Utilizing Machine Learning for Optimizing Cybersecurity Spending in Critical Infrastructures

George Stergiopoulos[1], Michalis Detsis[2], Sozon Leventopoulos[2] and Dimitris Gritzalis[2]
[1]*Dept. of Information & Communication Systems Engineering, University of the Aegean, Samos, Greece*
[2]*Dept. of Informatics, Athens University of Economics & Business, Athens, Greece*

Keywords: Risk Assessment, Risk Management, Artificial Intelligence, Graph Theory.

Abstract: This research paper presents a methodology and corresponding tool that aim to automate decision-making in prioritizing cybersecurity investments by identifying a minimal subset of assets based on their risk exposure, the protection of which would yield maximum risk reduction and cost efficiency. The presented method aims to assist in strategic security planning, offering significant savings while ensuring robust cyber defense mechanisms are in place. To achieve this, we developed an application that identifies and classifies critical assets within ICT networks using supervised machine learning, graph centrality measurements and cascading attack paths. We utilize over 100 randomly generated network models taken from existing companies to build a classifier able to determine ICT critical nodes. We use topological features and dependency risk graphs to simulate potential cyberattack paths.

1 INTRODUCTION

Cyber threats exploit vulnerabilities across information and communication technology (ICT) assets on an increased pace. To address these threats, organizations follow risk management practices that provide insights and recommendations on strengthening organization's cybersecurity posture. Nevertheless, the complexity of modern, decentralized networks complicates the risk assessment process, and the resulting investment prioritization.

Despite these advancements, balancing trade-offs during the implementation of measures is often manual. Effective resource allocation is crucial to protect against cyber vulnerabilities and maximize investments. This research explores the feasibility of using a supervised machine learning model to classify ICT assets based on their risk and position within the information system. The model represents ICT assets as nodes and their dependencies as edges and uses machine learning to prioritize investment during risk treatment.

1.1 Contribution

This research paper proposes a machine-learning model able to identify an arbitrary group of nodes within a network whose security enhancement leads to the greatest reduction of risk across the network. The targeted group is comprised of nodes characterized by their significant cumulative attack risks and notable positions within the network, indicated by high eigenvector centrality, functioning as critical connectors (indicated by high betweenness centrality), or being centrally located (highlighted by high closeness centrality).

The idea of using Centrality Measures in Dependency Risk Graphs (Stergiopoulos et al., 2015) is combined with the estimation of n-order dependency chains to be used as features and train a machine learning (ML) process with randomly generated networks that are formed over multiple bases. The model can identify and eliminate critical sub-net paths while maintaining the network's connectivity (Kotzanikolaou et al., 2013). Our contribution is summarized as follows:

- **Model for Investment Prioritization during Risk Treatment:** We introduce a novel approach by combining centrality measurements with cascading attack paths to train a machine learning model for investment prioritization during risk treatment.
- **Testing and Validation:** we test and validate the presented method on simulated ICT systems using

Stergiopoulos, G., Detsis, M., Leventopoulos, S. and Gritzalis, D.
Utilizing Machine Learning for Optimizing Cybersecurity Spending in Critical Infrastructures.
DOI: 10.5220/0012837300003767

In *Proceedings of the 21st International Conference on Security and Cryptography (SECRYPT 2024)*, pages 544-551
ISBN: 978-989-758-709-2; ISSN: 2184-7711

randomized simulations of real-world ICT environments based on company networks.

Results indicate that our approach can identify a minimal subset of critical assets for protection, significantly reducing overall risk and associated costs, thereby improving resource allocation and decision-making.

2 RELATED WORK

This work extends a previous framework (Stergiopoulos et al. 2020) for modelling the connections of ICT asset interdependencies on a company's business processes through dependency structural risks. Original work aimed at prioritizing assets based on their influence using dependency risk graphs, graph minimum spanning trees, and network centrality metrics. Attack graphs have been used in literature (Ray, 2005; Dewri et al., 2007) to model network devices and systems repeatedly for the purpose of prioritizing mitigation controls.

More recent related work from (Aksu et al., 2017) showcased a quantitative asset-centric risk assessment method based on attack-graph analysis, although their work does not tackle risk mitigation issues and prioritization. In (Shivraj et al., 2017) authors presented a model-driven risk assessment framework that was based on graph theory to model the flow graph and produce relevant attack trees according to the underlying ICT architecture. Still, this work addresses attack vectors and software state dependencies rather than risk assessment results on business processes.

