Poster: IoTSimSecure: Towards an IoT Simulator Supporting Cyber-Threat Detection Algorithms

Reham Almutairi [1,2], Giacomo Bergami [1], and Graham Morgan [1]

[1]School of Computing, Newcastle University, United Kingdom
Email: {r.m.s.almutairi2, Giacomo.Bergami, Graham.Morgan}@newcastle.ac.uk

[2]College of Computer Science and Engineering, University of Hafr Albatin, Saudi Arabia

***Abstract*—We introduce IoTSimSecure, a novel IoT simulator addressing the gap of IoT simulators for dealing with cyber-security threats, specifically battery draining attacks occurring in real-time. This will serve as a valuable testbed for evaluating current AI-driven security strategies. IoTSimSecure aims to address these needs by offering an architecture that combines signature-based and anomaly-based detection methods.**

***Index Terms*—IoT simulators, battery attacks, IoT security.**

I. Envisioning the IoTSimSecure Simulator

The proposed IoTSimSecure simulator would support the monitoring and analysing of network traffic patterns in real-time, allowing EWMA [1] and threshold-based techniques to detect anomalies indicative of potential flooding attacks. As this simulator should schedule activities for each agent and given that SimulatorBridger [2], the fittest simulator, does not simulate communications as comprehensively as required to mimic packet transmission correctly, we cannot extend this simulator despite its support for IoT battery consumption. We envision a new simulator supporting different malicious attacks and attack detection algorithms: Fig. 1 describes our proposed simulator IoTSimSecure using a dynamic evolutionary architecture [3].

A. Load Balancing Algorithm

Our network countermeasures include a flexible load-balancing algorithm for MELs in edge devices, designed to efficiently support Software-Defined Networking (SDN) within SimulatorBridger [2]. This algorithm allows for integrating either the Gateway Load Balancing Protocol (GLBP) or a threshold-based method for managing received requests. It ensures flexible switching between these methods in a plug-and-play manner to optimize load distribution. The algorithm redistributes overflow traffic to neighbouring edge devices when a device reaches its maximum capacity, using a hash table to manage edge information based on IoT device connections. For GLBP, it enables dynamic rerouting to maintain uninterrupted operations, requiring a backup agent for DNS routing adjustments. GLBP supports even traffic distribution across multiple gateways. Alternatively, the threshold-based approach prevents new requests at capacity limits, redirecting them to the nearest capable edge. This combined strategy aims to reduce load imbalances and improve energy efficiency, preventing potential battery depletion or increased energy consumption from overloads or attacks. SimulatorBridger currently supports a single gateway per network, indicating a need for architectural enhancements to implement these strategies fully.

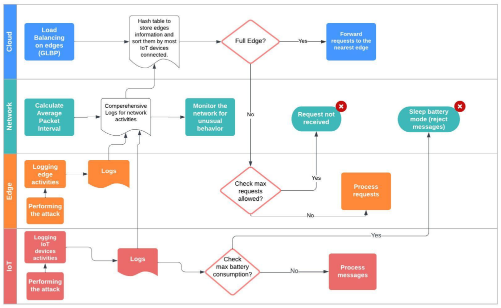

Cloud
Network
Edge
IoT
Load Balancing on edges (GLBP)
Hash table to store edges information and sort them by most IoT devices connected.
Full Edge?
Yes
Forward requests to the nearest edge
Calculate Average Packet Interval
Comperehensive Logs for network activities
Monitor the network for unusual behavior
No
Request not received
Sleep battery mode (reject messages)
Logging edge activities
Logs
Performing the attack
Check max requests allowed?
Yes
Process requests
No
Logging IoT devices activities
Logs
Performing the attack
Check max battery consumption?
Yes
No
Process messages

Fig. 1: The Proposed IoTSimSecure Architecture

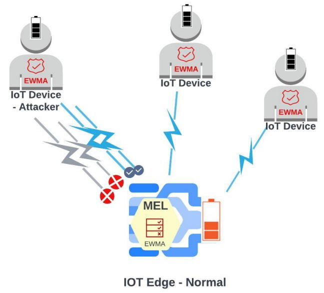
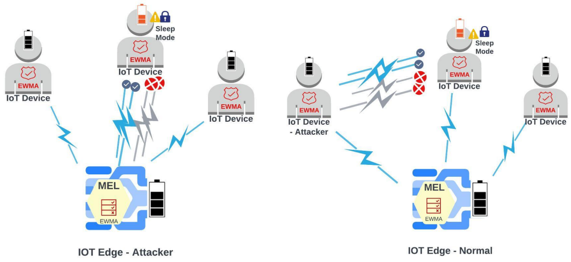


Fig. 2: Scenario 1: Attacking the battery of Edge device: implementing EWMA for number of requests received



(a) Edge device is attacking (b) IoT device is attacking

Fig. 3: Scenario 2: Attacking the battery of IoT device vs EWMA attack detection algorithm.

