Assessing Network Vulnerabilities: Implementing a Graph-Based Approach to Identify Critical Internet Infrastructure

Ahmad Fauzan Putra - 13524141
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: amanfauzan31@gmail.com , 13524141@std.stei.itb.ac.id

***Abstract*—This paper performs a connectivity analysis of the submarine cable network that serves as the foundation of the Internet, focusing on its vulnerability. This study shows the network's inherent fragility through a graph-based model to identify critical chokepoints. Using minimum cut set to examine and determine the criticality of certain submarine cables or landing points within the global network from the perspective of related countries.**

***Keywords*—Network Vulnerability, Graph Theory, Submarine Cables, Minimum Cut Set, Internet Connectivity.**

I. Introduction

The modern economy, communication, and social connectivity are heavily reliant on the flow of operation of the Internet. The Internet's physical reality is a vast network of critical submarine cables. International data traffic goes through these undersea cables, making them important to the digital world. This reliance creates a vulnerability that is often overlooked.

The fragility of this network is not just theoretical. Many incidents have demonstrated this reality. With one such event happening in 2020, where cables a part of the Falcon network were cut due to the anchor of a ship passing nearby[1]. This event caused disruptions to the internet of nearby countries, with Yemen being most affected with approximately 80% of its internet service being disrupted[1]. A catastrophic failure could effectively isolate entire nations and cripple their economies. This study aims to identify the combinations of points of failure that would lead to a complete loss of international connectivity for a given country.

This paper uses a graph-based method to identify those critical internet infrastructure. By assessing the global submarine cable network as a graph, with landing stations as nodes and cables as edges. Using minimum cut sets, to find the minimum number of edges or nodes that must be removed to disconnect a graph. By applying this analysis from the perspective of specific nations, we can identify the exact set of cables and landing points whose disruption would disconnect a country from the global internet.

II. Theoretical Foundations

A. Graph Theory

A graph is a mathematical structure used to model relations between two or more objects. It usually consists of pairs of sets, where one is a set of vertices/nodes and the other is a set of edges.[2]

Terminologies:

a) Vertex (Node): An element of set V, representing entities such as landing points.

b) Edge: An element of set E, connecting two vertices/nodes.

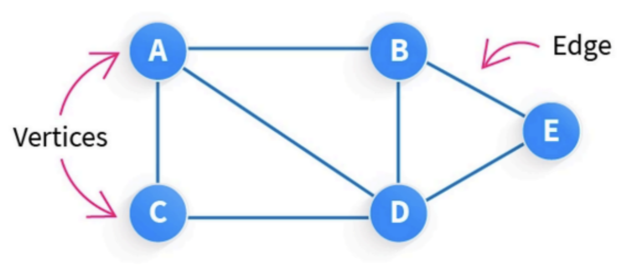

Fig. 2.1. Illustration of a graph's vertex and edge [Source: https://blog.stackademic.com/graph-basics-explained-from-vertex-to-ed ges-a0b240041fe7, Accessed: Jun. 19, 2025.]

.

c) Undirected Graph: A graph that only contains nodes and edges connecting those nodes.

d) Cut Set: A set of nodes or edges that, when removed, would disconnect a subgraph from the "main" graph. For instance (see Fig. 1), node B and node D would form a cut-set that would isolate node E from nodes A and C

B. *Minimum S-T Cut*

An s-t cut requires the node 's' and the node 't' to be in different subsets, and for there to be edges that form a connection between the source and the sink. [3]
A minimum s-t cut can be found through these algorithms:

1) Ford-Fulkerson Algorithm
2) Dinic's Algorithm

The problem contained within this paper can be transformed into the Minimum S-T Cut problem by bundling them into 2 groups commonly called the Source and the Sink.

III. Implementation

The Implementation is composed of three main stages: acquiring and storing network data, constructing a graph model, and performing a vulnerability analysis using minimum cut sets. This implementation uses Python for its extensive library and readability.

