UNIVERSITÀ
DEGLI STUDI
DI PALERMO

NDS
NETWORKING AND DISTRIBUTED SYSTEMS
RESEARCH GROUP

A Stackelberg Approach to Federated Learning for Malware Detection

Article

Accepted version

A. Augello, A. De Paola, M. Jestin, G. Lo Re

Proceedings of the Joint National Conference on Cybersecurity (ITASEC & SERICS 2025) - CEUR WORKSHOP PROCEEDINGS

Publisher: CEUR-WS

A Stackelberg Approach to Federated Learning for Malware Detection

Andrea Augello[1], Alessandra De Paola[1], Marena Jestin[1], and Giuseppe Lo Re[1]

Department of Engineering, University of Palermo, Italy
{andrea.augello01, alessandra.depaola, marena.jestin, giuseppe.lore}@unipa.it

Abstract

The widespread use of smart devices requires effective malware detection tools to ensure user security and privacy. The dynamic nature of the software ecosystem, characterized by data distribution changes, poses significant challenges to the long term sustainability of machine learning models for malware detection, requiring periodic updates to maintain their effectiveness. Additionally, collecting up-to-date information for training machine learning models in a centralized fashion is costly, time-consuming, and privacy-invasive. To address these shortcomings, this work proposes a Stackelberg game model to incentivize users to contribute to the training of a malware detection model through Federated Learning. The proposed model takes into account heterogeneous capabilities of the participants, allowing them to tune their contribution based on the quality and quantity of the data they can provide. Experimental results demonstrate that the proposed approach can ensure the effectiveness of the detection model over multiple years.

1 Motivation and Related Works

Smart devices are becoming increasingly integrated into our daily lives. As technology evolves, the scale and complexity of cyberattacks has made traditional security measures inadequate, necessitating the development of innovative strategies to effectively address emerging vulnerabilities [1]. Traditional malware detection techniques, such as signature-based and heuristic approaches, have become inadequate in addressing the complexities of modern cyber threats. These methods rely on predefined patterns or static code analysis, making them vulnerable to sophisticated attacks [19]. Specifically, polymorphic and metamorphic malware can alter their code or behavior to elude detection by such systems [7]. As a result, these conventional methods often produce high false-negative rates, especially when faced with new or dynamic forms of malware that have not been previously encountered [27].

Over the past decade machine learning algorithms have become the preferred threat detection approach [25], enabling systems to identify patterns and anomalies in data more efficiently than traditional methods [13]. By processing vast amounts of data, these systems can detect sophisticated attack patterns faster and more accurately [3]. Moreover, machine learning models are easier to adapt to emerging threats [20]. Despite their success, machine learning models frequently face a decline in performance after their deployment. This performance degradation is due to a phenomenon known as concept drift [2]. Concept drift arises when the statistical characteristics of the input data change over time, leading to discrepancies between the data the model was trained on and the data it encounters. Concept drift is a particularly serious issue in the cybersecurity domain, where the threat landscape is constantly evolving, and new malware variants are continually being developed [6]. This change can lead to a decrease in the model's accuracy, making it ineffective in detecting new threats [11]. Hence, it is essential to update the model periodically to ensure it remains effective in detecting new threats [21].

Furthermore, most existing malware detection systems that use machine learning rely on a centralized collection and annotation of data [9]. The process of labelling data is both expensive and time-intensive, creating a bottleneck in the development pipeline [22]. To reduce such overheads, crowdsourcing has been proposed as a viable solution for malware detection [4]. Users can collect and annotate data, which is then sent to a central server that elaborates the data to train the model. However, despite its potential, crowdsourcing introduces significant challenges regarding user privacy. Users may hesitate to share details with a central server about the applications installed on their devices since even a simple snapshot of installed apps can reveal sensitive personal information [26].

Decentralized alternatives, like Federated Learning, offer a solution by enabling collaborative model training without aggregating sensitive data on a central server [28]. In Federated Learning, each client retains its local training dataset, which is never shared with the server. Instead, clients compute updates to the global model maintained by the server, communicating only these updates rather than the raw data [17]. The effectiveness of such a learning model depends on the quality of updates provided by users, although it is not a given that users perceive the value of actively contributing to the functioning of the system [5]. By aligning clients' interests with the broader goal of collaborative model training, incentive mechanisms can help overcome the challenges of client engagement in Federated Learning systems [32].

Various studies have employed game theory to design incentive mechanisms that encourage clients to allocate their computational resources for Federated Learning [30]. For instance, Donahue et al. [10] analyzed the conditions under which rational client would be willing to participate in Federated Learning over multiple interactions with the server. While most works focus on ensuring that clients offer their computational capabilities to train a shared model [16, 31], this work leverages game theory to address the specific challenges of obtaining up-to-date training data to overcome concept drift in malware detection models. By incorporating a Stackelberg-based approach to designing an incentive mechanisms for federated learning, this work aims to ensure sustained client participation while adapting to concept drift. This approach enhances model robustness and accuracy in dynamic environments by aligning client incentives with the learning objectives of the system.

The main contributions of this work are:

- A Stackelberg game model for Federated Learning (FL) to prevent model ageing due to concept drift in malware detection.
- A novel game formulation that allows clients to tune their contribution based on the quantity of data they can provide and the effort associated with labelling it. Allowing clients to compensate for the potentially poor quality of labels they can supply with quantity ensures that even clients with limited labelling expertise can participate in the learning process.
- A novel strategy adaptation mechanism that allows the server to tune the reward in order to encourage client participation even in high-cost scenarios.
- The formal proof that proposed game model converges to a unique Nash equilibrium, ensuring system stability over time.
- Extensive experimental evaluations showing that, through the proposed incentive mechanism, it is possible to engage clients in the FL process for multiple years, maintaining the effectiveness of the model.