Similarly, (Hermanowski, 2018) used graphs to assess the risk of ICT using the MulVAL attack graph tool which adapts for risk assessment attack paths calculation against crucial assets of an IT system. Authors (Grigoriadis et al., 2021) proposed a situation -driven security management system to dynamically implement security controls specific to different use cases by producing dynamic risks for various situations. In (Stellios et al., 2021) authors proposed a graph-based analysis of risk assessment results over ICT systems.

In (Stergiopoulos et al., 2022), the authors proposed a method to automatically create complex attack graphs for enterprise networks, relating microservices, virtual system states, and cloud services as graph nodes using mathematical graph series and group clustering to prioritize vulnerabilities by analysing system states' effects on the overall network. This research is based on the relevant results to analyse graph paths and software state vulnerabilities but expands its focus, by building a machine learning classifier for decision support during risk management, aiming at cost reduction in implementing safeguards through machine learning.

3 BACKGROUND

3.1 Graph Structure and Node Predictions

According to Evans[1], there are two major ways that machine learning can be of service: (a) automate the functions that are easily understandable by humans, but hard for computers to comprehend, and (b) transform information on a large scale. Several methods have been developed to address the representation of graphs with complex structures in simple forms, such non-deterministic low-dimensional node embeddings.

Techniques like BFS[2] and DFS[3] are instrumental in generating embeddings that reflect these equivalences, with BFS aligning with structural equivalence and DFS with homophily equivalence. The effectiveness of these methods in labelling nodes is further enhanced by incorporating heuristic methods that consider structural characteristics and an "influence spread" factor (Zhang et al, 2016).

Adding "influence spread" to centrality measurements significantly improved model's performance, evidenced by an increase in the F1[4] score from 0.65 (not acceptable) to 0.86. Furthermore, we have identified that the applied methodology can isolate bridges by controlling the distance of influence and revealing closely interacting clusters (homophily equivalence). This factor, crucial for achieving accurate machine learning models, considers the potential of a subset of nodes to propagate information or malware through the network.

[1] https://www.ben-evans.com/benedictevans/2018/06/22/ways-to-think-about-machine-learning-8nefy

[2] Breadth First Search

[3] Depth First Search

[4] F1 score is an error metric used in classification, which measures per-formance by calculating the harmonic mean of precision and recall for the minority positive class. F1 score can be interpreted as a measure of overall model performance from 0 to 1, where 1 is the best.

3.2 Active Learning

Active learning (part of Machine Learning) is the process when a learning algorithm can interactively query a human (or other information) source (Settles B, 2009). This process helps label new data points with the desired outputs, provided that the information source has the required expertise and knowledge on the subject. The algorithm can actively query the information source for labels, thus minimizing the number of examples needed compared to a normal supervised learning model.

In this work, we use a Feedforward Neural Network (FNN) with ReLU activation functions for the input layers and SIGMOID activation functions for the output layers, maintaining a one-to-one correspondence between input values and classification labels. This is proven to be a promising setup for analyzing information networks of interconnected assets where information flow is degined by one-to-one relationships of different types of objects.

3.3 Simulating ICTs with Dependency Risk Graphs

The definition of ICT encompasses the role of unified communications a business and marketing concept regarding the integration of enterprise communication with non-real-time communication services, and that of information technology, such as enterprise software, critical business applications, and essential business development. The visual representation of these interconnections can be formulated using dependency graphs where the nodes represent assets and services of the NI, and their directed edges represent the potential risk that the destination node may suffer due to its dependency from the source node, in case of a failure being realized at the source node (Stergiopoulos, 2015) (Rinaldi et al., 2001).

Risk management helps organizations allocate limited resources (manpower and budget) to mitigating the most significant threats first rather than spending funds in less critical areas. A plethora of definitions on risk is available, such as the ISO Guide 73:2009, ISO 31000 series, or the NIST SP 800 series. For the purposes of this research, we used the definition provided by NIST[5].

