# Sending messages through a network

Network Analysis - Assignment 2

Enrico Pezzano

June 2025

## 1 Introduction

In this report, I will analyze the diffusion of messages through a network of political blogs, the same presented in the previous assignment. This time, the goal is to study how the network structure influences the spreading of information, and how malicious nodes can affect the diffusion process.

In the following sections, I will describe the methods used to simulate the diffusion process, via the threshold model, on 3 different network topologies: Erdős-Rényi, Watts-Strogatz, and Barabási-Albert, plus the said polblogs network from the previous assignment. The first 3 models have a relative small number of nodes ($n = 100$) in order to keep the firsts executions fast, while the polblogs network is larger ($n = 1224$) to better represent a real-world scenario.

I wanted to keep the images in the report small, so that we can see the effects of the contagion process at a glance, without having to scroll too much. Sadly, this means that the images are not very readable, especially for the heatmaps, which are quite small and hard to interpret. However, since this is a digital report, we can zoom in on the images to see the details more clearly. All the images are available in the appendix, in order to keep the report readable, and I will refer to them in the text.

## 2 Methodology

First of all, I generated the three network topologies cited in the introduction, using the `networkx` library in Python and a custom script.

```
def gen_network(model: str, n: int, **kwargs) -> nx.Graph:
    if model == 'Erd\H{o}s-R\'enyi':
        p = kwargs.get('p', 0.1)
        return nx.erdos_renyi_graph(n, p)
    elif model == 'Watts-Strogatz':
        k = kwargs.get('k', 4)
        p = kwargs.get('p', 0.1)
        return nx.watts_strogatz_graph(n, k, p)
```

```
    elif model == 'Barab\'asi-Albert':
        m = kwargs.get('m', 2)
        return nx.barabasi_albert_graph(n, m)
    elif model == 'polblogs':
        return nx.read_edgelist(POLBLOGS_PATH,
                                create_using=nx.Graph(),
                                nodetype=int,
                                comments='%')
    else:
       raise ValueError(f"Unknown model: {model}")
```

Listing 1: Network generation script

In my simulations, I ran various experiments with the help of a `for loop` to iterate over different parameters, i.e., the default message vector dimension (100), the default thresholds $\theta \in \{0.1, 0.3, 0.5\}$, the default percentage of malicious nodes $k_{\text{mal}} \in \{0.01, 0.05, 0.1\}$, the default percentage of initially honest nodes $k_{\text{seed}} \in \{0.01, 0.05, 0.1\}$, and of course the network models previously cited, with the addition of a special one `polblogs-attack`, where I sampled the malicious nodes from the hubs, i.e. the nodes with the highest degree.

Each experiment records the final adoption state of nodes (original, tampered, or none), visualizes the network, and computes cosine similarity matrices for final message vectors.

## 2.1 Cosine Similarity Heatmaps

The cosine similarity heatmaps plots, instead, are a visual representation of how similar the final messages received by nodes are with respect to each other.

In order to build each heatmap, I first assigned a vector to all nodes that encodes the type of message they adopted: if a node received the original message, it is represented by a vector of all ones, i.e. the original message; else, if it received a tampered message, I used a `noisy` version of the original vector; finally, if the node did not receive any message, it is encoded as a vector of all zeros. In the end, I computed the pairwise cosine similarity between all node vectors, resulting in the relative heatmap matrices.

These heatmaps show where the rows and columns both correspond to nodes (sorted by their IDs), and the cell at position $(i, j)$ encodes the similarity between the messages adopted by nodes $i$ and $j$. The color of each cell reflects the similarity value: bright yellow (close to 1) means the messages are very similar or identical; dark purple (close to 0) tells us that there is little or zero similarity, often because at least one of the nodes did not adopt a message; and intermediate shades such as green or blue suggest partial similarity, usually due to the noise of the tampering nodes.

Finally, I saved all the generated results in `.png` files and, in particular, I used the `spring_layout` and `tight_layout` functions to generate the layouts of the graphs, with their relative information about the state of the nodes and the messages that they are sending, and the heatmap for the cosine-similarity of the final messages for each combination of the parameters.

# 3 Three Network Models

In this section, I will briefly discuss the results of the contagion process on the three network models previously cited, mainly for reference and comparison between them and the `polblogs` network, which, of course, is the main focus of this report.

Note that all the images shows the **final state** of the contagion process, after all the nodes have adopted a message or not, and the colors represent different states of the nodes.

In particular, the red nodes are the malicious ones, i.e. that tampered the original message, the nodes with a bold black border are the initial seed nodes (the ones that started the diffusion process), the blue nodes are the honest nodes that adopted a tampered variant of the original message, and the gray nodes are the ones that did not adopt any message at all -even if we will see that this will almost never happen.

## 3.1 Erdős-Rényi Contagions

In the Erdős-Rényi graphs ($n = 100$, $p = 0.05$), links are uniformly random, so the diffusion is more homogeneous than in the small-world model but less clustered. We can see that with $\theta = 0.1$, the diffusion cascade rapidly engulfs the network, and as the number of malicious nodes grows, the tampered message (`orange` nodes) spreads more quickly than the original one (`blue`), leading to a near-complete takeover of the network.

Then, at $\theta = 0.3$, the diffusion process becomes more weak, we can see that with 1 and 5 initial seeds, a lot of the network never adopts any message (`gray` nodes), while with 10 initial seeds, the diffusion is complete. The tampered messages manage to spread partially only when `seeds` $= 10$ and $k_{\text{mal}} = 10$, dominating around half of the network.

Finally, at $\theta = 0.5$, the diffusion process is very weak, and we can see that only a handful of nodes become `blue` and the tampered messages never take over.

**Erdős-Rényi Cosine-Similarity Heatmaps** We can see that the cosine similarity heatmaps for the Erdős-Rényi model highlight how message diffusion and the malicious tampering evolve under varying conditions. When the threshold $\theta$ is low (e.g., $\theta = 0.1$), and there are few or no malicious nodes, nearly all nodes adopt a similar message—resulting in a uniformly bright yellow heatmap. As the number of malicious nodes increases, the matrix begins to show slight variation, with the emergence of intermediate shades reflecting partial similarity due to tampered content. Instead, when the threshold is high (e.g., $\theta = 0.5$), diffusion is severely limited: only a small number of nodes are activated, leading to isolated bright patches within an almost completely purple matrix, which indicates that most nodes did not adopt any message at all.

### 3.2 Watts-Strogatz Contagions

After the Erdős-Rényi network, I ran the contagion process on the Watts-Strogatz model, generated with $k = 4$ and $p = 0.1$.

Compared to the Erdős-Rényi model, the contagion process in the Watts-Strogatz network exhibits distinct dynamics due to its underlying small-world structure. Erdős-Rényi graphs are characterized by a uniform random edge distribution, but Watts-Strogatz networks introduce local clustering and short path lengths, leading to more localized diffusion; indeed, the message propagation begins within tightly connected clusters. This results in more resilient early diffusion, particularly when the number of seed nodes is small. Even at higher threshold values (e.g., $\theta = 0.3$), the local cohesion of neighborhoods often allows the original message to spread more effectively than in the Erdős-Rényi model.

Another key difference is the reduced sensitivity to malicious nodes, i.e. in Erdős-Rényi networks, tampered messages often spread quickly due to the absence of strong community boundaries, but in Watts-Strogatz networks there is some containment of tampering within specific neighborhoods, especially when the number of malicious nodes is low. This leads to a more heterogeneous diffusion pattern in Watts-Strogatz, as opposed to the more uniform or globally fragmented diffusion previous cited.

**Watts-Strogatz Cosine-Similarity Heatmaps** Then, the heatmaps under the Watts-Strogatz model showed a complicated relationship between local clustering and interference in messages. Unlike the Erdős-Rényi model, the Watts-Strogatz heatmaps display clearly demarcated regions of high similarity even in the presence of moderate and indeed high-level disturbances, which suggests that the contagion process will often be strongly localized within neighborhood areas, as we can see in the figures in the appendix.

Interestingly, partial diffusion became more evident as $k_{\text{seed}}$ increased, starting to show bright yellow blocks along the diagonal, i.e. block-diagonal pattern, indicating that many nodes adopted similar messages, but with some localized noise introduced by malicious nodes.

### 3.3 Barabási-Albert Contagions

Finally, the Barabási-Albert simulations showed to be the most vulnerable due to the presence of high-degree hubs, so that even a single seed node can trigger large cascades, especially under low threshold.

However, these same hubs also serve as critical points of failure: when compromised, they facilitate extensive tampering, visibly reducing message integrity; indeed, the Barabási-Albert network produced highly polarized diffusion outcomes (either near-complete spread or near-total corruption) depending or even splitted in half diffusion (tampered vs original) depending largely on whether hubs were initial seeds or malicious nodes.

**Barabási-Albert Cosine-Similarity Heatmaps** Finally, the heatmaps for the Barabási-Albert model showed high contrast, with well-defined clusters of high similarity surrounded by blurry or distorted regions. This is sintomatic of the central role played by hubs, thus, the heatmaps showed a block-diagonal pattern.