A. *Tools*

This project's implementation utilizes Python as the programming language for its readability and extensive library.
The following libraries are used within the program:

a. SQLite3: Enables the use of .db files that act as storage where information about the submarine cable networks is stored.
b. Collections: Enables the use of defaultdict, an improved dictionary that eases the handling of missing keys.
c. Itertools: Enables the use of a combinatorics function that helps with building the graph.
d. NetworkX: Introduces the ability to create a graph, and functions related to it, such as a function to find a minimum cut set.

B. *Data Acquisition and Storage*

To ensure the analysis is based on current infrastructure, real submarine cable network data is used and extracted by pulling live JSON data from submarinecablemap.com. This dataset contains detailed information on cable routes, landing points, and their interconnections. The extracted JSON data is then parsed and stored in a local SQLite Database. This approach allows for continuous use throughout the research process without having to continuously pull data from the website.

C. *Graph Model Construction*

The submarine cable network was modeled as a simple undirected graph, where nodes represent cable landing points, and edges represent submarine cables connecting these points.

The graph is constructed using the data contained within the database using Python. The following is the program used to fetch the data from the database and build the graph:

```python
def getData():
    conn = sqlite3.connect('cables.db')
    c = conn.cursor()
    c.execute('SELECT name, landing_point,
country FROM cables WHERE country != ""')
    data = c.fetchall()
    conn.close()
    return data
```

Fig 3.1 Data Fetching Function [Source: Author]

```python
def buildgraph(dbData):
    G = nx.Graph()
    lpToCountry = {}  # lp = landing point
    cableToLps = defaultdict(list)

    for cable, lp, country in dbData:
        lpToCountry[lp] = country
        G.add_node(lp, country=country)
        cableToLps[cable].append(lp)

    for cable, lps in cableToLps.items():
        for lp1, lp2 in
itertools.combinations(lps, 2):
            if lp1 != lp2:
                G.add_edge(lp1, lp2,
cable=cable)

    return G, lpToCountry
```

Fig 3.2 Graph Building Function [Source: Author]

D. Analysis Functions

The core of the analysis was performed using the NetworkX library. After constructing the graph, we utilized NetworkX's built-in functions to apply the primary analytical technique by finding the minimum cut sets. For a given nation, by identifying its associated landing point nodes and edges, we could then identify its minimum cut sets.
The following are the functions to find the minimum cut sets:

```
def MinCut(targetCountry,
cableToCountries):
   cutSet = set()
   for cable, countries in
cableToCountries.items():
       if targetCountry in countries
and len(countries) > 1:
           cutSet.add(cable)
   return sorted(list(cutSet))
```

Fig 3.3 Edges Minimum Cut Set Function [Source: Author]

This function's objective is to find the minimum cut set of the specified country through its edges (Cables connected through the country). The algorithm used here is a fairly simple one, as it identifies every cable that is connected to said country, and the minimum cut set is already found.
The following is what could be contained within the list returned by the function:

4 cables :
- Coral Sea Cable System (CS²)
- Hawaiki Nui 1
- Kumul Domestic Submarine Cable System
- PIPE Pacific Cable-1 (PPC-1)

By removing these 4 Cables, Papua New Guinea will not have a single cable connecting it to the outside world, practically isolating it from the whole world.
This result is in line with the figure shown below, where there are only 4 cables connecting Papua New Guinea with the outside world.

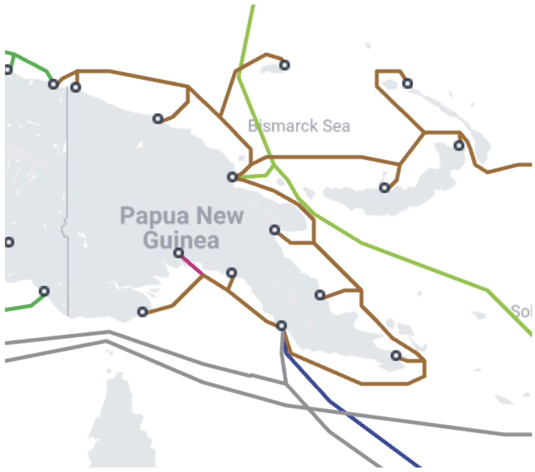

Fig 3.4 Submarine Cable Graph of Papua New Guinea (Source: https://www.submarinecablemap.com/, Accessed: Jun. 19, 2025.)

```
def internalcutNodes(targetCountry, dbData):
   cableToCountries = defaultdict(set)
   for cable, lp, country in dbData:
       cableToCountries[cable].add(country)

   internationalCables = {c for c, countries
in cableToCountries.items() if len(countries)
> 1}

   internalCutSet = set()
   for cable, lp, country in dbData:
       if country == targetCountry and cable
in internationalCables:
           internalCutSet.add(lp)

   return sorted(list(internalCutSet))
```