The remainder of this work is structured as follows: Section 2 describes the system model. Section 3 presents the optimal client strategy. Section 4 discusses the server reward policy. Section 5 provides the experimental evaluation. Finally, Section 6 concludes the paper.

2 System model

The system consists of a single central server and a set of N clients. The clients are responsible for training the local models using their own data and sending the local models to the server. The server is responsible for collecting the local models from the clients, aggregating them, and sending the global model back to the clients for further training according to the Federated Learning (FL) paradigm [18]. To incentivize the clients to collect high-quality data for training, the server offers them a monetary reward based on their contribution.

The interaction between the server and clients is modeled as a Stackelberg game, a game theory model useful in situations of sequential competition between two or more players, in which one player, named leader, moves first and the other players, named followers, respond to the first player's move [24]. The server acts as the leader and, in its opening move, announces a monetary reward for cooperative clients in terms of high-quality data to train their local model. The clients act as followers and respond by determining a suitable contribution to maximize their utility. The clients are fully rational agents, and engage in a non-cooperative subgame with the other clients to determine their optimal contribution.

The server and clients interact periodically, when the model needs to be updated on new data. For each of these updates, the server sets a reward and the clients decide how much data to contribute. The reward is given based on the quality of the data provided by the clients.

In the proposed model, the contribution θ of a client, referred to as their *quality index*, is linearly dependent on the size of the collected dataset $|D|$ and the quality of the labelling ρ, as shown in Eq. 1:

$$\theta = |D|(2\rho - 1). \tag{1}$$

This proposal is consistent with existing literature [12] showing that the generalization error introduced by label noise can be offset by increasing the dataset size. Experimental observations show that the accuracy of the global model is logarithmically dependent on the quality index, as shown in Fig. 1, which is in line with the results of previous studies [31]. Initially, increasing the quality index leads to significant improvements in the accuracy of the model, but these improvements become progressively less significant as the quality index increases.

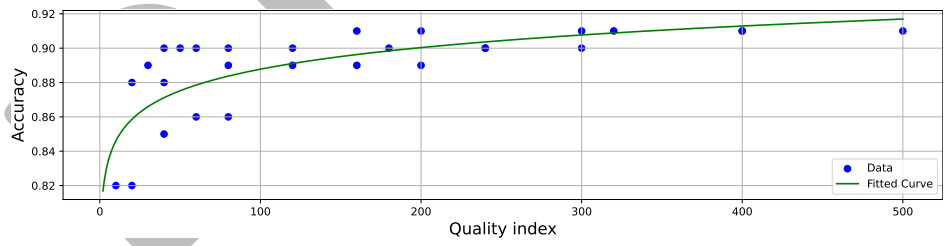

Figure 1: Relationship between global model's accuracy and quality index in the KronoDroid dataset [14]: $f(\boldsymbol{\theta}) = 0.0134 \log(8.1 \cdot 10^{26} \sum_{i=1}^{N} \theta_i + 7.69 \cdot 10^{27})$. The quality indexes are obtained by combining dataset size and label quality combinations such that $|D| \in [10, 500]$ and $\rho \in [0.5, 1]$.

3 Optimal client strategy

Each client contributes to the learning process by acquiring and labeling data, and receives a reward based on the quality and quantity of the data provided. The clients aim to maximize

their utility, defined as the difference between the reward obtained and the cost incurred. In their decision-making process, the clients can leverage knowledge from previous interactions.

3.1 Client policy

Client contribution cost model: The cost that client i incurs is given by the cost of acquiring data and the cost of labeling them. The cost of acquiring data scales linearly with the number or samples according to a parameter γ_i. Since not all samples are equally difficult to label, the cost of labeling an entire dataset is not linear to the target accuracy level [29]. The cost of labeling each sample is modeled as proportional to its difficulty. Assuming a maximum entropy distribution of the difficulty of the samples bounded in the $[m, M]$ range (i.e., an uniform distribution), the cost of labeling a fraction f of the samples, starting from the easiest, is proportional to definite the integral of the cumulative distribution function of the difficulty of the samples up to the f-th quantile:

$$\int_m^{f(M-m)+m} \int_m^x \frac{1}{M}\, dy\, dx = \frac{M-m}{2} f^2. \tag{2}$$

Since randomly assigning labels for a binary classification task yields 50% label accuracy, to ensure that a fraction ρ_i of samples have the correct label, the client will have to provide labels for a fraction of the samples equal to at least $2\rho_i - 1$, incurring in a cost proportional to $2(M-m)(\rho_i - 1/2)^2$. The resulting behavior can be interpreted as the client determining an upper bound on the difficulty of the samples they are willing to analyze. In practice, this means that at least a $2\rho_i - 1$ fraction of the samples undergoes costly manual annotation, while the remaining samples are labeled with an unreliable automatic procedure with negligible cost. The final cost of labeling a dataset of size $|D_i|$ up to a target accuracy of ρ is thus given by $|D_i|\beta_i(\rho_i - 1/2)^2$, with β_i being a proportionality constant that depends on the maximum difficulty of the samples and the client's labeling capabilities. Thus, the overall participation cost for client i is given by Eq. (3):

$$c_i(|D_i|, \rho_i) = |D_i|(\gamma_i + \beta_i(\rho_i - 1/2)^2). \tag{3}$$

Since the clients aim to maximize their utility, they will strive to choose the dataset size and the target accuracy that minimize the cost for any given target quality index θ_i. By fixing θ_i, Eq. (1) can be used to express the cost only in terms of the dataset size $|D_i|$:

$$c_i(|D_i|)|_{\theta_i} = |D_i|\gamma_i + \beta_i \frac{\theta_i^2}{4|D_i|}. \tag{4}$$