Risk is a metric defined as the sum of the Likelihood (L) of a threat occurrence on a business asset times the Impact (I) of the threat manifestation. It considers the possibility of the threat event (P), the vulnerability level (V) of an asset or object to that threat and the impact (I) to the business concept supported by the asset. We use typical Risk scales and input from Risk analysis on assets and systems to produce Risk metrics to embed onto the graph's edges and nodes.

$$Risk = L\ x\ I = (P\ x\ V)x\ I$$

For calculating the risk chain (attack path) we considered a path graph G', a subgraph of a graph G = (V, E) and a path of G. This path forms an attack path, or a risk chain, comprising n nodes numbered from N_1 to N_n, like the one depicted in the figure below:

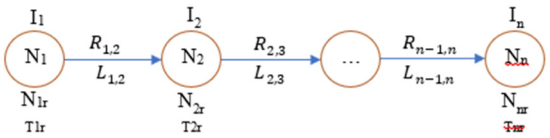

Figure 1: Attack path (risk chain).

Each node $N_i, i = 1, 2, \dots, n$, which is mapped to a vertex V_{N_i} of G, corresponds to a threat event that incurs an impact I_i with likelihood L_i, and each edge denotes a derivation of a node state, e.g., the likelihood $L_{j,j+1}, j = 1, 2, \dots, n-1$, to exploit node N_{j+1} from its predecessor node N_j, or the first-order **dependency risk $R_{j,j+1}$** node N_j exhibits on N_{j+1}.

4 APPROACH FORMULATION

4.1 Algorithmic Process

The whole application is executed using one to five stages. All stages are controlled by a configuration file (see snippet below). The five stages are:

1. Random Network Creation (optional),
2. Feature Extraction,
3. Training
4. Active Learning (optional)
5. Testing and validation

The combinations Feature Extraction → Active Learning and Feature Extraction → Testing form two ordered pairs that we will define as:

$$[\text{Feature Extraction}]_{[\text{Active Learning}]} \equiv (\text{Feature Extraction, Active Learning})$$

$$[\text{Feature Extraction}]_{\text{Testing}} \equiv (\text{Feature Extraction, Testing})$$

[5] https://csrc.nist.gov/glossary/term/risk

- **Random Network Creation:** Several random-generated network components (NIs) are created based on baseline models of real ICT company networks of assets.
- **[Feature Extraction]$_{Training}$:** Combined stage in which the centrality measurements and cumulative risks for all nodes are calculated. Then, this step evaluates the criticality of nodes using active learning methods. Both results will be used as inputs to the Training stage.
- **Training Stage:** Results generated at earlier stages are fed to the Machine Learning process, which is a feedforward neural network with multiple inputs and outputs.
- **[Feature Extraction]$_{Testing}$:** This is a combined stage in which one node is taken as input (the graph is stored in the Neo4j database), and using the Neo4j GDS library, we calculate centrality measurements and cumulative risks for all nodes.

4.2 Synthetic Dataset for Research

The first step in our approach is to create a random hierarchical network (tree network) where several different network components can be simulated. Random network generator also supports static network as templates that can be used as a base to create more complex networks. To simulate potential attack paths, most of the random network components allow input connections and output connections.

Impact values vary from 1 (low impact) to 9 (high impact). All components get a value based on attack graph participation and expert knowledge assigned to the node as an integer. The network components are connected according to possible attack paths. An attack path can originate from one component and be directed from one or more of its neighbor components to other network components. The risk dependency is calculated as the product of the impact of the attacked component and the value of the likelihood of the attack.

Within the framework of this research, several random Network Instances have been created. These instances were created to train the model on identifying the critical network assets, based on the estimated overall risk, and their interconnections,

4.3 Feature Extraction

Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set.

4.3.1 Feature Group 1: Centrality

For the purpose of calculating the significance of each node the following features where calculated:

- **Degree centrality** indicates a node's importance by counting its direct neighbours (Kumar et al., 2020).
- **Betweenness centrality** shows the criticality of connectedness. It is the number of the shortest paths between a pair of nodes.
- **Closeness centrality** is the reciprocal of the mean distance to all other nodes from the current node. The greater its value, the shortest the node distances to the rest of the graph.
- **Eigenvector centrality** measures the importance and the transitive influence of the node.

