Project plan, draft 1

- Beshir, John, Feraidon

Description:
Denial of Service (DoS) Attacks happen when a victim (host or network) is flooded by an abundant amount of malicious traffic until the victim cannot process any legitimate traffic or just crashes.

This project would include setting up a testing environment with a target and attacking machines. The target machine would include logging and metrics as a means to measure success of mitigation techniques. The target will have various mitigation techniques necessary for a complete analysis. Attackers will send malicious requests to the target machine to perform the DDoS (Distributed-Denial of Service) attack. Attacker machines will support various forms of DDoS attacks.

Learning Goals:

- How DDos traffic patterns differ by various ways
    - Identify differences between network-layer flooding and application-layer flooding
- What mitigation techniques work
    - Understand how common mitigation techniques(rate limiting, connection limits, filtering, etc) can reduce attack impact
- Practical Networking and Deployment
    - Gain experience deploying services via Raspberry Pi's
- How multiple mitigation techniques interact when deployed across layers
    - Understand how combining mitigations at different layers can improve or degrade effectiveness
- Analyze experimental data and present security findings
    - Interpret and assess mitigation effectiveness and communicate findings

Development Goals:

- Gain experience with security testing tools
- Develop practical networking skills
- Improve data-driven reasoning
- Learn to troubleshoot under real-time conditions
- Communicate Technical findings clearly and concisely

Testing:

To evaluate the effectiveness of our DDoS analysis and detection, we will use a controlled Raspberry Pi lab environment with clearly defined attacked, victim, and observer roles. Testing for correctness will focus on whether the tool accurately identifies malicious traffic patterns and logs relevant indicators of a DDoS attack. We will begin by establishing a baseline for a normal network behaviour under legitimate load. Then we will introduce simulated attack traffic from a single Pi and verify that the tool correctly distinguishes attack traffic (abnormal) versus normal activity. We will test edge cases by varying the number of attacking nodes, the request rate, and the duration of the attack to ensure the tool remains accurate under different conditions. Correctness will be evaluated by checking detection consistency, and whether alerts and logs reflect the actual lab conditions.

Performance Benchmark:

We will benchmark using repeatable standardized scenarios and compare system behavior. How does the mitigation technique perform under stress?

Benchmark Method:

- Fixed run duration
- Same workload profile each time
- Same stress workload profile each time
- Run each multiple times and report average

A mitigation would be considered effective if traffic performs better in the mitigated scenario than in the unmitigated attack scenario.

Data Analysis:

We will collect packet captures (through various techniques and platforms), system logs, and performance metrics from all roles (attackers , victim, observers). We will organize the data into tables and graphs to visualize trends, spikes, and unexpected changes. Conclusions will be based on measurable evidence rather than generalized assumptions.

Development Schedule:

Task	Deadline
(MVP) Manager (Start Stop, Minimal config)	1 ½ Week (01/28) - John
(MVP) Target Machine (static endpoints)	1 ½ Week (01/28) - Feraidon
(MVP) Attacking Machine (HTTP L7)	1 ½ Week (01/28) - Feraidon
(MVP) Proxy (Forwards, no mitigation)	1 ½ Week (01/28)-Beshir
(MVP) Above nodes work together E2E	1 ½ Week (01/28) - ALL
(MVP) Manager can edit config across nodes (Standardized)	1-2 Week - John (Will be blocked by node development)
(MVP) Node Monitoring (Basic metrics from each node)	1 ½ Week -John
(MVP) Monitoring Endpoint (Node metrics sent to Prometheus + Grafana, basic dashboard)	1 Week - John
Manager Improvements (Add Grafana Dashboard, improved manager controlling)	1 Week - John
Proxy (Rate Limit, IP filter, etc)	~ 1 Week (Subject to change) - Beshir
Proxy (SYN Backlog, SYN Cookies, etc, L3/L4 Mitigations)	1 Week - John
Target Machine Caching Setup	1 week - Feraidon
Target Machine Mitigations (Similar to proxy)	~ 1 Week (Subject to change) - Beshir
Proxy (Any additional, Subject to change)	~ 1 Week (Subject to change) - John
Attacker (L3/4 SYN-TCP)	1 week -Feraidon
Attacker (L7/3 Slowloris)	-Feraidon
Node monitors in depth logging	-Beshir
Monitor in depth Dashboard (Prometheus + Grafana)	½ Week - Feraidon

Target 2 implemented (Micro service, DB/Background processes)	- 1-2 Week: Beshir
Target 2 Caching and Mitigations	~ 1 - 1 ½ Weeks - Feraidon (Subject to change)