The cost function is minimized for $|D_i| = \theta_i \sqrt{\beta_i/(4\gamma_i)}$, where the first derivative of c_i is equal to zero and the second derivative is positive. The corresponding fraction of accurately labeled samples can be obtained by substituting the optimal $|D_i|$ in Eq. (1) and solving for ρ_i given a target quality index, resulting in $\rho_i = \frac{1}{2} + \frac{1}{\sqrt{\beta_i/\gamma_i}}$. In the following, it is assumed that $4\gamma_i \leq \beta_i$ so that $\frac{1}{\sqrt{\beta_i/\gamma_i}} \leq \frac{1}{2}$, as a labeling accuracy ρ_i above 100% is meaningless. In scenarios where this condition is not met, i.e. labeling one sample is cheaper than collecting four, the label accuracy is set as the maximum possible value $\rho_i = 1$. By substituting this value of ρ_i in Eq. (1) the dataset size to achieve a target quality index θ_i is $|D_i| = \theta_i$. Fig. 2 shows the cost of a unitary increase in the quality index as the labeling cost increases compared to the collection cost. If all clients were required to achieve a quality index of 1, the cost would increase linearly. Instead, through the proposed dynamic tuning of the quality parameters, the same quality index can be achieved with a sublinear dependency on the labeling cost.

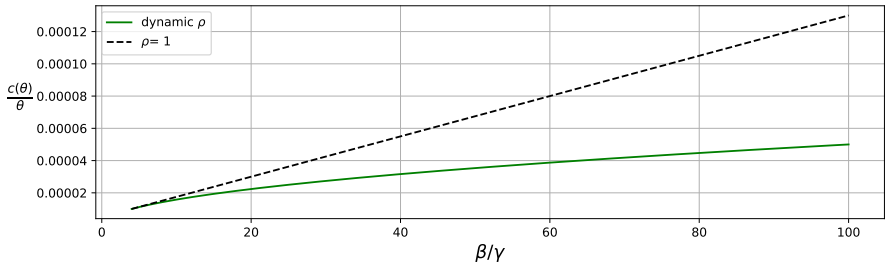

Figure 2: If all clients were required to achieve $\rho = 1$, the unitary contribution cost $\frac{c(\theta)}{\theta}$ would increase linearly with the labeling/collecting cost ratio β/γ. Instead, under the proposed formulation the same quality index can be achieved at a sublinear cost.

Optimal client contribution: Given the contribution θ_i of a client, its utility is defined as the difference between the reward obtained and the cost incurred, as shown in Eq. (5), with R being the total reward given by the server and $\theta = \sum_{j=1}^{N} \theta_j$ the total contribution of all clients:

$$u_i(\theta_i) = R\frac{\theta_i}{\theta} - c_i(\theta_i). \tag{5}$$

To find the optimal quality index of the i-th client, the derivative of the utility function with respect to θ_i is set as equal to zero. Since θ can be rewritten as $\theta = \theta_i + \theta_{-i}$, where θ_{-i} is the sum of the contributions of clients other than i, i.e., $\theta_{-i} = \sum_{j \neq i} \theta_j$, it is possible to isolate the contribution of individual clients:

$$\frac{du_i}{d\theta_i} = \frac{R\theta_{-i}}{(\theta_i+\theta_{-i})^2} - \sqrt{\gamma_i\beta_i} = 0, \quad \text{satisfied by} \quad \theta_i = \theta_{-i}\left(\sqrt{\frac{R}{\theta_{-i}\sqrt{\gamma_i\beta_i}}} - 1\right) \quad \text{if} \quad \frac{R}{\sqrt{\gamma_i\beta_i}} > \theta_{-i}. \tag{6}$$

Intuitively, the condition in Eq. (6) means that the i-th client should only participate if the current expected reward per unit of contribution is greater than its unitary contribution cost, otherwise the expected utility would be negative. This is illustrated in Fig. 3, where the cost for a client to achieve a unitary quality index is plotted against the average reward per unit of contribution from the other clients across multiple client interactions. For all the interactions where the unit reward is lower than the unit cost, the client abstains from participating.

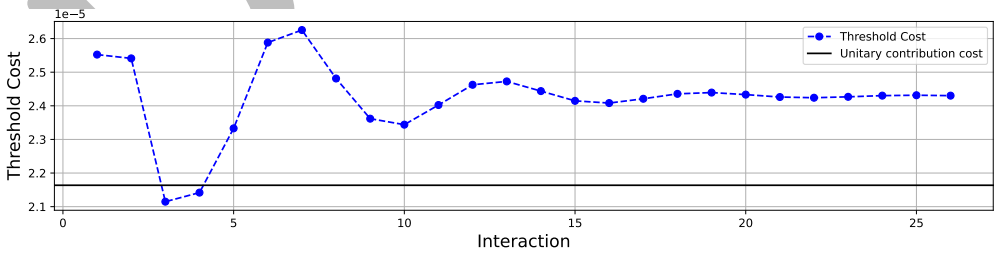

Figure 3: When the cost per unit of contribution of a client exceeds the unit reward, the client's utility becomes negative, disincentivising participation.

