When Two-Layer Federated Learning and Mean-Field Game Meet 5G and Beyond Security: Cooperative Defense Systems for 5G and Beyond Network Slicing

Hichem Sedjelmaci*, Abdelwahab Boualouache**
(*) Ericsson R&D Security, France
(**) FSTM, University of Luxembourg, Luxembourg
Email: (*) hichem.sedjelmaci@ericsson.com, (**) abdelwahab.boualouache@uni.lu

Abstract—Cyber security for 5G and Beyond (5GB) network slicing is drawing much attention due to the increase of complex and dangerous cyber-attacks that could target the critical components of network slicing, such as radio access and core network. This paper proposes a new cyber defense approach based on two-layer Federated Learning (FL) to protect 5GB network slicing from the most dangerous network attacks and a mean-field game to safeguard the FL-enabled defense system from poisoning attacks. Our proposed distributed defense systems cooperate, intending to detect internal and external attacks targeting the critical components of 5GB network slicing and detecting infected parts in the 5GB defense system. Our experimental results show that our cooperative defense systems exhibit high accuracy detection rates against network attacks, namely (distributed) denial of service and botnets while being robust against poisoning attacks and requiring a few overheads generated by defense systems. To the best of our knowledge, we are the first to propose lightweight and accurate cooperative defense systems based on two-layer FL and non-cooperative games to enhance security against attackers in 5GB network slicing.

Index Terms—5G and Beyond; Network slicing; Security; Privacy; Federated learning; Mean field game;

1 Introduction

Network slicing is one of the key enablers of the fifth generation and Beyond (5GB) networks [1]. This new paradigm allows the creation of logical networks on shared physical infrastructure to decrease the end-to-end latency while considering the network constraints such as bandwidth and packet loss. This is achieved by integrating several technologies, mainly Software-Defined Networking (SDN), Network Function Virtualization (NFV), and Multi-Access Edge Computing (MEC). More specifically, SDN and NFV work together to enable several network slice services, while MECs place these services closer to the end users for decreasing latency [2]. The 3rd Generation Partnership Project (3GPP) standards define several 5GB services provided by network slicing, including Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), and Massive Machine-Type Communication (mMTC) [3]. However, various security threats come with advantages offered by network slicing [4]. Breaking network services becomes easier with network slicing since attackers only target slice software components to get services broken. Thus, securing the network slicing is mandatory, and failure to achieve this will disturb the acceptance of network slicing in 5GB networks.

Over the past few years, various attack detection techniques based on cooperative Machine Learning (ML) approaches have been proposed to secure the 4/5G networks [5–9]. These approaches leverage popular cooperative ML, such as deep and reinforcement learning algorithms, to detect known and zero-day attacks. Although these approaches exhibit high attack detection accuracy, they suffer from many overhead and privacy issues. Indeed, a huge amount of relevant training data is exchanged between the learning nodes, causing high computation and communication overheads during the training process. In addition, sharing datasets between learning nodes causes critical privacy issues in case of personal data leakage. To cope with these issues, a cooperative Federated Learning (FL) approach has been proposed. In the FL approach, distributed and centralized nodes running ML algorithms exchange between each other only the parameters of ML models, which drastically reduces communication overhead in training global models, as well as lowering privacy risks [10]. However, the FL-based defense systems should be carefully designed to handle the design requirement of 5GB network slicing, such as isolation, elasticity, and end-to-end optimization [11]. On the other hand, these systems could be hacked by attackers (such as the poisoning attack) to alter the training models used by the defense systems and hence lead these systems to provide false judgments against the monitored target, i.e., the malicious target is a legitimate node and vice versa [12].

To this end, this paper proposes a cooperative defense approach based on an FL approach carefully tailored to 5GB network slicing for securing the main slices' elements, such

as gNodeB, edge servers, and the core network functions. Specifically, distributed and hierarchical defense systems cooperate during the training and detection processes to detect the (Distributed) Denial of Service (DDoS) and Botnet attacks targeting the 5GB network slicing. Moreover, we propose a new security game model based on a mean-field approach to detect accurately the malicious defense systems infected by poisoning attacks. Our experimentation results show that cooperative defense systems based on the FL paradigm and mean-field game achieve a high level of efficiency and robustness against DDoS, Botnet, and poisoning attacks.

The main contributions of this paper can then be summarized as follows:

- We propose a two-layer FL-based architecture for detecting the most advanced network attacks targeting 5GB network slicing attacks while keeping security overhead low. The first FL layer consists of defense systems activated at gNodeB nodes as FL clients and defense systems activated at edge servers as FL servers, while the second layer consists of defense systems activated at edge servers as FL clients and defense systems activated at Access and Mobility Management Function (AMF) as an FL server to aggregate the global training model.
- We formulate a new security model based on mean field games to detect malicious defense systems launching poisoning attacks.
- We evaluate the detection of our defense systems in training to detect DDoS and Botnet 5GB slicing attacks and in deployment to detect the same attack, under-poisoning attacks, and time overhead.

The rest of this research work is organized as follows. Section 2 summarizes the current research works and highlight their advantages and weakness. Section 3 describes our proposed trusted 5GB network slicing defense systems based on two-layer FL architecture and mean-field game mechanisms. The performance evaluation results are presented in Section 4. A conclusion is given in Section 5.

2 Related Work

2.1 Cyber security systems based on cooperative machine learning algorithms

In [13], Liu et al. developed a secure FL algorithm based on a blockchain system to secure the communication between the distributed and centralized nodes to prevent external threats from executing an attack against the FL. In addition, they proposed a privacy detection technique to prevent the malicious distributed nodes from overhearing the sensitive communication exchanged between the legitimate nodes. In their simulation results, the proposed secure system can detect attacks targeting the messages exchanged between the cooperative nodes, improving the FL's security. In [14], Chai et al. aimed to combine the FL algorithm and blockchain system to secure the communication between the vehicles in the internet of vehicles network. The authors developed a new FL algorithm based on a non-cooperative game. This solution models the interaction between the security systems and attackers to get reliable and secure knowledge sharing that is used during the training process by the centralized and distributed nodes. Simulation results show that almost all attacks targeting knowledge sharing are detected by combining the FL algorithm and blockchain-enabled system. However, the drawback of the works [13, 14] is that the proposed secure systems cannot detect the internal threats that launch attacks against the centralized node. Therefore, when the attackers infect the centralized node, the training data shared between the cooperative nodes will be altered, and hence the accuracy of attack detection will be decreased promptly. In [15], Vinayakumar et al. proposed an intrusion detection framework based on a hierarchical deep neural network algorithm. The proposed detection framework is equipped with network and host Intrusion Detection Systems (IDSs) (N-IDS, H-IDS) to monitor and detect the attacks targeting the network and distributed devices. N-IDS and H-IDS hierarchically cooperate during training and detection, aiming to exchange relevant training data and enhance the attack detection rate over time. The performance of a detection framework is evaluated under various attack data sets. According to their simulation results, the framework exhibits a high classification accuracy and true positive rate against network and host attacks while reacting promptly against the detected attacks. Nguten et al. [16] proposed a distributed and cooperative anomaly detection framework to detect cyber-attacks targeting Internet of Things (IoT) devices. The detection framework relies on an unsupervised learning algorithm to build normal behavior during the training process. In the detection process, the proposed unsupervised learning algorithm aims to detect any deviation from that normal behavior to determine a new category of attacks. In the simulation results, almost of attacks targeting IoT devices are detected. The major weakness of the works [15, 16] is that the distributed detection systems deployed within the network are assumed to be trusted nodes. However, in a real case, these detection systems could be hacked and infected by the attackers, and hence they could provide false detection.

2.2 Attack detection framework based on AI systems to protect the 5G core and access networks

In [17], Wang et al. focused on detecting spoofing attacks targeting the radio access network of 5G architecture. The authors proposed a rule-based detection technique that relies on Euclidean distance to identify the spoofing devices targeting wireless communication. In their simulation, the authors proved that the accuracy detection against the proposed technique's spoofing attacks outperforms the current detection techniques. However, the rule-based detection technique cannot detect unknown spoofing attacks as the related attack signatures are not defined in the detection technique. In [18], Abdulqadder et al. proposed a new attack detection and prediction framework against the sophisticated attacks targeting the Software-Defined Networking (SDN), Network Function Virtualization (NFV), and edge computing of 5G architecture. The proposed detection and prediction technique relies on game theory, reinforcement learning algorithm, and Shannon entropy method to identify spoofing, overloading, Denial of Service (DoS), and hi-

jacking attacks. The network simulator NS3 is used to evaluate the performance of their framework. According to their simulation results, almost of all attacks are detected with high accuracy, i.e., high detection and low false-positive rates. However, the detection and prediction framework could require a high computation overhead to achieve this high level of security, which may not be suitable for the real-time use cases such as 5G network slicing for vehicular ad-hoc networks as the attacks should be detected promptly, i.e., with a low computation overhead. In [8], Sedjelmaci developed a new reinforcement learning algorithm adapted to the 5G core network to detect the DoS and botnet attacks. The author proposed a cooperative defense system based on the reinforcement learning algorithm to prevent the occurrence of collaborative and distributed DoS and botnet attacks. According to its simulation results, the accuracy protection rate is high; specifically when the number of DoS and botnets increases. In [19], Hachimi et al. focused on securing the 5G radio access network against jamming attacks. They developed an intrusion detection framework based on supervised Deep Learning (DL) and support vector machine algorithms to accurately detect attacks that jam the 5G wireless communication while considering the false positive rate. According to their experimental results, almost all jamming attacks are detected with low false positive and false negative rates. The weakness of the works [8, 19] is the authors did not consider the fact that the attackers could infect the distributed attack detection/prediction systems. Hence, a high false-positive could be generated.