Fig 3.5 Internal Nodes Minimum Cut Set Function [Source: Author]

This Function returns a list of nodes that form a minimum cut set, the nodes considered here are only the nodes contained within the specified country. It works by identifying cables that are connected to multiple countries, and if such a cable is connected to a landing point, that would mean the landing point serves as a connection to the rest of the world and would therefore be added to the cut set. But by only considering internal nodes, it isn't forming a true minimum cut set.
The following is what could be contained within the returned list:

Internal Cut
Nodes to disable: 14
- Alotau, Papua New Guinea
- Arawa, Papua New Guinea
- Daru, Papua New Guinea
- Kavieng, Papua New Guinea
- Kerema, Papua New Guinea
- Kimbe, Papua New Guinea
- Kokopo, Papua New Guinea
- Lae, Papua New Guinea
- Lorengau, Papua New Guinea
- Madang, Papua New Guinea
- Popondetta, Papua New Guinea
- Port Moresby, Papua New Guinea
- Vanimo, Papua New Guinea
- Wewak, Papua New Guinea

```
def globalMinCut(G, targetCountry, lpToCountry):
   H = G.copy()
   internal = {lp for lp, country in lpToCountry.items() if country == targetCountry}
   external = {lp for lp, country in lpToCountry.items() if country != targetCountry}

   if not internal or not external or not any(nx.has_path(H, u, v) for u in internal for v in external):
       return []

   source, sink = 'SOURCE', 'SINK'
   H.add_node(source)
   H.add_node(sink)

   for node in internal: H.add_edge(source, node)
   for node in external: H.add_edge(node, sink)

   cutSet = nx.minimum_node_cut(H, source, sink)
   return sorted(list(cutSet))
```

Fig 3.6 Global Nodes Minimum Cut Set Function [Source: Author]

This function utilizes the Minimum S-T Cut problem by effectively grouping every node within the country into a single node "source" by connecting every node in the country to the node "source" and every node outside the country into the node "sink", it transform the problem into a Minimum S-T Cut problem that could quickly be solved by finding the cut set that would separate the 2 nodes "source" and "sink". The Program utilizes the minimum_node_cut function within the NetworkX library, with H being the graph, and source and sink as the 2 nodes that need to be disconnected

The following is what could be contained within the returned list:

Nodes to disable: 3
- Jayapura, Indonesia (Indonesia)
- Madang, Papua New Guinea (Papua New Guinea)
- Port Moresby, Papua New Guinea (Papua New Guinea)

This result is in line with the figure shown below, where those 3 nodes, if disabled, would disconnect Papua New Guinea from the rest of the world. The figure below is the result of choosing the 3 landing points in submarinecablemap.com.

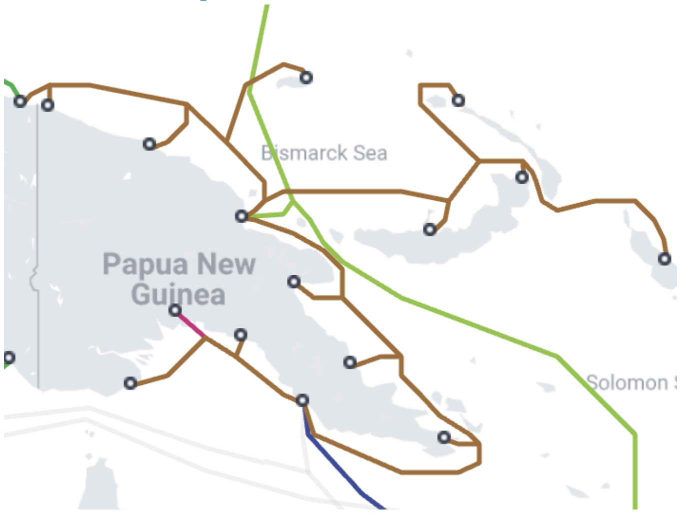

Fig 3.7 Submarine Cable Graph of Papua New Guinea from the perspective of 3 landing points (Source: https://www.submarinecablemap.com/, Accessed: Jun. 19, 2025.)

