

Candidate: Giuseppe Piombino s280117
Title: Intelligent forensics for the automatic anomaly detection in distributed infrastructures
Supervisor: Cataldo Basile, Andrea Atzeni, Borja Bordel Sanchez

The thesis "Intelligent forensics for the automatic anomaly detection in distributed infrastructures" explores the realm of the digital forensics, in search for solution for the automatic recognition of the Denial of Service (DoS) attacks, in a distributed infrastructure. The impact of the rapid technological advancements, that characterised the digital age, led to the development of an ultra-connected work, especially in sectors as IoT, finance, healthcare, e-commerce with special mention for smart cities and electric vehicles. Together with this advancement, came the negative impact of the progress: the evolution of crime, in this particular context, of cybercrime. Among the various types of cybercrimes, DoS attacks are particularly challenging. These attacks disrupt the normal functioning of systems, by overwhelming them with illegitimate requests. With the evolution of the DoS attacks, the detection and recognition process became harder and more complex. The solution proposed in this thesis, consist in a system that employs an AI to automate the forensic process of identifying DoS attacks, reducing the complexity and time request for manual detection.

The existing literature offers different hints about the direction the project could take. Previous works, for example, highlighted the importance of the collection of the log activity in a permanent way, saving them from a possible loss. Another, suggested the use of the Random Forest algorithm as baseline to test another machine learning algorithm, which in this thesis, have been found in the MultiLayer Perceptrons.

Initially, for a better comprehension of the phenomenon, an examination of the DoS attack is performed, detailing their nature, impact, types and specific examples. A DoS attack is a malicious attempt to disrupt the normal functioning of targeted servers, services or networks by overwhelming them with a flood of Internet traffic. The attacks exploit resource limitation and vulnerabilities, leading to significant losses in productivity, finance and reputation. Even an hour of downtime can result in substantial financial losses, as highlighted by statistical data. Three types of attacks have been distinguished: volume-based attacks, protocol attacks and application layer attacks. The volumetric attacks flood the target with massive amounts of traffic to consume all available bandwidth and resources, preventing legitimate users from accessing the services. The protocol attacks exploit vulnerabilities in network protocols to exhaust the target's resources. Finally, the application layer attacks target a specific application to disrupt the service. A special attention has been dedicated to ICMP flood, SYN flood, HTTP Flood and slow HTTP attacks, offering a valid explanation for each of the categories mentioned. Furthermore, to better understand how to approach the problem, a part of the study is dedicated to the detection and mitigation of the attacks, exploring solutions oriented to the traffic analysis, the resource performances, the log files, honeypots and more.

At the same time, raised the necessity to obtain a testing and development environment. The best option appeared to be the creation and utilisation of a virtual network environment, aiming to the test and development of an AI model to recognise DoS attacks. The environment simulates a real-world network condition, providing a controlled space to study attack behaviours and refine detection methodologies. The virtual network has been build using Docker, exploiting the

key features of containerisation. In this way the environment dependencies, configurations and standard operations are automatic, portable and consistent across different systems. The use of the virtual network offers several advantages, like: isolation to provide a safe environment to simulate and study DoS attacks without risking any consequence, legal included; reproducibility, providing the condition to repeat the experiments and obtain consistent results; flexibility, making the infrastructure easy to scale and configure. Docker-compose is employed to define and manage multi-container Docker applications, simplifying the setup and configuration of the virtual network.

The network has a tree topology, in which as root is positioned a router called routerB and as leafs client representative nodes, which include an attacker node. On top of routerB there is the routerA, which connects the tree to the server network and an administrator node. The network routing is handled by FRRouting (FRR), an open-source IP routing protocol suite, designed to facilitate routing protocols like BGP, OSPF and RIP. It ensures efficient data packet routing across the network, mimicking the routing behaviours of real-world network. The administrator node, instead, is employed as a log collector and is meant to monitor the entire network. In fact, every node of the net sends its log to the administrator node. The log collection and aggregation activity is performed in the admin node, by means of the open-source data collector Fluentd. Together with the logs of the net, Fluentd collects the necessary information to understand if a DoS attack happened.

