determined by time (temporal) or environment. The metrics in this group are Attack Vector, Attack Complexity, Privileges Required, User Interaction, Scope, Confidentiality Impact, Integrity Impact, and Availability Impact. The temporal group of metrics describes the time required for the vulnerability to be exploited. The environmental metrics describe how the attack is realized, such as over a remote connection, local network, or direct physical access [19].

While CVSS is an effective tool for analyzing vulnerabilities, it is less useful in threat attribution. It is certainly possible to relate a CVSS score to an actual intrusion. Such an analysis would provide some limited insight into the attack vector the threat actor used. However, the CVSS process was not designed for threat attribution and is not effective in that application.

In addition to general attack modeling methods, there are methods for documenting Indicators of Compromise (IoC). Any attack will be detected via one or more IoC's. Thus, documenting and communicating such IoC's in a formal manner is of great importance. Three of the most widely used IoC methods are STIX, TAXII, and CybOx [20] [21] [22]. These methods are effective in documenting and communicating specific, individual IoC's. However, by themselves, these methods do not provide a robust method of attack attribution.

Another threat modeling technique is Visual, Agile, Simple, Threat Modeling (VAST). This threat modeling method is used to enumerate and prioritize threats. VAST is focused on software threat modeling, particularly in Agile programming [23]. VAST works with two concurrent types of models. The application threat model and the operational threat model. Threats are reviewed from both perspectives. Process flow diagrams are used to examine application threats. Data flow diagrams are used to examine operational threats [24]. While useful in analyzing software vulnerabilities, this methodology is not applicable to general cyber-attacks, nor to attack attribution.

What a review of the current literature demonstrates, is that there is a gap in the literature. There are numerous methods for evaluating vulnerabilities, analyzing cyber-attacks, and describing indicators of compromise. What is currently lacking is a reliable attack attribution methodology. Such a methodology must be reliable, and thus should be based on well-established, rigorous mathematics.

3 Problem Solution

As was previously discussed in this paper, graph theory provides a robust modeling tool that has been applied to a wide range of problem domains. The solution proposed in this paper is to utilize graph auto-morphisms in order to both fully understand the parameters of a given attack, and to compare the attack against other attacks to determine attribution.

Our approach to this problem starts with a complete equivalence of two graphs, an isomorphism. A simple definition of isomorphism between two

graphs is if both graphs have the same number of vertices, the same number of edges, and identical degree matrices. While these conditions are necessary for isomorphism, they are not sufficient. For instance, consider two graphs Va and Vb. The added requirement is that a vertex function f from Va and Vb preserves both adjacency and non-adjacency values. Essentially this requires that the two graphs have not only identical vertex sets, edge sets, and degree matrices, but that their structure is retained. Put more formally there must exist a structure-preserving vertex bijection f: Va -> Vb in order for these two graphs to be isomorphic [25]. Even two attacks executed by the same threat actor using the same attack vector are unlikely to be isomorphic. This is due to the target systems having different structure and data flow.

Graph homomorphisms are better suited to attack attribution. A graph homomorphism is a structure preserving mapping between two graphs. If it is a directed graph, even the origins and tails of arcs are preserved [26]. Put a bit more formally, and referring to directed graphs:

Let $G_1 = (V_1, E_1, o_1, t_1)$ and $G_2 = (V_2, E_2, o_2, t_2)$
Where V_x is the set of vertices in the graph, E_x is the set of edges, o_x is the origin of the arcs, and t_x is the tail.
$\theta_V: V_1 \rightarrow V_2$
$\theta_E: E_1 \rightarrow E_2$

Such that the origins and tails maintain their structure for all e ε E, this is a strong homomorphism.

This may sound quite similar to an isomorphism; and it is true that a strong graph homomorphism, that is also bijective, is an isomorphism. However, there are variations of graph homomorphisms.

A weak homomorphism, also called a graph egamorphism [27] has the same edge set, but not necessarily the same origin to tail relationship. Put more formally, an egamorphism is a relationship for G1 and G2 such that:
if (a,b) ε E and f(a) ≠ f(b)

Thus, one tool available with mapping a network intrusion is to determine if the attacks form an egamorphism. Of course, one would check to see if the attacks are isomorphic, but that is highly improbable.

The next area to explore, when comparing two attacks, is to consider induced subgraphs. In general, terms an induced subgraph is formed form a subset of the vertices of another graph with the edges connecting the vertices in that subset. Put more formally if $G = (V, E)$ and $S \subset V$ of G, then a graph H whose vertex set is S and which includes all of the edges that have both endpoints in S is an induced subset of G [28]. Induced subgraphs are important in examining network intrusions. If one treats the affected devices as an induced subset of the network graph, one can compare the induced subset from another attack to analyze similarities in attack vectors, targets, and other aspects of the incident.