979-8-3503-1515-8/23/$31.00 ©2023 Crown

B. Average packet interval

In SimulatorBridger, devices are set to communicate at specific intervals, defined during cloud App configuration. To identify potential malicious activity, such as devices sending requests more frequently than usual, the system calculates an average packet interval that is considered normal based on historical data from real-world network patterns, including different times and days of the week. Logging the number of requests per second at regular intervals helps establish this baseline of normal behaviour. From this baseline, thresholds are set to flag deviations that may indicate malicious behaviour, with upper and lower limits to minimize false positives. Exceeding these thresholds could signal an intrusion or attack, prompting further investigation or response to secure the network.

C. Attacking the edge device

To enhance the simulation capabilities of SimulatorBridger in this regard, several key modifications are proposed: performing (EWMA) algorithm in MELs and setting the threshold above which traffic is considered anomalous or potentially malicious. This method is proposed to monitor network traffic effectively, identifying unusual spikes or patterns indicative of an attack by calculating moving averages. The threshold will be set for the number of requests a MEL in the edge can accept so the edge can be prevented from becoming overwhelmed by a draining battery attack. The EWMA algorithm will be dynamically adjusted based on the volume of incoming requests and the number of IoT devices connected to the edge, providing a flexible approach to traffic management. MELs will log traffic information within a sliding window, ensuring that only the most relevant, recent data is considered, optimizing memory usage and focusing on current traffic patterns. The scenario in Fig. 2 illustrates an attack on an edge device's battery, where an IoT device sends excessive requests to the edge, exceeding the established threshold. This scenario demonstrates the mechanism where the MEL will only process requests up to a specific limit, indicated by (✓), and reject any requests that exceed this capacity, indicated by (✗). This strategy highlights the need for capabilities to access and update secondary memory for traffic log management, which is not currently in place within the current simulator.

D. Logging procedures

To detect battery-draining attacks in IoT networks, an advanced logging system is essential for comprehensively recording all network activities, including communication patterns, request frequencies, and transmission anomalies. Obtaining access to these extensive logs is necessary for the IoT attack prevention algorithm, enabling it to identify changes from established network behaviour and potential security threats by analysing unexpected data request spikes or irregular communication patterns. The algorithm's effectiveness relies on its ability to distinguish between normal operations and possible attacks, which allows it to make critical decisions regarding whether to continue monitoring the messages or launch a response protocol, such as suspending message monitoring for further analysis or adjusting network parameters. This flexibility to re-schedule is crucial for addressing different levels of attacks and maintaining network integrity dynamically.

E. Support for other algorithms in addition to EWMA

The IoTSimSecure framework should be flexible, allowing different algorithms to be plugged in and used for traffic analysis. To detect attacks, this tool could provide an interface for algorithms to access traffic logs and perform the necessary computations. The plugged-in algorithm would analyze the data within the window to calculate the EWMA/thresholds and other statistical measures. If the traffic within the current window varies significantly from the calculated average or expected pattern, it could be flagged as abnormal. The thresholds in the proposed IoTSimSecure system are two: one is in the MELs in the edge devices to calculate the number of requests allowed to be received by the edge, and the other threshold calculation is in IoT devices that identify the maximum power consumption it is capable of consuming. The threshold value in IoT devices will be fixed and set at the configuration step before the simulation starts. The scenario for attacking the battery of an IoT device is illustrated in Fig. 3. If an attacker, an edge device or an IoT device, tries to deplete an IoT device battery, the battery consumption will reach the maximum limit/threshold. Hence, this device is set to sleep battery mode to protect the battery from being depleted so it will not receive the rest of the packets from the attacker. This demonstrates the need for the simulator to mimic IoT nodes transmitting packets and receiving them from the network. The current SimulatorBridger does not encompass this, as it does not fully implement the IPv6 communication layers as in Omnet++.

II. Conclusion

Our future efforts will implement a novel simulator, IoTSimSecure: by requiring a massive refactoring of the current state-of-the-art simulator, SimulatorBridger, this postulates the definition of a novel simulator architecture, less monolithic and more modular, which could allow further possible extensions in the future similar to Omnet++, while considering a broader range of IoT-specific attacks. This goes hand in hand with choosing data meshes [3] as the preferred storage for logging data, thus allowing us to provide better analytics, including time series monitoring.

References

[1] M. Perry, *The Exponentially Weighted Moving Average*, 06 2010.

[2] R. Almutairi, G. Bergami, G. Morgan, and R. Gillgallon, "Platform for energy efficiency monitoring electrical vehicle in real world traffic simulation," in *2023 IEEE 25th Conference on Business Informatics (CBI)*. IEEE, 2023, pp. 1–8.

[3] G. Bergami, "Towards automating microservices orchestration through data-driven evolutionary architectures," *Service Oriented Computing and Applications*, vol. 18, pp. 1–14, In Press 2024.