Having the infrastructure and the basic knowledge to approach a DoS attack, the AI model have been realised. The model aims to automate the forensic process, reducing the complexity and time requirements for manual detection of the cyber threats. The dataset used to train the model is based on the Canadian Institute of Cybersecurity intrusion detection dataset (CIC-IDS2017). It includes a wide range of network traffic data, both benign and malicious, providing a comprehensive base for training the AI model. The dataset came from a previous work of the team of the co-relator Bordel, and have been preprocessed by importing the PCAP data on Wireshark and later elaborated in order to obtain the metric interested to recognise the DoS attack. Specifically, the metric searched are the same obtainable by the monitoring tool tcpLife and tcpTracer, offered by the BCC python library. The thesis compares two algorithms, the MultiLayer Perceptrons (MLP) and the Random Forest (RF). The MLP is a type of neural network composed of multiple layers of nodes, each connected to the next layer. MLPs are capable of capturing complex patterns in data, making them suitable for identifying intricate behaviours associated with DoS attacks. RF is a learning method that operates constructing multiple decision trees during training. It is known for its accuracy and robustness in various application, for this reason it has been used as baseline to evaluate the accuracy of the MLP. To train the model, the dataset has been divided in training set and testing set. The model has been configured with 3 hidden layers, in which the first two have six nodes each with a ReLu activation function, while the last layer has only one node with a sigmoid activation function, allowing the model to perform as a classification algorithm. Finally, the models have been evaluated focusing on the accuracy, precision, recall and F1-score. The RF resulted slightly more precise, but in optic of a future bigger database the deep learning option has been chosen.

In order to use the model in a real-world or emulated context, it is vital to define how to collect the data to be fed to the AI model. Data collection is crucial for gathering real-time and historical network traffic information and perform an

accurate analysis of data, ensuring an ideal functioning of the model. The focus of the recognition has been set on the server, for this reason the collection unit is located in the server. The collection system relies on three main script: `1clock.py`, `tcpLife` and `tcpTracer`. The first one serves as manager of the seconds, which are indeed the responsible for the collection of information. The tools `tcpLife` and `tcpTracer` are offered by the BCC (BPF Compiler Collection) library that provides several tools for various layer of system monitoring and most of all the possibility to compile, load and attach eBPF programs by means of a python script. EBPF (extended Berkeley Packet Filter) is a technology that allows user-space programs to execute custom code within the Linux kernel without needing to modify and recompile the kernel itself. `TcpLife` tool gather information related to the byte transmitted and received, and the duration of the TCP connection. `TcpTracer` instead gather information related to the number of opened and closed connections, and the time difference between one open/close/accept event and the other. To make the recognition more functional, the data have been collected in a specific time interval, in order to extract mean and variance of each feature. The manager script task is to synchronise the data coming from the two tools and send the result to the data collector `Fluentd`. The synchronisation happens by means of a timer and of the signal `SIGUSR1` sent to both the tools. The inter process communication between the tools and the `1clock`, happens by means of sockets. Once the `SIGUSR1` signal is received, the tool sends a divisor string.

The final step for the creation of a real-world environment is the creation of an Attacker node. It is defined how to execute the attacks by means of different tools, such as `slowhttptest`, `HULK` (HTTP unbearable Load King), `hping3` and an ad-hoc script to perform the Ping of Death attack. `Slowhttptest` can simulate several types of attacks, including the `Slowloris`, `Slow HTTP POST`, `Slow read` and `Range Header`. Two versions of `HULK` are presented: the original one by Barry Shteiman and a more modern version, that allows the creation of a local botnet and perform a `DDoS HTTP POST` attack. Finally, the `hping3` tool, allowed the performing of `SYN flood` and `ICMP flood` attacks.

The outcome of the digital forensic practice are exposed. The process, from data acquisition to model deployment and analysis, is discussed. The key points are how the node logs and TCP connections detail were extracted by means of `syslog-ng` and the TCP tools already mentioned, and how they are collected by `Fluentd`. It is explained how to export and import the AI model and how the analysis have been done. Finally, the result of the tests are shown: the attacks `Slowloris`, `Slow HTTP POST`, `Slow read`, `Range Header` and the two versions of `HTTP flood` were recognised by the model. The `SYN flood`, `ICMP flood` and `Ping of Death` were not recognised.

Concluding, the thesis underscores the complexity and time-consuming nature of manually recognising these attacks. The automated solution proposed in fact offers an alternative for the recognising of several DoS attacks in a digital forensic environment. Furthermore, the choice of using a virtual network offers a pedagogic instrument for the exploration of the DoS attacks, of resources like eBPF and tools like `Fluentd`.