Analyzing the Impact of Malicious Network Traffic: Initial Implementation

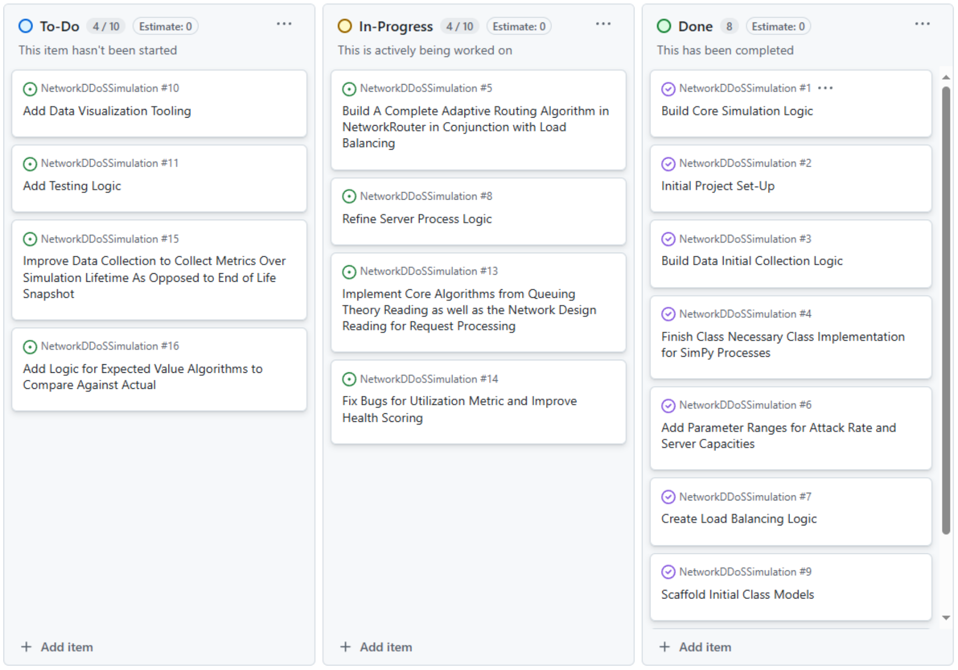
ALEXANDER M. VALENTIN, Kennesaw State University, USA

CCS Concepts: • **Networks** → **Network simulations**; **Network performance modeling**; *Network performance analysis*; *Network experimentation.*

Additional Key Words and Phrases: Distributed Denial of Service (DDoS), Botnet, Load-Balancing, Distributed Adaptive Routing, Bandwidth Exhaustion, Resource Depletion, Queuing Theory, Network Topology

ACM Reference Format:
Alexander M. Valentin. 2025. Analyzing the Impact of Malicious Network Traffic: Initial Implementation. *ACM Trans. Access. Comput.* 1, 1, Article 1 (September 2025), 4 pages.

1 Project Board Screenshots



To-Do 4 / 10 Estimate: 0
This item hasn't been started

- NetworkDDoSSimulation #10 — Add Data Visualization Tooling
- NetworkDDoSSimulation #11 — Add Testing Logic
- NetworkDDoSSimulation #15 — Improve Data Collection to Collect Metrics Over Simulation Lifetime As Opposed to End of Life Snapshot
- NetworkDDoSSimulation #16 — Add Logic for Expected Value Algorithms to Compare Against Actual

In-Progress 4 / 10 Estimate: 0
This is actively being worked on

- NetworkDDoSSimulation #5 — Build A Complete Adaptive Routing Algorithm in NetworkRouter in Conjunction with Load Balancing
- NetworkDDoSSimulation #8 — Refine Server Process Logic
- NetworkDDoSSimulation #13 — Implement Core Algorithms from Queuing Theory Reading as well as the Network Design Reading for Request Processing
- NetworkDDoSSimulation #14 — Fix Bugs for Utilization Metric and Improve Health Scoring