4.3.2 Feature Group 2: Node Risk Metrics

As mentioned previously, the outcome of a risk assessment of a given asset (system, data type or process) is a risk score assigned to a particular vulnerability, calculated by considering its likelihood and impact. Each node has an **overall node risk N_{ir}**, which is the sum of all cascading risks that *target* node N_i and its mapped vertex V_{N_i}, and a **total hit count T_{ir}** of cascading risks that target node N_i. The **average node risk AN_{ir}** is defined as the ratio of *overall node risk* over *total hit count*:

$$AN_{ir} = \frac{N_{ir}}{T_{ir}}, if\ T_{ir} > 0, else\ 0 \quad (1)$$

The **cumulative dependency risk $CR_{1\ldots n}$** is the overall risk produced by the nth-order dependency of the attack path:

$$CR_{1\ldots n} = \sum_{i=1}^{n} R_{1\ldots i} = \sum_{i=1}^{n} \left(\prod_{j=1}^{i} L_j \right) I_i \quad (2)$$

The **cumulative attack risk CR_N** for node N of graph G is defined as the sum of all cumulative dependency risks of the attack paths that *start* from this node.

$$(N_{ir})_{new} = (N_{ir})_{old} + R_{1\ldots i} \quad (3)$$

The **overall attack graph risk G_r** is the sum of the cumulative dependency risk for each nth-order dependency of graph G:

$$G_r = \sum CR_{1\ldots n} \quad (4)$$

4.3.3 Feature Group 3: Dependency Risk Chains Metrics

With the identification of the dependency chains (threat vector) the following key findings are extracted:

- Find a subset of nodes that affect many critical dependency paths. Decreasing the probability of failure in these nodes by selectively applying security controls may have a greater overall benefit in risk reduction.
- Identify nodes of high importance outside the most critical dependency paths that concurrently affect many other nodes in these paths or impact the overall dependency risk of the entire structure/graph.

Using a simple SI epidemic model (Barabási et al., 1999) and as per the VoteRank description (Zhang et al., 2016), a node can be in one of two statuses: Susceptible (S) or Infected (I). At the outset of the process, all nodes are deemed susceptible except for a designated group of r infected nodes that serve as source spreaders. During each interval, an infected node endeavors to infect one of its neighboring nodes with a likelihood of μ and a consequential impact of I. This impact is determined by multiplying the probability of infection (μ) by the influence of the infected node (I), which ultimately produces the Risk value (Risk = μ * I). Once the infection has been successfully transmitted, the neighboring node's vote count will increase by one.

Furthermore, nodes that exceed a designated risk level are classified as critical only under specific circumstances. These include being deemed as "significant" (having a high Eigenvector centrality), serving as bridges or bridging nodes (having a high Betweenness centrality), or functioning as central nodes (having a high Closeness centrality).

4.3.4 Critical Node Classification

To programmatically determine the criticality of each node, we used “active learning” (Ricci et al., 2015). Each node determined as critical is labelled with “1”, while the non-critical ones with “0”. Specifically, for each centrality measure, the top-ranking nodes are selected that have a rank greater or equal to the midpoint of the extrema values and are classified as possibly critical. Then after the calculation of attack paths:

- the *overall node risk* (when **attackPathStrategy=TargetNode**), or
- the *cumulative attack risk* (when **attackPathStrategy=SourceNode**)

are calculated and if its ratio over the maximum value of all nodes is greater than or equal to a threshold value (**thresholdRiskRatio), the node is also classified as possibly critical**. The final estimation of the criticality of the node is the combination of *overall node risk* (or *cumulative attack risk*) criticality and one or more of the *Betweenness*, *Closeness* and *Eigenvector* centralities, which provide best results in determining the criticality of a node.

5 EXPERIMENTAL RESULTS

5.1 Feature Extraction

Using the GDS plugin we calculated several metrics which then were used for training the model. We therefore could identify “possibly critical” nodes (i.e., node that exceed the midpoint value of each applicable metric), attacks paths and their associated risk based on the OWASP Severity Risk Levels. Furthermore, the same approach was used to calculate various risk levels (e.g., the overall, and average node risk, etc.), which were fed to the training model.

Using as threshold the **thresholdRiskRatio** value, all the *overall node risks* or *cumulative attack risks* which have values above the $thresholdRiskRatio * max\{N_{1r}, N_{2r}, \dots, N_{nr}\}$ are labelled as possibly critical and all the others as not critical. A node is classified as definitely critical if it is labelled possibly critical in all measurements.

Finally, using the DFS algorithm (with limited depth if **attackPathMaxDepth > 0**, or with unlimited depth if **attackPathMaxDepth=-1**), all the attack paths are determined, and the *overall node risk*, *average node risk* and *cumulative attack risk* are calculated for each node of the projected graph.