In Table 1, we assess the research works cited above based on the following criteria: attack detection rate, false-positive rate, computation overhead, and privacy preservation. The rating medium, low and high are defined as follows: Attacks detection rates between [45%, 70% [, [70% , 80%[and [80%, 100%[are categorized respectively as Low, Medium, and High. False positive rates between [3%, 8%[, [8% , 18%[and [18%, 30%[are categorized respectively as Low, Medium and High. In this research work, we aim to circumvent the main issues cited above and propose a novel defense system relying on a trust-based two-layer federated learning algorithm to secure the 5G network slicing while considering the issues of false positive rates and computation overhead.

3 Cooperative Defense Systems Based on a Trusted Federated Learning Algorithm

Cooperative machine learning algorithms are classified into two techniques: centralized and distributed learning algorithms. In a centralized learning algorithm, the distributed nodes send their training data to the centralized node. This latter aggregates the training data of distributed nodes with its local training data. Then a global training model will be shared with cooperative nodes, i.e., centralized and distributed nodes. However, the major weakness of the centralized learning approach is the lack of data privacy and high communication overhead that the cooperative nodes could generate during the training process [20]. In a distributed learning algorithm, the centralized and distributed nodes cooperatively learn a shared training global model while keeping the training data at each node locally. The FL belongs to distributed learning as the most suitable learning algorithm that ensures the privacy of training data and guarantees a low communication overhead at each cooperative node [21]. The first subsection presents our hierarchical defense systems for 5GB network slicing based on a two-layer FL architecture against DDoS and Botnet attacks. In the second subsection, we focus on securing the defense system against poisoning attacks that aim to target the training data of the cooperative defense systems to lead them to provide false decisions.

3.1 Hierarchical defense systems based on a federated learning algorithm in 5G network slicing

As illustrated in Figure 1, we have three kinds of attack defense systems: First-Attacks Defense System (F-ADS), Second-Attacks Defense System (S-ADS), and Third-Attacks Defense System (T-ADS) that are deployed at gNodeB, edge server, and Access and Mobility Management Function (AMF) respectively. These defense systems cooperate to protect the 5G network slicing from external attacks (i.e., targeting wireless communication) and internal attacks (i.e., targeting the edge and core network functions). F-ADS monitors the link between the user equipment and gNodeB to detect malicious user equipment that targets wireless communication and gNodeB. S-ADS monitors the communication link between gNodeB and the edge server to detect the malicious gNodeB that hacks the wireless communication and edge server. T-ADS protects the AMF and legitimate edge servers from malicious edge servers. Specifically, T-ADS is activated at AMF since it is the first core network function that communicates with gNodeB and has a link with other 5G core network functions such as Session Management Function (SMF), Network Slice Selection Function (NSSF), Policy Control Function (PCF), and Unified Data Management (UDM). The security of 5G core network function is mandatory due to the sensitive data that they manage. Thereby, the cooperation detection process between F-ADS, S-ADS, and T-ADS prevents the execution of attacks targeting the core network. Hence, AMF, SMF, UDM, and PCF security are hardened.

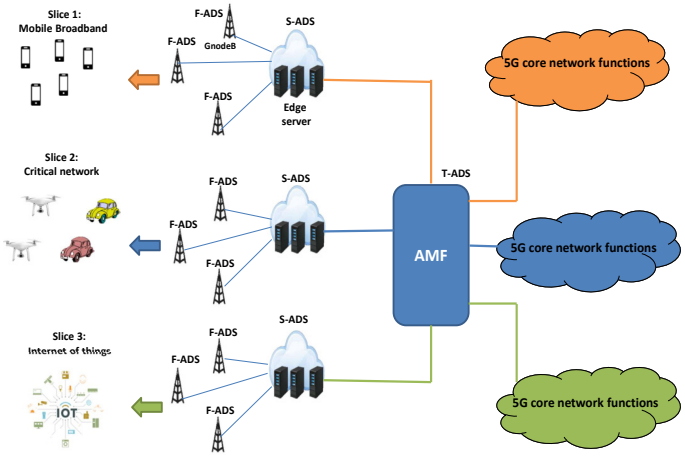


Fig. 1: Security architecture of 5GB network slicing

TABLE 1: Comparison among state-of-art attack detection works

Research work	Attack detection rate	False positive rate	Computation overhead	Privacy preservation
Liu et al. [13]	Medium	Low	Medium	Yes
Chai et al. [14]	Medium	Low	Medium	Yes
Vinayakumar et al. [15]	High	Medium	Medium	No
Nguten et al. [16]	High	Medium	Medium	No
Wang et al. [17]	Low	Low	Low	No
Abdulqadder et al. [18]	High	Low	High	No
Sedjelmaci [8]	High	Medium	Medium	No
Hachimi et al. [19]	High	Medium	Medium	No
Our work	High	Low	Low	Yes

3.1.1 Two-layer Federated Learning Security Model

As illustrated in Figure 2, F-ADSs, S-ADSs, and T-ADS cooperatively execute the FL algorithm to detect the attackers targeting the main segments of network slices, such as radio access, edge computing, and core network functions. The architecture we propose consists of two layers. The first FL layer consists of F-ADSs as FL clients and S-ADS as an FL server, while the second layer consists of S-ADSs as FL clients and T-ADS as an FL server to aggregate the global model. S-ADS and T-ADS provide two different aggregation results, as illustrated in Figure 2. Specifically, The FL models trained locally at F-ADS and S-ADS are called the first training and second training models. Moreover, the FL model aggregated at the S-ADS using the first training models of F-ADSs is called the Local training model. Similarly, the FL model generated at the T-ADS using the second training model of S-ADSs is called the Global training model. In our FL security model, each F-ADS and S-ADS have respectively the monitored data's matrices $M_l^i = [m_1^i, ..., m_{l_i}^i]$ and $M_{l'}^{i'} = [m_1^{i'}, ..., m_{l_{i'}}^{i'}]$, which are considered as the inputs matrix of FL algorithm, during the training process. Here, i and i' vary respectively from $1, .., S$ and $1, .., S'$, where S and S' are the maximum numbers of F-ADSs and S-ADSs deployed within the 5G network slicing. l_i is the number of simple data monitored by F-ADS and $l_{i'}$ is the number of simple data monitored by S-ADS. $M'^i_l = [m'^i_1, ..., m'^i_{l_i}]$ and $M'^{i'}_{l'} = [m'^{i'}_1, ..., m'^{i'}_{l_{i'}}]$ are respectively the outputs data's matrices of F-ADS' FL and S-ADS' FL algorithms, δ_i is the parameter weight vector of the first training model and $\delta'_{i'}$ is the parameter weight vector of the second training model. The objective function of the FL algorithm in the training process at F-ADS, S-ADS, and T-ADS levels are defined in equations 1 and 2 as in [21, 22]:

$$\arg\min_{\delta} \frac{1}{L} \sum_{i=1}^{S} \sum_{l=1}^{l_i} v(\delta_i, m_l^i, m'^i_l) \tag{1}$$

$$\delta_1 = \delta_2 = ... = \delta_S = \omega$$

$$\arg\min_{\delta'} \frac{1}{L'} \sum_{i'=1}^{S'} \sum_{l'=1}^{l_{i'}} v(\delta'_{i'}, m_l^{i'}, m'^{i'}_l) \tag{2}$$

$$\delta'_1 = \delta'_2 = ... = \delta'_{S'} = \omega'$$

Here $L = \sum_{i=1}^{S} l_i$ and $L' = \sum_{i'=1}^{S'} l_{i'}$ are the total size of training data used by the F-ADSs and S-ADSs, respectively. $v(\delta_i, m_l^i, m'^i_l)$ and $v(\delta'_{i'}, m_l^{i'}, m'^{i'}_l)$ are the error functions and defined as the accurate classification of the FL; in [21–23], the authors defined a set of error functions of the FL algorithms. ω is the parameter weight vector of the Local training model generated by the S-ADS and ω' is the parameter weight vector of the Global training model computed by the T-ADS as shown in Figure 2. To solve the equation (1) and (2), at each iteration t, each F-ADS uploads from the S-ADS the parameter weight vector ω^t and runs the gradient algorithm such as Stochastic Gradient Descent (SGD) [23] to generate the weight vector $\omega 1^{t+1}$ and computes the updated parameter weight, which is defined as $\delta_i^{t+1} = \omega 1^{t+1} - \omega^t$. δ_i^{t+1} is sent back to S-ADS to compute the updated parameter ω^{t+1} by averaging the parameters weights of F-ADSs, which is computed as shown in equation 3 [23]. Similarly, Each S-ADS uploads from the T-ADS the parameter weight vector ω'^t and executes the algorithm SGD to determine the vector $\omega 2^{t+1}$. Afterward, $\delta'^{t+1}_{i'}$ which is equal to $\omega 2^{t+1} - \omega'^t$ is sent to T-ADS to determine the updated parameter weight ω'^{t+1} as shown in equation 4.