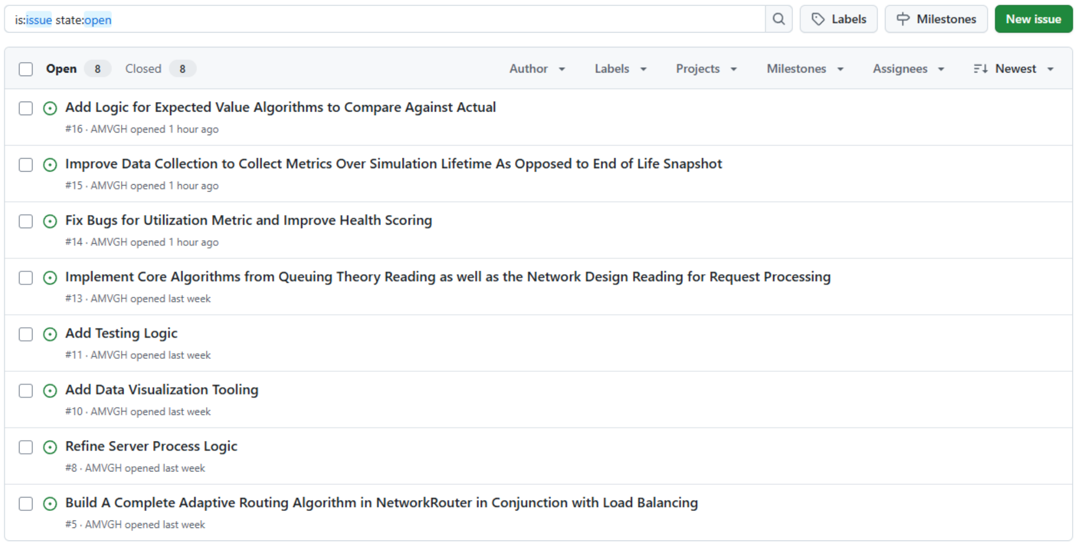
Done 8 Estimate: 0
This has been completed

- NetworkDDoSSimulation #1 — Build Core Simulation Logic
- NetworkDDoSSimulation #2 — Initial Project Set-Up
- NetworkDDoSSimulation #3 — Build Data Initial Collection Logic
- NetworkDDoSSimulation #4 — Finish Class Necessary Class Implementation for SimPy Processes
- NetworkDDoSSimulation #6 — Add Parameter Ranges for Attack Rate and Server Capacities
- NetworkDDoSSimulation #7 — Create Load Balancing Logic
- NetworkDDoSSimulation #9 — Scaffold Initial Class Models

Fig. 1. Overview of the malicious network traffic simulation project board with columns.

Author's Contact Information: Alexander M. Valentin, Kennesaw State University, Marietta, Georgia, USA.

2025. ACM 1936-7236/2025/9-ART1
https://doi.org/


is:issue state:open

Labels | Milestones | New issue

Open 8 | Closed 8 | Author | Labels | Projects | Milestones | Assignees | Newest

- Add Logic for Expected Value Algorithms to Compare Against Actual
  #16 · AMVGH opened 1 hour ago
- Improve Data Collection to Collect Metrics Over Simulation Lifetime As Opposed to End of Life Snapshot
  #15 · AMVGH opened 1 hour ago
- Fix Bugs for Utilization Metric and Improve Health Scoring
  #14 · AMVGH opened 1 hour ago
- Implement Core Algorithms from Queuing Theory Reading as well as the Network Design Reading for Request Processing
  #13 · AMVGH opened last week
- Add Testing Logic
  #11 · AMVGH opened last week
- Add Data Visualization Tooling
  #10 · AMVGH opened last week
- Refine Server Process Logic
  #8 · AMVGH opened last week
- Build A Complete Adaptive Routing Algorithm in NetworkRouter in Conjunction with Load Balancing
  #5 · AMVGH opened last week

Fig. 2. List of project issues numbers and titles with current status.

