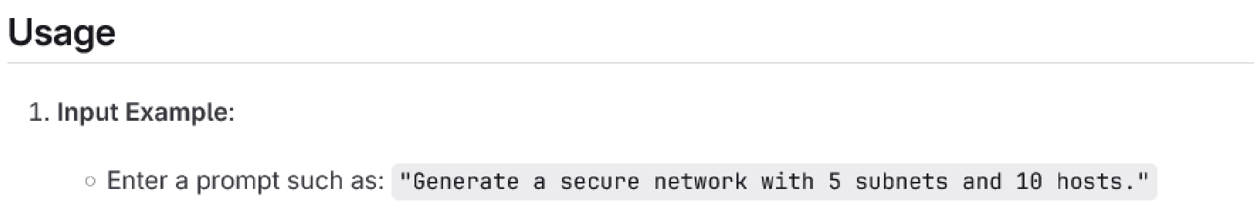
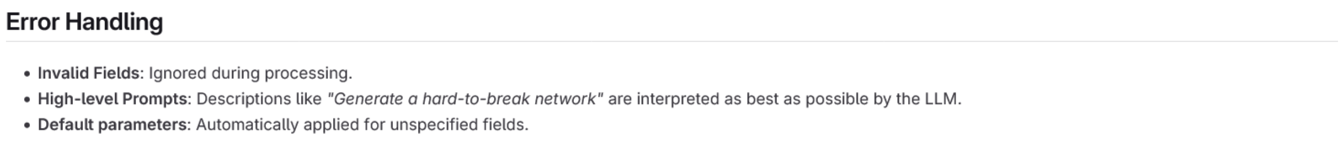


Usage

1. **Input Example:**
   - Enter a prompt such as: `"Generate a secure network with 5 subnets and 10 hosts."`

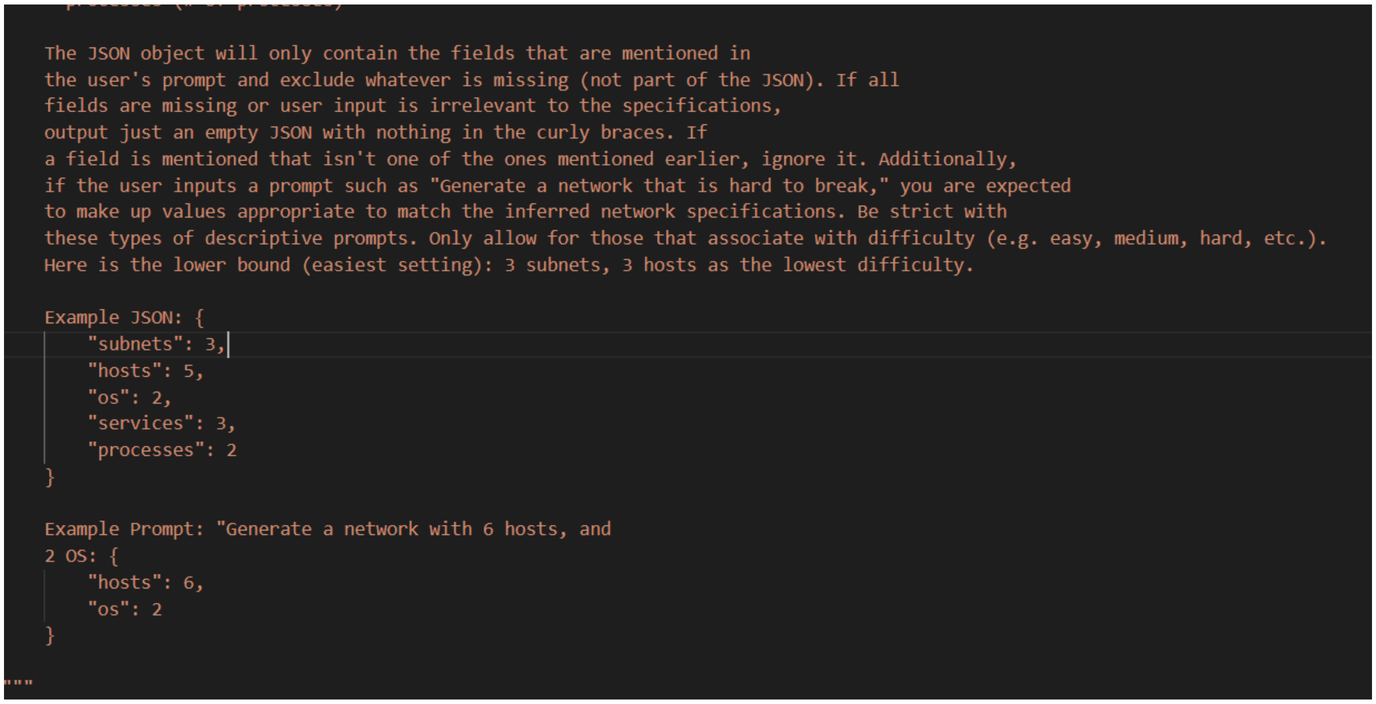
Above, the documentation depicts a prompt similar to scenarios 3, 4, and 5. The tool does not seem to comprehend the usage of secure networks on a wider scale, as the topologies returned during the testing were seen for three different security prompts. It is believed that the tool only follows two extremes, which are insecure and secure, and even when the tool returns topologies for this, they tend to repeat, showing a lack of understanding. There is no reasoning behind the tool returning a topology for insecure prompts to secure prompts. The belief is that the topologies are completely random, and descriptions for a topology are not taken into account by the tool.