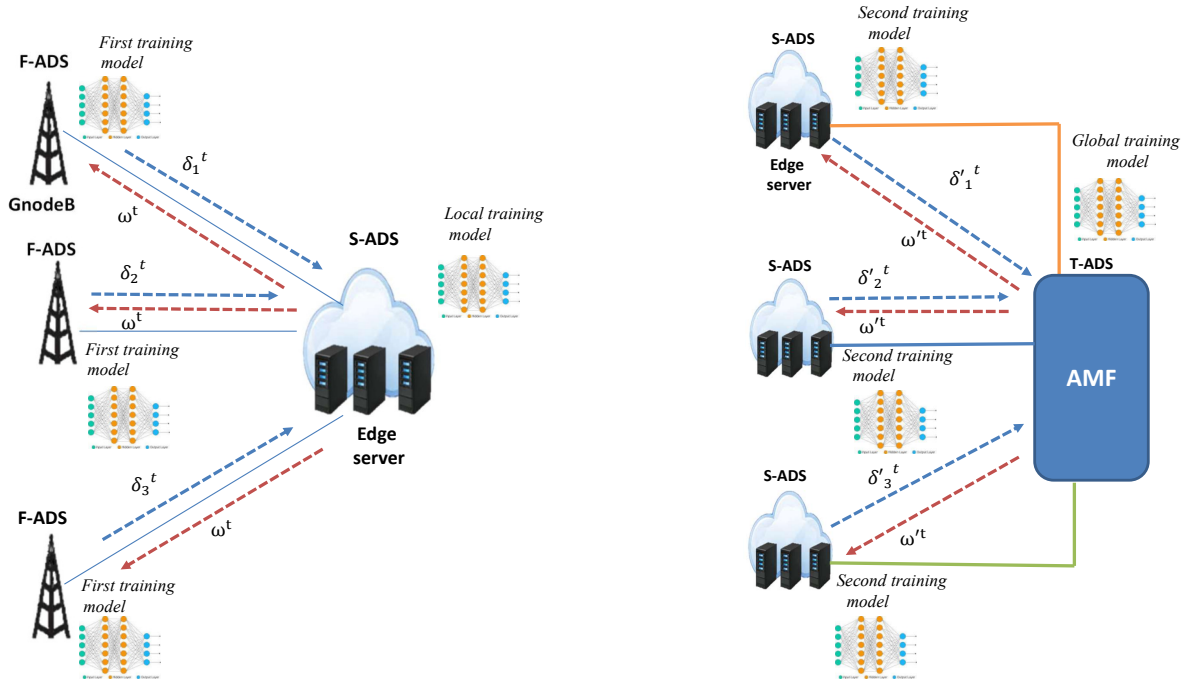
$$\omega^{t+1} = \sum_{i=1}^{S} \frac{l_i \delta_i^{t+1}}{L} \tag{3}$$

$$\omega'^{t+1} = \sum_{i'=1}^{S'} \frac{l_{i'} \delta'^{t+1}_{i'}}{L'} \tag{4}$$

3.1.2 Hierarchical and distrusted attacks detection

In this research, we focus on protecting 5GB network slicing from the most dangerous and complex attacks that could target the 5G, namely the DDoS and Botnet attacks. These attacks could be launched at radio access and edge server levels to create a group of devices targeting the main components of 5G architecture, such as gNodeB, edge servers, and 5G core network functions. The impact of these attacks can be catastrophic. Indeed, DDoS and botnet attacks can drop signaling messages, inject wrong data into messages, and send many packets to flood the edge servers and AMF for breaking critical 5GB network slicing services. For example, malicious gNodeB can generate a high signal strength to deceive the legitimate user equipment close to the destination that user equipment is looking for and frequently sends the unwanted packets to gNodeB and edge servers nodes to increase the computation overhead at these nodes.

F-ADS agents execute a multi-class DL algorithm for detecting malicious user equipment and infected gNodeB that run cyber-attacks. Here, we should mention security experts periodically intervene to label local data sets of F-ADS. When the F-ADS detects a suspected attack, an Anomaly message is forwarded to S-ADS, which includes



(a) between F-ADSs and S-ADS

(b) between S-ADSs and T-ADS

Fig. 2: Hierarchical defense systems based on FL algorithm: federated learning process

the identities of gNodeB and the suspected node targeted by an attacker, the detection time, and the features related to the suspected attack. The F-ADS, S-ADS, and T-ADS agents also use a multi-class DL algorithm for attack detection. This latter is based on a Deep Neural Network (DNN), where the network topology and optimal network parameters are defined in the performance evaluation section. The choice of this DNN algorithm lies in the fact that it exhibits a high accuracy classification, specifically when the amount of training data is important. This is the case with AMF and edge servers, as these network elements handle a huge amount of data. It is noted that F-ADS, S-ADS, and T-ADS classify the new incoming data into normal or attack (s) according to the Global training model determined during the training process of the FL algorithm, described in the previous subsection.

As indicated in the previous subsection, S-ADS monitors the behaviors of gNodeB nodes (located within its area) and edge servers (where the S-ADS is activated). In case an attack is detected at the edge server or/and gNodeB nodes, S-ADS executes a reaction action by informing the cyber defense center managed by the security experts to take a mitigation decision. The security experts analyze the attack information sent by S-ADS and make a final decision. In case when security experts confirm the attack, the malicious gNodeB or/and the infected edge server are removed from the network. The DL algorithm executed by S-ADS could categorize the target node as anomalous, i.e., it could be a malicious or legitimate node. Hence, S-ADS sends T-ADS an Attack message for further detection. The Attack message includes the identities of suspected gNodeB and edge server nodes, attack features, and detection time.

S-ADS analyzes the Anomaly message to verify whether the attack detected by F-ADS is valid or misclassified. S-ADS assigns a reputation value to each F-ADS, which is calculated as $R^{F-ADS} = \frac{R_{+}^{F-ADS} - R_{-}^{F-ADS}}{N^{F-ADS}}$, where R_{+}^{F-ADS} and R_{-}^{F-ADS} are the number of time that S-ADS agrees and does not agree on the attacks detection provided by F-ADS, respectively. N^{F-ADS} is the number of F-DASs interacting with S-ADS. The reputation value varies over time. However, the F-ADS is reported as malicious, and its detection judgment will not be considered when its reputation value is below a certain threshold during a specific time, defined by the security expert. As indicated in the previous subsection, the primary purpose of T-ADS is to protect the AMF and edge server from internal attacks. This is achieved by using a robust multi-class DL algorithm that monitors the traffic from edge servers and locally monitors the behavior of AMF. T-ADS blacklists the malicious edge server infected by the internal attacks. If an attack is detected at AMF, the T-ADS interacts with the cyber defense center for the final detection and decision-making toward the suspected AMF. In addition, T-ADS analyzes the Attack message sent by S-ADS to verify whether the detected anomaly is an attack. Similarly, T-ADS computes the reputation value related to each S-ADS that interacts with them, which is defined as $R^{S-ADS} = \frac{R_{+}^{S-ADS} - R_{-}^{S-ADS}}{N^{S-ADS}}$. This reputation value is used as an entry feature for the DL algorithm to assess the trust level of S-ADS when the Attack message is sent. S-ADS is detected as a malicious agent when the value of its reputation is below a certain threshold (defined by a security expert). The thresholds for judging the F-ADS and S-ADS as malicious agents can vary depending on the security level the security experts seek. For example, a high false positive rate could be generated when the threshold is high.

However, the attack detection rate will be very low when the threshold is low. Hence, security experts should ensure a trade-off between the false positive rate and the attack detection rate when they select the threshold. To prevent the passive attack from overhearing the security messages (Anomaly and Attack messages) exchanged between F-ADS, S-ADS, and T-ADS, the security mechanism based on Elliptic Curve Cryptography [24] is used to encrypt these security messages. The pseudocode of cooperative attack detection is shown in Algorithm 1.