5.2 Training

First, we created complex ICT networks as graphs by formulating an isometric topology to the ICT service network characteristics. These NIs were then fed to the classifier for training the Machine Learning algorithm based on the following features, (i) the normalized values of the centrality measures, (ii) the overall node risk, (iii) cumulative attack risk and (iv) dependency path risk. To find all potential nth-order dependencies in G and calculate the overall node risks, all overall node risks are first initialized to zero.

Next, for every node $N \in V$, a DFS (Depth-First Search) algorithm is applied, limiting the maximum depth to n (when **attackPathMaxDepth** configuration property has a positive integer value), or without

limiting the depth (when **attackPathMaxDepth = -1)**. Then, all attack paths that start from N are determined and the *cumulative dependency risk* is calculated for each attack path, along with the new values of the *overall node risks* of the nodes belonging to the attack path. Finally, the *cumulative attack risk* related to node N is calculated. The resulted data were fed to RELU-activated inputs of a Feedforward Neural Network (FNN) and the classification labels to an equal number of SIGMOID-activated outputs with 1-1 pairing to the input values.

The resulting FNN contains two hidden layers, one with 64 neurons and one with 32 neurons, and has a retain probability of neurons of 90%. The optimization algorithm that it uses is the Stochastic Gradient Descent. The following figure showcases the model and training information and the parameter ratios and standard deviations:

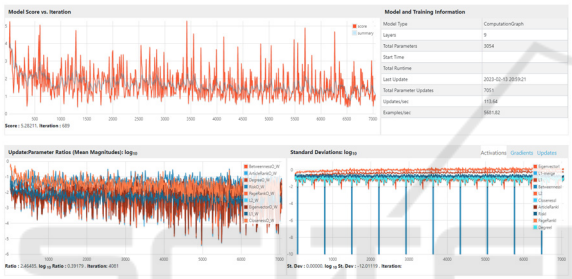

Figure 2: Overview page with information regarding the training model.

5.3 Testing

For testing the ML algorithm, we used new ICT networks, transposed them as graphs and extracted the normalized values of the centrality measures and the overall node risk metrics. The model represents ICT assets as nodes and their dependencies as edges, using centrality measures and dependency chains (Stergiopoulos, 2015) These were fed to the previously trained ML algorithm to classify the criticality of each node based on whether the normalized value of the overall node risk, or cumulative attack risk, is critical and over the threshold value specified by:

- thresholdRiskRatio: [Overall Node Risk or Cumulative Attack Risk is critical], and
- [Normalized Overall Node Risk or Cumulative Attack Risk >= thresholdRiskRatio].

5.3.1 Test Scenario #1

For the purposes of the first test scenario, we used the network proposed by (Dedousis P., 2019), which in turn, is based on a real-life industry network implementation from the Greek private sector. We identified (as per the output log, see Figure 6) that **Node A5** is critical, as it is the bridging and central node with the most dependency chains. This means that:

1. Threat attacks originating from node A5 quickly traverse the whole NI, posing a great risk for the assets and services.
2. In our example**,** node A5 constitutes a single point of failure (SPOF) for the communication network between assets. SPOFs are undesirable in any system with a goal of high availability or reliability, be it a business practice, software application, or other industrial system.
3. Nodes A15 and A32 are bridging nodes serving information to two network subnets. A disruption of the connection between them affects the business carried out by both departments.

2023-03-02 10:26:00 INFO_Main:157 - **A15 is CRITICAL**, due to risk AND (importance or betweenness or closeness), is bridge, is central

2023-03-02 10:26:00 INFO_Main:157 - **A5 is CRITICAL**, due to risk AND (importance or betweenness or closeness), is bridge, is central

2023-03-02 10:26:00 INFO_Main:157 - **A32 is CRITICAL**, due to risk AND (importance or betweenness or closeness)

Figure 3: Extract from the output log, showcasing critical nodes.

5.3.2 Test Scenario #2

For the purposes of this test scenario, a network simulating a real-world scenario of a complex structure, was used. Three large nodes were created, representing the various departments of a modern-day organization, with the appropriate interconnections as necessitated by the organization's business processes.

2023-03-02 10:36:33 INFO_Main:157 - **Router_2 is CRITICAL**, due to risk AND (importance or betweenness or closeness), is important, is bridge, is central

2023-03-02 10:36:33 INFO_Main:157 - **Switch_5 is CRITICAL**, due to risk AND (importance or betweenness or closeness), is important, is bridge, is central

Figure 4: Extract from the output log, showcasing critical nodes.