The rest of the program below are functions that are not a part of the analysis and only serve to help the user of the program use the critical functions mentioned before.

```python
def byCountry(analysisType, targetCountry, dbData, graphData):
    graph, lpToCountry, cableToCountries = graphData
    print(f"\nAnalyzing: {targetCountry}")
    print("="*50)

    if analysisType == 'cable':
        minCutCables = MinCut(targetCountry, cableToCountries)
        print(f"{len(minCutCables)} cables :")
        if not minCutCables:
            print("  None")
        else:
            for cable in minCutCables:
                print(f"  - {cable}")

    elif analysisType == 'landing_point':
        internalCut = internalcutNodes(targetCountry, dbData)
        print("Internal Cut")
        print(f"  Nodes to disable: {len(internalCut)}")
        if not internalCut:
            print(f" ")
        else:
            for node in internalCut:
                print(f"  - {node}")

        globalCut = globalMinCut(graph, targetCountry, lpToCountry)
        print(f"\nMinimum Global Cut (Includes external nodes if needed to achieve smallest possible cut)")
        print(f"  Nodes to disable: {len(globalCut)}")
        if not globalCut:
            print("  This country is already isolated or has no international connections.")
        else:
            for node in globalCut:
                nodeCountry = lpToCountry.get(node, "Unknown")
                print(f"  - {node} ({nodeCountry})")

    print("-" * 50 + "\n")

def Everything(analysisType, allCountries, dbData, graphData):
    for country in allCountries:
        byCountry(analysisType, country, dbData, graphData)
```

```python
def main():
    dbData = getData()
    if not dbData:
        return

    graph, lpToCountry = buildgraph(dbData)
    cableToCountries = defaultdict(set)
    for cable, lp, country in dbData:
        cableToCountries[cable].add(country)
    allCountries = sorted(list(set(lpToCountry.values())))
    graphData = (graph, lpToCountry, cableToCountries)
```

```python
    while True:
        print("1. Edge Cut (Cables)")
        print("2. Node Cut (Landing Points)")
        print("3. Exit")
        choice = input()

        if choice == '1':
            analysisType = 'cable'
        elif choice == '2':
            analysisType = 'landing_point'
        elif choice == '3':
            print("Exiting.")
            break
        else:
            print("Please enter 1, 2, or 3.")
            continue

        while True:
            print("\n")
            print("1. Single country")
            print("2. List all countries")
            print("3. Back to main menu")

            subChoice = input("Enter your choice (1, 2, or 3): ")

            if subChoice == '1':
                countryToCheck = input(f"\nCountry Name: ")
                if countryToCheck not in allCountries:
                    print(f"\nCountry '{countryToCheck}' not found.")
                else:
                    byCountry(analysisType, countryToCheck, dbData, graphData)
            elif subChoice == '2':
                Everything(analysisType, allCountries, dbData, graphData)
            elif subChoice == '3':
                break
            else:
                print("Invalid choice.")

if __name__ == "__main__":
    main()
```

Fig 3.8 Other Functions [Source: Author]

The functions (see Fig. 3.7) help the user determine whether the output will be based on either nodes or edges, and it also prints out the list that will be returned by the analysis functions.

IV. Results

This chapter shows the results found through the methods in Chapter III. The primary objective is to demonstrate the use of minimum cut set to identify and determine the vulnerabilities of a nation's internet connectivity. This analysis focuses on a specific case study: Papua New Guinea. The results are in three parts, examining the nation's vulnerability from the perspective of the connecting cables (edge cut sets), its internal landing points, or a global set of landing points (node cut sets).