Error Handling

- **Invalid Fields:** Ignored during processing.
- **High-level Prompts:** Descriptions like *"Generate a hard-to-break network"* are interpreted as best as possible by the LLM.
- **Default parameters:** Automatically applied for unspecified fields.

Due to the error handling describing the tool's lack of understanding for high-level prompts, the scenarios featured two prompts using different adjectives for the prompts to see the variation in results.



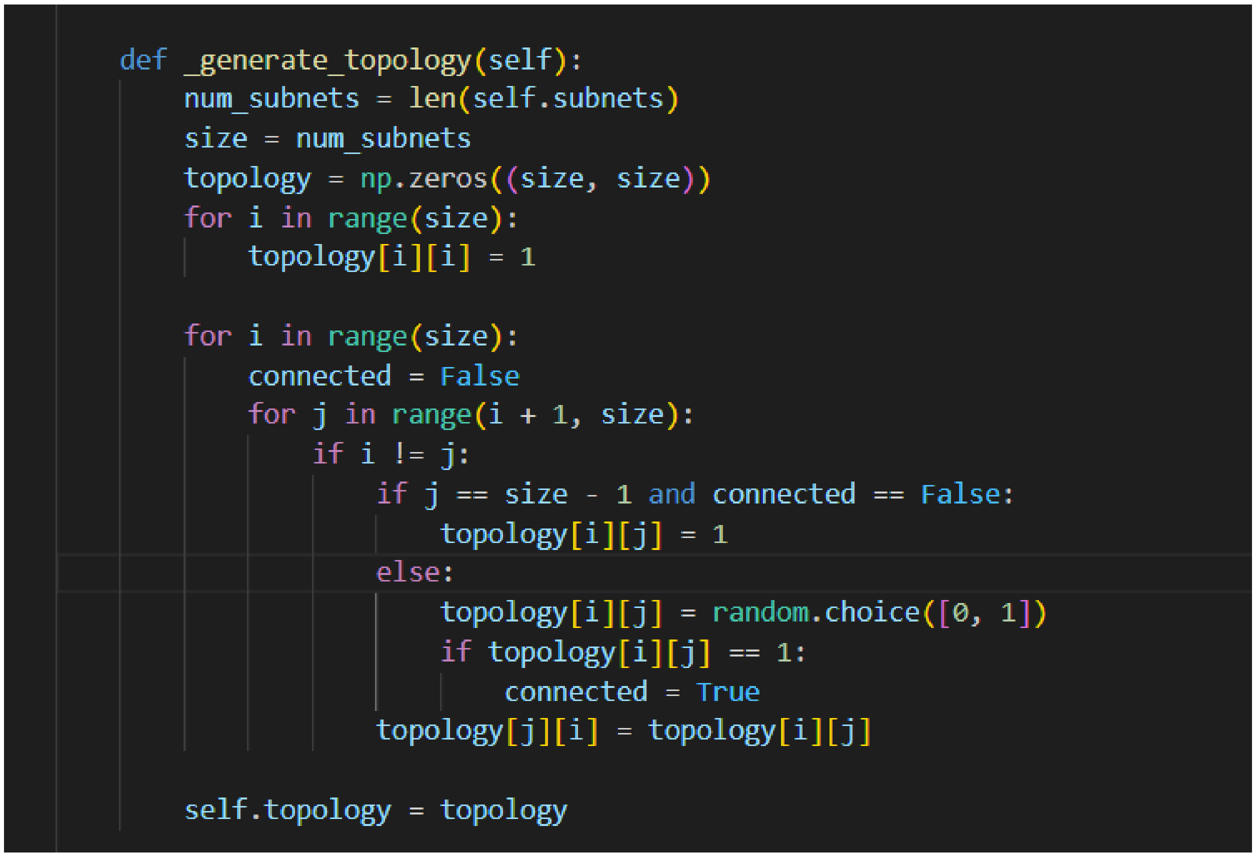
```
    The JSON object will only contain the fields that are mentioned in
    the user's prompt and exclude whatever is missing (not part of the JSON). If all
    fields are missing or user input is irrelevant to the specifications,
    output just an empty JSON with nothing in the curly braces. If
    a field is mentioned that isn't one of the ones mentioned earlier, ignore it. Additionally,
    if the user inputs a prompt such as "Generate a network that is hard to break," you are expected
    to make up values appropriate to match the inferred network specifications. Be strict with
    these types of descriptive prompts. Only allow for those that associate with difficulty (e.g. easy, medium, hard, etc.).
    Here is the lower bound (easiest setting): 3 subnets, 3 hosts as the lowest difficulty.

    Example JSON: {
        "subnets": 3,
        "hosts": 5,
        "os": 2,
        "services": 3,
        "processes": 2
    }

    Example Prompt: "Generate a network with 6 hosts, and
    2 OS: {
        "hosts": 6,
        "os": 2
    }

"""
```

The above code details the way the tool interprets the prompt. The tool only selects the integer before the highlighted word within the prompt. Adjectives used to describe a topology are not understood by the tool, therefore, the tool will not comprehend the terms “insecure” or “secure.”