At this point in time, completed work items mainly encapsulate the initial-project setup as well as fleshing out the End-to-End implementation of core simulation logic. Work items such as Initial Project Set-Up and Scaffold Initial Class Models were completed very early as they were necessary in facilitating later development. The remaining work items such as: Build Core Simulation Logic, Build Initial Data Collection Logic, Finish Necessary Class Implementation Logic for SimPy Processes, Add Parameter Ranges for Attack Rate and Server Capacities, Create Load Balancing Logic, and Build Core Simulation Functions were all created in response to previous project feedback and the necessity to build out a functioning simulation pipeline early. The remaining To-Do and In-Progress items are designed to promote further refinement of the simulation, in order to better capture the behaviors of a real network under load. Furthermore, these items capture functionality that was outlined in the initial proposal, such as data visualization, that were not necessary for implementing the core simulation logic.

2 Implementation Evidence

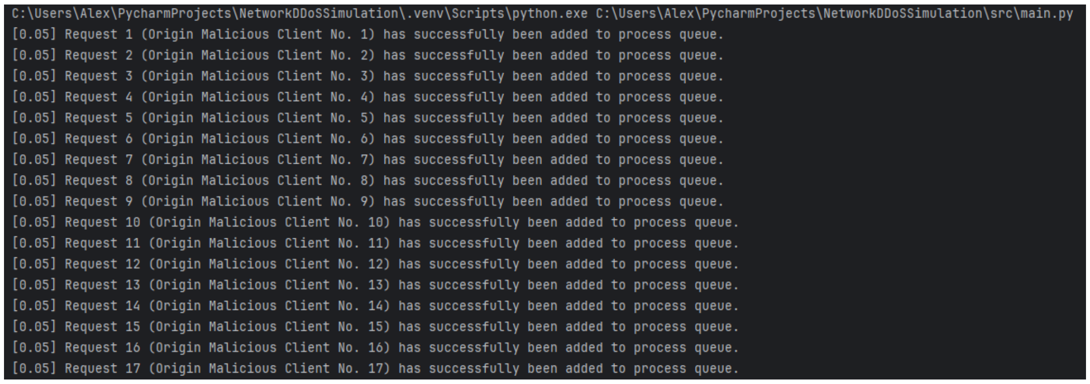
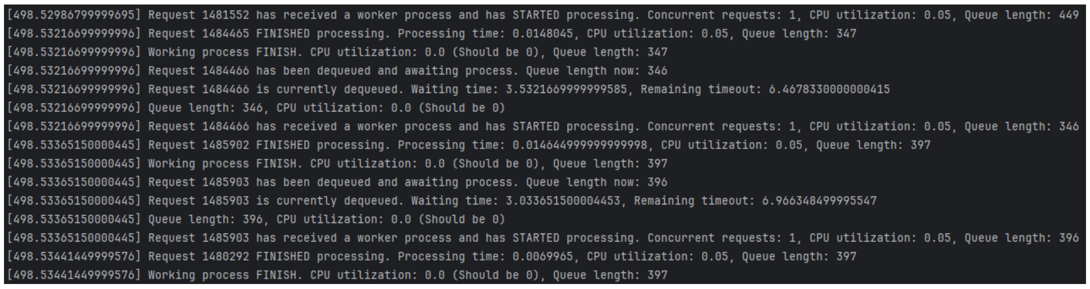
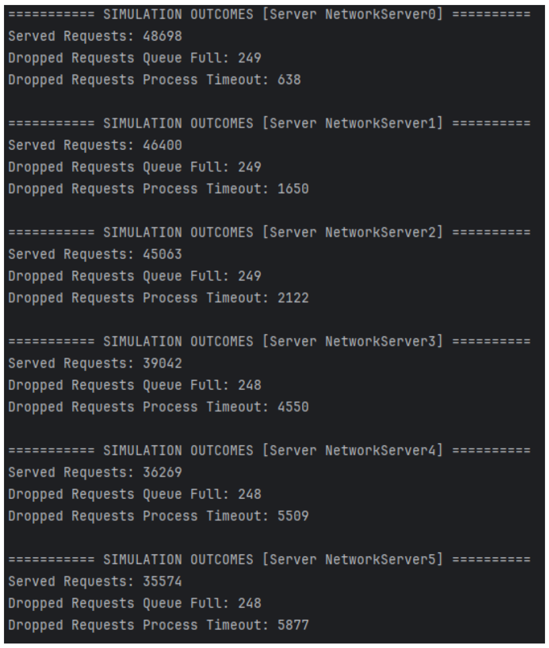

```
C:\Users\Alex\PycharmProjects\NetworkDDoSSimulation\.venv\Scripts\python.exe C:\Users\Alex\PycharmProjects\NetworkDDoSSimulation\src\main.py
[0.05] Request 1 (Origin Malicious Client No. 1) has successfully been added to process queue.
[0.05] Request 2 (Origin Malicious Client No. 2) has successfully been added to process queue.
[0.05] Request 3 (Origin Malicious Client No. 3) has successfully been added to process queue.
[0.05] Request 4 (Origin Malicious Client No. 4) has successfully been added to process queue.
[0.05] Request 5 (Origin Malicious Client No. 5) has successfully been added to process queue.
[0.05] Request 6 (Origin Malicious Client No. 6) has successfully been added to process queue.
[0.05] Request 7 (Origin Malicious Client No. 7) has successfully been added to process queue.
[0.05] Request 8 (Origin Malicious Client No. 8) has successfully been added to process queue.
[0.05] Request 9 (Origin Malicious Client No. 9) has successfully been added to process queue.
[0.05] Request 10 (Origin Malicious Client No. 10) has successfully been added to process queue.
[0.05] Request 11 (Origin Malicious Client No. 11) has successfully been added to process queue.
[0.05] Request 12 (Origin Malicious Client No. 12) has successfully been added to process queue.
[0.05] Request 13 (Origin Malicious Client No. 13) has successfully been added to process queue.
[0.05] Request 14 (Origin Malicious Client No. 14) has successfully been added to process queue.
[0.05] Request 15 (Origin Malicious Client No. 15) has successfully been added to process queue.
[0.05] Request 16 (Origin Malicious Client No. 16) has successfully been added to process queue.
[0.05] Request 17 (Origin Malicious Client No. 17) has successfully been added to process queue.
```