It is also advantageous to examine the neighborhood of any vertex that is identified as being relevant to the attack in question. A neighborhood of vertex v in a graph G is a subgraph of G induced by all vertices adjacent to v [29]. It is clear that multiple induced subgraphs will be found by studying the neighborhood of vertices of interest in a given attack.

The process begins with creating a graph model of the two attacks. Then the graphs are analyzed to see if they are isomorphic or egamorphic. Either of which would be a strong indicator of identical threat actors using the same threat vector on similar target networks. Certainly, isomorphic graphs demonstrate that the exact same attack was used, with the same threat vector, on a substantially similar network topology. That could only occur with an identical attacker. Egamorphic graphs are not entirely identical as isomorphic graphs are but have enough overlap to clearly point to the same threat actor. Assuming that the induced graphs are neither isomorphic or egamorphic, the next portion of the analysis is to identify induced subgraphs that are homomorphic (even weakly homomorphic) and analyze those. Particular attention should be paid to the neighborhood induced graphs of key vertices in both attacks.

Of particular interest would be the state wherein the neighborhood induced subgraph of G is a covering graph of a neighborhood induced subgraph of H. A covering graph is a covering map from the vertex set of G to H. More formally, a covering map is a surjection and an isomorphism. It is even more relevant to comparing two attacks if the graphs are multigraphs. A covering graph of two induced sub-multigraphs is a strong indicator of identical threat actors and attack vectors.

Each induced subgraph that represents a homomorphism would be a point of match between the two attacks. This provides a method that is analogous to fingerprints. In fingerprint analysis, points of similarity are identified. The more matching points between fingerprints, the strong the identification is considered to be [30] [31].

The strength of the relationship in the induced subgraphs would be weighted, such that an isomorphism is weighted more than a weak homomorphism. A proposed weighting is shown in table 1.

Table 1 Relationship Weighting

Weight	Relationship
3	Isomorphism
2	Strong homomorphism
1	Weak homomorphism/ egamorphism

It should be obvious that the weighting is relevant to the total size of the graph. If one has two graphs, each of only four vertices, three of which form an isomorphic subgraph yielding a very strong match. However, the same three vertices from each graph forming an isomorphic subgraph, when the entirety of each graph is 100 vertices, is not a very strong relationship. Therefore, the degree of similarity is calculated by the weight divided by the total number of vertices. This is shown in equation 1.

$$\theta = \frac{w}{Tv} \quad (1)$$

The degree of matching is represented by the theta symbol θ. The w represents total weight assigned to subgraphs multiplied by 2. This is done because each original graph contains an induced subgraph that is being compared to an induced subgraph in the other complete graph. The Vt are the total number of vertices in the two graphs. This is normally described as the combined order of the two graphs. This produces a number between 0 and 1, quantifying the similarity. Given the fact that networks can be very different, one would expect relatively low values for θ. Values even above 0.25 would be considered strong matches.

3.1 A Case Study

In order to test this fingerprinting process, a comparative experiment was conducted. Two different virtual systems were breached. In both cases the same attack vector was used, by the same threat actor. However, the individual virtual environments were different. In these case studies, directed graphs are utilized. The arrow points from the origin of the communications, towards the target of communications.

In virtual environment 1, a Windows 7 virtual machine was connected to the internet and internally connected to a Windows 2012 server running SQL Server 2008. There was also a Windows 10 computer that was hardened with very few services running on it. There were two other Windows 10 virtual machines that were not scanned or breached. However, these machines were connected to the Windows 2012 server, specifically to the SQL Server database.

In virtual environment 2, a Windows 10 virtual machine was connected to the internet, and internally connected to a Windows 2019 server and an Ubuntu Linux web server running Apache. There were also three Windows 10 virtual machines that were not scanned or breached. However, these machines were connected to the Apache web server.

In both cases, the internet computer was sent a malicious payload using the Kali Linux msfvenom tool [32]. The payload was disguised as a PDF file and was sent as an attachment to an email. Upon clicking the file, it was designed to create an HTTPS reverse shell back to the listening Kali machine. The payload for the virtual environment 2 was obfuscated using architecture modifications and encryption [33].