The optimal quality index in Eq. (6) depends on the total contribution of the other clients θ_{-i}. However, since the clients are not cooperating, the i-th client does not have direct access to this information and must estimate it. When the system converges, each client can estimate

the total contribution of the other clients by observing the reward received compared to the total reward in the previous interaction.

For the initial estimate, the client has no information about the other clients, so it assumes that all clients are homogeneous and adopt the same policy to determine their quality index. In this case, in each client's formulation, $\theta_{-i} = (N-1)\theta_i$ and $u_i(\theta_i) = \frac{R\theta_i}{(N-1)\theta_i} - c_t(\theta_i)$, hence and the quality index maximizing u_i will be given by Eq. (7):

$$\theta_i = \frac{R(N-1)}{N^2\sqrt{\gamma_i\beta_i}}. \tag{7}$$

3.2 Nash Equilibrium

The interaction among the clients is a non-cooperative game, where each client acts independently without collaborating with others, with the aim of maximizing his own utility. Therefore, it is essential to determine whether a Nash equilibrium exists for the clients' utilities. Nash equilibrium is a stable point in the game where no follower can improve his utility by changing his strategy, assuming that the other followers keep their strategies unchanged.

Theorem 1. *At least one Nash Equilibrium exists for the client's utility function in Eq. (5).*

Proof. For a Nash Equilibrium to exist, the utility function of the client must be concave with respect to the client's strategy, i.e., the choice of quality index θ_i [23]. The first and second derivative of the utility function with respect to θ_i are computed to prove the concavity:

$$\frac{du_i}{d\theta_i} = \frac{R\theta_{-i}}{(\theta_i+\theta_{-i})^2} - \sqrt{\gamma_i\beta_i} \quad \text{and} \quad \frac{d^2u_i}{d\theta_i^2} = -\frac{2R\theta_{-i}}{(\theta_i+\theta_{-i})^3}. \tag{8}$$

Since second derivative of $u_i(\theta_i)$ with respect to θ_i, given by Eq. (8), is negative for all the admissible (positive) values of θ_i, the utility function is concave. □

In order to ensure that the clients will converge to the Nash equilibrium, there must be a unique equilibrium point. Let $g_i(\boldsymbol{\theta})$ be the response function representing the optimal quality index of client i given the quality indexes of the other clients, following Eq. (6). If $g_i(\boldsymbol{\theta})$ is a standard function then the proposed game has a unique Nash equilibrium [8].

Definition 1. A function $g_i(\boldsymbol{\theta})$ is standard if, for all $\boldsymbol{\theta} \geq 0$, the following properties are satisfied:

1. Positivity: $g_i(\boldsymbol{\theta}) > 0$.
2. Monotonicity: $\boldsymbol{\theta} \geq \boldsymbol{\theta'} \rightarrow g_i(\boldsymbol{\theta}) \geq g_i(\boldsymbol{\theta'})$; this means that, in reaction to an increase in contribution performed by all the other clients, the i-th client also increases its contribution.
3. Scalability: $\forall\lambda > 1, \lambda g_i(\boldsymbol{\theta}) \geq g_i(\lambda\boldsymbol{\theta})$, when all clients uniformly scale up their contribution, the i-th client's response function will increase less than linearly.

Lemma 1. $\sqrt{\frac{R}{\sqrt{\gamma_i\beta_i}}} - 2\sqrt{\theta_{-i}} > 0$ *always holds if the following condition holds:*

$$\sum_{j=1}^{N}\sqrt{\gamma_j\beta_j} > 2(N-1)\sqrt{\gamma_i\beta_i}. \tag{9}$$

Proof. By setting the first derivative of the client's utility function in Eq. (6) to zero, the following relationship between the quality indexes of the clients can be obtained:

$$\frac{\theta_{-i}}{(\theta_{-i}+\theta_i)^2} = \frac{1}{R}\sqrt{\beta_i\gamma_i}. \tag{10}$$

Summing up on both sides of Eq. (10) for all clients yields the relationship in Eq. (11):

$$\sum_{j=1}^{N} \frac{\theta_{-j}}{(\theta_{-j}+\theta_j)^2} = \sum_{j=1}^{N} \frac{1}{R}\sqrt{\beta_j \gamma_j}. \tag{11}$$

Hence, the total contribution in terms of the costs of all the clients is rewritten as Eq. (12):

$$\theta = \frac{R(N-1)}{\sum_{j=1}^{N}\sqrt{\beta_j\gamma_j}}. \tag{12}$$

By substituting Eq. (12) into Eq. (10), it is possible to derive that

$$R^2(N-1)^2\sqrt{\beta_i\gamma_i} = R\theta_{-i}\left(\sum_{j=1}^{N}\sqrt{\beta_j\gamma_j}\right)^2. \tag{13}$$

If Eq. (9) holds, by computing the square root of both sides of the equation, it is possible to prove that $\sqrt{R\sqrt{\beta_i\gamma_i}}(N-1) > 2\sqrt{\theta_{-i}}(N-1)\sqrt{\beta_i\gamma_i}$, which simplifies to $\sqrt{\frac{R}{\sqrt{\beta_i\gamma_i}}} - 2\sqrt{\theta_{-i}} > 0$ □

Theorem 2. *The client response function is standard, and the proposed game has a unique Nash Equilibrium if the condition in Eq. (9) holds.*

Proof. The positivity is obviously satisfied by the client response function under the condition in Eq. (6).