# 4 Polblogs Contagions

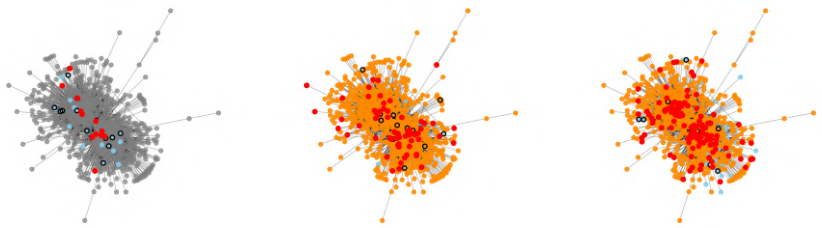
Firstly, I removed from the `polblogs` network the only disconnected component, which was formed by only two nodes, since it was not relevant for the contagion process nor interesting for the analysis in this report.

Then, I ran the contagion process with the same parameters as before, but this time the results showed a more resilient diffusion pattern.

Indeed, the `polblogs` network is characterized by a strong community structure, which plays a crucial role in shaping how information spreads.

We can see that, at low threshold ($\theta = 0.1$) and $k_{\text{seed}} = 1\%$, every node in the network adopts a message (either tampered or original), but the malicious nodes have achieved the complete diffusion of their tampered message, leaving only a handful of nodes with still the original message. This is caused mainly by the low threshold, which allows the diffusion process to be very effective, even with a small number of initial seed nodes.

**1% Initial seed**

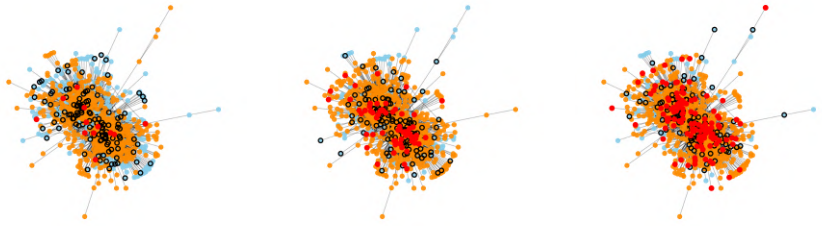


1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**



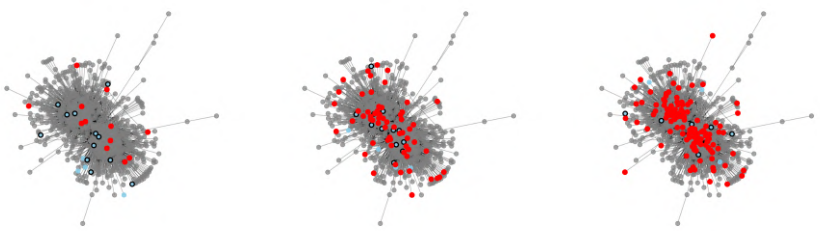
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 1: Polblogs contagions at $\theta = 0.1$.

As $\theta$ increases to 0.3, (as shown in the figure below) the message adoption becomes much more difficult, and therefore confined to the political faction of each initial seed; only a handful of adjacent communities—those with the strongest inter-group ties—see any spread at all. It is worth noting that at this threshold, the original message is still able to spread, but only within the executions with $k_{\text{seed}} = 10\%$ and in some rare cases with $k_{\text{seed}} = 5\%$ with

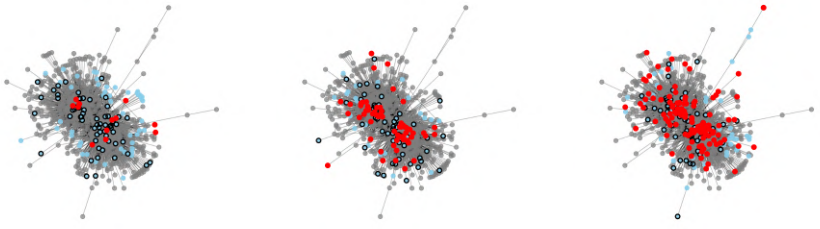
$k_{\text{mal}} = 10\%$, otherwise the majority of the nodes will never receive any message.

**1% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 2: Polblogs contagions at $\theta = 0.3$.

At the highest threshold ($\theta = 0.5$) (figure below), diffusion stalls almost entirely, with only immediate neighbors of the initial seeds ever adopting the message, obviously in this case the threshold is too high for the contagion process to be effective.

**1% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

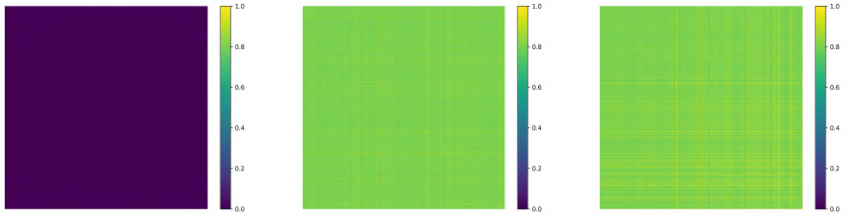
Table 3: Polblogs contagions at $\theta = 0.5$.

**PB Cosine-Similarity Heatmaps** As seen before in the synthetic networks, also in this case the cosine similarity heatmaps reflected the results of the contagion process.

More in detail, the `polblogs` cosine-similarity matrices exhibit what it was already hypothesized: resilience between communities, leading to "stubborn" clusters of nodes that adopt nearly identical messages within their own political factions, while inter-community diffusion remains minimal.

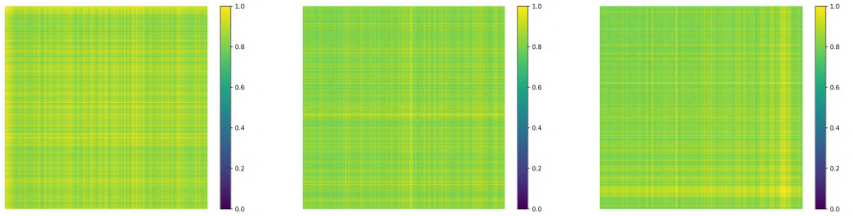
In particular, the `polblogs` cosine-similarity matrices exhibit a very dense block-diagonal structure when $\theta = 0.1$, which is a clear indication of the strong community structure of the network. Indeed, we can see in the figure below that the majority of the matrices is green, due to the tampered messages.

**1% Initial seed**



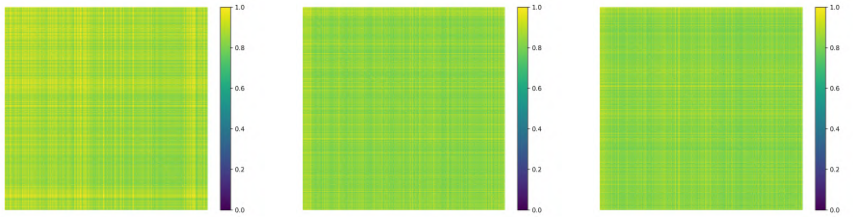
1% Malicious node 5% Malicious nodes 10% Malicious nodes

**5% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

**10% Initial seed**



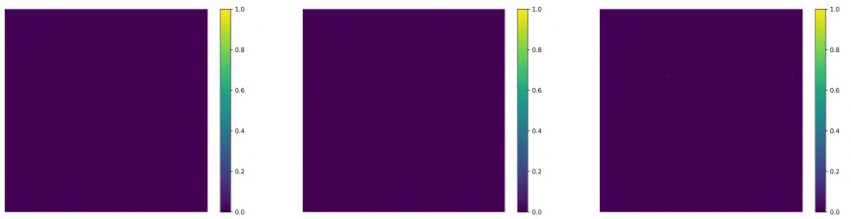
1% Malicious node 5% Malicious nodes 10% Malicious nodes

Figure 1: Polblogs heatmap at $\theta = 0.1$.

As the threshold increases to $\theta > 0.1$, the heatmaps reflect the increasing difficulty of message adoption by the inner-communities, with all of the matrices fading to dark purple, with only a few yellow isolated zones when $k_{\text{mal}}$ grows,

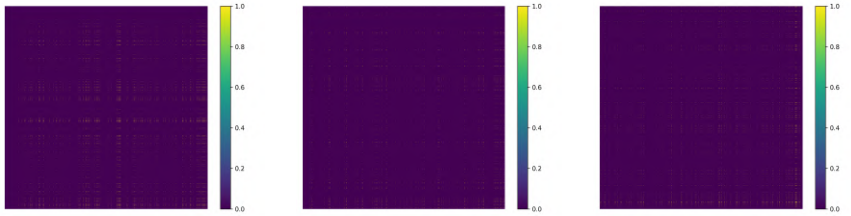
indicating that many nodes never adopted any message at all, and showcasing the strong community structure of the network and the innate "stubbornness" and isolation of the political factions.