Once a tunnel was established, the attacking machine attempted to determine other targets in the network, and to pivot to those machines. In virtual environment 1, the attack was able to successfully log on to the Windows 2012 server, but not the hardened Windows 10 computer. In virtual environment 2, the attacker attempted to connect to both the Windows server 2019 and the Ubuntu web server but was not able to breach the target system. In both virtual environments, the attempts to breach secondary machines were preceded by identical metasploit scans.

Virtual environment 1, is depicted as a graph in figure 1.

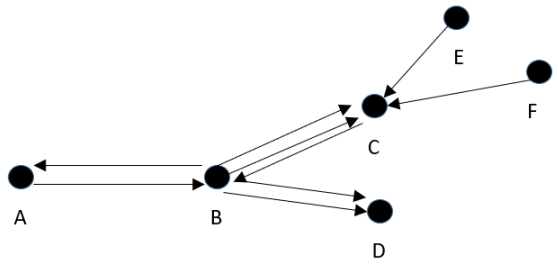

Fig. 1: The graph of virtual environment 1

The vertices in figure 1 are described in table 2.

Table 2 Virtual Environment 1

Vertex	Description
A	Kali Linux attack virtual machine
B	Wndows 7 virtual machine
C	Windows server 2012
D	Hardened Windows 10 vritual system
E	Windows 10 virtual system
F	Windows 10 virtual system

The directionality of the arcs indicates communication initiated from the origin to the tail. If a breach is made, then there is communication from the target, to the attack machine.

Virtual environment 2, is depicted as a graph in figure 2.

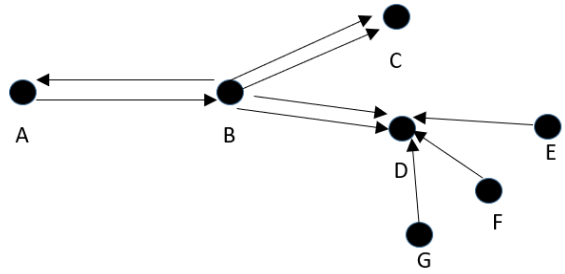

Fig. 2 : The graph of virtual environment 2

The vertices in figure 2 are described in table 3.

Table 2 Virtual Environment 2

Vertex	Description
A	Kali Linux attack virtual machine
B	Wndows 10 virtual machine
C	Windows server 2019 virtual machine
D	Ubuntu – Apache virtual machine
E	Windows 10 virtual system
F	Windows 10 virtual system
G	Windows 10 virtual system

As is expected, the two environments are different. In this scenario, the vertices A, B, C, and D do form an induced subgraph. This is depicted in figure 3.

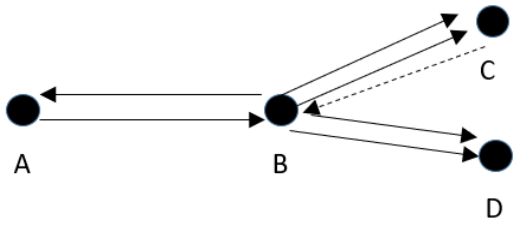

Fig. 3: Induced subgraph

The dotted line from vertex C to B represents the one arc present in one induced subgraph but not the other. One good further divide this into two induced subgraphs. The A-B-D subgraph is isomorphic in both virtual environments. The A-B-C subgraph is not, due to the arc present in one environment and not the other. However, the A-B-C subgraph is a weak homomorphism.

Using the weighting previously discussed in table 1, the two attacks have a match weighted at 4. This represents a 3 for the isomorphism + 1 for the weak homomorphism. Using equation 1, previously described, would provide a matching score of $\theta = \frac{8}{13}$ which is .615.

In this case study, it is already known that the two attacks were carried out by the same threat actor, using the same attack vector, and the same entry point. Thus, it is expected that the level of similarity would be high. However, it is necessary to compare this against an attack that was not done with the same attack vector.

The third experiment utilized virtual environment 2, with one change. In this case the attack was not with an MSFVenom package delivered via email, but rather an Excel spreadsheet with a malicious macro embedded and uploaded to a web server. The target machines were all sent a link encouraging them to download the attachment.

Table 3 Virtual Environment 1 Attack 2

Vertex	Description
A	Kali Linux attack virtual machine
B	Web Server for malicious file
C	Wndows 10 virtual machine
D	Windows server 2019 virtual machine
E	Ubuntu – Apache virtual machine
F	Windows 10 virtual system
H	Windows 10 virtual system
H	Windows 10 virtual system

Machines F and H did open the malicious spreadsheet initiating a connection back to machine A. The graph produced by this attack is shown in figure 4.