• The client response function is monotone for $\theta_i > 0$ if Eq.(9) holds, i.e., $\boldsymbol{\theta} \geq \boldsymbol{\theta'} \rightarrow g_i(\boldsymbol{\theta}) - g_i(\boldsymbol{\theta'}) \geq 0$:

$$\begin{aligned} g_i(\boldsymbol{\theta}) - g_i(\boldsymbol{\theta'}) &= \sqrt{\tfrac{R}{\sqrt{\beta_i\gamma_i}}}(\sqrt{\theta_{-i}} - \sqrt{\theta'_{-i}}) - (\theta_{-i} - \theta'_{-i}) \\ &= (\sqrt{\theta_{-i}} - \sqrt{\theta'_{-i}})\left(\sqrt{\tfrac{R}{\sqrt{\beta_i\gamma_i}}} - (\sqrt{\theta_{-i}} + \sqrt{\theta'_{-i}})\right) \geq (\sqrt{\theta_{-i}} - \sqrt{\theta'_{-i}})\left(\sqrt{\tfrac{R}{\sqrt{\beta_i\gamma_i}}} - 2\sqrt{\theta_{-i}}\right), \end{aligned} \tag{14}$$

which is positive according to Lemma 1.

• The client response function is scalable: $\forall \lambda > 1, \lambda g_i(\boldsymbol{\theta}) - g_i(\lambda\boldsymbol{\theta}) \geq 0$:

$$\lambda g_i(\boldsymbol{\theta}) - g_i(\lambda\boldsymbol{\theta}) = \lambda\sqrt{\frac{R\theta_{-i}}{\sqrt{\beta_i\gamma_i}}} - \lambda\theta_{-i} - \sqrt{\frac{R\lambda\theta_{-i}}{\sqrt{\beta_i\gamma_i}}} + \lambda\theta_{-i} = (\sqrt{\lambda}-1)\sqrt{\frac{R\theta_{-i}}{\sqrt{\beta_i\gamma_i}}}. \tag{15}$$

Since $\lambda > 1$, both terms are positive, thus $\lambda g_i(\boldsymbol{\theta}) \geq g_i(\lambda\boldsymbol{\theta'})$ and the response function is scalable. Hence the client response function $g_i(\boldsymbol{\theta})$ is a standard function which, as demonstrated by Deligiannis et al. [8], is a sufficient condition for the existence of a unique Nash Equilibrium. □

4 Server reward policy

The server aims to maximize its utility by setting an appropriate reward. The utility, denoted as V, is defined as the difference between the satisfaction derived from the aggregated global model and the total reward R distributed to the followers. The server's satisfaction depends on the accuracy of the new global model, which grows concavely as the total contribution of the followers increases. To represent this dependency, a logarithmic function is employed to describe the relationship between the model's accuracy and the followers' quality indexes:

$$V(R) = \sigma \log\left(1 + \lambda\sum_{i=1}^{N}\theta_i(R)\right) - R. \tag{16}$$

The Stackelberg equilibrium can be found by solving the above non-linear optimization problem. Let θ_i^* be unique Nash equilibrium obtained by the clients when the server offers a reward R. The server has to maximize Eq. (16) a priori to find its unique optimal reward R^* and announce it to the clients. Given a reward R, each client sets their contribution θ_i^* according to Eq. (6). The server, however, does not have direct access to the parameters of the clients' cost function, i.e., γ_i and β_i. Instead, assuming that the costs have not changed significantly, the server can estimate the unitary contribution cost of client i, that is $\sqrt{\beta_i \gamma_i}$, from the behavior of the clients in the previous two rounds, denoted with the $(r-1)$ and $(r-2)$ subscripts respectively, through Eq. (17).

$$\sqrt{\beta_i \gamma_i} = \frac{R_{(r-1)} \theta_{-i,(r-2)}}{\left(\theta_{i,(r-1)} + \theta_{-i,(r-2)}\right)^2}. \tag{17}$$

Hence, the server can estimate the expected utility given a reward R solely by relying on the previous interactions by substituting Eq. (6) into Eq. (16), with the cost estimated by Eq. (17):

$$V(R) = \sigma \log \left(1 + \lambda \frac{\theta_{(r-1)}}{N} \sqrt{\frac{R}{R_{(r-1)}}} \sum_{i=1}^{N} \sqrt{\frac{\theta_{-i,(r-1)}}{\theta_{-i(r-2)}}}\right) - R. \tag{18}$$

The server can then find the optimal reward R^* by maximizing the utility function $V(R)$. Since the utility function is concave (negative second derivative), the optimal reward can be found by calculating the first derivative, shown in Eq. (19), and setting it equal to zero.

$$\frac{dV}{dR} = \frac{\sigma \frac{\lambda \theta_{(r-1)}}{N} \sqrt{\frac{R}{R_{(r-1)}}} \sum_{i=1}^{N} \sqrt{\frac{\theta_{-i,(r-1)}}{\theta_{-i(r-2)}}}}{2 \frac{\lambda \theta_{(r-1)}}{N} R \sqrt{\frac{R}{R_{(r-1)}}} \sum_{i=1}^{N} \sqrt{\frac{\theta_{-i}}{\theta_{-i,(r-2)}}} + 2R} - 1. \tag{19}$$