The critical nodes reported are shown in bold and these are: Router_2 and Switch_5. They are bridging and central nodes (also, Firewall_1 is bridge, but has less cumulative attack risk than Switch_5 which is connected to it). They are central due to the maximum attack path limit set (**attackPathMaxDepth=4**).

5.4 Validation

To validate our approach, we created an unprotected network as shown in Figure 9. Our approach is based

on a real-world scenario, and while it lacks complexity, it does represent a typical networking approach of an organization. The network components are depicted below:

Table 1: Network Components.

Asset name	Asset type	Likelihood	Impact
Server_1	HTTP Server	8	7
Server_2	Database Server	7	8
Server_3	Email Server	6	6
Switch_1	Switch	4	3
PC_1	PC	5	2
PC_2	PC	5	2
PC_3	PC	5	2

Using results returned from the model, we can make **a uniform decision** regarding the placemenand type of the safeguard, and then rerun the test. The expected result should be that the model will identify **no critical node** on the protected network. We identified that in the unprotected network, **Switch_1** is the critical asset due to its high **cumulative attack risk** and at least one of the *Eigenvector, Betweenness* and *Closeness* values. To mitigate the risk, we applied the following security controls, based on best practices:

1. Insert a firewall in gateway mode, connecting all the servers and the switch to the firewall and all the rest assets (the PCs) to the switch.
2. Lower the impact of Switch_1 by 2 units since no critical assets are connected to it.

The respective results clearly show the improvement in the criticality of Switch_1,:

Cumulative Attack Risk	0.19955817452512156	NON-CRITICAL
PageRank	**0.3886785997858035**	
ArticleRank	**0.3786797072154877**	
Eigenvector	**0.3871648907036537**	
Betweenness	**0.4090909090909091**	
Degree	**0.3749994478736234**	
Closeness	**0.40104059612122567**	

Figure 5: Extract from the output log.

The following figure shows the ex-post and ex-ante network configuration. The latter encompasses the recommendations from the AI model.

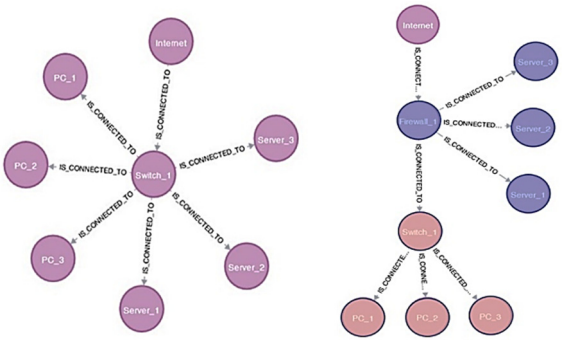

Figure 6: Network configuration, prior and after the implementation of the proposed safeguards as per the AI model.

6 CONCLUSIONS

We have developed and propose a machine learning model, that can automatically identify and classify assets within ICT networks, irrespective of their size or complexity. The model utilizes centrality measures, dependency chains, and machine learning to provide a predictive risk estimation that can effectively support the decision-making process in regards to allocating funds towards the implementation of the most effective security measures on the most critical network assets.

The validation of our model confirmed its benefits, demonstrating quick and efficient identification of optimal safeguards, without affecting network connectivity and performance. The model can scale rapidly, without any issues identified. The testing results highlighted the feasibility of our model, especially in cybersecurity risk management scenarios, which are particularly valuable for companies with limited resources.

This model is a significant advancement in predicting and prioritizing cybersecurity investments, since it can optimise resource allocation, focus on network assets with topological significance and thus enhancing the cybersecurity posture of the organizations basing their business models on ICT infrastructure.

REFERENCES

Aksu, M. U., Dilek, M. H., Tatli, E. I., Bicakci, K., Dirik, H. I., Demirezen, M. U., & Aykir, T. (2017). *A quantitative CVSS-based cyber security risk assessment methodology for IT systems*, IEEE. https://doi.org/10.1109/ccst.2017.8167819

Barabási, A.-L., & Albert, R. (1999). *Emergence of Scaling in Random Networks*. In Science (Vol. 286, Issue 5439,

pp. 509–512). American Association for the Advancement of Science. https://doi.org/10.1126/ science. 286.5439.509

Burr Settles. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648, University of Wisconsin–Madison. 2009.