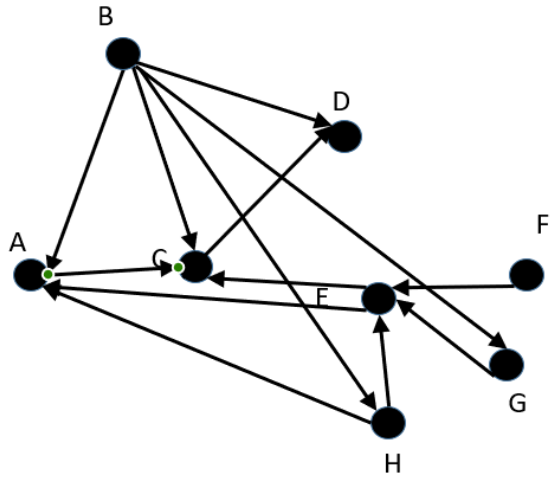

Fig. 4: Virtual Environment 2-Attack 2

Even though the initial attack machine is the same, and the target network is the same, even an elementary visual comparison of the graph in figure 4 verses the graph in figure 2 are clearly quite different. Now the two graphs are compared to seek out any homomorphic, non-trivial subgraphs. In scenario 3, the attacking machines are a and b. Vertex b did not exist in scenario 2, and in scenario 2, and vertex a only connected directly to B and C. This leads to a situation wherein there only one induced subgraph including an attacking machine, shown in figure 5.

Fig. 4: Virtual Environment 2-Attack 2-subgraph

As this subgraph does not retain the head-tail relationship, it is egamorphic. That provides matching score of $\theta = \frac{1}{15}$ which is 0.0666.

Thus, when two attacks involve the same attack vector, the same methodology, on very similar networks, the θ value was 0.615. When two attacks use a completely different attack vector and methodology, even on identical networks, the θ value was θ value was 0.615. Thus, the θ value from analyzing homomorphic induced subgraphs was effective in attack attribution and can provide a digital fingerprint for cyber-attacks.

4 Conclusion

In this study a methodology was proposed and tested with case studies. That method utilized homomorphisms of induced subgraphs provides a method that is equivalent to biological fingerprint matching. The various points (i.e. subgraphs) are compared to determine how many matching points exist in the two graphs. This allows a network intrusion wherein the threat actor and attack vector are known to be compared to an unknown attack to aid in attack attribution. The method outlined in the current study improves threat attribution to a quantifiable, mathematically precise practice.

More work in this area is recommended. The most obvious avenue for further research would be additional case studies involving known attacks. It may also be advantages to explore additional points of comparison including weightings and incidence functions. It may also be advantageous to explore additional levels of homomorphisms such as quasi-strong homomorphisms and locally strong homomorphisms.

Yet another area of possible extension for this current study is to explore variations in graph theory. Fractional graphs deal with non-integer

values and may be applicable to network attacks. Fuzzy graph theory integrates non-binary logic into graph theory. Both fractional graph theory and fuzzy graph theory are areas that should be explored in reference to the current study.

References:

[1] Hansen, L. P. The Spy Who Never Has to Go Out Into the Cold: Cyber Espionage. In Encyclopedia of Criminal Activities and the Deep Web (pp. 258-270). IGI Global. 2020.

[2] Easttom, C. The role of weaponized malware in cyber conflict and espionage. In Proc. 13th Int. Conf. Cyber Warfare Secur.(ICCWS) (p. 191). 2018.

[3] Easttom. An Examination of the Operational Requirements of Weaponized Malware. Journal of Information Warfare 17 (2). 2018.

[4] Vasiu, I., & Vasiu, L. Malicious Cyber Activity Distribution, Attribution, and Retribution. Advanced Cyberlaw and Electronic Security, 9-19. 2017.

[5] Cook, A., Nicholson, A., Janicke, H., Maglaras, L. A., & Smith, R. Attribution of cyber-attacks on industrial control systems. EAI Endorsed Trans. Indust. Netw. & Intellig. Syst., 3(7), e3. 2016.

[6] Edwards, B., Furnas, A., Forrest, S., & Axelrod, R. Strategic aspects of cyberattack, attribution, and blame. Proceedings of the National Academy of Sciences, 114(11), 2825-2830. 2017.

[7] Casper, S. P. Cyberspace and International Affairs: Nation-state Cyber-attacks and Normative Behavior (Doctoral dissertation, Utica College). 2019.

[8] Dörfler F, Simpson-Porco JW, Bullo F. Electrical networks and algebraic graph theory: Models, properties, and applications. Proceedings of the IEEE Vol.106, No. 5, pp. 977-1005. 2018.