**1% Initial seed**



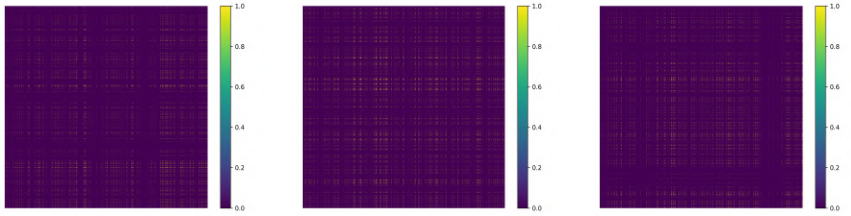
1% Malicious node 5% Malicious nodes 10% Malicious nodes

**5% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

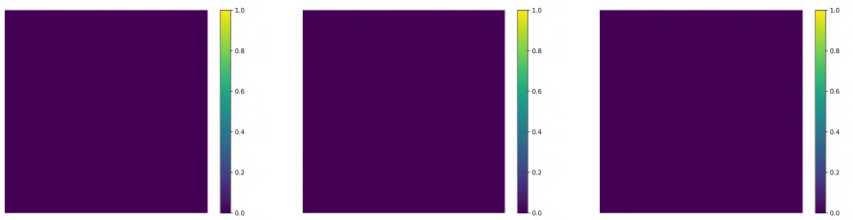
**10% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

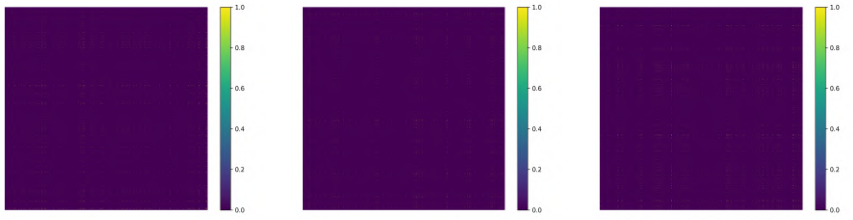
Figure 2: Polblogs heatmap at $\theta = 0.3$.

**1% Initial seed**



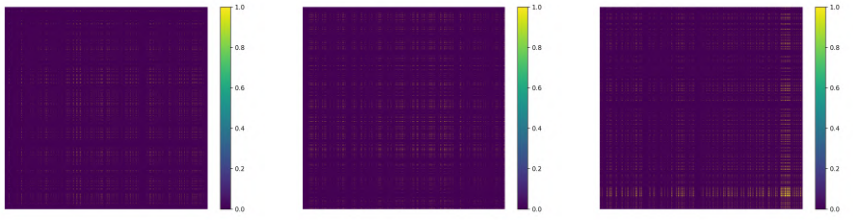
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Figure 3: Polblogs heatmap at $\theta = 0.5$.

# 5 How does the network structure influence the spreading of the information?

In the `polblogs` graph, the strong modular community structure is the primary factor shaping diffusion. as we have seen in the previous assignment, political blogs are grouped into tightly knit clusters —each with high internal density and

relatively few links to other clusters—so once a message (original or tampered) enters a community it quickly reaches most of its members, but rarely escapes that community's boundary. The sparse "bridge" links between factions act as choke points: unless multiple such links carry the message (which requires either a low threshold or multiple seeds), inter-community transmission stalls.

Moreover, key hub-like blogs within each cluster amplify intra-community spread, but their influence remains largely local. Even when malicious nodes lie on high-degree blogs, their altered content propagates deeply only inside their home cluster, as we will see in the next sections. This combination of high clustering and few long-range ties explains why the threshold experiments on the `polblogs` network did not returned a single, global cascade.

# 6 With a given threshold in the network, how many malicious nodes are necessary to have more than half of the messages corrupted?

In order to measure how many malicious nodes are required to corrupt the majority of the `polblogs` network, I ran all the possible combinations of the parameters, i.e. the thresholds $\theta \in \{0.1, 0.3, 0.5\}$, the percentage of malicious nodes $k_{\text{mal}} \in \{0.01, 0.05, 0.1\}$, and the percentage of initial seed nodes $k_{\text{seed}} \in \{0.01, 0.05, 0.1\}$.

As we already saw in the previous section, the contagion process, in the case of the `polblogs` network, dependes strongly by the value of the threshold $\theta$, rather than the number of malicious nodes, which is a common feature of many real-world networks, e.g. fake news spreading on social media, or in political blogs, as in this case.

Indeed, the results of the contagion process show that, under low threshold ($\theta = 0.1$), regardless of the number of malicious nodes, the majority of the network will adopt a tampered message, even with only 1% of malicious nodes and 1% of initial seeds. However, as the threshold increases, the number of malicious nodes required to corrupt the majority of the network also increases, as we can see in the heatmaps with higher $k_{\text{mal}}$ values, which shows some yellow or green blocks.

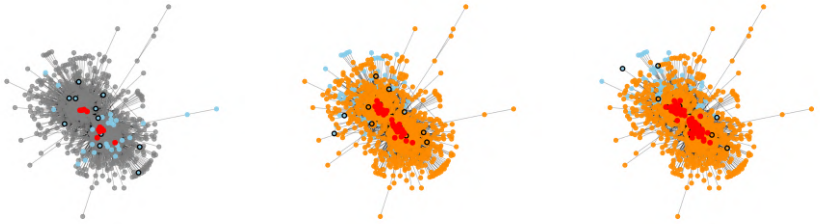
# 7 What happens if instead of selecting malicious nodes at random, we target specific nodes in the network?

In order to answer this question, I "attacked" the `polblogs` network by sampling the malicious nodes from the nodes with the highest degree, rather than randomly.

This is a common strategy in real-world scenarios, where malicious actors often target influential nodes to maximize their impact.

In this case, the resultant plots shows that for a low-threshold regime ($\theta = 0.1$), the contagion process in more or less the same, regardless of this tiny set of high-degree nodes that try to hijack the diffusion, due to the fact that the threshold is so low that the contagion process is very effective. See the figure below for the contagion process with malicious hubs at $\theta = 0.1$.

**1% Initial seed**

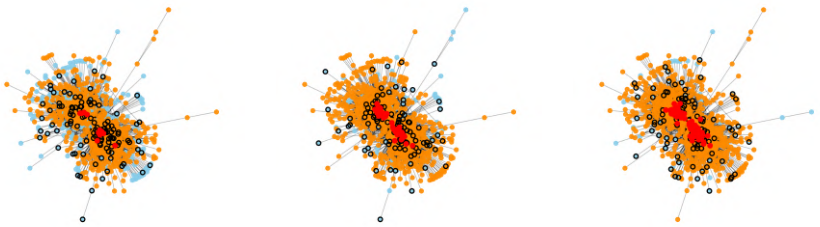


1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

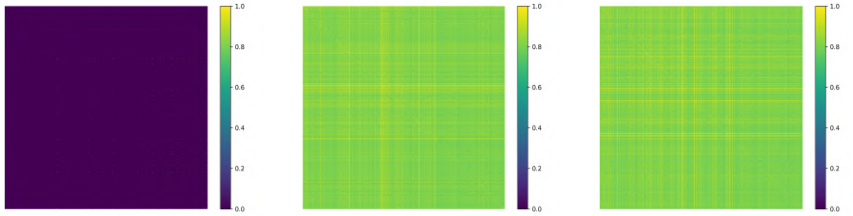


1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 4: Polblogs contagions with malicious hubs at $\theta = 0.1$.

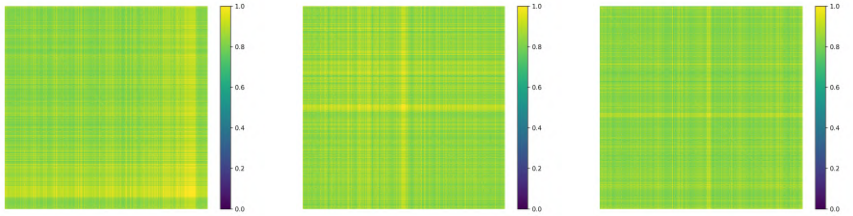
The relative heatmap at $\theta = 0.1$, shown below, reflects the same results as in the previous section, but every block-diagonal cluster is slightly more green, indicating clearly that the malicious hubs have managed to tamper the original message more effectively than random malicious nodes.

**1% Initial seed**



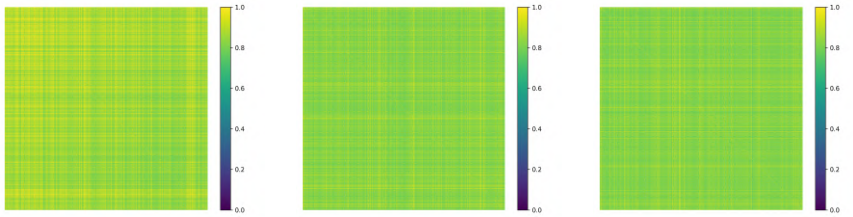
1% Malicious node 5% Malicious nodes 10% Malicious nodes

**5% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

**10% Initial seed**

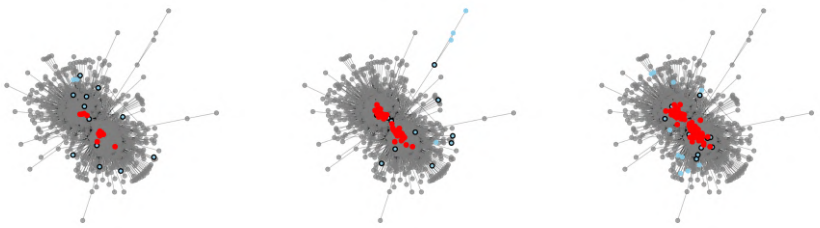


1% Malicious node 5% Malicious nodes 10% Malicious nodes

Figure 4: Polblogs with malicious hubs heatmap at $\theta = 0.1$.