Fig. 3. Console output capturing network requests being generated by network clients and being added to network server process queues in the target network.



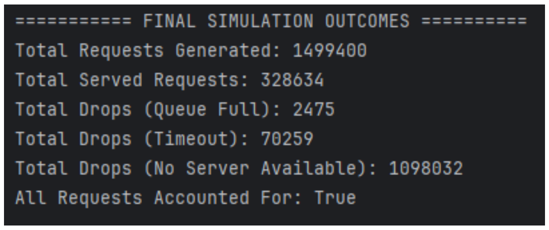
```
[498.52986799999695] Request 1481552 has received a worker process and has STARTED processing. Concurrent requests: 1, CPU utilization: 0.05, Queue length: 449
[498.53216699999996] Request 1484465 FINISHED processing. Processing time: 0.0148045, CPU utilization: 0.05, Queue length: 347
[498.53216699999996] Working process FINISH. CPU utilization: 0.0 (Should be 0), Queue length: 347
[498.53216699999996] Request 1484466 has been dequeued and awaiting process. Queue length now: 346
[498.53216699999996] Request 1484466 is currently dequeued. Waiting time: 3.5321669999999585, Remaining timeout: 6.4678330000000415
[498.53216699999996] Queue length: 346, CPU utilization: 0.0 (Should be 0)
[498.53216699999996] Request 1484466 has received a worker process and has STARTED processing. Concurrent requests: 1, CPU utilization: 0.05, Queue length: 346
[498.53365150000445] Request 1485902 FINISHED processing. Processing time: 0.014644999999999998, CPU utilization: 0.05, Queue length: 397
[498.53365150000445] Working process FINISH. CPU utilization: 0.0 (Should be 0), Queue length: 397
[498.53365150000445] Request 1485903 has been dequeued and awaiting process. Queue length now: 396
[498.53365150000445] Request 1485903 is currently dequeued. Waiting time: 3.033651500004453, Remaining timeout: 6.966348499995547
[498.53365150000445] Queue length: 396, CPU utilization: 0.0 (Should be 0)
[498.53365150000445] Request 1485903 has received a worker process and has STARTED processing. Concurrent requests: 1, CPU utilization: 0.05, Queue length: 396
[498.53441449999576] Request 1480292 FINISHED processing. Processing time: 0.0069965, CPU utilization: 0.05, Queue length: 397
[498.53441449999576] Working process FINISH. CPU utilization: 0.0 (Should be 0), Queue length: 397
```