Dedousis P. (2019) "*Development of software for the automatic restructuring of corporate networks using graphs*" [Master Thesis]. Greece: Athens University of Economics and Business (online: https://pyxida. aueb.gr/index.php?op=view_object&object_id=7047)

Dewri, R., Poolsappasit, N., Ray, I., Whitley, D. (2007). *Optimal security hardening using multi-objective optimization on attack tree models of networks*. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 204–213.

Grigoriadis, C., Laborde, R., Verdier, A., & Kotzanikolaou, P. (2021). *An adaptive, situation-based risk assessment and security enforcement framework for the maritime sector. Sensors*, 22(1), 238.

Hermanowski, D., & Piotrowski, R. (2021). *Network Risk Assessment Based on Attack Graphs*. In Theory and Engineering of Dependable Computer Systems and Networks (pp. 156–167). Springer International Publishing. https://doi.org/10.1007/978-3-030-76773-0_16

Kotzanikolaou, P., Theoharidou, M., & Gritzalis, D. (2013). *Cascading Effects of Common-Cause Failures in Critical Infrastructures*. In Critical Infrastructure Protection VII (pp. 171–182). Springer. https://doi.org/10.1007/ 978-3-642-45330-4_12

Kumar, S., & Panda, B. S. (2020). *Identifying influential nodes in Social Networks: Neighborhood Coreness based voting approach*. In Physica A: Statistical Mechanics and its Applications (Vol. 553, p. 124215). Elsevier. https://doi.org/10.1016/j.physa.2020.124215

Kurzenhauser, S. (2003). *Natural frequencies in medical risk communication: Applications of a simple mental tool to improve statistical thinking in physicians and patients*. Ph.D. thesis, Freie Universitat Berlin

Ray, I., Poolsappasit, N. (2005). *Using attack trees to identify malicious attacks from authorized insiders*. In: Proceedings of ESORICS, Italy, pp. 231–246.

Ricci, F., Rokach, L., & Shapira, B. (Eds.). (2015). *Recommender Systems Handbook*. Springer. https://doi.org/ 10.1007/978-1-4899-7637-6

S. M. Rinaldi, J. P. Peerenboom and T. K. Kelly. Dec. (2001). *Identifying, understanding, and analyzing critical infrastructure interdependencies*, in IEEE Control Systems Magazine, vol. 21, no. 6, pp. 11-25, https://doi:10.1109/37.969131

Shivraj, V. L., Rajan, M. A., & Balamuralidhar, P. (2017). *A graph theory based generic risk assessment framework for internet of things (IoT)*. IEEE. https://doi.org/10.1109/ants.2017.8384121

Stellios, I., Kotzanikolaou, P., Psarakis, M., & Alcaraz, C. (2021). *Risk assessment for IoT-enabled cyber-physical systems.* Advances in Core Computer Science-Based Technologies: Papers in Honor of Professor N. Alexandris, 157-173.

Stergiopoulos, G., Dedousis P., & Gritzalis G. (2022). *Automatic analysis of attack graphs for risk mitigation and prioritization on large-scale and complex networks in Industry 4.0*. International Journal of Information Security 21.1: 37-59.

Stergiopoulos, G., Dedousis, P., & Gritzalis, D. (2020). *Automatic network restructuring and risk mitigation through business process asset dependency analysis*. In Computers & Security (Vol. 96, p. 101869). Elsevier BV. https://doi.org/10.1016/j.cose.2020.101869

Stergiopoulos, G., Theocharidou, M., Kotzanikolaou, P., & Gritzalis, D. (2015). *Using Centrality Measures in Dependency Risk Graphs for Efficient Risk Mitigation*. In IFIP Advances in Information and Communication Technology (pp. 299–314). Springer International Publishing. https://doi.org/10.1007/978-3-319-26567-4_18

Stergiopoulos, Y. (2015). *Securing critical infrastructures at software and interdependency levels*. National Documentation Centre (EKT). https://doi.org/10.126 81/eadd/36753

Wirth, A. (2017). *The Economics of Cybersecurity. In Biomedical Instrumentation & Technology* (Vol. 51, Issue s6, pp. 52–59). Association for the Advancement of Medical Instrumentation. https://doi.org/10.2345/ 0899-8205-51.s6.52

Zhang, J.-X., Chen, D.-B., Dong, Q., & Zhao, Z.-D. (2016). *Identifying a set of influential spreaders in complex networks*. In Scientific Reports (Vol. 6, Issue 1). Springer. https://doi.org/10.1038/srep27823