For the sake of conciseness, the following parameters are used: $A = \frac{\theta_{(r-1)}}{N}$, which is the average contribution in the previous round, and $B = \sum_{i=1}^{N} \sqrt{\frac{\theta_{-i,(r-1)}}{\theta_{-i(r-2)} R_{(r-1)}}}$. Eq. (19) is thus rewritten as $\frac{\sigma \lambda A B \sqrt{R}}{2 \lambda A B \sqrt{R} + 2R} - 1$, which has a unique positive solution represented in Eq. (20), chosen as a reward by the server to maximize its utility:

$$R^* = \frac{\sigma}{2} + \frac{R_{(r-1)} + \sqrt{R_{(r-1)} \left(2\sigma\lambda^2 A^2 B^2 + R_{(r-1)}\right)}}{2\lambda^2 A^2 B^2}. \tag{20}$$

Since in the first interaction the server has no information about the clients, it assumes that all clients are homogeneous and utilizes an estimate of the average costs instead. The utility is thus formulated as $V(R) = \sigma \log \left(1 + \lambda \sum_{i=1}^{N} \frac{R(N-1)}{N^2 \sqrt{\beta\gamma}}\right) - R$, which is concave with respect to R and can easily be maximized analytically.

5 Experimental evaluation

To assess the proposed approach, a multi-layer perceptron was considered as model to be learned. The data sampled by clients are derived from the KronoDroid dataset [14], a publicly available real-world dataset covering the period from 2008 to 2020. The KronoDroid dataset, consisting of 28,745 Android malware samples and 35,256 benign samples, sorted according to their "last modification" date, provides a rich and dynamic environment for assessing the

system's capabilities in maintaining model performances over multiple years. The dataset was divided into 26 non-overlapping subsets covering three months each, from 2012 to 2018. After each three-month period, the server announces a reward to the ten clients, who then decide their contribution effort acquiring samples from the current subset and labeling them. The model accuracy and the server's utility are computed according to the total client contribution.

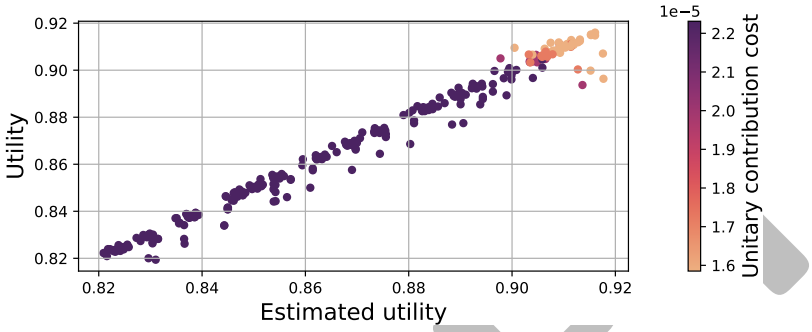

Figure 4: The server's expected utility from Eq. (18) is consistent with the actual utility obtained from the clients' contributions.

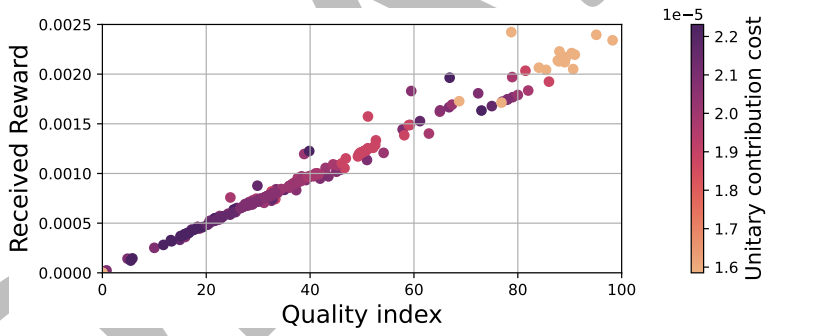

Figure 5: As their unitary contribution cost decreases, clients contribute more and achieve higher rewards.

Model soundness: In order to ensure that the server can effectively estimate the client's contributions and provide fair and consistent rewards, the mathematical model was validated by comparing the theoretical utility expected by the server with the actual utility (Fig. 4), and the reward obtained by the clients with their provided quality index (Fig. 5). As a general trend, clients with lower data acquisition costs achieve higher quality indexes, enabling them to obtain higher rewards. Additionally, if all clients have high costs, the server will obtain lower utility. The linear relationship in these plots confirms that the server estimates are consistent with the actual outcomes, and that the clients receive consistent and fair rewards for their contributions. The difference in reward outcome for heterogeneous clients is further highlighted in Fig. 6, where it is possible to observe that lower data acquisition costs lead clients to provide higher quality indexes while incurring in lower costs, thus achieving a reward inversely proportional to their unitary cost.

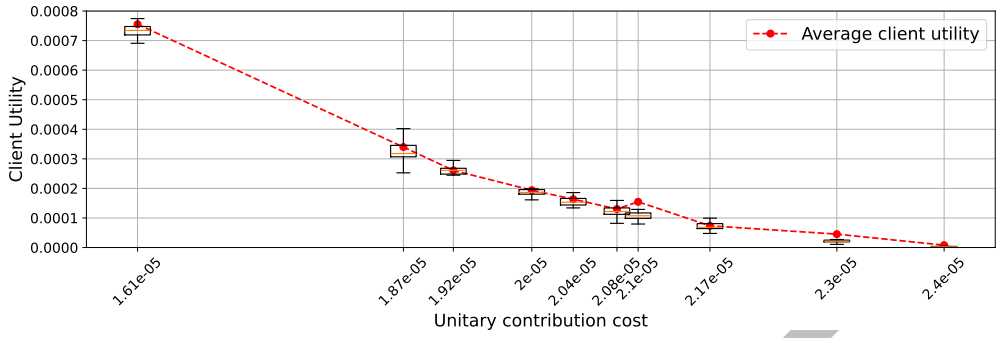

Figure 6: Clients with lower unit costs achieve higher quality indexes, leading to higher rewards, as highlighted in Fig. 5, and greater client utility. This is because lower costs enable more efficient contributions, which in turn improve system performance and incentivize clients with better rewards.