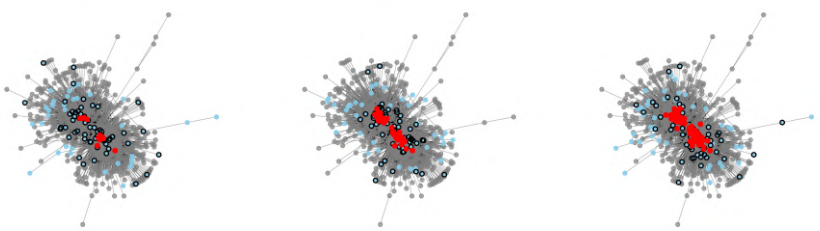
When the threshold is increased to $\theta = 0.3$ or $\theta = 0.5$, the contagion process becomes more resilient also to the malicious hubs, as we can see in the figures below.

**1% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



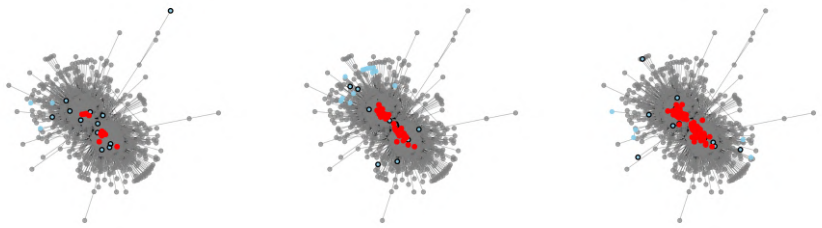
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

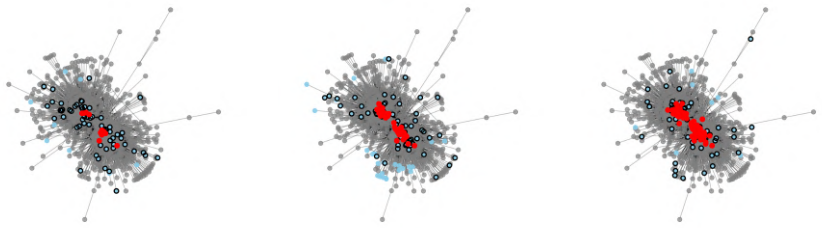
Table 5: Polblogs contagions with malicious hubs at $\theta = 0.3$.

**1% Initial seed**



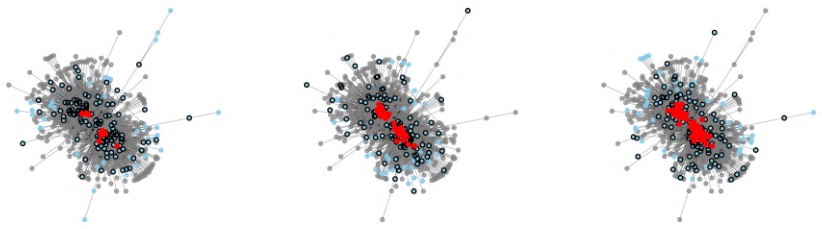
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

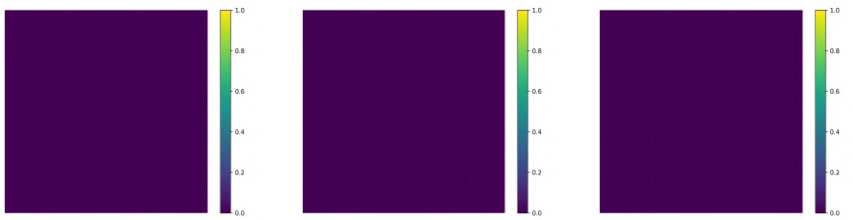


1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 6: Polblogs contagions with malicious hubs at $\theta = 0.5$.

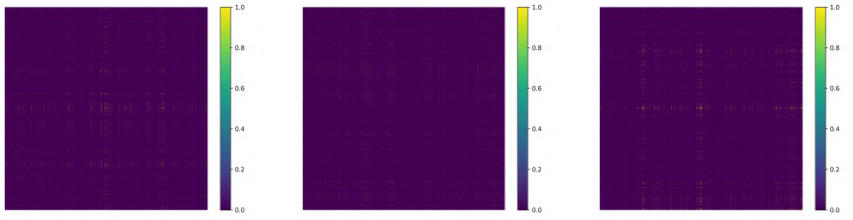
However, both of their relative heatmaps, shown below, even though they are still block-diagonal, we can observe a little more colored blocks, i.e. more yellow and green blocks, which indicates that the malicious hubs have managed to tamper the original message more effectively than random malicious nodes, but only when the percentage of malicious nodes is high enough.

**1% Initial seed**



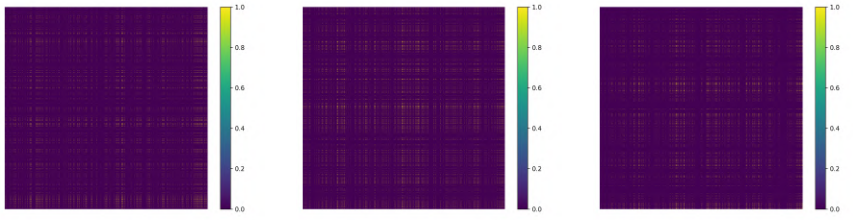
1% Malicious node 5% Malicious nodes 10% Malicious nodes

**5% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

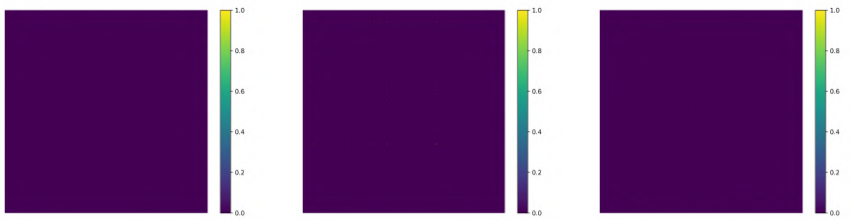
**10% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

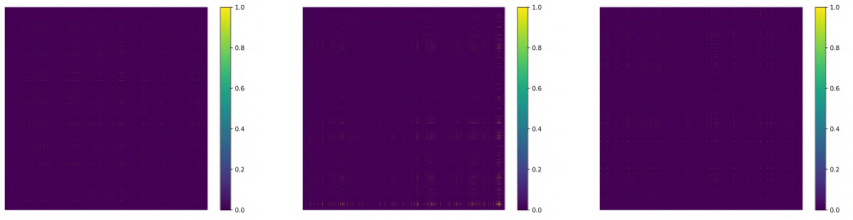
Figure 5: Polblogs with malicious hubs heatmap at $\theta = 0.3$.

**1% Initial seed**



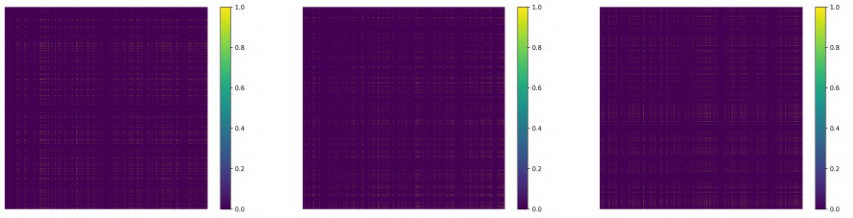
1% Malicious node 5% Malicious nodes 10% Malicious nodes

**5% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

**10% Initial seed**



1% Malicious node 5% Malicious nodes 10% Malicious nodes

Figure 6: Polblogs with malicious hubs heatmap at $\theta = 0.5$.

In summary, these results taken together, tell us that targeting the handful of highest-degree nodes is a little more destructive than random attacks—and while increasing individual thresholds can blunt this effect, it cannot fully immunize a heterogeneous network against a determined hubs-only campaign. The real-world explanation stands in the nature of this network, where there are a lot of hubs with a not so high degree, reflecting the isolation and clustering inside the two main political sides, resulting in a very strong community structure, which

is the main factor that shapes the diffusion of messages in this network, in a typical political-blogs fashion.

# 8 Conclusions

In this report, I have explored the processes by which information spreads over a variety of network topologies, ranging from the randomly linked structures typical of Erdős-Rényi graphs to the densely interconnected clusters represented in Watts-Strogatz models, and the hub-and-spoke topology found in Barabási-Albert networks, as well as the empirical `polblogs` network selected for my assignments.