Fig. 4. Console output that captures network requests being processed simultaneously by multiple network server process queues. Relevant server processing information is also listed.



```
=========== SIMULATION OUTCOMES [Server NetworkServer0] ==========
Served Requests: 48698
Dropped Requests Queue Full: 249
Dropped Requests Process Timeout: 638

=========== SIMULATION OUTCOMES [Server NetworkServer1] ==========
Served Requests: 46400
Dropped Requests Queue Full: 249
Dropped Requests Process Timeout: 1650

=========== SIMULATION OUTCOMES [Server NetworkServer2] ==========
Served Requests: 45063
Dropped Requests Queue Full: 249
Dropped Requests Process Timeout: 2122

=========== SIMULATION OUTCOMES [Server NetworkServer3] ==========
Served Requests: 39042
Dropped Requests Queue Full: 248
Dropped Requests Process Timeout: 4550

=========== SIMULATION OUTCOMES [Server NetworkServer4] ==========
Served Requests: 36269
Dropped Requests Queue Full: 248
Dropped Requests Process Timeout: 5509

=========== SIMULATION OUTCOMES [Server NetworkServer5] ==========
Served Requests: 35574
Dropped Requests Queue Full: 248
Dropped Requests Process Timeout: 5877
```

Fig. 5. Console output that captures relevant network server statistics post-simulation run. Metrics such as served requests and dropped requests can be observed.



```
=========== FINAL SIMULATION OUTCOMES ==========
Total Requests Generated: 1499400
Total Served Requests: 328634
Total Drops (Queue Full): 2475
Total Drops (Timeout): 70259
Total Drops (No Server Available): 1098032
All Requests Accounted For: True
```

Fig. 6. Console output that captures relevant network simulation system statistics post-simulation run. Metrics such as total requests generated, total served requests, and total request drops can be observed.

3 Status Summary

At the current moment, class models have been sufficiently implemented to facilitate and capture core simulation functionality. Users are able to configure the simulation parameters found in config.py and get simulation results regarding the total requests generated, total served requests, and total request drops with their causes. Furthermore, the user is provided these metrics per server, in order to visualize how these metrics may deviate between server instances. While the core implementation logic has been captured, there is still further refinement that must be made in order to accurately model a real world network under load. While the simulation implementation still requires work in order to reach the proposed end goal, the core simulation logic as well as the initial load-balancing and data collection mechanisms have been key accomplishments since M1.

The biggest challenge when implementing such a network simulation came in architectural design. Unfortunately, there were instances during the course of development where I had realized that the architectural design I had outlined in my UML Class Diagram was either ineffective in capturing the behavior I wished to emulate, or could be further refined and was not optimal for what I was trying to achieve. These issues however, did not impede my ability to implement the system I wished to emulate, and instead caused me to reevaluate design choices and improve my class models as a whole. While the current state of the simulation serves as an excellent foundation and validates that my current approach of implementation is correct, there are still many work items that need to be completed in order to reach the final implementation goal. Over the coming weeks I plan to refine data collection techniques and expand them to better capture simulation metrics, implement improved routing techniques to better capture network routing in real world systems, refine server processing, implement data visualization tooling, add testing logic, and implement core algorithms to predict simulation outcomes in order to generate a baseline for comparison.

4 GitHub

The simulation implementation can be found at: https://github.com/AMVGH/NetworkDDoSSimulation and the simulation project board can be found at https://github.com/users/AMVGH/projects/2