Effectiveness over time: The model's accuracy and the server's utility were evaluated over multiple interactions to assess the system's ability to maintain the model's performance over time. Each client's contribution converges over the interactions to a value that depends on their data acquisition and labeling costs, as shown in Fig. 7. As can be seen in Fig. 8, the model's accuracy remains stable over time, despite the concept drift in the dataset, thanks to the clients' contributions. If acquiring labeled data is too expensive the performance of the model will settle on a lower accuracy (∼86%), while if the clients can provide high-quality data at a lower cost, the model will achieve a higher accuracy (∼92%). These values are consistent with state-of-the-art results with this dataset when concept drift is effectively managed [15], thus showing the feasibility of using an incentive mechanism for performing data collection through crowdsourcing to maintain the effectiveness of malware detection models over time.

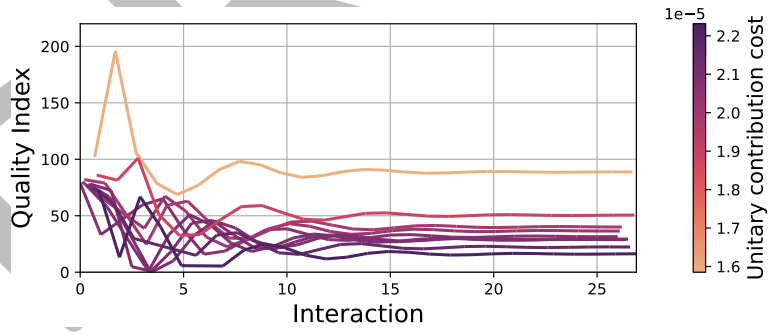

Figure 7: The quality index of the clients converges over the interactions. Lower unitary costs let the clients achieve higher quality indexes.

6 Conclusions

It has been widely demonstrated that machine learning models are effective at detecting malware, but lose effectiveness over time due to concept drift. To maintain the effectiveness of the model, it

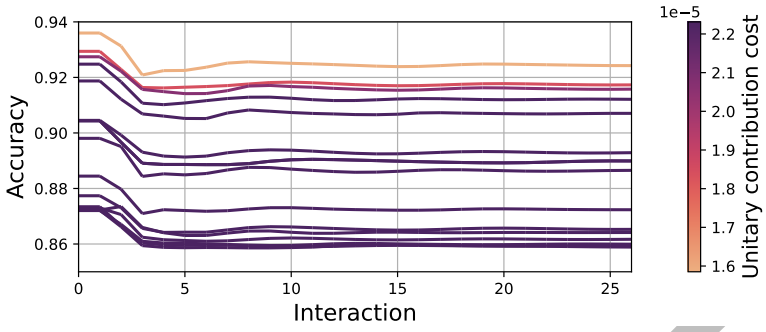

Figure 8: The model accuracy remains stable over time thanks to the client contributions. Higher client costs result in lower server utility.

is necessary to continuously update the model with new data. Distributed computing paradigms such as Federated Learning can be adopted to directly involve users in the data collection process required to maintain the model up-to-date. However, without adequate incentives, the clients may not be willing to provide high-quality data, leading to model degradation. This work proposes a Stackelberg game model for Federated Learning to incentivize the clients to collect high-quality data, enabling the long-term feasibility of employing machine learning models for malware detection despite the concept drift issue. Clients can decide how much data to collect and how much effort to put into labeling it to maximize their net utility, while the server decides the overall reward to maximize the contribution of the clients. The Stackelberg game was shown to possess a stable Nash equilibrium, and the model was experimentally shown to be capable of preserving the effectiveness of the model over multiple years. In the current version of the system, the server relies on honest clients to provide accurate quality indexes. Future work will investigate strategies that allow the server to infer the quality indexes of the clients, and meta-learning strategies to extract more value from poorly-annotated data.

Acknowledgements This work was partially supported by the AMELIS project, within the project FAIR (PE0000013), and by the ADELE project, within the project SERICS (PE00000014), both under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

Declaration on Generative AI The authors have not employed any Generative AI tools.

References

[1] Vipindev Adat and Brij B Gupta. Security in internet of things: issues, challenges, taxonomy, and architecture. *Telecommunication Systems*, 67:423–441, 2018.

[2] Vincenzo Agate, Alessandra De Paola, Salvatore Drago, Pierluca Ferraro, and Giuseppe Lo Re. Enhancing iot network security with concept drift-aware unsupervised threat detection. In *2024 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2024.

[3] Mostofa Ahsan, Kendall E Nygard, Rahul Gomes, Md Minhaz Chowdhury, Nafiz Rifat, and Jayden F Connolly. Cybersecurity threats and their mitigation approaches using machine learning—a review. *Journal of Cybersecurity and Privacy*, 2(3):527–555, 2022.

[4] Andrea Augello, Alessandra De Paola, and Giuseppe Lo Re. M2FD: Mobile malware federated detection under concept drift. *Computers & Security*, page 104361, 2025.