The simulations, show that both the quality and extent of diffusion are influenced simultaneously by the adoption thresholds of the individual nodes and the underlying structure of the network. If the threshold is low, even a small set of initial seeds will initiate fast, system-wide cascades; but as the threshold rises, the diffusion process usually stalls, breaking into isolated clusters that rarely merge into a large-scale epidemic, or even not spreading any type of message (tampered or original), leading to a mostly gray graph.

Additionally, not only is the rate at which information spreads predetermined by the topology of the network, but so is system resiliency to specific attacks. Scale-free networks, characterized by extremely well-linked hubs, have the ability to carry both positive and negative information to comparably great effect. In small-world graphs, characterized by intense local interlinking and occasional long-distance linking, it takes an even greater widespread initiation to achieve even an equal level of reach. In the `polblogs` network, the experiments showed that the network is obviously vulnerable to cascades of malicious tampering for very low thresholds, but as the threshold increases, the diffusion process becomes more resilient to malicious nodes, even when they are sampled from the hubs, thanks to the innate isolation and clustering of the political factions, which is the main factor that shapes the diffusion of messages in this network. Indeed, even a targeted attack on the hubs, does not guarantee that the tampered message will spread throughout the network, as we saw in the previous sections.

It is clear that the inner diffusion and clustering isolation in political blogs is a dangerous fact, as fake news, misinformation, or even extremists and radicalizing content can spread rapidly within a community, while remaining largely contained from the outside. This is a common feature of many real-world networks, such as social media, where the strong community structure and the clustering of nodes can lead to the rapid spread of misinformation, while also making it difficult to contain or stop it.

For some future implementations and improvements, it could be interesting to run the experiments having every node at an arbitrary or random threshold. Another implementation could include a tentative of recovery of the original message by the honest nodes, if the network has been tampered with within a certain percentage of the total number of nodes.

# A Tables and Heatmaps

As mentioned in the sections before, for the sake of space, I will include here all the tables resultant from my `Python` experiments, which visually summarize the contagion process for each network model and threshold, in addition with their relative heatmaps.

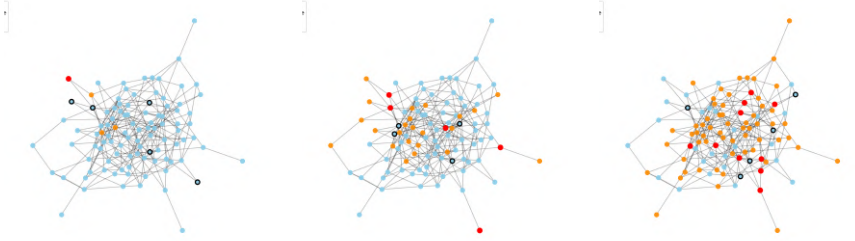
Additionally, I included the output of the `Python` script in the `results.txt` file.

## A.1 Erdős-Rényi Contagion Tables and Heatmaps

**1% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



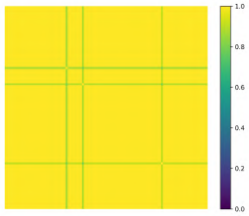
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

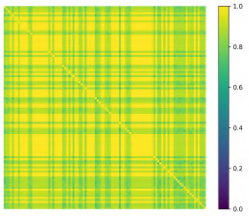
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 7: Erdős-Rényi contagions at $\theta = 0.1$.

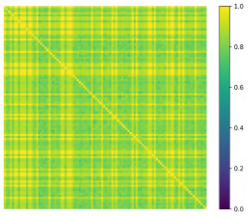
**1% Initial seed**



1% Malicious node

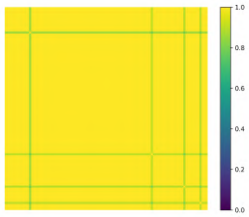


5% Malicious nodes

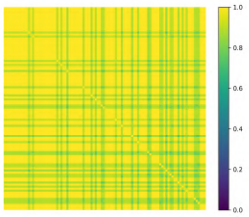


10% Malicious nodes

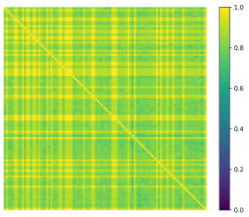
**5% Initial seed**



1% Malicious node

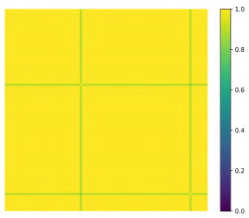


5% Malicious nodes

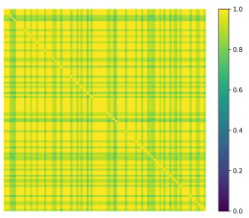


10% Malicious nodes

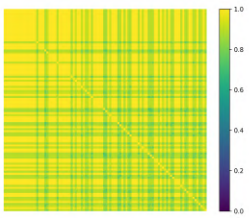
**10% Initial seed**



1% Malicious node



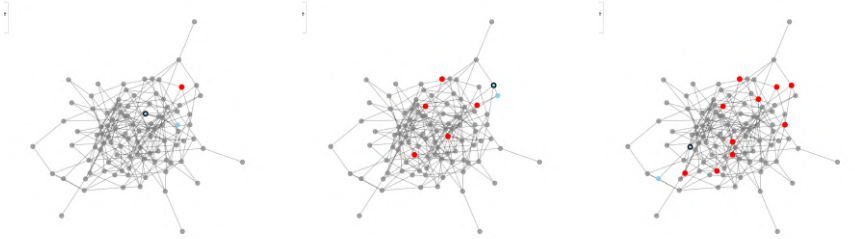
5% Malicious nodes



10% Malicious nodes

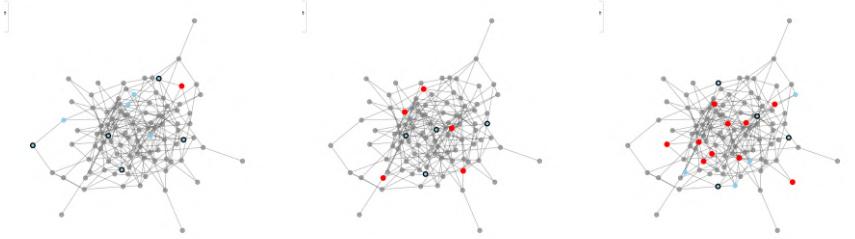
Figure 7: Erdős-Rényi heatmap at $\theta = 0.1$.

**1% Initial seed**



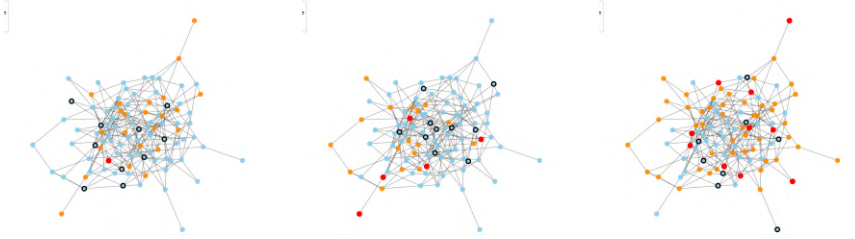
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

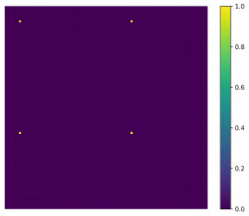
**10% Initial seed**



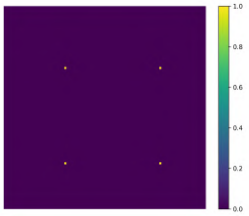
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 8: Erdős-Rényi contagions at $\theta = 0.3$.

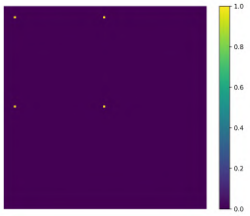
**1% Initial seed**



1% Malicious node

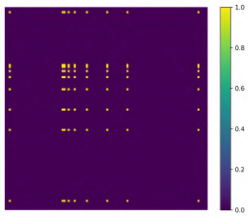


5% Malicious nodes

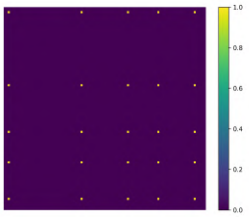


10% Malicious nodes

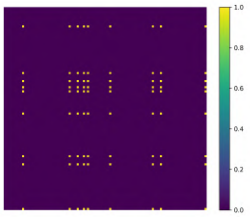
**5% Initial seed**



1% Malicious node

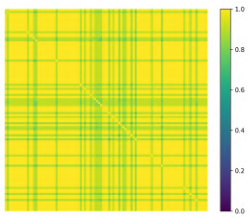


5% Malicious nodes

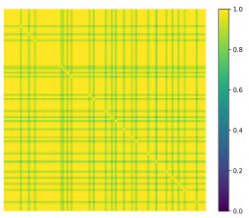


10% Malicious nodes

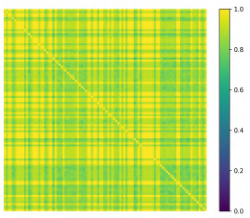
**10% Initial seed**



1% Malicious node



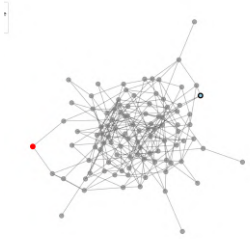
5% Malicious nodes



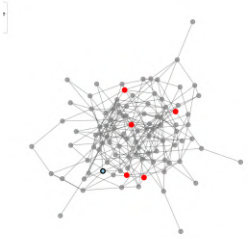
10% Malicious nodes

Figure 8: Erdős-Rényi heatmap at $\theta = 0.3$.