Algorithm 1: Cooperative attacks detection process

```
Begin (at t=0) ;
F-ADS, S-ADS, and T-ADS monitor their neighborhood areas. ;
Classify the new incoming features according to the FL Global training model (with parameter ω'^{t+1}). ;
if (F-ADS detects an attack) then
    Forwards Anomaly message to S-ADS.;
    if (R^{F-ADS} > 0.5 & S-ADS confirms the attack) then
        Reaction is executed by S-ADS;
    else
        if (S-ADS does not confirm the attack) then
            R^{F-ADS} is decreased ;
        else
            if (R^{F-ADS} < 0.5) then
                F-ADS is removed;
            end
        end
    end
end
if (S-ADS detects attack/anomaly) then
    S-ADS executes reaction action against an attack, and an Attack message is forwarded to T-ADS for anomaly checking.;
    if (R^{S-ADS} > 0.5 & T-ADS detects new attack) then
        R^{S-ADS} is increased and T-ADS interacts with security experts against the detected attack. ;
    else
        if (T-ADS does not confirm the attack) then
            R^{S-ADS} is decreased ;
        else
            if (R^{S-ADS}) < 0.5) then
                S-ADS is removed ;
            end
        end
    end
end
```

3.2 Trusted security systems based on a mean field game

The defense systems, F-ADS, S-ADS, and T-ADS could be infected by the attackers. Hence, these defense systems could provide false judgments against the legitimate target, i.e., the legitimate monitored target is malicious and vice versa. As explained by the authors in [20], in an FL approach, it is hard for the malicious nodes (e.g., malicious F-ADS, S-ADS, and T-ADS agents) to provide a false training model (e.g., first and second training models) without being detected by the defense systems. However, security experts from Stanford University [22] developed a new kind of threat against FL algorithm named poisoning attack that the malicious defense systems could execute to send fake training models to centralized defense systems, e.g., S-ADS and T-ADS. To execute the poisoning attack at F-ADS and S-ADS, for instance, the malicious F-ADS and S-ADS agents attempt first to determine the parameters ω^{t+1} of the Local training model and ω'^{t+1} of the Global training model respectively. However, the non-legitimate defense systems can't get the current values of the parameters, ω^{t+1} and ω'^{t+1}, due to the private nature of the FL approach. In this case, the malicious F-ADS and S-ADS agents estimate respectively the values of parameters $\omega^{t+1^{\phi}}$ and $\omega'^{t+1^{\phi}}$, which are equal respectively to $f(\vartheta^t)$ and $f(\vartheta'^t)$ [22], where ϑ^t and ϑ'^t are the information related to the training models that the malicious F-ADS and S-ADS agents have at time t, respectively. The function f used to determine the estimated parameters is defined in [22]. Based on the obtained parameters $\omega^{t+1^{\phi}}$ and $\omega'^{t+1^{\phi}}$, the defense system injects fake training models that are not detected by the FL algorithm and hence wrong Local and Global training models will be generated respectively at legitimate F-ADS and S-ADS. Therefore, malicious defense systems could provide false judgments against legitimate targets without being detected by F-ADSs, S-ADSs, and T-ADS. To detect accurately the malicious defense systems that run the poisoning attack, a new security model based on a mean field game is proposed. The mean field game theory is a useful mathematical tool that models the interactions between a large number of players and determines the optimal decisions of the players, e.g., the attackers attack or stay in idle mode, and defense systems react or stay in idle mode. In this subsection, we first define the proposed security model based on a mean field game. Afterward, we present the best responses of the competitive players, i.e., the defense systems attempt to secure the network, and the malicious defense systems launch the poisoning attacks against the FL algorithms executed by the legitimate F-ADSs, S-ADSs, and T-ADS. In the end, we prove the existence of an equilibrium between the competitive players and propose a detection process algorithm against the poisoning attack.

3.2.1 Security game model

The security model is modeled as an N + M mean field game, where N is the number of trusted defense systems and M is the number of malicious defense systems infected by the poisoning attack. In this security game, we assume that we have N trusted defense systems, where the security expert frequently monitors their detection process to ensure that the attackers do not infect these systems. ψ_i^1 and ψ_j^2 are respectively the trusted and malicious detection players, where $i = 1, \ldots, N$ and $j = 1, \ldots, M$. $S^1 = s_1^1, \ldots, s_k^1$ and $A^1 = A_1^1, \ldots, A_l^1$ are respectively the states and strategies of player ψ_i^1. S^1 corresponds to the monitoring states of player ψ_i^1 against suspected players ψ_j^2, i.e., activating the

monitoring process or switching to the idle mode. The strategies A^1 are the reactions of player ψ_i^1 against the suspected players ψ_j^2, i.e., ψ_j^2 are detected as attackers that carry out a poisoning attack. It is noted that the suspected players are the defense systems that could be acting as malicious agents (by executing poisoning attacks) or legitimate systems. $S^2 = s_1^2, \ldots, s_{k'}^2$ and $A^2 = A_1^2, \ldots, A_{l'}^2$ are respectively the states and strategies of player ψ_j^2. S^2 corresponds to the attack states of player ψ_j^2 against players ψ_i^1, i.e., activating the attack process or staying in idle mode. A^2 are the poisoning attacks executed by player ψ_j^2. u_i^1 is the payoff of the player ψ_i^1, which is computed as $\alpha.X - \beta.Y$, in which X is the number of poisoning attacks (executed by ψ_j^2) detected by the players ψ_i^1 and Y is the false detection rate against the legitimate players ψ_i^1. α and $\beta \in$]0,1] are the weights parameters. The payoff u_j^2 of player ψ_j^2 is equal to $-u_i^1$. The average payoffs of the players ψ_i^1 and ψ_j^2 in the mean-field game were defined respectively as $U^1(t)$ (see equation 5) and $U^2(t)$(see equation 6).

$$U^1(t) = (U_1^1(t), \ldots, U_l^1(t)) \tag{5}$$

$$U^2(t) = (U_1^2(t), \ldots, U_{l'}^2(t)) \tag{6}$$

Where, $U_l^1(t) = \dfrac{(\sum_{i=1}^{N} u_i^1(t))}{N}$ and, $U_{l'}^2(t) = \dfrac{(\sum_{j=1}^{M} u_j^2(t))}{M}$

We define $\rho^1(a^1|b^1, c^1)$ and $\rho^2(a^2|b^2, c^2)$ as the transition probabilities of players, ψ_i^1 and ψ_j^2, respectively as shown in equations 7 and 8.

$$\rho^1(a^1|b^1, c^1) = P(u_i^1(t+1) = a^1 | u_i^1(t) = b^1, A_i^1(t) = c^1) \quad \text{Where } a^1, b^1 \in S^1 \text{ and } c^1 \in A^1. \tag{7}$$

$$\rho^2(a^2|b^2, c^2) = P(u_j^2(t+1) = a^2 | u_j^2(t) = b^2, A_j^2(t) = c^2) \quad \text{Where } a^2, b^2 \in S^2 \text{ and } c^2 \in A^2. \tag{8}$$

The utility function of the player ψ_i^1 is defined as $\phi^1(u_i^1(t), A_i^1(t), S_i^1(t), U^1(t), U^2(t))$ and $\phi^2(u_j^2(t), A_j^2(t), S_j^2(t), U^2(t), U^1(t))$ is the utility function for player ψ_j^2. The average payoff $U^2(t)$ impacts the utility function ϕ^1 since when the malicious players (ψ_j^2) are not detected accurately, $U^2(t)$ increases rapidly and hence leads to the decrease of $U^1(t)$ and ϕ^1. Similarly, $U^1(t)$ impacts the utility function ϕ^2, specifically when almost all malicious players ψ_j^2 are detected accurately.

3.2.2 Best response of players in the mean-field game

The best responses of players ψ_i^1 and ψ_j^2 correspond to the maximization of their respective utility functions, ϕ^1 and ϕ^2, as shown in the mean-field equation system, in equations 9 and 10.

$$\phi^{*1}(u_i^1(t), U^1(t), U^2(t)) = \max_{A_i^1(t) \in A^1, S_i^1(t) \in S^1} \phi^1(u_i^1(t), A_i^1(t), S_i^1(t), U^1(t), U^2(t)) \tag{9}$$

$$\phi^{*2}(u_j^2(t), U^2(t), U^1(t)) = \max_{A_j^2(t) \in A^2, S^{2_j}(t) \in S^2)} \phi^2(u_j^2(t), A_j^2(t), S_j^2(t), U^2(t), U^1(t)) \tag{10}$$

From equations 9 and 10, it is apparent that to determine the best responses, the players should estimate the expected values of the average payoffs $U^2(t)$ and $U^1(t)$, in addition to their respective payoffs' values, $u_i^1(t)$ and $u_j^2(t)$. However, the computation of functions $U^2(t)$ and $U^1(t)$ are very difficult and require a high computation overhead for the defense system to determine the values of these functions, especially when the number of players ψ_i^1 and ψ_j^2 is high, i.e., N and M $\to \infty$. This is mainly due to the scalability of 5G network slicing, as it is not easy for the defense systems to promptly determine the value of $U^2(t)$ when the number of edge servers is high. Therefore, to overcome the computation overhead, the average payoffs $U^2(t)$ and $U^1(t)$ are approximated to the functions ξ^1 and ξ^2 as demonstrated in Theorem 1. ξ^1 and ξ^2 are defined as limiting processes [25].