A. *Case Study: Papua New Guinea*

Papua New Guinea was selected as a random case study, featuring a low number of international connections. This analysis utilizes the functions MinCut, internalcutNodes, and globalMinCut to determine the specific nodes and edges that are critical to Papua New Guinea's connectivity.

B. *Edge-Based Vulnerability: Cable Cut Set*

This Analysis identifies the minimum set of edges whose removal would disconnect Papua New Guinea from the rest of the world using the MinCut function. The analysis identified a minimum edge cut consisting of four cables.

```
4 cables :
 - Coral Sea Cable System (CS²)
 - Hawaiki Nui 1
 - Kumul Domestic Submarine Cable System
 - PIPE Pacific Cable-1 (PPC-1)
```

The results show that simultaneous failure of these four cables would effectively isolate Papua New Guinea from the global internet. This finding is visually confirmed through Figure 3.4.

C. *Node-Based Vulnerability: Nodes Cut Set*

Differing with edge-based analysis, Node-Based analysis focuses on the landing points instead of the cables, often a more practical perspective due to landing points being points of convergence, where it could be holding multiple cables connecting a country to the global network.

Due to the local nature of the internalcutSet function, we will be skipping its usage and utilizing the more accurate globalMinCut function, which will provide us with the true minimum cut set.

```
Nodes to disable: 3
 - Jayapura, Indonesia (Indonesia)
 - Madang, Papua New Guinea (Papua New Guinea)
 - Port Moresby, Papua New Guinea (Papua New Guinea)
```

This is the most important finding of this analysis, it reveals that the most efficient way to disconnect Papua New Guinea also includes a landing point of a neighboring country. This highlights a crucial dependency towards a foreign country, a hallmark of the modern world.

D. *Discussions*

The analysis of Papua New Guinea demonstrates the utility of the graph-based approach. By comparing the edge-based and node-based approaches, a more detailed understanding of vulnerability is achieved. The analysis reveals that an adversary would need to target four distinct cable systems, but only three specific landing points, to achieve the same outcome of isolating the nation.

The discovery that the true minimum node cut includes an external node (Jayapura, Indonesia) underscores that a nation's digital sovereignty isn't just a domestic issue but is linked to the stability of the international network.

V. Conclusion

This study successfully demonstrated the application of a graph-based method for identifying the vulnerabilities within the global submarine cable network. By modelling the network as a graph with landing points as nodes and cables as edges, a minimum cut set analysis proved to be effective at finding critical chokepoints or critical sets of nodes of specific countries that, if disabled, would paralyze a country's digital world. The critical finding that a critical node of a country could be located outside of said country also highlighted a reason for countries to uphold stable international relations.

In conclusion, this paper confirms that graph theory isn't just a theoretical tool but a powerful lens through which vulnerabilities of our modern world can be exposed.

VI. Video Link at YouTube

A video explanation and demonstration of the program used within this paper is available at [Link]

VII. Acknowledgment

The author would like to express sincere gratitude and appreciation to Dr. Ir. Rinaldi Munir, M.T., as the lecturer for the IF1220 Discrete Math Class, for the guidance and knowledge provided throughout the whole semester.

VIII. Appendix

The complete source code of this paper, including Data Acquisition and analysis functions, can be found at [Github Link]

References

[1] S. Uqba, "Yemen in Focus: Cable damage wipes out 80% of country's internet connectivity," *The New Arab*, Jan. 15, 2020. [Online]. Available: https://www.newarab.com/analysis/yemen-focus-cable-damage-wipes-out-internet-connectivity. [Accessed: Jun. 19, 2025].

[2] R. Munir, "Graf (Bag. 1)", 2024. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf [Accessed: Jun. 19, 2025].

[3] "Minimum Cut in a Directed Graph," GeeksforGeeks, May 15, 2024. [Online]. Available: https://www.geeksforgeeks.org/dsa/minimum-cut-in-a-directed-graph/. [Accessed: Jun. 19, 2025].

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 1 Juni 2025

Ahmad Fauzan Putra dan 13524141