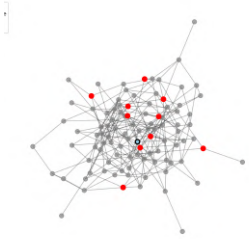
**1% Initial seed**



1% Malicious node

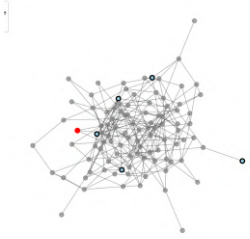


5% Malicious nodes

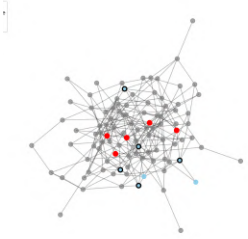


10% Malicious nodes

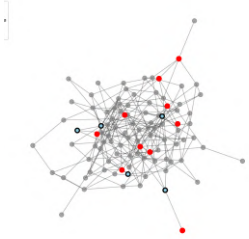
**5% Initial seed**



1% Malicious node



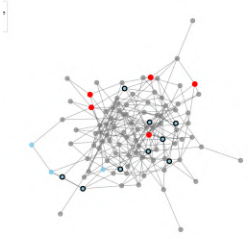
5% Malicious nodes



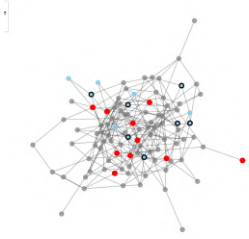
10% Malicious nodes

**10% Initial seed**

1% Malicious node



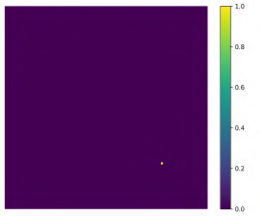
5% Malicious nodes



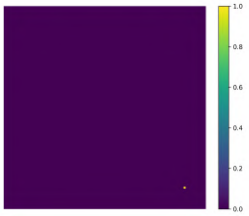
10% Malicious nodes

Table 9: Erdős-Rényi contagions at $\theta = 0.5$.

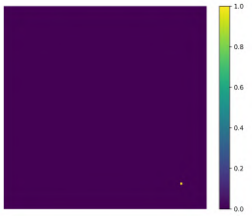
**1% Initial seed**



1% Malicious node

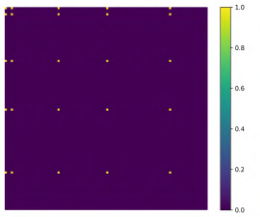


5% Malicious nodes

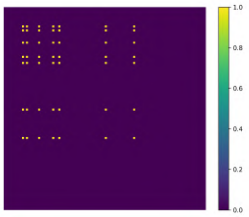


10% Malicious nodes

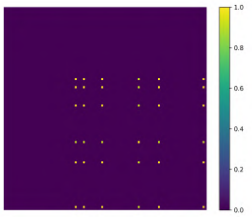
**5% Initial seed**



1% Malicious node

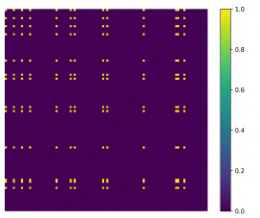


5% Malicious nodes

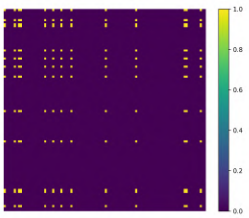


10% Malicious nodes

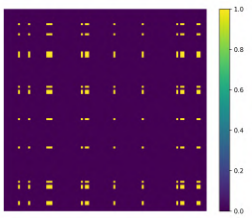
**10% Initial seed**



1% Malicious node



5% Malicious nodes

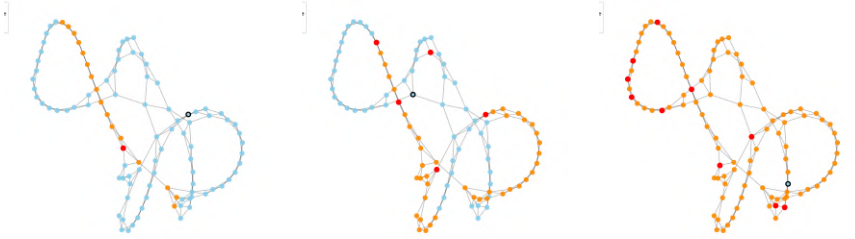


10% Malicious nodes

Figure 9: Erdős-Rényi heatmap at $\theta = 0.5$.

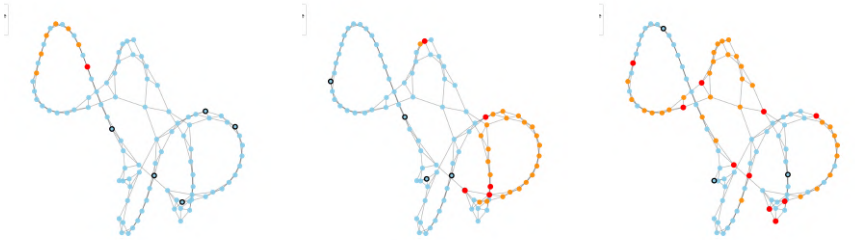
## A.2 Watts-Strogatz Contagion Tables and Heatmaps

**1% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



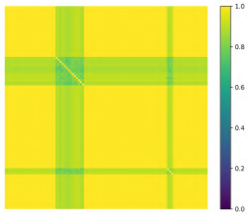
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**10% Initial seed**

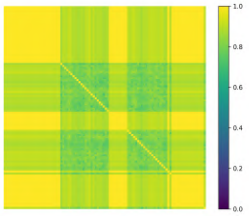
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 10: Watts-Strogatz contagions at $\theta = 0.1$.

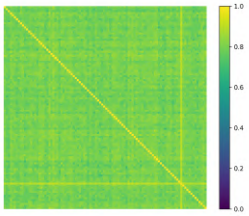
**1% Initial seed**



1% Malicious node

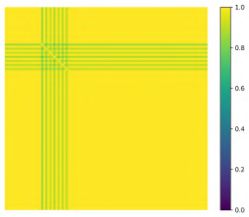


5% Malicious nodes

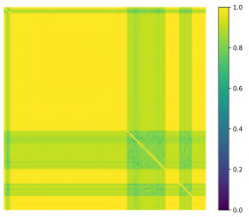


10% Malicious nodes

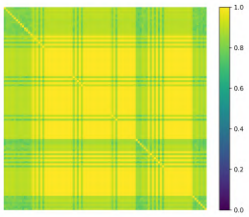
**5% Initial seed**



1% Malicious node

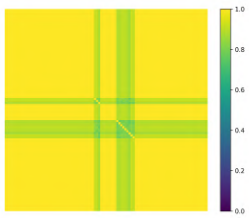


5% Malicious nodes

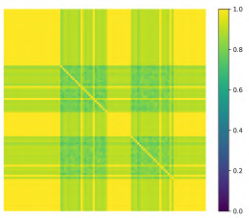


10% Malicious nodes

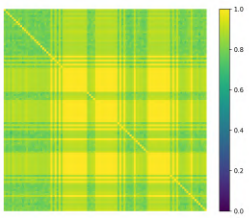
**10% Initial seed**



1% Malicious node



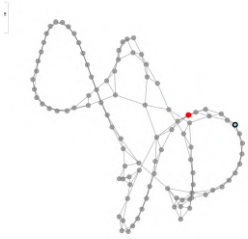
5% Malicious nodes



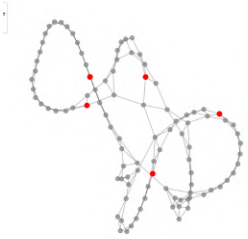
10% Malicious nodes

Figure 10: Watts-Strogatz heatmap at $\theta = 0.1$.