Theorem 1. *$A^{*1} = A_1^1, \ldots, A_{l1}^1$ and $A^{*2} = A_1^2, \ldots, A_{l2}^2$ are respectively the optimal strategies spaces of the players, ψ_i^1 and ψ_j^2; where states $i^{'} = 1, \ldots, l1 \in S^1$ and states $j^{'} = 1, \ldots, l2 \in S^2$. $\xi^1 = (\xi_1^1, \ldots, \xi_{l1}^1)$ and $\xi^2 = (\xi_1^2, \ldots, \xi_{l2}^2)$. When N and $M \to \infty$ the average payoffs $U^2(t)$ and $U^1(t)$ are equal respectively to equations 11 and 12.*

$$U^1(t) = \sum_{i^{'}=1}^{l1} \xi_{i^{'}}^1 . \rho^1(1|i^{'}, A^{*1}), ..., \sum_{i^{'}=1}^{l1} \xi_{i^{'}}^1 . \rho^1(l1|i^{'}, A^{*1}) \tag{11}$$

$$U^2(t) = \sum_{j^{'}=1}^{l2} \xi_{j^{'}}^2 . \rho^2(1|j^{'}, A^{*2}), \ldots, \sum_{j^{'}=1}^{l2} \xi_{j^{'}}^2 . \rho^2(l2|j^{'}, A^{*2}), \tag{12}$$

Proof. $u_i^1(t)$ and $u_i^1(t+1)$ are conditionally independent for $i = 1$ to N. Similarly, $u_j^2(t)$ and $u_j^2(t+1)$ are conditionally independent for $j = 1$ to M. At states $i^{'} = l1$ and $j^{'} = l2$, the optimal payoffs $u_i^{*1}(t)$ and $u_j^{*2}(t)$ (with the related strategies spaces, A^{*1} and A^{*2}) are determined by the players ψ_i^1 and ψ_j^2 with transition probabilities equal respectively to $\rho^1(l1|i^{'}, A^{*1})$ and $\rho^2(l2|j^{'}, A^{*2})$. The probabilities $\rho^1(l1|i^{'}, A^{*1})$ and $\rho^2(l2|j^{'}, A^{*2})$are considered respectively as one of $N.U^1(t)$ and $M.U^2(t)$ distributions. Thus, the conditional means E^1 and E^2 of the players ψ_i^1 and ψ_j^2 are defined in equations 13 and 14.

$$E^1(t) = \sum_{i^{'}=1}^{l1} u_{i^{'}}^1 . \rho^1(1|i^{'}, A^{*1}), \ldots, \sum_{i^{'}=1}^{l1} u_{i^{'}}^1 . \rho^1(l1|i^{'}, A^{*1}) \tag{13}$$

$$E^2(t) = \sum_{j^{'}=1}^{l2} u_{j^{'}}^2 . \rho^2(1|j^{'}, A^{*2}), \ldots, \sum_{j^{'}=1}^{l2} u_{j^{'}}^2 . \rho^2(l2|j^{'}, A^{*2}) \tag{14}$$

The subtractions of U^1 (t) with E^1 (t) and U^2 (t) with E^2 (t) are equal to zero when N and $M \to \infty$. Thus,

the solutions of equations 11 and 12 are verified. From equations 13 and 14, we get equations 15 and 16.

$$\rho^{*1}(A^{*1}) = \begin{bmatrix} \rho^1(1|1, A^{*1}) & ... & \rho^1(l1|1, A^{*1}) \\ \vdots & \ddots & \vdots \\ \rho^1(1|l1, A^{*1}) & ... & \rho^1(l1|l1, A^{*1}) \end{bmatrix} \quad (15)$$

$$\rho^{*2}(A^{*2}) = \begin{bmatrix} \rho^2(1|1, A^{*2}) & ... & \rho^2(l2|1, A^{*2}) \\ \vdots & \ddots & \vdots \\ \rho^2(1|l2, A^{*2}) & ... & \rho^2(l2|l2, A^{*2}) \end{bmatrix} \quad (16)$$

The average optimal payoffs U^{*1} (t) and $U^{*2}(t)$ of the players ψ_i^1 and ψ_j^2 when N and M $\to \infty$ are equal respectively to equations 17and 18.

$$U^{*1}(t) = \sum_{i'=1}^{l} 1\xi_{i'}^1 . \rho^{*1}(A^{*1}) \quad (17)$$

$$U^{*2}(t) = \sum_{j'=1}^{l2} \xi_{j'}^2 . \rho^{*2}(A^{*2}) \quad (18)$$

From equations 9 and 10, the best responses of players ψ_i^1 and ψ_j^2 are formulated as in equations 19 and 20.

$$\phi^{*1}(u_i^1(t), U^{*1}(t), U^{*2}(t)) = \max_{(A_{i'}^{*1}(t) \in A^{*1}, S_i^{*1}(t) \in S^{*1})} \phi^1(u_i^1(t), A_{i'}^{*1}(t), S_i^{*1}(t), U^{*1}(t), U^{*2}(t)) \quad (19)$$

$$\phi^{*2}(u_j^2(t), U^{*2}(t), U^{*1}(t)) = \max_{A_{j'}^{*2}(t) \in A^{*2}, S_j^{*2}(t) \in S^{*2}} \phi^2(u_j^2(t), A_{j'}^{*2}(t), S_j^{*2}(t), U^{*2}(t), U^{*1}(t)) \quad (20)$$

□

3.2.3 Mean-field equilibrium in the security game

The equilibrium in a mean-field game, named Nash-Mean Field Equilibrium (MFE), is the approximation of Nash equilibrium in a non-cooperative game when the number of competitive players is large. Determining a Nash equilibrium in a mean-field game requires a high computation overhead. The major advantage that Mean Field Game (MFG) offers is the simplification of the equilibrium's computability at a large scale game, i.e., the number of players is large, and hence a low computation overhead is required for the player, e.g. defense system to determine the equilibrium. Such approximation of Nash equilibria works in two phases. First the optimization of the game with N and M players, then the passage to the limit with N and M $\to \infty$.

Theorem 2. *The strategies' couple $(A_{i'}^{*1}(t), A_{j'}^{*2}(t))$ and the average payoffs' couple $(U^{*1}(t), U^{*2}(t))$ constitute the MFE.*

Proof. According to [26] MFE exists if: $(A_{i'}^{*1}(t), A_{j'}^{*2}(t))$ is an optimal strategies' couple given $(U^{*1}(t), U^{*2}(t))$ and the average payoffs' couple $(U^1(t), U^2(t))$ is a steady state distribution of $(A_{i'}^{*1}(t), A_{j'}^{*2}(t))$. The utility functions ϕ^1 and ϕ^2 can respectively be expressed in equations 21 and 22 as:

$$\phi^1(u_i^1(t), A_i^1(t), S_i^1(t), \xi^1(t), \xi^2(t)) = f_i(u_i^1(t), A_i^1(t), S_i^1(t)) + \xi^1(t). \sum_{i=1}^{N} u_i^1 - \xi^2(t). \sum_{j=1}^{M} u_j^2 \quad (21)$$

$$\phi^2(u_j^2(t), A_j^2(t), S_j^2(t), \xi^2(t), \xi^1(t)) = g_j(u_j^2(t), A_j^2(t), S_j^2(t)) + \xi^2(t). \sum_{j=1}^{M} u_j^2 - \xi^1(t). \sum_{i=1}^{N} u_i^1 \quad (22)$$

The poisoning attacks could target several defense systems and the legitimate defense systems could detect at the same time the poisoning attacks, so the payoff of players ψ_i^1 and ψ_j^2 are computed as $\sum_{i=1}^{N} u_i^1$ and $\sum_{j=1}^{M} u_j^2$, respectively.