[5] Andrea Augello, Ashish Gupta, Giuseppe Lo Re, and Sajal K Das. Tackling selfish clients in federated learning. In *27th European Conference on Artificial Intelligence (ECAI 2024)*, pages 1888–1895. IOS Press, 2024.

[6] Federico Barbero, Feargus Pendlebury, Fabio Pierazzi, and Lorenzo Cavallaro. Transcending transcend: Revisiting malware classification in the presence of concept drift. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 805–823. IEEE, 2022.

[7] Christian Catalano, Andrea Chezzi, Mario Angelelli, and Franco Tommasi. Deceiving ai-based malware detection through polymorphic attacks. *Computers in Industry*, 143:103751, 2022.

[8] Anastasios Deligiannis, Anastasia Panoui, Sangarapillai Lambotharan, and Jonathon A Chambers. Game-theoretic power allocation and the nash equilibrium analysis for a multistatic mimo radar network. *IEEE Transactions on Signal Processing*, 65(24):6397–6408, 2017.

[9] Xiaoheng Deng, Zhe Wang, Xinjun Pei, and Kaiping Xue. Transmalde: An effective transformer based hierarchical framework for iot malware detection. *IEEE Transactions on Network Science and Engineering*, 2023.

[10] Kate Donahue and Jon Kleinberg. Optimality and stability in federated learning: A game-theoretic approach. *Advances in Neural Information Processing Systems*, 34:1287–1298, 2021.

[11] Damien Warren Fernando and Nikos Komninos. Fesad ransomware detection framework with machine learning using adaption to concept drift. *Computers & Security*, 137:103629, 2024.

[12] Ruijiang Gao and Maytal Saar-Tsechansky. Cost-accuracy aware adaptive labeling for active learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2569–2576, 2020.

[13] Daniel Gibert, Carles Mateu, and Jordi Planes. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. *Journal of Network and Computer Applications*, 153:102526, 2020.

[14] Alejandro Guerra-Manzanares, Hayretdin Bahsi, and Sven Nõmm. Kronodroid: time-based hybrid-featured dataset for effective android malware detection and characterization. *Computers & Security*, 110:102399, 2021.

[15] Alejandro Guerra-Manzanares, Marcin Luckner, and Hayretdin Bahsi. Android malware concept drift using system calls: detection, characterization and challenges. *Expert Systems with Applications*, 206:117200, 2022.

[16] Suhan Jiang and Jie Wu. A reward response game in the federated learning system. In *2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, pages 127–135. IEEE, 2021.

[17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[18] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[19] Andreas Moser, Christopher Kruegel, and Engin Kirda. Limits of static analysis for malware detection. In *Twenty-third annual computer security applications conference (ACSAC 2007)*, pages 421–430. IEEE, 2007.

[20] Ali Bou Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Mohamad Dakalbab. Machine learning for anomaly detection: A systematic review. *IEEE Access*, 9:78658–78700, 2021.

[21] Feargus Pendlebury, Fabio Pierazzi, Roberto Jordaney, Johannes Kinder, and Lorenzo Cavallaro. {TESSERACT}: Eliminating experimental bias in malware classification across space and time. In *28th USENIX security symposium (USENIX Security 19)*, pages 729–746, 2019.

[22] Valerian Rey, Pedro Miguel Sánchez Sánchez, Alberto Huertas Celdrán, and Gérôme Bovet.

Federated learning for malware detection in iot devices. *Computer Networks*, 204:108693, 2022.

[23] J Ben Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica: Journal of the Econometric Society*, pages 520–534, 1965.

[24] Tim Roughgarden. Stackelberg scheduling strategies. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 104–113, 2001.

[25] Aya H Salem, Safaa M Azzam, OE Emam, and Amr A Abohany. Advancing cybersecurity: a comprehensive review of ai-driven detection techniques. *Journal of Big Data*, 11(1):105, 2024.

[26] Suranga Seneviratne, Aruna Seneviratne, Prasant Mohapatra, and Anirban Mahanti. Predicting user traits from a snapshot of apps installed on a smartphone. *ACM SIGMOBILE Mobile Computing and Communications Review*, 18(2):1–8, 2014.

[27] Jagsir Singh and Jaswinder Singh. A survey on machine learning-based malware detection in executable files. *Journal of Systems Architecture*, 112:101861, 2021.

[28] Yongxin Tong, Yansheng Wang, and Dingyuan Shi. Federated learning in the lens of crowdsourcing. *IEEE Data Eng. Bull.*, 43(3):26–36, 2020.

[29] Sudheendra Vijayanarasimhan and Kristen Grauman. What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *2009 IEEE conference on computer vision and pattern recognition*, pages 2262–2269. IEEE, 2009.

[30] Rongfei Zeng, Chao Zeng, Xingwei Wang, Bo Li, and Xiaowen Chu. Incentive mechanisms in federated learning and a game-theoretical approach. *IEEE Network*, 36(6):229–235, 2022.

[31] Yufeng Zhan, Peng Li, Zhihao Qu, Deze Zeng, and Song Guo. A learning-based incentive mechanism for federated learning. *IEEE Internet of Things Journal*, 7(7):6360–6368, 2020.

[32] Yufeng Zhan, Jie Zhang, Zicong Hong, Leijie Wu, Peng Li, and Song Guo. A survey of incentive mechanism design for federated learning. *IEEE Transactions on Emerging Topics in Computing*, 10(2):1035–1044, 2021.