**1% Initial seed**



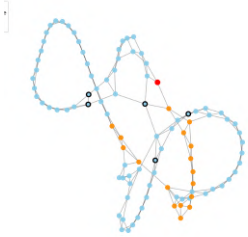
1% Malicious node



5% Malicious nodes

10% Malicious nodes

**5% Initial seed**

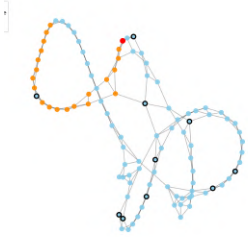


1% Malicious node

5% Malicious nodes

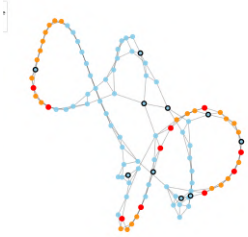
10% Malicious nodes

**10% Initial seed**



1% Malicious node

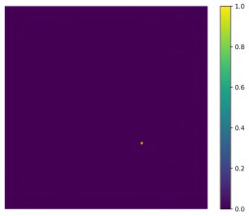
5% Malicious nodes



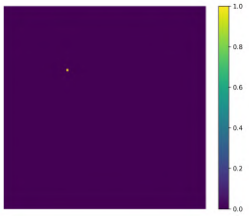
10% Malicious nodes

Table 11: Watts-Strogatz contagions at $\theta = 0.3$.

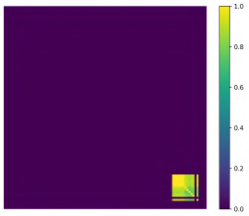
**1% Initial seed**



1% Malicious node

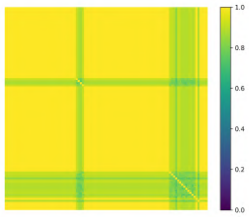


5% Malicious nodes

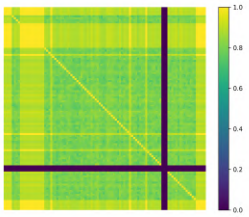


10% Malicious nodes

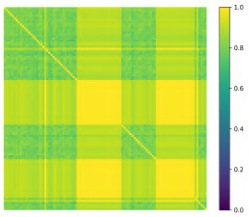
**5% Initial seed**



1% Malicious node

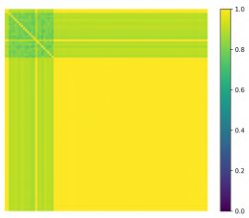


5% Malicious nodes

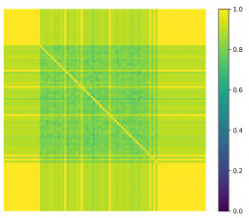


10% Malicious nodes

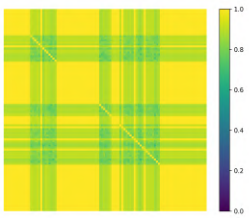
**10% Initial seed**



1% Malicious node



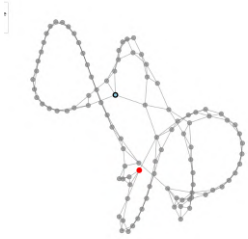
5% Malicious nodes



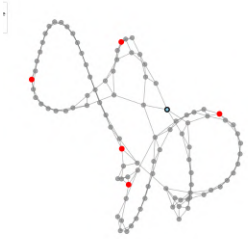
10% Malicious nodes

Figure 11: Watts-Strogatz heatmap at $\theta = 0.3$.

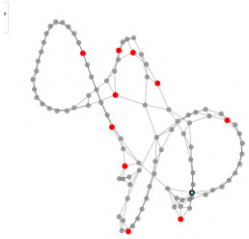
**1% Initial seed**



1% Malicious node

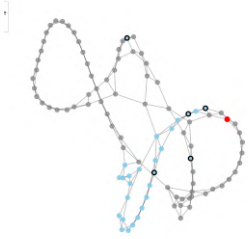


5% Malicious nodes

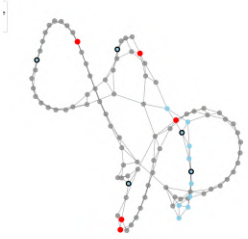


10% Malicious nodes

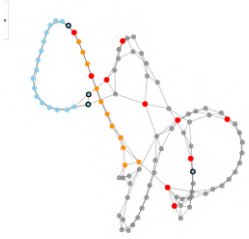
**5% Initial seed**



1% Malicious node

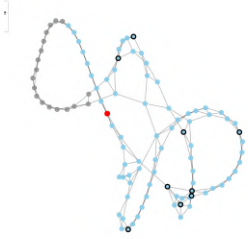


5% Malicious nodes



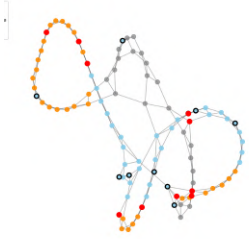
10% Malicious nodes

**10% Initial seed**



1% Malicious node

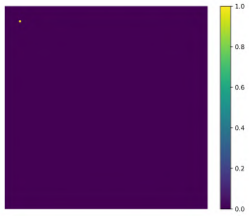
5% Malicious nodes



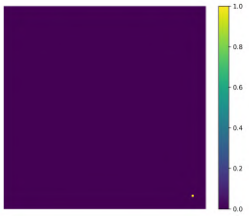
10% Malicious nodes

Table 12: Watts-Strogatz contagions at $\theta = 0.5$.

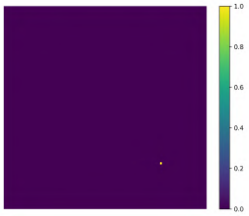
**1% Initial seed**



1% Malicious node

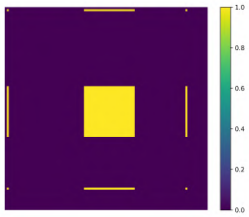


5% Malicious nodes

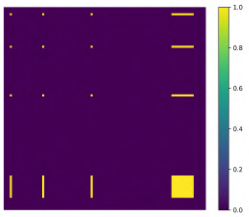


10% Malicious nodes

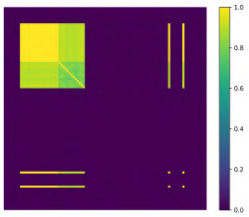
**5% Initial seed**



1% Malicious node



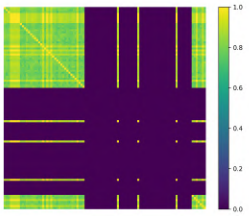
5% Malicious nodes



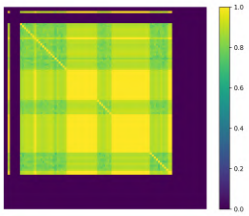
10% Malicious nodes

**10% Initial seed**

1% Malicious node



5% Malicious nodes

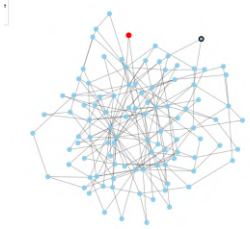


10% Malicious nodes

Figure 12: Watts-Strogatz heatmap at $\theta = 0.5$.

## A.3 Barabasi-Albert Contagion Tables and Heatmaps

**1% Initial seed**

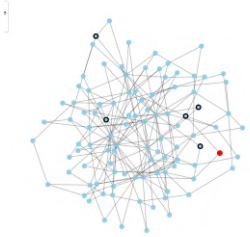


1% Malicious node

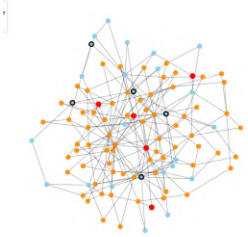
5% Malicious nodes

10% Malicious nodes

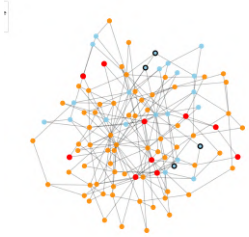
**5% Initial seed**



1% Malicious node

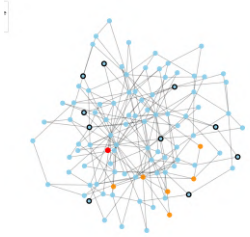


5% Malicious nodes

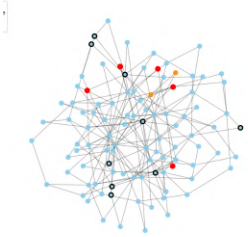


10% Malicious nodes

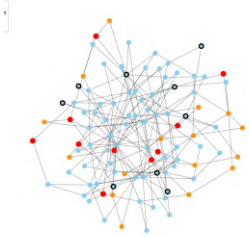
**10% Initial seed**



1% Malicious node



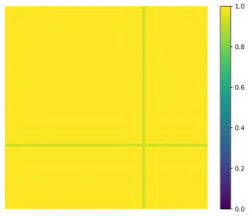
5% Malicious nodes



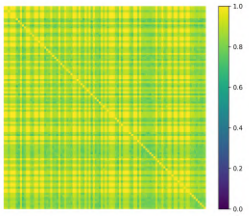
10% Malicious nodes

Table 13: Barabási-Albert contagions at $\theta = 0.1$.