To solve the mean field equation systems, equations 9 and 10 and equations 21 and 22, we use the dynamic programming method. From equations 9 and 21, the strategies of players ψ_i^1 are obtained as $A_i^1(t) = A_{i'}^1(t) = A_1^1, \ldots, A_{l1}^1$, where $i^{'} = 1, \ldots, l1$. Similarly, from Eqs. (10) and (22), the strategies of players ψ_j^2 are obtained as $A_j^2(t) = A_{j'}^2(t) = A_1^2, \ldots, A_{l2}^2$, where $j^{'} = 1, \ldots, l2$. Therefore, the strategies $(A_{i'}^{*1}(t), A_{j'}^{*2}(t))$ are the optimal strategies' couple of players ψ_i^1 and ψ_j^2 given the average payoffs' couple ($U^{*1}(t)$, $U^{*2}(t)$). This latter is approximated to ($\xi^1(t)$, $\xi^2(t)$).

From equations 17 and 18, the average optimal payoffs could be expressed as shown in equations 23 and 24, as $\sum_{i'=1}^{l1} \xi_{i'}^1$ and $\sum_{j'=1}^{l2} \xi_{j'}^2$ are the approximations of $U^{*1}(t)$ and $U^{*2}(t)$, respectively.

$$U^{*1}(t) = U^{*1}(t) . \rho^{*1}(A^{*1}) \quad (23)$$

$$U^{*2}(t) = U^{*2}(t) . \rho^{*2}(A^{*2}) \quad (24)$$

Therefore, we claim that $U^{*1}(t)$ is a steady state distribution of $A_{i'}^{*1}(t)$ and $U^{*2}(t)$ is a steady state distribution of $A_{j'}^{*2}(t)$.

□

As a conclusion, the MFE exists, and when this equilibrium is reached the optimal strategies of players ψ_i^1 and ψ_j^2 are defined respectively as $A_1^1, \ldots, A_{l1}^1$ and $A_1^2, \ldots, A_{l2}^2$. F-ADSs, S-ADSs, and T-ADS categorize the suspected ψ_j^2 as a malicious defense system executing a poisoning attack when MFE is reached, i.e., $U^1(t) = U^{*1}(t)$ and $U^2(t) = U^{*2}(t)$, and $u_j^2(t) = \max_Y(\beta.Y - \alpha.X) = u_j^{*2}(t)$. The pseudocode of the detection process against poisoning attack based on a mean field game is illustrated in Algorithm 2

Algorithm 2: Poisoning attack detection process

Begin (at t=0);
repeat
  Players ψ_i^1 (legitimate defense systems) compute their $u_i^1(t)$;
  if $(u_i^1(t) > 0)$ **then**
    Each players ψ_i^1 computes $\sum_{i'=1}^{l1} \xi_{i'}^1$ and estimates $\sum_{j'=1}^{l2} \xi_{j'}^2 \cdot$;
    if $(\sum_{i'=1}^{l1} \xi_{i'}^1 > 0$ *and* $\sum_{j'=1}^{l2} \xi_{j'}^2 > 0)$ **then**
      ψ_j^2 is suspected to carry out a poisoning attack;
      if $(\xi^1(t) = \xi^{*1}(t)$ *and* $\xi^2(t) = \xi^{*2}(t))$ **then**
        Players ψ_i^1 compute $u_j^2(t)$;
        if $(u_j^2(t) = \max(\beta.Y - \alpha.X))$ **then**
          Player ψ_j^2 is a malicious defense system that carries out a poisoning attack;
        end
      end
    end
  end
until *the end of the mean-field security game*;

4 Performance evaluation

This section evaluates the performance of our defense systems against 5GB network slicing attacks. We first evaluate the performance of our FL collaborative training model. We then deploy the model and evaluate attack detection accuracy and overhead while considering poisoning attacks based on mean field algorithms.

4.1 Training results

To evaluate the training performance of our scheme, we have implemented a two-layer FL architecture with three network slices using Tensorflow and Keras Python libraries. Specifically, we set up two configurations. The first configuration (Config 1) consists of one T-ADS (layer 2) and two S-ADSs (layer 1) with three F-ADS each. The second configuration (Config 2) consists of one T-ADS and three S-ADSs with five F-ADS each. We also limited the FL rounds for S-ADSs and T-ADS to 100. The global models of S-ADSs and T-ADS were trained on the Google Colab platform using Compute Engine backend (TPU). The F-ADSs have been implemented as Tensorflow instances running local models. For the multi-class classification, we use a DL model. We selected the CSE-CIC-IDS-2018 dataset [27] to train our DL model since it adequately covers different types of network attacks ((D)DoS, Botnet), addressed in this paper. To train our DL model, we have a dataset containing a total of $208,186$ instances (rows) split as follows (i) $49,971$ instances are Benign, (ii) $28,619$ instances are Botnet, and (iii) $129,596$ instances are D(DoS) including $34,300$ instances of DDoS HOIC attack, $28,809$ instances of DDoS LOIC-HTTP attack, $13,989$ instances of DoS SlowHTTPtest attack, $41,508$ instances of DoS GoldenEye attack, and $10,990$ instance of DoS Slowloris attack.

The dataset initially included 80 features on network flows. This number became 85 after converting the timestamp feature to the date format. We also rescaled the dataset in the range of [0,1] using MinMaxScaler to speed up the training process. The dataset was split into training, validation, and test sub-datasets. Specifically, we have chosen 10% of the whole dataset as a validation dataset and 10% as a test dataset. We have also tested two scenarios. The first scenario is when the data is Independent and Identically Distributed (IID) over the network slices. The second scenario is the non-IID, specifically:

- Slice 1 comprises 80% of DoS attacks-GoldenEye. The remaining 20% is, for slice two in Config 1 and equally shared between slices two and three in Config 2.
- Slice 2 comprises 80% of DDOS attack-HOIC. The remaining 20% is, for slice one in Config 1 and equally shared between slices one and three in Config 2.
- Slice 3 (applicable only for Config 2) comprises 80% of DoS Bot attacks. The remaining 20% is equally shared between slices one and two.
- All the remaining attacks are equally shared over the slices.

Our model consists of (i) an inputs layer with 84 neuron nodes, (ii) two hidden layers with 85 and 42 hidden nodes for each of them, respectively, with a dropout rate set to 0.75, (iii) an output layer with 7 nodes based on one hot encoding to detect and identify attacks. The ReLU activation function is used for the hidden nodes, while the softmax function is used for the output layer. We have used the Stochastic Gradient Descent (SGD) with a learning rate of 0.01 to calculate local models' weights. After each round, the Federated Averaging is used to compute the global model's weights. Table 2 lists the hyperparameters of the model. We note that each F-ADS (FL-client) handles around $11,103$ instances in the training phase. Also, it takes around 2 hours to train S-ADSs simultaneously and 1 hour to train T-ADS.

TABLE 2: Training parameters of the global model

Parameter	Value
The ratio of validation/test dataset	10%
Learning rate	0.01
Batch size	32
Optimizer	SGD
Dropout	0.75
# Rounds	100

We selected accuracy, precision, recall, and F1-score as evaluation metrics. The training process performance based on IID and non-IID dataset-splitting for Config 1 is illustrated in Figures 3a and Figures 3b. The accuracy and loss values versus the number of rounds are shown in Figure 3a (a) (3b (a)) and Figure 3a (b) (3b (b)), respectively. Figure 3a (b) (3b (b)) also compares the loss values at T-ADS with those at S-ADS 1 and S-ADS 2, respectively. It can be observed that the loss decreases as the number of rounds increases in general. Moreover, it can be seen that the loss at T-ADS drops rapidly since the first rounds and reaches

the lowest values compared with S-ADSs (1 and 2). Furthermore, it can be seen in Figure 3a (a) (3b (a)) that all S-ADSs (1,2) achieve above 87% accuracy at the end of training. It can also be seen that S-ADS 2 (S-ADS 1 in 3b (a)) performs better than the other S-ADS. This is because of the quality of data used by the two F-ADSs connected to the S-ADS 2 (S-ADS 1 in 3b (a)), which results in building good local models and then having a good global model at S-ADS 2 (S-ADS 1 in 3b (a)) at the first aggregation phase. Moreover, it can be seen that T-ADS performs better than all S-ADSs (1, 2) since at the second aggregation stage, T-ADS uses all models built at S-ADSs to build the global model achieved above 97% at the end of the training stage. Figures 3c and Figures 3d show the performance evaluation of the training process based on IID and non-IID dataset-splitting. Figure 3c (a) (3d (a)) and Figure 3c (b) (3d (b)) show the obtained accuracy values and loss values, respectively, versus the number of rounds. Figure 3c (b) (3d (b)) compares the loss values at T-ADS with those at S-ADS 1, S-ADS 2, and S-ADS 3, respectively. As a general observation, we can see that the loss decreases as the number of rounds increases. In addition, we can see that loss at T-ADS quickly decreases since the first rounds and achieves the lowest values compared with S-ADSs (1, 2, and 3). On the other hand, we can see in Figure 3c (a) (3d (a)) that all S-ADSs (1,2,3) achieve at least 87% (91% in 3d (a)) accuracy at the end of training. We can also see that S-ADS 2 (S-ADS 1 in 3d (a)) outperforms the two other S-ADSs. This is due to the quality of data used by the five F-ADSs connected to the S-ADS 2 (S-ADS 1 in 3d (a)), which leads to building good local models and then having a good global model at S-ADS 2 (S-ADS 1 in 3d (a)) at the first aggregation phase. Moreover, we can see that T-ADS outperforms all S-ADSs (1, 2, 3) since at the second aggregation stage, T-ADS exploits all models built at S-ADSs to build the global model achieved more than 97% at the end of the training stage.