[9] Rangaswamy KD, Gurusamy M. Application of Graph Theory Concepts in Computer Networks and its Suitability for the Resource Provisioning Issues in Cloud Computing-A Review. JCS. Vol., pp. 163-72. 2018.

[10] Tiwari, A., Boachie, M., & Gupta, R. (2019). Network Analysis of Economic and Financial Uncertainties in Advanced Economies: Evidence from Graph-Theory (No. 201982). 2019.

[11] Kulkarni, S. J. (2017). Graph theory: Applications to chemical engineering and chemistry. Galore International Journal of Applied Sciences and Humanities, 1(2). 2017.

[12] Easttom, C. A Systems Approach To Indicators Of Compromise Utilizing Graph Theory. IEEE International Symposium on Technologies for Homeland Security. 2018.

[13] Easttom, C. On the Application of Algebraic Graph Theory to Modeling Network Intrusions. 2020 IEEE 10th Annual Computing and Communication Conference.

[14] Easttom, C. Adda, M. An Enhanced View of Incidence Functions for Applying Graph Theory to Modeling Network Intrusions. WSEAS Transactions On Information Science And Applications. DOI: 10.37394/23209.2020.17.12 2020.

[15] Kijewski, P., Jaroszewski, P., Urbanowicz, J. A., & Armin, J. The never-ending game of cyberattack attribution. In Combatting Cybercrime and Cyberterrorism (pp. 175-192). Springer, Cham. 2016.

[16] Rowe, N. C. The attribution of cyber warfare. In Cyber Warfare (pp. 75-86). Routledge. 2015.

[17] N. Miloslavskaya, "Remote Attacks Taxonomy and their Verbal Indicators." Procedia Computer Science, 123, 278-284, 2018.

[18] Mell P, Scarfone K, Romanosky S. Common vulnerability scoring system. IEEE Security & Privacy. Nov;4(6):85-9. 2006.

[19] Johnson P, Lagerström R, Ekstedt M, Franke U. Can the common vulnerability scoring system be trusted? a bayesian analysis. IEEE Transactions on Dependable and Secure Computing. Dec 23;15(6):1002-15.2016.

[20] van de Kamp, A. Peter, M. Everts, & W. Jonker, W. "Private sharing of IOCs and sightings." In Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security (pp. 35-38). ACM. 2016.

[21] S. Barnum, "Standardizing cyber threat intelligence information with the Structured Threat Information eXpression (STIX)." MITRE Corporation, vol 11, pp 1-22, 2012.

[22] C. Eoghan, G. Back, & S. Barnum "Leveraging CybOX™ to standardize representation and exchange of digital

forensic information." Digital Investigation, vol. 12, pp S102-S110, 2015.
[23] Shevchenko N, Chick TA, O'Riordan P, Scanlon TP, Woody C. Threat modeling: a summary of available methods. Carnegie Mellon University Software Engineering Institute Pittsburgh United States. 2018.
[24] Mead NR, Shull F, Vemuru K, Villadsen O. A hybrid threat modeling method. Carnegie MellonUniversity-Software Engineering Institute-Technical Report-CMU/SEI-2018-TN-002. 2018.
[25] N. Deo. "Graph theory with applications to engineering and computer science." Courier Dover Publications. 2017.
[26] Godsil C, Royle GF. Algebraic graph theory. Springer Science & Business Media. 2013.
[27] Knauer U, Knauer K. Algebraic graph theory: morphisms, monoids and matrices. Walter de Gruyter GmbH & Co KG. 2019.
[28] J. Gross, J. Yellen, & P. Zhang. Handbook of graph theory. Chapman and Hall/CRC. 2013.
[29] Boutrig R, Chellali M, Haynes TW, Hedetniemi ST. Vertex-edge domination in graphs. Aequationes mathematicae;90(2):355-66. 2016.
[30] Fang, G., Srihari, S. N., Srinivasan, H., & Phatak, P. (2007, April). Use of ridge points in partial fingerprint matching. In Biometric Technology for Human Identification IV (Vol. 6539, p. 65390D). International Society for Optics and Photonics. 2007.
[31] Jain, Anil, Arun Ross, and Salil Prabhakar. "Fingerprint matching using minutiae and texture features." In Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205), vol. 3, pp. 282-285. IEEE, 2001.
[32] O'Leary, M. Malware and Persistence. In Cyber Operations (pp. 507-566). Apress, Berkeley, CA 2019.
[33] Jaswal, N. *Mastering Metasploit*. Packt Publishing Ltd. 2016.