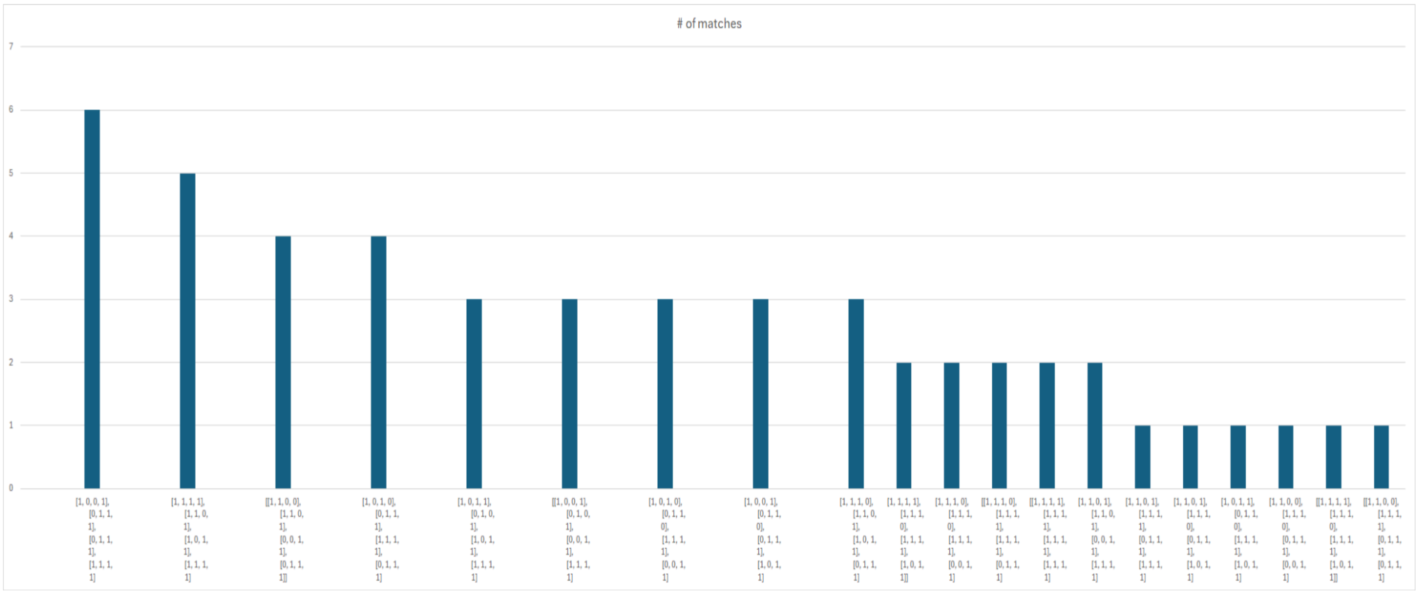


```python
def _generate_topology(self):
    num_subnets = len(self.subnets)
    size = num_subnets
    topology = np.zeros((size, size))
    for i in range(size):
        topology[i][i] = 1

    for i in range(size):
        connected = False
        for j in range(i + 1, size):
            if i != j:
                if j == size - 1 and connected == False:
                    topology[i][j] = 1
                else:
                    topology[i][j] = random.choice([0, 1])
                    if topology[i][j] == 1:
                        connected = True
                topology[j][i] = topology[i][j]

    self.topology = topology
```

The above code dictates how the topology is generated. The code generates the topology at random with random zeroes and ones throughout. This explains why the tool generates similar topologies for several prompts.



Summary:

In summary, the tool randomly generates topologies that tend to be unaffected by prompts. Above, you can see that across 50 trials, there were many repetitions even with differing prompts. Comparing the code screenshotted above and with the analysis of the graph, you can see that the topologies are randomly generated, and the wording of the documentation of the tool makes it seem like a topology will be created based on the words “secure” or “insecure”. The tool does appear to understand the extremes of secure and insecure, but the code is created to automatically filter out any words that do not correlate with the number of hosts, subnets, or OS wanted for the network. This makes user prompts typically useless, and while the extremes are sometimes recognized, there were also topologies created for an “insecure” prompt that appeared in the “secure” prompt trials. The documentation of the NetGen should be updated to reflect this error, and it should be made clear to the user that topologies are not created for a specific prompt and are random instead.