TABLE 3: Attack detection results

Scenario	Model	Accuracy	Precision	Recall	F1-score
Config 1 (IID)	**S-ADS 1**	0.90	0.93	0.83	0.86
	S-ADS 2	0.95	0.96	0.93	0.94
	T-ADS	**0.97**	**0.97**	**0.96**	**0.96**
Config 1 (non-IID)	**S-ADS 1**	0.92	0.94	0.90	0.91
	S-ADS 2	0.91	0.94	0.88	0.89
	T-ADS	**0.97**	**0.97**	**0.96**	**0.96**
Config 2 (IID)	**S-ADS 1**	0.87	0.78	0.78	0.77
	S-ADS 2	0.95	0.95	0.92	0.93
	S-ADS 3	0.89	0.78	0.78	0.78
	T-ADS	**0.97**	**0.97**	**0.96**	**0.96**
Config 2 (non-IID)	**S-ADS 1**	0.96	0.96	0.94	0.95
	S-ADS 2	0.94	0.95	0.91	0.92
	S-ADS 3	0.91	0.95	0.88	0.90
	T-ADS	**0.97**	**0.97**	**0.96**	**0.96**

Table 3 gives performance measures of both of the two configurations (Config 1 and Config 2) for all S-ADSs together with T-ADS on the test dataset for IID and non-IID splitting methods. As we can see, for both configurations, T-ADS outperforms all S-ADSs, achieving 97% and 96% accuracy and F1-score, respectively, similarly in both cases (IID and non-IID). These results demonstrate how effectively our scheme recognizes attack occurrences that haven't been observed previously.

4.2 Detection and overhead results

F-ADS, S-ADS, and T-ADS are executed at gNodeB, edge server, and AMF levels. In our case study, the number of attack defense systems deployed within the network equals six F-ADS agents, two S-ADS agents, and one T-ADS agent. To execute the poisoning attacks, we are inspired by the work in [22]. We consider a set of malicious devices that target the training models of FL by modifying their training vectors and labels. The main metrics that are used in the evaluation process are defined as follows:

- **Accuracy Detection Rate (ADR):** is the number of attacks detected minus the number of false detection generated by the defense systems, i.e., false positive and false negative rates. The average accuracy detection rate is computed as $A = \frac{(\sum_{i=1}^{S}(\alpha^x.D_i - \beta^x.F_i))}{(S.TL)}$, where D_i and F_i are respectively the attack detection and false detection rates generated by the defense system i, S is the number of defense systems and TL is the total number of attacks. Here, α^x and $\beta^x \in]0,1]$ are the weight factors of detection and β^x=1-α^x. The value of α^x depends on the severity of the detected attack. In this study, we assume that the poisoning attack is a more severe threat than (D)DoS and botnet, where α^x=1 for poisoning attack detection while for (D)DoS and botnet attacks' detection α^x is equal respectively to 0,7 and 0,5.
- **Time Overhead (TO)**: is defined as the required time of the defense systems to detect the (D)DoS, botnet, and poisoning attacks accurately. The average time overhead is computed as $O = \frac{\sum_{i=1}^{S} O_i)}{S.TL}$, where O_i is the time overhead the defense system i generates.

4.2.1 Accuracy detection

Figures 4 and 5 show respectively results of accuracy detection against the (D)DoS, botnet, and poisoning attacks of our cooperative defense systems while comparing with two relevant state-of-the-art works [8, 13] exhibit.

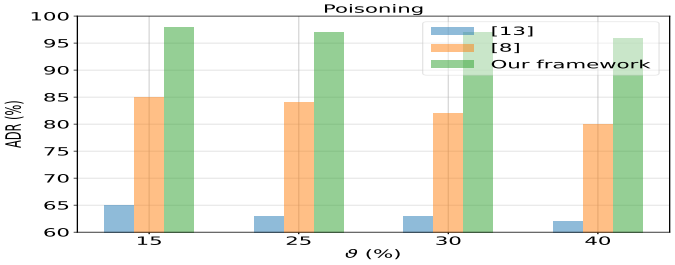
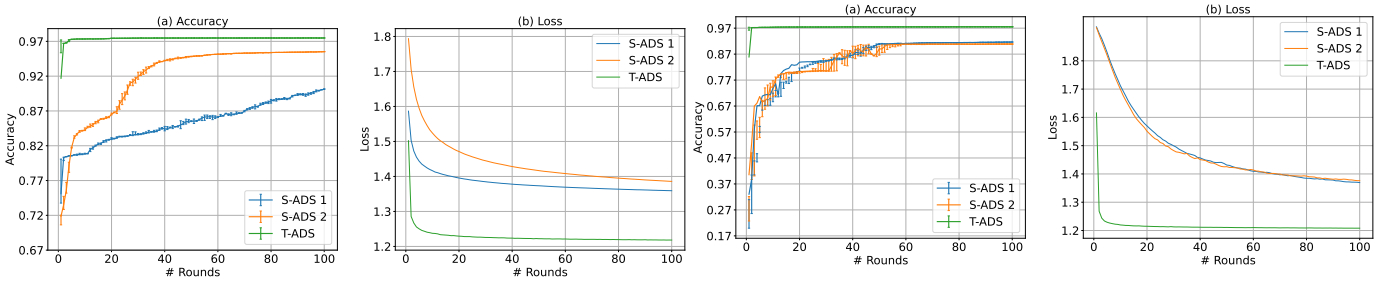


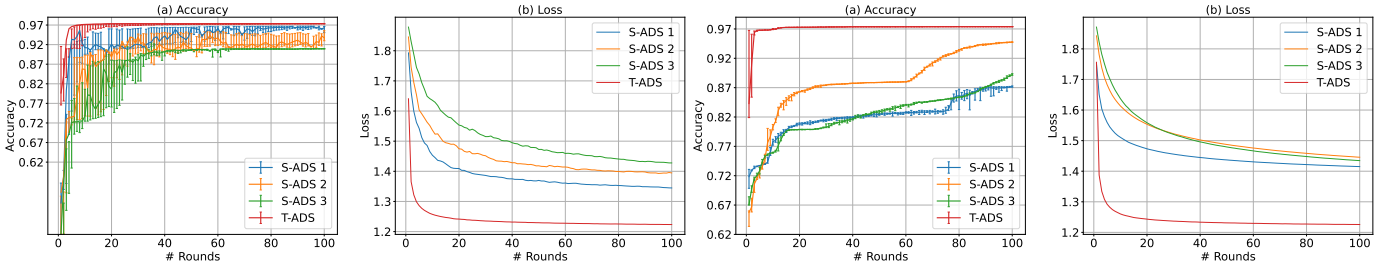
Fig. 5: Accuracy detection rates in the presence of poisoning attacks

Figure 5 shows results of accuracy detection against poisoning attacks. We define ϑ as the percentage of the information (training vectors and labels) modified by the attackers, where ϑ varies from 5%, 10%,20%, and 25%. As we can see, our proposed cooperative defense system exhibits a high accuracy in detecting poisoning attacks compared to related works. Figure 4 shows results of accuracy detection against (D)DoS and botnet network attacks. Here, we vary the amount of malicious traffic instigated by the



(a) Config 1 (the IID case)

(b) Config 1 (the non-IID case)



(c) Config 2 (the IID case)

(d) Config 2 (the non-IID case)