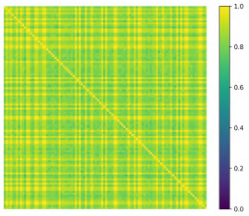
**1% Initial seed**



1% Malicious node

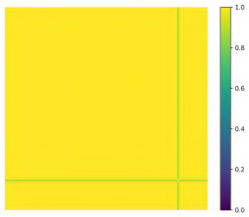


5% Malicious nodes

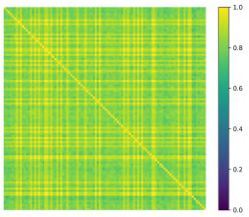


10% Malicious nodes

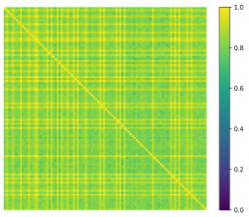
**5% Initial seed**



1% Malicious node

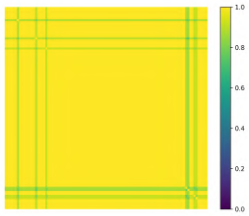


5% Malicious nodes

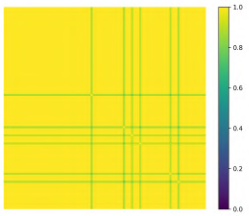


10% Malicious nodes

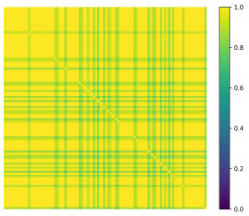
**10% Initial seed**



1% Malicious node



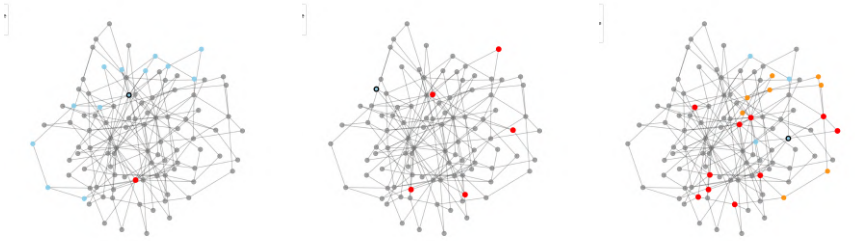
5% Malicious nodes



10% Malicious nodes

Figure 13: Barabási-Albert heatmap at $\theta = 0.1$.

**1% Initial seed**

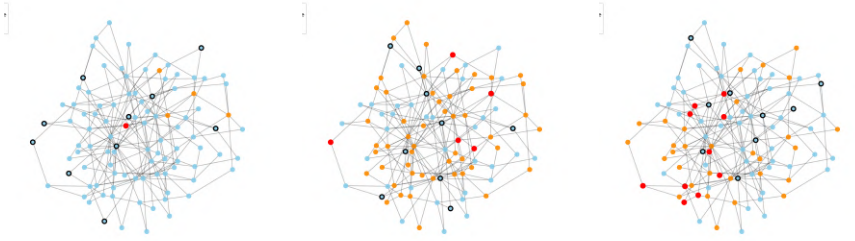


1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**

1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

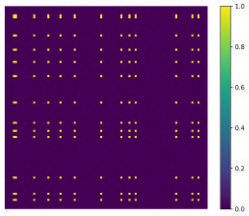
**10% Initial seed**



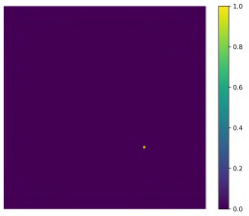
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 14: Barabási-Albert contagions at $\theta = 0.3$.

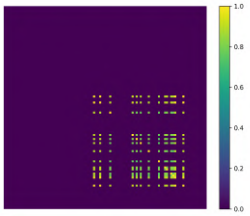
**1% Initial seed**



1% Malicious node

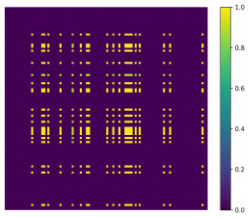


5% Malicious nodes

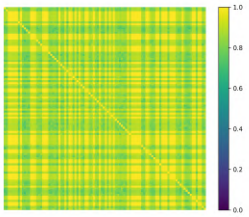


10% Malicious nodes

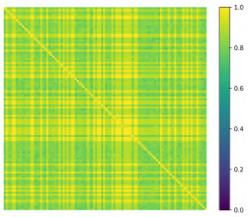
**5% Initial seed**



1% Malicious node

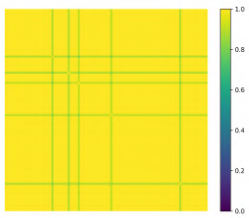


5% Malicious nodes

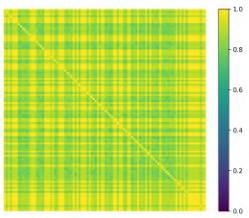


10% Malicious nodes

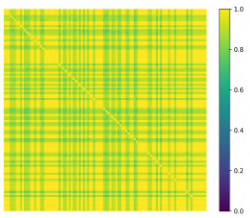
**10% Initial seed**



1% Malicious node



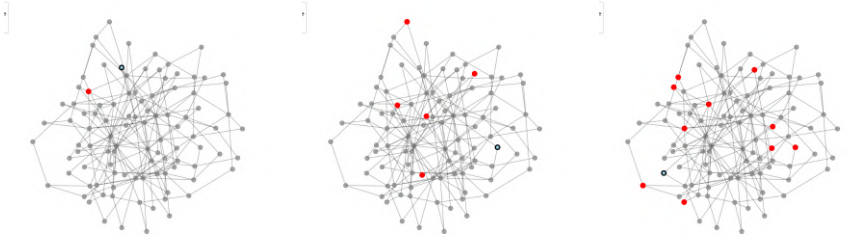
5% Malicious nodes



10% Malicious nodes

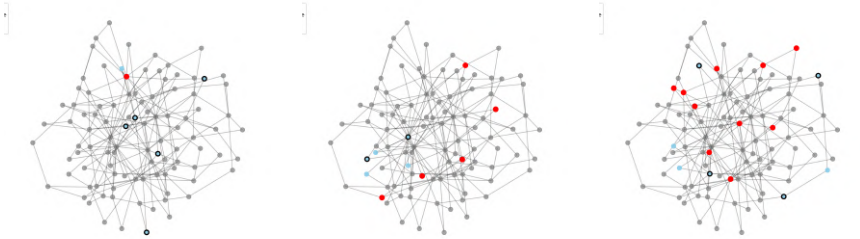
Figure 14: Barabási-Albert heatmap at $\theta = 0.3$.

**1% Initial seed**



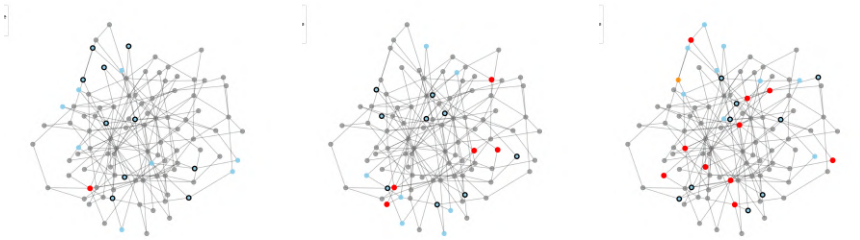
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

**5% Initial seed**



1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

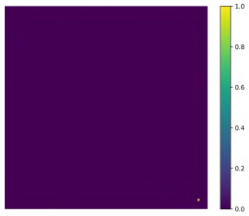
**10% Initial seed**



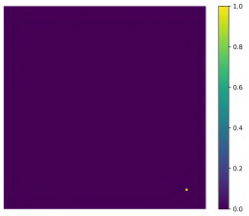
1% Malicious node | 5% Malicious nodes | 10% Malicious nodes

Table 15: Barabási-Albert contagions at $\theta = 0.5$.

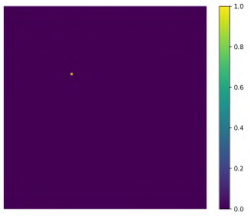
**1% Initial seed**



1% Malicious node

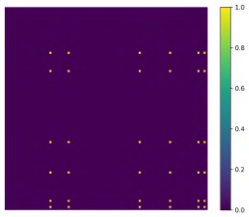


5% Malicious nodes

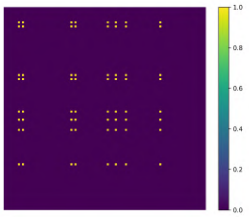


10% Malicious nodes

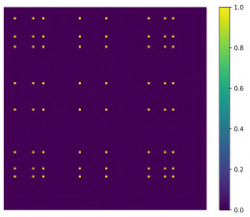
**5% Initial seed**



1% Malicious node

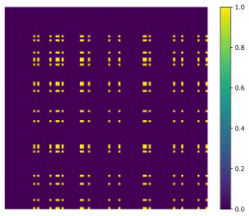


5% Malicious nodes

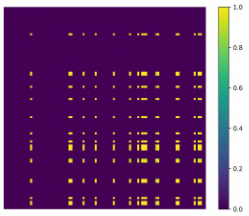


10% Malicious nodes

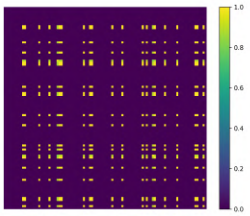
**10% Initial seed**



1% Malicious node



5% Malicious nodes



10% Malicious nodes

Figure 15: Barabási-Albert heatmap at $\theta = 0.5$.