Fig. 3: Model performance in the training phase

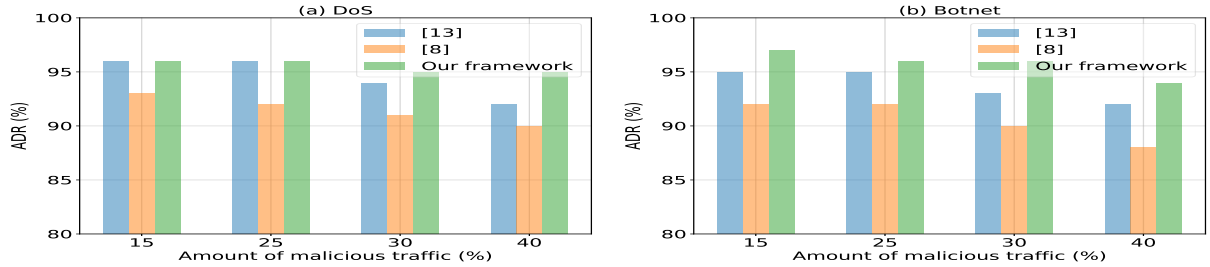


Fig. 4: Accuracy detection rates in the presence of a) (D)DoS and b) Botnet

(D)DoS and botnet attacks from 15% to 40% of the total traffic. As illustrated in Figures 4 and 5, even though the accuracy detection rates of all works are high when (D)DoS and botnet attacks occur, the accuracy detection rates of the related works [8, 13] are low under poisoning attacks as compared to our cooperative systems. These results are achieved for the following reasons: (i) Cooperative attack detection process executed by F-ADSs, S-ADSs, and T-ADS aims to monitor the main elements of network slicing to detect internal and external attacks while minimizing the false positive rates. More specifically, applying a two-layer FL algorithm in the cyber detection process gradually reduces misclassifications errors generated from false positives and false negatives, and (ii) in complementary, the proposed poisoning attack detection algorithm based on a non-cooperative game identifies the malicious defense systems that exhibit false detections against the legitimate monitored target, i.e., detect the normal targets as attackers and vice versa. Specifically, as explained in subsection 3.2, the poisoning attack could infect the training vectors and labels of the ML algorithm executed by the defense system, leading this system to perform a malicious action. As demonstrated in Theorem 2, when the equilibrium MFE is reached, almost all malicious defense systems are detected. Hence, the internal poisoning attack executed within the malicious defense system is detected. The optimum accuracy detection rates of (D)DoS, botnet, and poisoning attacks correspond to the tradeoff between high detection rates and low false positive rates.

4.2.2 *Overhead*

As shown in Figure 6, we evaluate the time overhead generated by our proposed cooperative defense systems while comparing it with related works [8, 13] by injecting the (D)DoS, botnet, and poisoning attacks. Here, we vary the amount of malicious traffic instigated by (D)DoS, botnet, and poisoning attacks from 15% to 40% of the total traffic. As shown in Figure 6, when the malicious traffic increases, our defense systems require low time to detect the malicious traffic accurately. However, even in a worst-case (i.e., 40% of traffic is malicious), the cooperative defense systems require a low time overhead to secure the network. From Figure 6, when the number of malicious traffic increases, our cooperative defense systems and the solution in [8] require low time overheads during the detection process compared to the solution in [13]. The low time overhead that our cooperative defense systems require to detect accurately (D)DoS, botnet, and poisoning attacks is mainly due to the

cooperation security process achieved between the different defense systems, namely F-ADSs, S-ADSs, and T-ADS.

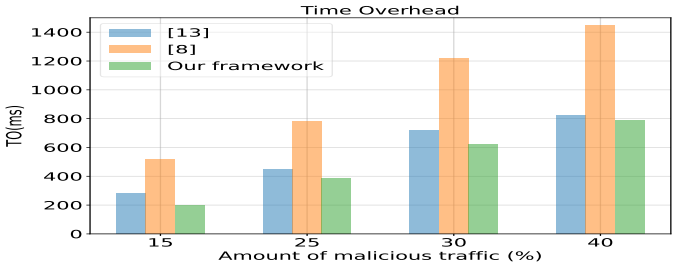


Fig. 6: Generated time overhead

5 Conclusion

In this research article, we have proposed trusted cooperative defense systems based on a two-layer FL and mean-field game to protect the 5G and beyond network slicing from the most dangerous attacks, such as DDoS and botnet, and enable robustness against poisoning attacks. Our experimental results show high accuracy detection against network and poisoning attacks, while low overhead is generated in training and deployment. As future work, we will further address the impact of non-IID on detecting poisoning attacks.

References

[1] M. Mekki, S. Arora, and A. Ksentini, "A Scalable Monitoring Framework for Network Slicing in 5G and Beyond Mobile Networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 413–423, 2021.

[2] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[3] 3GPP TS 23.501, " System Architecture for the 5G System; Stage 2," Mar 2018.

[4] R. F. Olimid and G. Nencioni, "5g network slicing: A security overview," *IEEE Access*, vol. 8, pp. 99 999–100 009, 2020.

[5] L. F. Maimó, Á. L. P. Gómez, F. J. G. Clemente, M. G. Pérez, and G. M. Pérez, "A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks," *Ieee Access*, vol. 6, pp. 7700–7712, 2018.

[6] A. Gupta, R. K. Jha, P. Gandotra, and S. Jain, "Bandwidth Spoofing and Intrusion Detection System for Multistage 5G Wireless Communication Network," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 618–632, 2017.

[7] H. Fang, X. Wang, and S. Tomasin, "Machine Learning for Intelligent Authentication in 5G and Beyond Wireless Networks," *IEEE Wireless Communications*, vol. 26, no. 5, pp. 55–61, 2019.

[8] H. Sedjelmaci, "Cooperative attacks detection based on artificial intelligence system for 5G networks," *Computers & Electrical Engineering*, vol. 91, p. 107045, 2021.

[9] Y. Hu, Y. Gao, and B. An, "Accelerating Multiagent Reinforcement Learning by Equilibrium Transfer," *IEEE transactions on cybernetics*, vol. 45, no. 7, pp. 1289–1302, 2014.

[10] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated Learning for Wireless Communications: Motivation, Opportunities, and Challenges," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 46–51, 2020.

[11] L. U. Khan, I. Yaqoob, N. H. Tran, Z. Han, and C. S. Hong, "Network slicing: Recent advances, taxonomy, requirements, and open research challenges," *IEEE Access*, vol. 8, pp. 36 009–36 028, 2020.

[12] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[13] Y. Liu, J. Peng, J. Kang, A. M. Iliyasu, D. Niyato, and A. A. Abd El-Latif, "A Secure Federated Learning Framework for 5G Networks," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 24–31, 2020.

[14] H. Chai, S. Leng, Y. Chen, and K. Zhang, "A Hierarchical Blockchain-enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 3975–3986, 2020.

[15] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," *Ieee Access*, vol. 7, pp. 41 525–41 550, 2019.

[16] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A Federated Self-learning Anomaly Detection System for IoT," in *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 756–767.

[17] N. Wang, J. Tang, and K. Zeng, "Spoofing Attack Detection in Mm-Wave and Massive MIMO 5G Communication," in *2019 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2019, pp. 1–5.

[18] I. H. Abdulqadder, S. Zhou, D. Zou, I. T. Aziz, and S. M. A. Akber, "Multi-layered intrusion detection and prevention in the SDN/NFV enabled cloud of 5G networks using AI-based defense mechanisms," *Computer Networks*, vol. 179, p. 107364, 2020.

[19] M. Hachimi, G. Kaddoum, G. Gagnon, and P. Illy, "Multi-stage Jamming Attacks Detection using Deep Learning Combined with Kernelized Support Vector Machine in 5G Cloud Radio Access Networks," in *2020 international symposium on networks, computers and communications (ISNCC)*. IEEE, 2020, pp. 1–5.

[20] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu, "Multi-Task Network Anomaly Detection using Federated Learning," in *Proceedings of the tenth international symposium on information and communication technology*, 2019, pp. 273–279.

[21] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "Performance Optimization of Federated Learning over Wireless Networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[22] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Model poisoning attacks in federated learning," in *Proc. Workshop Secur. Mach. Learn.(SecML) 32nd Conf. Neural Inf. Process. Syst.(NeurIPS)*, 2018, pp. 1–23.

[23] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated Optimization: Distributed Machine Learning for On-Device Intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[24] Z. Shang, M. Ma, and X. Li, "A Secure Group-Oriented Device-to-Device Authentication Protocol for 5G Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7021–7032, 2020.

[25] Y. Wang, F. R. Yu, M. Huang, A. Boukerche, and T. Chen, "Securing vehicular ad hoc networks with mean field game theory," in *Proceedings of the third ACM international symposium on Design and analysis of intelligent vehicular networks and applications*, 2013, pp. 55–60.

[26] S. Adlakha and R. Johari, "Mean field equilibrium in dynamic games with strategic complementarities," *Operations Research*, vol. 61, no. 4, pp. 971–989, 2013.

[27] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

Dr. Hichem Sedjelmaci is a Senior Developer System at Ericsson R&D France, working in Standardization and leading some security R&D activities, such as External Research Collaboration and Artificial Intelligence (AI) security activities. His research interests include Cyber defense systems (such IDS/IPS), Game theory, and Machine learning in the context of 5G, 6G, Vehicular, Drone, and lightweight IoT Networks.

Dr. Abdelwahab Boualouache is a research associate at the Faculty of Science, Technology, and Medicine (FSTM), University of Luxembourg. His current research covers security and privacy in mobile and wireless networks, mainly focusing on topics related to 5G and beyond 5G cellular networks and connected and automated vehicles.